A Lösungen der Übungsaufgaben

Im Folgenden sind Lösungen für die am Ende der einzelnen Kapitel formulierten Aufgaben zusammengestellt. Es handelt sich dabei nur um Lösungsvorschäge ohne Anspruch auf Vollständigkeit zur Anregung weiterer eigener Überlegungen.

A.1 Kapitel 1

Lösung A.1.1: Durch einfaches Ausmultiplizieren ergibt sich:

a) $f(h) = 4(h^2+h)^2 - 4h^4 = 4h^4 + 8h^3 + 4h^2 - 4h^4 = O(h^2)\,.$

b) $g(n) = 4(n^2+n)^2 - 4n^4 = 4n^4 + 8n^3 + 4n^2 - 4n^4 = O(n^3)\,.$

c) Anwendung der Taylor-Formel ergibt:

$$f(h) = \frac{e^h - e^{-h}}{2h} - 1 = \frac{1}{2h}\int_{-h}^{h} \{e^x - 1\}\,dx = \frac{1}{2h}\int_{-h}^{h} \Big\{x + \frac{x^2}{2}e^{\eta}\Big\}\,dx = O(h^2).$$

d) Durch Ausmultiplizieren sieht man

$$G_n(x) := \frac{1-e^{-nx}}{1-e^{-x}} = 1 + e^{-x} + e^{-2x} + \cdots + e^{-(n-1)x}$$

und

$$G_n'(x) = -e^{-x} - 2e^{-2x} - \cdots - (n-1)e^{-(n-1)x} \le 0, \quad x \ge 0.$$

Die Funktion $G_n(x)$ ist also monoton fallend und folglich ergibt sich (Regel von l'Hospital):

$$G_n(x) \le G_n(0) = \lim_{x \searrow 0} \frac{1-e^{-nx}}{1-e^{-x}} = n.$$

Dies liefert $g(n) = n = O(n)\,.$

e) Mit der Regel von l'Hospital erhalten wir für jedes $\alpha > 0$:

$$\lim_{h\to 0} h^{\alpha}\ln(h) = \lim_{h\to 0} \frac{\ln(h)}{h^{-\alpha}} = \lim_{h\to 0} \frac{h^{-1}}{-\alpha h^{-\alpha-1}} = \lim_{h\to 0} \frac{h^{\alpha}}{-\alpha} = 0,$$

aber für $\alpha = 0$:

$$\lim_{h\to 0} \ln(h) = \infty.$$

Folglich ist $1/\ln(h) = o(1)\,.$

Lösung A.1.2: a) Nach Definition der Konditionszahlen gilt mit $f(x_1, x_2) = x_1/x_2$:

$$k_1 = \frac{\partial f}{\partial x_1}\frac{x_1}{f} = \frac{1}{x_2}\frac{x_1 x_2}{x_1} = 1, \quad k_y = \frac{\partial f}{\partial y}\frac{y}{f} = -\frac{x_1}{x_2^2}\frac{x_2^2}{x_1} = -1, \quad \alpha = 1.$$

$$\left|\frac{\Delta x_i}{x_i}\right| \le \varepsilon \ \ (i=1,2) \quad \Rightarrow \quad \left|\frac{\Delta f}{f}\right| \le 2\varepsilon + o(\varepsilon).$$

Die Division ist also generell gut konditioniert.

b) Analog folgt mit $f(x_1,x_2) = x_1^{x_2} = e^{x_2 \ln(x_1)}$:

$$k_1 = \frac{\partial f}{\partial x_1}\frac{x_1}{f} = x_2 x_1^{x_2-1}\frac{x_1}{x_1^{x_2}} = x_2, \qquad k_2 = \frac{\partial f}{\partial x_2}\frac{x_2}{f} = \ln(x_1) f \frac{x_2}{f} = \ln(x_1)x_2.$$

Die Potenzbildung ist schlecht konditioniert für $x_2 \gg 1$ oder im Fall $x_2 \neq 0$ für $x_1 \ll 1$ oder $x_1 \gg 1$.

Die Operation $f(x) = 1/x$ ist stets gut konditioniert. Die Operation $f(x) = \sqrt{x}$ ist ebenfalls gut konditioniert (Fehlerdämpfung).

Lösung A.1.3: a) Die Konditionionierung der Auswertung der Funktion $f(x)$ ist unabhängig von ihrer Darstellung in der Form $a(x)$ oder $b(x)$:

$$k = \frac{\partial f(x)}{\partial x}\frac{x}{f(x)} = \left|2 - \frac{3x+4x^2}{(1+x)(1+2x)}\right| \approx 2, \quad 0 < |x| \ll 1.$$

Die numerische Aufgabe, $f(x)$ zu berechnen, ist also für Argumente $0 < |x| \ll 1$ gut konditioniert.

b) Der Regel "Schlecht konditionierte Operationen vor gut konditionierten ausführen!" folgend, sollte die Darstellung $b(x)$ verwendet werden. Bei $a(x)$ droht Auslöschung.

Lösung A.1.4: Mit der Maschinengenauigkeit eps gilt für $x, y \in A$:

$$x \oplus y = x, \quad \text{für } |y| \le \frac{|x|}{2}\text{eps}.$$

Dies legt den folgenden Test nahe: $k = 1,2,3,\ldots$:

$$1 + 2^{-k} = 1 \quad \Rightarrow \quad 2^{1-k} \approx \text{eps}.$$

Z. B. ergibt sich so auf einem (ehemaligen) Atlon PC-Prozessor (1.4 GHz Taktung) unter MATLAB bei Verwendung des Types "double precision" (wie zu erwarten) eps $\approx 2^{-53} \approx 10^{-16}$.

Lösung A.1.5 (Praktische Aufgabe): a) Die Maschinengenauigkeit wird bestimmt aus der Beziehung:

$$x \circledast y = (x * y)(1+\varepsilon), \quad x, y \in A, \ \ |\varepsilon| \le \text{eps}.$$

Dies legt den folgenden Test nahe: $k = 1,2,3,\ldots$:

$$1 + 2^{-k} = 1 \quad \Rightarrow \quad 2^{1-k} \approx \text{eps}.$$

Auf einem (alten) Atlon 3500+ Prozessor unter MATLAB ergibt sich so bei Verwendung des Types "double precision" (wie zu erwarten) eps $\approx 2^{-53} \approx 10^{-16}$.

MATLAB-Programm: Bestimmung der Maschinengenauigkeit

```
clear
a = 1.;
c = 1.;
i = 0;
while c ~= 0
   i = i+1
   b = 10^(-i);
   c = a + b;
   c = c - a
end
```

b) Taylor-Summen der Exponentialfunktion

$$e^x \approx T_n(x) = \sum_{k=0}^{n} \frac{x^k}{k!}.$$

Relativer Fehler für $n \in [0, 20]$ für die Argumente $x \in \{10, 1, -1, -10\}$: Die schlechten Ergebnisse für negative x-Werte sind dadurch bedingt, dass für $x = -10$ $e^{-10} \ll 1$ ist. Daher bietet sich in diesem Fall die Auswertung

$$e^{-x} \approx (T_n(x))^{-1}$$

an, welche eine akzeptable Approximation für alle x-Werte liefert.

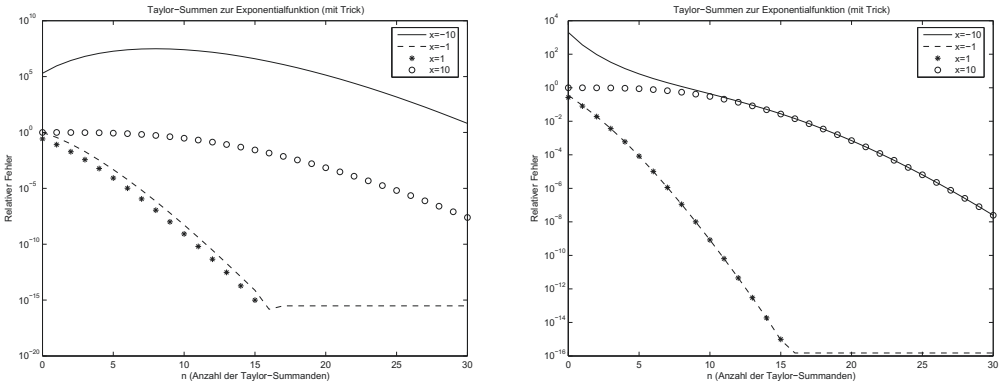


Abbildung A.1: *Relative Fehler der Taylor-Approximation:* $T_n(x)$ *(links),* $T_n(-x)^{-1}$ *(rechts)*

MATLAB-Programme:

a) Normale Taylor-Summe

```
    function erg = taylorsumme(x,n)
% Berechne den Vektor x^k, k=1,...,n:
    N = 0:n;
    X = x.^N;
% Teile die Eintraege durch k!:
    X = X./factorial(N);
% Erhalte das Ergebnis als Summe ueber alle Eintraege:
    erg = sum(X);
    return
```

b) Reziproke Taylor-Summe

```
     function erg = trickytaylorsumme(x, n)
% Berechne den Vektor x^k, k=1,...,n:
     N = 0:n;
     if x<0
        X = (-x).^N;
     else
        X = x.^N;
     end
% Teile die Eintraege durch} k!:
     X = X./factorial(N);
% Erhalte das Ergebnis als Summe ueber alle Eintraege}:
     erg = sum(X);
     if x<0
        erg = 1./erg;
     end
     return
```

c) Relativer Fehler der Taylor-Approximation

```
     x = [-10,-1,1,10];
     n = 1:20;
     for k=1: length(x)
        for i=1:20
        rel(k,i) =  abs(taylorsumme(x(k),i) - exp(x(k)))/exp(x(k));
        end
     end
     semilogy(n,rel)
     xlabel('n (Anzahl der Taylor-Summanden)')
     ylabel('Relativer Fehler')
     title('Taylor-Summen zur Exponentialfunktion')
     legend('x=-10','x=-1','x=1','x=10')
```

Lösung A.1.6: a) Wir setzen $\varphi(x) := \sin(x)$ und finden mit der Taylor-Formel:

$$\varphi(x+h) = \varphi(x) + h\varphi'(x) + \tfrac{1}{2}h^2\varphi''(x) + \tfrac{1}{6}h^3\varphi'''(x) + \tfrac{1}{24}h^4\varphi^{(iv)}(\xi_+),$$
$$\varphi(x-h) = \varphi(x) - h\varphi'(x) + \tfrac{1}{2}h^2\varphi''(x) - \tfrac{1}{6}h^3\varphi'''(x) + \tfrac{1}{24}h^4\varphi^{(iv)}(\xi_-).$$

Dies ergibt

$$\varphi''(x) = \frac{\varphi(x+h) - 2\varphi(x) + \varphi(x-h)}{h^2} + \underbrace{\tfrac{1}{24}h^2\big\{\varphi^{(iv)}(\xi_+) + \varphi^{(iv)}(\xi_-)\big\}}_{O(h^2)}.$$

Unter beachtung von $\varphi''(x) = -\sin(x)$ ergibt sich so für $x = 1$:

$$f(h) = \frac{\sin(1+h) - 2\sin(1) + \sin(1-h)}{h^2} + \sin(1) = O(h^2).$$

b) Es gilt

$$c_h := \frac{1}{\ln(h)} \to 0 \quad (h \to 0), \qquad f(h) = \frac{h}{\ln(h)} = c_h h = o(h).$$

Lösung A.1.7: Die Datenfehler sind

$$\frac{|\Delta P|}{P} \le \frac{0,01}{2} \le 0.005, \quad \frac{|\Delta V|}{V} \le \frac{0,2}{10} \le 0.02, \quad \frac{|\Delta T|}{T} \le \frac{0,5}{200} \le 0.00025.$$

Mit den Konditionszahlen $k_P = k_V = k_T = 1$ ergibt sich so

$$\left|\frac{\Delta m}{m}\right| \le 0,005 + 0,02 + 0,00025 = 0,02525.$$

Um den relativen Fehler weiter unter 1%, d. h. unter $0,01$, zu drücken, muss die Messung von V verfeinert werden.

Lösung A.1.8: a) Mit dem Ansatz $a(h) = a + ch^\alpha$ ergibt sich

$$\frac{a(h) - a}{a(h/2) - a} = \frac{ch^\alpha}{c(h/2)^\alpha} = 2^\alpha$$

und folglich

$$\alpha = \frac{1}{\log(2)} \log\left(\left|\frac{a(h) - a}{a(h/2) - a}\right|\right).$$

Wenn der Limes a nicht bekannt ist, schreiben wir

$$\frac{a(h) - a(h/2)}{a(h/2) - a(h/4)} = \frac{a(h) - a + a - a(h/2)}{a(h/2) - a + a - a(h/4)} = \frac{ch^\alpha - c(h/2)^\alpha}{c(h/2)^\alpha - c(h/4)^\alpha}$$
$$= \frac{1 - 2^{-\alpha}}{2^{-\alpha} - 4^{-\alpha}} = 2^\alpha$$

und erhalten

$$\alpha = \frac{1}{\log(2)} \log\left(\Big| \frac{a(h) - a(h/2)}{a(h/2) - a(h/4)} \Big| \right).$$

b) Für die gegebenen Folgen ergibt sich:

$$\alpha \approx \frac{1}{\log(2)} \log\left(\Big| \frac{a(h) - a}{a(h/2) - a} \Big| \right) \approx 1,$$

und

$$\alpha \approx \frac{1}{\log(2)} \log\left(\Big| \frac{b(h) - b(h/2)}{b(h/2) - b(h/4)} \Big| \right) \approx 2.$$

Lösung A.1.9: a) Fehlerabschätzung

$$\Big| \frac{\Delta f}{f} \Big| \dot{\leq} k \Big| \frac{\Delta x}{x} \Big|, \qquad k = \Big| \frac{df}{dx} \frac{x}{f} \Big| \quad \text{(Konditionszahl)}.$$

$$\frac{df}{dx} = \frac{x\sin(x) - 1 + \cos(x)}{x^2},$$
$$k = \Big| \frac{x\sin(x) - 1 + \cos(x)}{x^2} \frac{x^2}{1 - \cos(x)} \Big| = \Big| \frac{x\sin(x)}{1 - \cos(x)} - 1 \Big|.$$

Also ist die Auswertung von $f(x)$ gut konditioniert für $|x| \ll 1$ und schlecht konditioniert für $|x| \gg 1$.

b) Es gilt die Maschinenoperationsregel $x \circledast y = (x * y)(1+\varepsilon)$ mit $|\varepsilon| \leq \text{eps}$. Im Folgenden wird ε als „generische'" Konstante verwendet, welche variieren kann, aber stets $|\varepsilon| \leq \text{eps}$ erfüllt.

Algorithmus A: Nach Voraussetzung ist

$$u = \cos(x)(1+\varepsilon), \quad v = 1 \ominus u, \quad \tilde{f}(x) = v \oslash x,$$

und demnach für $x \neq 0$:

$$\begin{aligned} \tilde{f}(x) &= \frac{v}{x}(1+\varepsilon) = \frac{1-u}{x}(1+\varepsilon)(1+\varepsilon) = \frac{1-\cos(x)(1+\varepsilon)}{x}(1+\varepsilon)(1+\varepsilon) \\ &= \frac{1-\cos(x)}{x}(1+\varepsilon+\varepsilon) - \frac{\cos(x)}{x}\varepsilon + O\Big(\frac{\text{eps}^2}{x}\Big). \end{aligned}$$

Also gilt für $x \neq 0, 2\pi, 4\pi, \ldots$:

$$\Big| \frac{\tilde{f}(x) - f(x)}{f(x)} \Big| = \Big| \varepsilon + \varepsilon - \frac{\cos(x)}{1-\cos(x)}\varepsilon \Big| + O\Big(\frac{\text{eps}^2}{1-\cos(x)}\Big) \;\to\; \infty \quad (|x| \to 0).$$

Algorithmus A wird also mit $|x| \to 0$ zunehmend schlechter konditioniert.

Algorithmus B: Umformung

$$f(x) = \frac{1-\cos(x)}{x} = \frac{1-\cos(x)^2}{x(1+\cos(x))} = \frac{\sin(x)^2}{x(1+\cos(x))}.$$

Nach Voraussetzung ist dann wieder

$$u_1 = \sin(x)(1+\varepsilon), \quad u_2 = u_1 \oplus u_1, \quad v_1 = \cos(x)(1+\varepsilon),$$
$$v_2 = 1 \oplus v_1, \quad v_3 = x \odot v_2, \quad \tilde{f} = u_2 \oslash v_3,$$

und demnach

$$\begin{aligned}\tilde{f} &= \frac{u_2}{v_3}(1+\varepsilon) = \frac{u_1 u_1(1+\varepsilon)}{x v_2(1+\varepsilon)}(1+\varepsilon) \\ &= \frac{\sin(x)^2(1+\varepsilon)^2(1+\varepsilon)(1+\varepsilon)}{x(1+v_1)(1+\varepsilon)(1+\varepsilon)} = \frac{\sin(x)^2(1+\varepsilon)^2(1+\varepsilon)(1+\varepsilon)}{x(1+\cos(x)(1+\varepsilon))(1+\varepsilon)(1+\varepsilon)}.\end{aligned}$$

Also gilt für $x \in (0,\pi)$:

$$\begin{aligned}\Big|\frac{\tilde{f}(x)-f(x)}{f(x)}\Big| &= \Big|1 - \frac{\sin(x)^2(1+\varepsilon)^2(1+\varepsilon)(1+\varepsilon)}{x(1+\cos(x)(1+\varepsilon))(1+\varepsilon)(1+\varepsilon)}\frac{x(1+\cos(x)}{\sin(x)^2}\Big| \\ &= \Big|1 - \frac{(1+\varepsilon)^2(1+\varepsilon)(1+\varepsilon)}{1+\varepsilon\frac{\cos(x)}{1+\cos(x)}(1+\varepsilon)(1+\varepsilon)}\Big| \\ &\le \Big|1 - \frac{(1+\varepsilon)^2(1+\varepsilon)(1+\varepsilon)}{(1+\varepsilon)(1+\varepsilon)(1+\varepsilon)}\Big| \\ &= \big|1 - (1+2\varepsilon+\varepsilon+\varepsilon)\,\big| + O(\mathrm{eps}^2) = O(\mathrm{eps}).\end{aligned}$$

Algorithmus B ist also insbesondere für $|x| \ll 1$ gut konditioniert.

Lösung A.1.10 (Praktische Aufgabe):

MATLAB-Programm: Auswertung der Taylor-Summe

```
     clear
     n = 3*(1:10);
% Aufgabenteil 1:
     for i=1:length(n)
        fehler1(i) = abs((taylorsumme(-5.5, n(i)) - exp(-5.5))/eps(-5.5);
     end
% Aufgabenteil 2:
     for i=1:length(n)
        fehler2(i) = abs((trickytaylorsumme(-5.5, n(i)) - exp(-5.5))/eps(-5.5);
     end
% Aufgabenteil 3:
     for\; i=1:length(n)
```

```
    fehler3(i) = abs((taylorsumme(-0.5, n(i))^11 - exp(-5.5))/eps(-5.5);
end
semilogy(n, fehler1, n, fehler2, n, fehler3)
legend('Teil 1)', 'Teil 2)', 'Teil 3)')
title('Taylor-Summen zur Exponentialfunktion')
xlabel('n (Anzahl der Taylor-Summanden)')
ylabel('Absoluter Fehler')
```

Auswertung der Approximationen für die Exponentialfunktion ergibt:

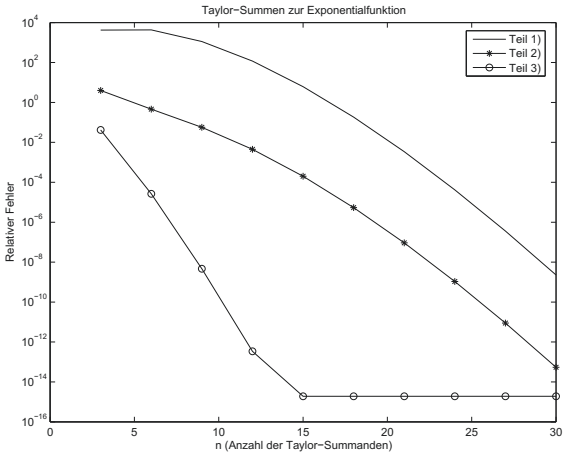


Abbildung A.2: Relative Fehler der Taylor-Approximation: $T_n(x)$

Lösung A.1.11: Das Horner Schema zur Polynomauswertung $p(x) = \sum_{i=0}^{m} a_i x^i$,

$$a'_n = a_n\,, \quad a'_k = a_k + x a'_{k+1}\,, \quad k = n-1, \ldots, 0,$$

liefert den Funktionswert $p(x_0) = a'_0$. Formale Anwendung auf das Matrixpolynom

$$p(A) = \sum_{i0}^{m} a_i A^i = (\ldots(a_n A + a_{n-1})A + a_{n-2})\ldots)A + a_0 I$$

ergibt den Algorithmus:

$$A'_n = a_n I, \qquad k = n-1, \ldots, 0: \;\; A'_k = A'_{k+1} A + a_k I,$$

mit $p(A) = A'_0$. Seine Realisierung in der Form

$$B = a_n I, \qquad k = n-1, \ldots, 0: \;\; C = BA,\; B = C + a_k I, \qquad p(A) = B,$$

erfordert die Speicherung von $A, B, C \in \mathbb{R}^{n\times n}$, $c \in \mathbb{R}^{n+1}$, $b \in \mathbb{R}^n$, d. h.: $3n^2 + O(n)$ Speicherplätze und $n^4 + O(n^2)$ a. Op.

Am Speicherplatz lässt sich durch eine geschicktere Schreibweise des Algorithmus noch etwas verbessern. Seien

$$A = [a_1, \ldots, a_n]^T \quad \text{Zeilenvektoren}, \quad B = [b_1, \ldots, b_n] \quad \text{Spaltenvektoren}$$

Damit schreiben wir den Algorithmus in der Form

$$B := a_n I, \qquad k = n-1, \ldots, 0,\ j = 1, \ldots, n: \quad b = \begin{bmatrix} a_i \cdot b_j \\ \vdots \\ a_n \cdot b_j \end{bmatrix} + a_k \begin{bmatrix} \delta_{1j} \\ \vdots \\ \delta_{nj} \end{bmatrix}, \quad b_j := b,$$

und schließlich $p(A) = B$. Diese Variante erfordert dieselbe Anzahl von a. Op. wie die obige, aber nur $2n^2 + O(n)$ Speicherplätze.

Lösung A.1.12: Durch Taylor-Entwicklung um die Nullstelle $z \neq 0$ erhält man

$$p(\tilde{z}) = p(z) + p'(z)(\tilde{z} - z) + O(|\tilde{z} - z|^2) = p'(z)(\tilde{z} - z) + O(|\tilde{z} - z|^2),$$

und weiter durch Umformung

$$\Big|\frac{\tilde{z} - z}{z}\Big| \leq \Big|\frac{p(\tilde{z})}{p'(z)z}\Big| + O(|\tilde{z} - z|^2).$$

Lösung A.1.13: Nach Vorausetzung ist $\tilde{f}(x_i) = f(x_i) + \varepsilon_i$ mit $|\varepsilon_i| \leq 10^{-3}|f(x_i)|$ und somit

$$\begin{aligned} \frac{\tilde{f}(x_{i+1}) - \tilde{f}(x_{i-1})}{2h} &= \frac{f(x_{i+1}) + \varepsilon_{i+1} - f(x_{i-1}) - \varepsilon_{i-1}}{2h} \\ &= \frac{f(x_{i+1}) - f(x_{i-1})}{2h} + \frac{\varepsilon_{i+1} - \varepsilon_{i-1}}{2h}. \end{aligned}$$

Da $f(\cdot)$ linear ist, ergibt sich für $\varepsilon_{i+1} = 10^{-3} f(x_{i+1})$ und $\varepsilon_{i-1} = -10^{-3} f(x_{i-1})$ und speziell für $i = 1$:

$$\frac{\tilde{f}(x_{i+1}) - \tilde{f}(x_{i-1})}{2h} = 1 + \frac{f(x_{i+1}) + f(x_{i-1})}{2} = 1 + \frac{1 + 2h + 1}{2} = 1 + 1 + 10^{-3}.$$

Der relative Approximationsfehler ist in diesem Fall also größer als 100%.

Lösung A.1.14 (Praktische Aufgabe): Für die verschiedenen Datensätze $\{a, b, c\}$ ergeben sich die folgenden Ergebnisse:

1. Parameter $\{0, 0, 0\}$: Es gibt unendlich viele Lösungen.
2. Parameter $\{0, 0, 1\}$: Es gibt keine Lösung.

3. Parameter $\{0,1,0\}$: $x_{1/2} = 0$; Genauigkeit: 0.
4. Parameter $\{0,2,1\}$: $x_{1/2} = -0.5$; Genauigkeit: 0.
5. Parameter $\{2,0,0\}$: $x_{1/2} = 0$; Genauigkeit: 0.
6. Parameter $\{2,0,1\}$: Es gibt komplexe Lösungen.
7. Parameter $\{2,0,-4\}$: $x_{1/2} = \pm 1.4142$; Genauigkeit: $1.1102e-16$.
8. Parameter $\{4,2,0\}$: $x_1 = 0,\ x_2 = -0.5$; Genauigkeit: 0.
9. Parameter $\{1,2,-3\}$: $x_1 = 1,\ x_2 = -3$; Genauigkeit: 0.
10. Parameter $\{1,2,1\}$: $x_{1/2} = -1$; Genauigkeit: 0.
11. Parameter $\{1,2,2\}$: Es gibt komplexe Lösungen.
12. Parameter $\{-1,0,0\}$: $x_{1/2} = 0$; Genauigkeit: 0.
13. Parameter $\{-1,0,-1\}$: Es gibt komplexe Lösungen.
14. Parameter $\{-4,0,2\}$: $x_{1/2} = \pm - 0.7071$; Genauigkeit: $1.1102e-16$.
15. Parameter $\{-1,2,0\}$: $x_1 = 0,\ x_2 = 2$; Genauigkeit: 0.
16. Parameter $\{-4,8,12\}$: $x_1 = -1,\ x_2 = 3$; Genauigkeit: 0.
17. Parameter $\{-1,2,-1\}$: $x_{1/2} = 1$; Genauigkeit: 0.
18. Parameter $\{-1,2,-5\}$: Es gibt komplexe Lösungen.
19. Parameter $\{2.5 \cdot 10^9, -10^5, 1\}$: $x_{1/2} = 2 \cdot 10^{-5}$; Genauigkeit: **0.76294**.

MATLAB-Programme:

```
% Hauptprogramm
para = [0 0 0 0 2 2 2 4 1 1 1 -1 -1 -4 -1 -4 -1 -1 2.5e9;
        0 0 1 2 0 0 0 2 2 2 2 0 0 0 2 8 2 2 -1.e5;
        0 1 0 1 0 1 -4 0 -3 1 2 0 -1 2 0 12 -1 -5 1];
for i = 1:max(size(para))
   disp([int2str(i),') Parameter:'])
   disp(para(:,i)')
   disp('Loesung: ')
   sol = mitternacht(para(:,i));
   if (isempty(sol))
      disp('Es gibt komplexe Loesungen')
      disp(' ')
      disp(' ')
   elseif (sol == inf)
      disp('Es gibt beliebig viele Loesungen')
      disp(' ')
      disp(' ')
   elseif (sol == -inf)
      disp('Es gibt keine Loesung')
      disp(' ')
      disp(' ')
```

```
   else
      if (checkgenauigkeit(sol, para(:,i)) == 1)
         disp(sol)
         disp('Genauigkeit: ')
         if (length(sol) == 1)
            disp(num2str(genauigkeit(sol,para(:,i))))
         else
            disp([num2str(genauigkeit(sol(1),para(:,i))), '   ',
            num2str(genauigkeit(sol(2),para(:,i)))])
         end
         disp(' ')
         disp(' ')
      else
         disp(['Loesung waere ', num2str(sol), ', aber '])
         if (length(sol) == 1)
            ausgabe = ['Genauigkeit von ',
            num2str(genauigkeit(sol,para(:,i))), ' nicht ausreichend'];
         else
            ausgabe = ['Genauigkeit von ',
            num2str(genauigkeit(sol(1),para(:,i))), ' und ',
            num2str(genauigkeit(sol(2),para(:,i))), ' nicht ausreichend'];
         end
         disp(ausgabe)
         disp(' ')
      end
   end
end

% Genauigkeitstest
function erg = checkgenauigkeit(z,para)
erg = 1;
for i = 1:length(z)
   if (genauigkeit(z(i),para) > 1.e-12)
      erg = 0;
   end
end
return

% Ableitungsberechnung
function erg = dp(x,para)
a = para(1);
b = para(2);
c = para(3);
erg = 2*a*x+b;
return
```

```
% Fehlerschaetzer
function erg = genauigkeit(z,para)
if (z == 0 | dp(z,para) == 0)
    erg = 0;
else
    erg = abs(p(z,para) / (dp(z,para)*z));
end
return

% Loesungsformel
function erg = mitternacht(para)

a = para(1);
b = para(2);
c = para(3);
if (a == 0)
   if (b == 0)
      if (c == 0)
         erg = inf;
      else
         erg = -inf;
      end
   else
      erg = -c/b;
   end
else
   if (b^2 < 4*a*c)
      erg = [];
   elseif (b^2 == 4*a*c)
      erg = -b/(2*a);
   else
      erg = [-b+sqrt(b^2-4*a*c), -b-sqrt(b^2-4*a*c)] / (2*a);
   end
end
return

% Funktionsauswertung
function erg = p(x,para)
a = para(1);
b = para(2);
c = para(3);
erg = a*x^2+b*x+c;
return
```

A.2 Kapitel 2

Lösung A.2.1: 0) Nach Konstruktion habe die Lagrangeschen Polynome den Grad n und die Eigenschaft

$$L_i^{(n)}(x_j) = \delta_{ij}, \quad i,j = 0,\ldots,n. \quad \text{(Kronecker-Symbol)}$$

Damit ergibt sich die Implikation

$$\sum_{i=1}^{n} \alpha_i L_i^{(n)}(x) = 0, \;\; x \in \mathbb{R}, \quad \Rightarrow \quad 0 = \sum_{i=1}^{n} \alpha_i L_i^{(n)}(x_j) = \alpha_j, \quad j = 0,\ldots,n.$$

Also sind die $n+1$ Polynome $\{L_i^{(n)}, i = 0,\ldots,n\}$ linear unabhängig und bilden eine Basis des P_n.

i) Das Polynom $q(\cdot) = \sum_{i=0}^{n} L_i^{(n)} \in P_n$ ist in den $n+1$ Stellen $x_i, i = 0,1,\ldots,n$, nach Konstruktion gleich dem Polynom $q \equiv 1$. Wegen des Eindeutigkeit des Lagrange-Interpolationspolynoms muss also $q \equiv 1$ sein (Anwendung des Satzes von Rolle).

ii) Das Polynom $p_k(\cdot) = \sum_{i=0}^{n} x_i^k L_i^{(n)} \in P_n$ stimmt in den $n+1$ Stellen x_i mit dem Monom $q(x) = x^k$ überein. Für $k = 1,\ldots,n$ ist also wieder wegen der Eindeutigkeit des Lagrange-Interpolationspolynoms $p_k \equiv x^k$ und damit $p_k(0) = 0$.

iii) Im Fall $k = n+1$ ist $p_k(\cdot) = \sum_{i=0}^{n} x_i^{n+1} L_i^{(n)} \in P_n$ das Lagrange-Interpolationspolynom des Monoms $p_{n+1}(x) = x^{n+1}$ zu den $n+1$ Punkten $x_0,\ldots,x_n$. Die allgemeine Darstellung des Fehlers bei der Lagrange-Interpolation,

$$f(x) - p(x) = \frac{f^{(n+1)}(\xi_x)}{(n+1)!} \prod_{i=0}^{n} (x - x_i),$$

ergibt angewendet auf die vorliegende Situation:

$$x^{n+1} - p_{n+1}(x) = \frac{1}{(n+1)!} \frac{d^{n+1}x^{n+1}}{dx^{n+1}} \prod_{i=0}^{n} (x - x_i) = \prod_{i=0}^{n} (x - x_i).$$

Also ist wie behauptet

$$p_{n+1}(0) = -\prod_{i=0}^{n} (-x_i) = (-1)^n \prod_{i=n}^{n} x_i.$$

Lösung A.2.2: Die Rekursionsformel des Neville-Algorithmus zur Punktauswertung des Lagrange-Interpolationspolynoms zu den Stützstellen $\{x_0,\ldots,x_n\}$ lautet

$$p_{i,i+k}(\xi) = p_{i,i+k-1}(\xi) + \frac{p_{i,i+k-1}(\xi) - p_{i+1,i+k}(\xi)}{\frac{\xi - x_{i+k}}{\xi - x_i} - 1}, \quad i = 0,\ldots,n, \; k = 1,\ldots,n-i.$$

Der Funktionswert des Interpolationspolynoms ist dann $p_n(\xi) = p_{0,n}(\xi)$. Das Neville-

Schema lautet:

x_i	$p_{i,i}$	$p_{i,i+1}$	$p_{i,i+2}$	$p_{i,i+3}$	$p_{0,4}$
$55,7$	$17,47$	$19,06$	$19,46$	$19,49$	$19,46$
$57,7$	$18,00$	$19,30$	$19,50$	$19,44$	
$59,3$	$18,52$	$19,55$	$19,39$		
$62,6$	$19,93$	$19,14$			
$65,6$	$22,57$				

Die Tageslänge am Ort F ist also $19\text{h}\ 27,6\text{m}$.

Lösung A.2.3: Das Integrationsintervall $[0,\pi]$ sei äquidistant unterteilt mit den Stützstellen $0 = x_0 < x_1 < \ldots < x_n = \pi$ mit $x_i - x_{i-1} = h = \pi/n$. Der Auswertungsfehler in beliebigem $x \in [0,\pi]$ setzt sich zusammen aus dem Interpolationsfehler und dem Rundungsfehler in den zur Interpolation verwendeten Stützwerten $\tilde{f}(x_i) = f(x_i) + \varepsilon$.

i) Abschätzung des Interpolationsfehlers: Ausgangspunkt ist die Fehlerdarstellung für die kubische Lagrange-Interpolation zu den jeweils vier Stützstellen $\{x_{i-1}, x_i, x_{i+1}, x_{i+2}\}$:

$$f(x) - p_3(x) = \frac{f^{(iv)}(\xi_x)}{4!} \prod_{j=-1}^{2} (x - x_{i+j}), \quad x \in [x_i, x_{i+1}], \quad i = 1, \ldots, n-1.$$

Mit $f^{(iv)}(x) = -8\sin(x)\cos(x) = -4\sin(2x)$ ist $\max_{x\in[0,\pi]} |f^{(iv)}(x)| = 4$. Weiter ist

$$\max_{x_i \le x \le x_{i+1}} \prod_{j=-1}^{2} (x - x_{i+j}) = \max_{-h/2 \le x \le h/2} (x + 3h/2)(x + h/2)(x - h/2)(x - 3h/2) = \frac{9}{16} h^4,$$

wobei das Maximum in $x = 0$ angenommen wird. Damit ergibt sich die Fehlerabschätzung:

$$\max_{x\in[0,\pi]} |f(x) - p_3(x)| \le \frac{4}{4!}\frac{9}{16} h^4 = \frac{3}{32} h^4.$$

ii) Abschätzung des Rundungsfehlereinflusses: Aus der Lagrangeschen Darstellung des Interpolationspolynoms folgt

$$\max_{x_i \le x \le x_{i+1}} |p_3(x) - \tilde{p}_3(x)| \le \sum_{j=-1}^{2} |f(x_{i+j} - \tilde{f}(x_{i+j})| \max_{x_i \le x \le x_{i+1}} |L_{i+j}^{(3)}(x)|.$$

Etwas Rechnerei ergibt

$$\max_{-h/2 \le x \le h/2} \frac{(x^2 - h^2/4)(x - 3h/2)}{6h^3} \le \frac{1}{12}, \quad j = -1, 1.$$
$$\max_{-h/2 \le x \le h/2} \frac{(x^2 - 9h^2/4)(x - h/2)}{2h^3} \le 1, \quad j = 0, 2,$$

und somit

$$\max_{x\in[0,\pi]} |p(x) - \tilde{p}(x)| \leq 4 \max_{x\in[0,\pi]} |f(x_i) - \tilde{f}(x_i)| \leq 4 \cdot 0,5 \cdot 10^{-9}.$$

iii) Für den Gesamtfehler ergibt sich:

$$\max_{0\leq x\leq\pi} |f(x) - \tilde{p}_3(x)| \leq \frac{3}{32} h^4 + 2 \cdot 10^{-9} \approx 10^{-1} h^4 + 2 \cdot 10^{-9}.$$

Für 250 Stützstellen ist $h = \pi/249 \approx 0,01262$ und folglich $h^4 \approx 2,5 \cdot 10^{-8}$. Mit 250 Stützstellen läßt sich die Approximationsgenauigkeit von $5 \cdot 10^{-9}$ also garantieren.

Lösung A.2.4: Es ist $f^{(n+1)}(x) = \lambda^{n+1} e^{\lambda x}$. Aus der Fehlerdarstellung der Lagrange-Interpolation in Punkten $\{x_0, \ldots, x_n\}$,

$$f(x) - p_n(x) = \frac{f^{(n+1)}(\xi_x)}{(n+1)!} \prod_{i=0}^{n} (x - x_i),$$

erhält man also in diesem Fall die Abschätzung

$$|e^{\lambda x} - p_n(x)| \leq \frac{|\lambda|^{n+1} e^{\lambda b}}{(n+1)!} (b-a)^{n+1}.$$

Wegen

$$\frac{|\lambda|^n (b-a)^n}{n!} \to 0 \quad (n \to \infty)$$

folgt die behauptete gleichmäßige Konvergenz $\max_{x\in[a,b]} |f(x) - g(x)| \to 0 \ \ (n \to \infty)$.

Für die „glatte“ Funktion $f(x) = (1 + x^2)^{-1}$ gilt

$$|f^{(n+1)}(x)| \approx n!,$$

so dass das obige Argument nicht gilt.

Lösung A.2.5: i) Der Beweis der (eindeutigen) Lösbarkeit der Hermite-Interpolationsaufgabe folgt ganz analog wie für die Lagrange-Interpolationsaufgabe mit Hilfe des Satzes von Rolle.

ii) Für $x \in \{x_0, \ldots, x_m\}$ gilt die behauptete Restgieddarstellung trivialerweise. Sei also $x \in [a, b] \setminus \{x_0, \ldots, x_m\}$. Wir verwenden wieder die Funktionen

$$l(x) := \prod_{i=0}^{m} (x - x_i)^2, \quad c(x) := \frac{f(x) - p(x)}{l(x)}.$$

Die Funktion $F(t) := f(t) - p(t) - c(x) l(t)$ hat $n + 2$ Nullstellen im Intervall $[a, b]$, nämlich die $x_i, i = 0, \ldots, m$, mit Vielfachheiten 2 und den betrachteten Punkt x. Mit Hilfe des Satzes von Rolle folgt hieraus, daß die $(n+1)$-te Ableitung $F^{(n+1)}$ eine Nullstelle

ξ_x im von den Punkten $\{x, x_0, \ldots, x_m\}$ aufgespannten Intervall besitzt, d. h.:

$$0 = F^{(n+1)}(\xi_x) = f^{(n+1)}(\xi_x) - p^{(n+1)}(\xi_x) - c(x)l^{(n+1)}(\xi_x) = f^{(n+1)}(\xi_x) - c(x)(n+1)!.$$

Dies impliziert die behauptete Fehlerdarstellung.

Lösung A.2.6 (Praktische Aufgabe): Die folgenden Bilder zeigen die relativen Fehler der Lagrange-Interpolation der Funktionen $f(x) = (1 + 25x^2)^{-1}$ und $g(x) = \sqrt{|x|}$. Es zeigt sich, dass für $n \to \infty$ keine Konvergenz vorliegt; insbesondere am Intervallrand treten zunehmende Oszillationen auf. Das Problem ist also offenbare nicht das „singuläre“ Verhalten von g bei $x = 0$ sondern das Verhalten der höheren Ableitungen von g für $x \to \pm 1$.

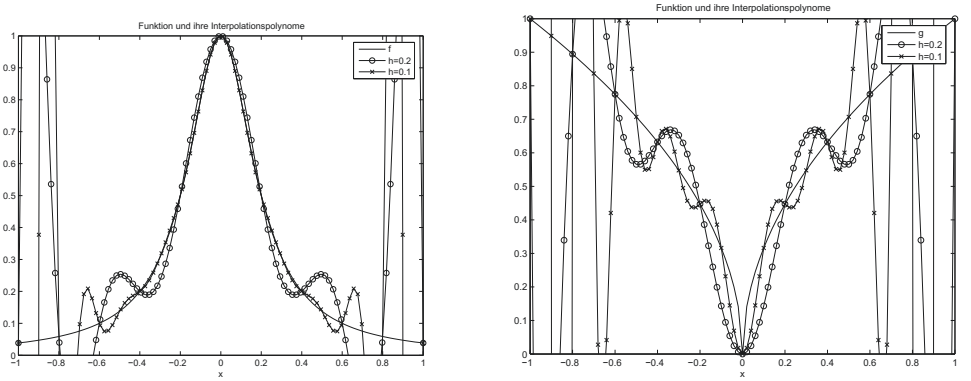


Abbildung A.3: Lagrange-Interpolationspolanome $p_n(x)$ zur Funktion $f(x) = (1 + 25x^2)^{-1}$ (links) und $g(x) = \sqrt{|x|}$ (rechts)

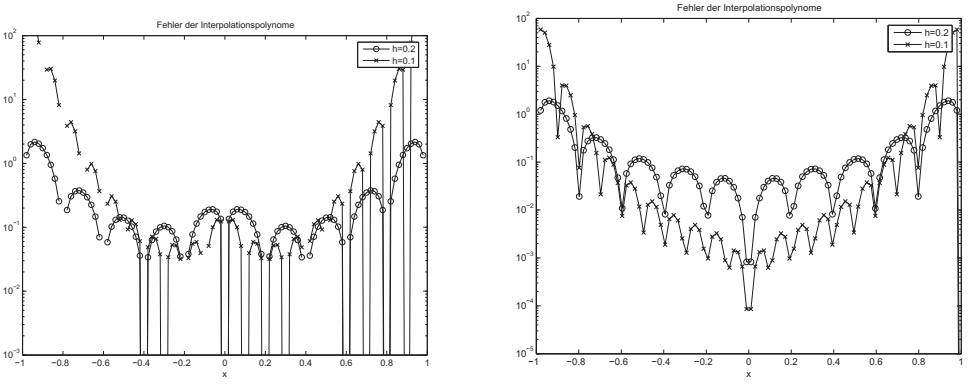


Abbildung A.4: Relativer Fehler der Lagrange-Interpolation zur Funktion $f(x) = (1 + 25x^2)^{-1}$ (links) und $g(x) = \sqrt{|x|}$ (rechts)

MATLAB-Programm:

```
% Hauptprogramm
     clear;
     h = [.2, .1];
%
% Plotte die Funktion:
     figure(1)
     hold off;
     clf;
     fun = 'g';
     x = linspace(-1, 1, 101);
     y1 = feval(fun, x);
     plot(x, y1, 'k-');
%
% Berechne die Interpolationspolynome nach dem Neville-Schema:
     N = length(h);
     y2 = zeros(N,length(x));
     for k = 1:N
        stuetzstellen = -1:h(k):1;
        for i = 1:length(x)
           y2(k,i) = neville(x(i), stuetzstellen, fun);
        end
     end;
%
% Plotte die Interpolationspolynome:
     hold on;
     plot(x, y2(1,:), 'k-o');
     plot(x, y2(2,:), 'k-x');
     title('Funktion und ihre Interpolationspolynome');
     xlabel('x')
     legend(fun, ['h=', num2str(h(1))], ['h=', num2str(h(2))]);
     axis([-1 1 0 1])
     print -deps2 polynome.eps
     figure(2)
     hold off;
     clf;
     fehler = y2;
     for k = 1:N
        fehler(k,:) = abs(y1 - y2(k,:));
     end;
     semilogy(x, fehler(1,:), 'k-o');
     hold on
     semilogy(x, fehler(2,:), 'k-x');
     title('Fehler der Interpolationspolynome');
```

```
     xlabel('x')
     legend(['h=', num2str(h(1))], ['h=', num2str(h(2))])
     print -deps2 fehler.eps

     function erg = f(x);
     erg = 1. ./ (1. + 25*x.^2);
     return

     function erg = neville(x, stuetz, fun);
%
% Berechne das Lagrangepolynom der Funktion fun an der Stelle x mit den
% Stuetzstellen stuetz
     n = length(stuetz);
     p = zeros(n);
     p(:,1) = (feval(fun, stuetz))';
     for i = 2:n
        p(1:n-i+1,i) = p(1:n-i+1,i-1) + (x-stuetz(1:n-i+1))'...
     .* (p(2:n-i+2,i-1) - p(1:n-i+1,i-1)) ./ (stuetz(i:n)...
      - stuetz(1:n-i+1))';
     end
     erg = p(1,n);
```

Lösung A.2.7: (i) Für das quintische Lagrange-Interpolationspolynom $p_5^{(k)} \in P_5$ auf einem der Teilintervalle $I_k = [x_{k-1}, x_k],\ k = 1, \ldots, N$, gilt die Fehlerabschätzung

$$|f(x) - p_5^{(k)}(x)| \leq \frac{|f^{(6)}(\xi_x)|}{6!} \prod_{i=0}^{5} (x - x_{k-1} - jh_k/5), \quad \xi_x \in [x_{k-1}, x_k],$$

und somit wegen $|x_k - x_{k-1}| \leq h$:

$$\max_{x \in [x_{k-1}, x_k]} |f(x) - p_n^{(k)}(x)| \leq \frac{h^6}{720} \max_{x \in [x_{k-1}, x_k]} |f^{(6)}(x)|.$$

Zusammensetzung dieser „lokalen“ Abschätzungen für $k = 1, \ldots, N$ ergibt dann die behauptete Abschätzung auf dem ganzen Intervall I.

(ii) Für das quintische Hermite-Interpolationspolynom $p_5^{(k)} \in P_5$ auf einem der Teilintervalle $I_k = [x_{k-1}, x_k],\ k = 1, \ldots, N$ gilt die Fehlerabschätzung

$$|f(x) - p_5^{(k)}(x)| \leq \frac{|f^{(6)}(\xi_x)|}{6!} (x - x_{k-1})^3 (x - x_k)^3, \quad \xi_x \in [x_{k-1}, x_k],$$

und somit analog zu eben:

$$\max_{x \in [x_{k-1}, x_k]} |f(x) - p_n^{(k)}(x)| \leq \frac{h^6}{720} \max_{x \in [x_{k-1}, x_k]} |f^{(6)}(x)|.$$

Zusammensetzung dieser „lokalen“ Abschätzungen für $k = 1, \ldots, N$, ergibt dann wie bei (i) die behauptete Abschätzung auf dem ganzen Intervall I.

Lösung A.2.8: Es ist $f(x) = \cosh(x)$ mit $f'(0.6) \approx 0.63665358\ldots$. Aus den Tabellenwerten lassen sich die folgenden zentralen Differenzenquotienten 1. Ordnung bilden (Diese sind wegen ihrer höheren Ordnung den einfachen vorwärts- oder rückwärts genommenen Differenzenquotienten vorzuziehen.):

$$\begin{aligned} h_0 = 0.08: \quad a(h_0) &= \frac{f(0.68) - f(0.52)}{2 \cdot 0.08} = \underline{0.63}73329 \\ h_1 = 0.04: \quad a(h_1) &= \frac{f(0.64) - f(0.56)}{2 \cdot 0.04} = \underline{0.6368}234\,. \end{aligned}$$

Der Fehler für den zentralen Differenzenquotienten erlaubt eine Entwicklung nach *geraden* Potenzen von h. Das zugehörige Extrapolationsschema lautet:

$$\begin{aligned} a_{i,0} &= a(h_i), \quad i = 0, 1, 2, \ldots \\ a_{i,k} &= a_{i,k-1} + \frac{a_{i,k-1} - a_{i-1,k-1}}{(h_{i-k}/h_i)^2 - 1}, \quad i = 1, 2, \ldots, \quad k = 1, \ldots, i. \end{aligned}$$

Dies ergibt

$$a_{1,1} = \frac{4a(h_1) - a(h_0)}{3} = \underline{0.6366535}$$

Nach einem Extrapolationsschritt ist die erreichte Genauigkeit der Theorie gemäß $O(h^4)$. Dies entspricht der Bildung des zentralen Differenzenquotienten 4. Ordnung:

$$\frac{-f(x+2h) + 8f(x+h) - 8f(x-h) + f(x-2h)}{12h} = f'(x) + O(h^4).$$

Lösung A.2.9: Für die Extrapolation zulässige Indexfolgen $(n_i)_{i\in\mathbb{N}}$ sind charakterisiert durch die Eigenschaft

$$\sup_{i\in\mathbb{N}} \frac{n_i}{n_{i+1}} \leq \gamma < 1.$$

(i) Die Folge $(2i-1)_{i\in\mathbb{N}}$ ist nicht zulässig wegen

$$\sup_{i\in\mathbb{N}} \frac{n_i}{n_{i+1}} = \sup_{i\in\mathbb{N}} \frac{2i-1}{2(i+1)-1} = 1.$$

(ii) Die Folge $(3^i)_{i\in\mathbb{N}}$ ist zulässig wegen

$$\sup_{i\in\mathbb{N}} \frac{3^i}{3^{i+1}} = \frac{1}{3} < 1.$$

(iii) Die Folge $(i^2)_{i\in\mathbb{N}}$ ist nicht zulässig wegen

$$\sup_{i\in\mathbb{N}} \frac{i^2}{(i+1)^2} = 1.$$

Lösung A.2.10: a) i) Wegen $f''(0) = -2$ liegt kein natürlicher, kubischer Spline vor.
ii) Es ist $f(x) = 8 - 4x$ und folglich $f''(0) = f''(2) = 0$. Also liegt ein natürlicher, kubischer Spline vor.
iii) Es ist $f_{|[0,1]} = -x^3/2$ und $f_{|[1,2]} = (x-1)^3 - x^3/2$ und folglich $f''(0) = 0$ und $f''(2) = 0$. Also liegt ein natürlicher, kubischer Spline vor.

b) Die Stützstellen sind $x_0 = 0, x_1 = 1, x_2 = 2$ mit den Stützwerten $y_0 = 0, y_1 = 1, y_2 = 8$. Zur Konstruktion des interpolierenden, kubischen Splines machen wir den Ansatz:

$$s_2(x) = \begin{cases} a_0^{(1)} + a_1^{(1)}(x-1) + a_2^{(1)}(x-1)^2 + a_3^{(1)}(x-1)^3, & 0 \le x \le 1, \\ a_0^{(2)} + a_1^{(2)}(x-1) + a_2^{(2)}(x-1)^2 + a_3^{(2)}(x-1)^3, & 1 \le x \le 2. \end{cases}$$

Die Interpolationsbedingung impliziert: $a_0^{(1)} = y_1 = 1$, $a_0^{(2)} = y_2 = 8$. Berücksichtigung der Randbedingungen ergibt $a_0^{(2)} = 0$. Die Gleichung für $a_2^{(1)}$ (s. Vorlesung) lautet bei Beachtung von $h_1 = h_2 = 1$:

$$h_1 a_2^{(0)} + 2(h_1 + h_2)a_2^{(1)} + h_2 a_2^{(2)} = 3\Big(\frac{y_2 - y_1}{h_2} - \frac{y_1 - y_2}{h_1}\Big),$$

was $a_2^{(1)} = 9/2$ impliziert. Weiter folgt

$$a_3^{(1)} = \frac{a_2^{(1)} - a_2^{(0)}}{3h_1} = \frac{3}{2}, \qquad a_3^{(2)} = \frac{a_2^{(2)} - a_2^{(1)}}{3h_2} = -\frac{3}{2},$$
$$a_1^{(1)} = \frac{y_1 - y_0}{h_1} + \frac{h_1}{3}(2a_2^{(1)} + a_2^{(2)}) = 4, \qquad a_1^{(2)} = \frac{y_2 - y_1}{h_2} + h_2 a_2^{(2)} - h_2^2 a_3^{(2)} = \frac{17}{2}.$$

Hiermit gewinnen wir die Darstellung

$$s_2(x) = \begin{cases} 1 + 4(x-1) + \frac{9}{2}(x-1)^2 + \frac{3}{2}(x-1)^3, & 0 \le x \le 1, \\ 8 + \frac{17}{2}(x-1) - \frac{3}{2}(x-1)^3, & 1 \le x \le 2. \end{cases}$$

Im Fall der inhomogenen Randbedingungen ergibt sich natürlich (wegen der Eindeutigkeit des interpolierenden Splines) $s_2(x) = x^3$.

Lösung A.2.11 (Praktische Aufgabe): Wir setzen $h = 1/n$ und

$$a(h) = \frac{2}{h}\sin(\pi h) = \frac{2}{h}\sum_{i=0}^{\infty} \frac{(\pi h)^{2i+1}}{(2i+1)!},$$

d. h.: Die Funktion $a(h)$ besitzt eine Entwicklung nach geraden Potenzen von h. Es ist

$$c_2 = a(1/2) = 4\sin(\pi/2) = 4$$
$$c_3 = a(1/3) = 6\sin(\pi/3) = 5.1962\ldots$$
$$c_6 = a(1/6) = 12\sin(\pi/6) = 6.$$

Das Extrapolationsschema unter Verwendung der h^2-Entwicklung lautet:

$$a_{k,i} = a_{k-1,i} + \frac{a_{k-1,i} - a_{k-1,i-1}}{(h_{i-k}/h_i)^2 - 1}$$

1/2	4		
1/3	5.1962	6.1532	
1/6	6	6.2679	<u>6.2806</u>

Das Ergebnis der Extrapolation ist $2\pi \approx 6.2806$.

Historische Bemerkungen: Mit Hilfe der Längen des ein- und des umbeschriebenen n-Ecks gilt die Einschließung

$$T_n := n \sin(\pi/n) < \pi < n \tan(\pi/n) =: U_n.$$

Zur Berechnung dieser Längen kann die folgende Rekursionsformel verwendet werden:

$$T_6 = 6, \qquad T_{2n} = 2\sqrt{2n^2 - n\sqrt{4n^2 - T_n^2}}.$$

- *Archimedes* (282-212 v. Chr.) erhielt mit Hilfe des ein- und des umbeschriebenen 96-Ecks die Einschließung (3 Stellen):

$$3\tfrac{10}{71} < \pi < 3\tfrac{1}{7}.$$

- *Ptolemaios* (150 v. Chr.) fand die Näherung (4 Stellen):

$$\pi \approx 3,141.$$

- *Lin Hui* (263) erhielt mit Hilfe des 3072-Ecks die Näherung (6 Stellen):

$$\pi \approx 3,14159.$$

- *Al-Kasi* (1437) erhielt mit Hilfe des $3 \cdot 2^{28}$-Ecks die Näherung (17 Stellen):

$$\pi \approx 3,1415\,9265\,3589\,7932\,.$$

- *L. van Ceulen* (1600) erhielt mit Hilfe des $3 \cdot 2^{60}$-Ecks die Näherung (35 Stellen):

$$\pi \approx 3,1415\,9265\,3589\,7932\,3846\ldots.$$

- *Huygens* (1654) erhielt mit Hilfe der auf heuzristischem Wege gefundenen „Extrapolationsformel“ $S_n := \frac{1}{3}(4T_n - T_{n/2})$ die Näherung

$$\pi \approx S_{96} = 3,141592\,.$$

Hiermit lassen im Prinzip beliebig gute Näherungen zu π gewinnen.

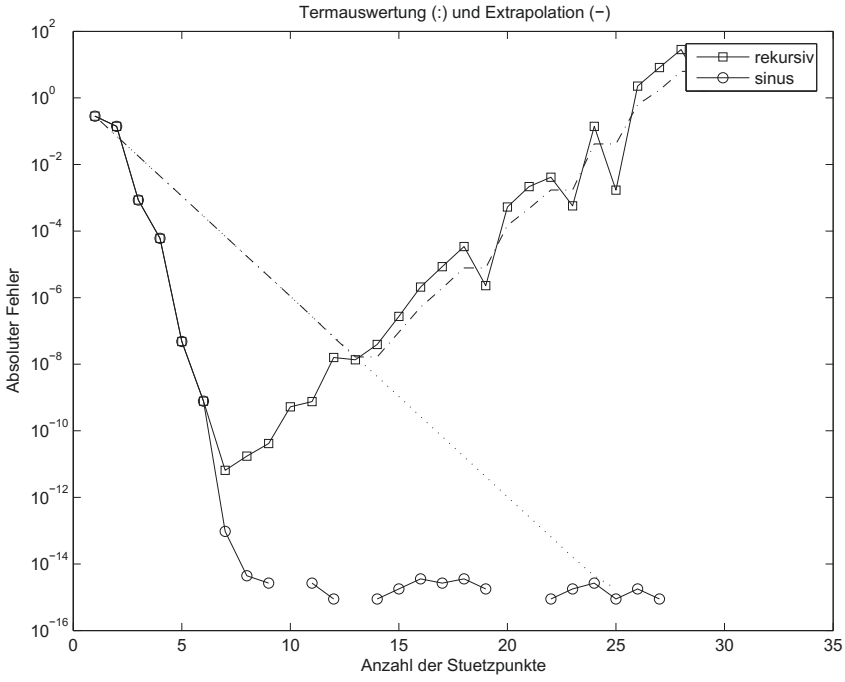


Abbildung A.5: *Fehler der Richardson-Extrapolation zur Berechnung von π auf der Basis der Rekursionsformel und der Polygonzugformel im Vergleich zur direkten Approximation mit diesen Formeln*

MATLAB-Programm:

```
clear
format long
kmax = 30;
n = 6*2.^(0:kmax);
Arek = richardson(n, 'T');
Asin = richardson(n, 'T2');

semilogy(abs(diag(Arek) - 2*pi), 'b')
hold on
semilogy(abs(diag(Asin) - 2*pi), 'r')
semilogy(abs(Arek(:,1) - 2*pi), 'b:')
semilogy(abs(Asin(:,1) - 2*pi), 'r:')
```

```
legend('rekursiv', 'sinus')
xlabel('Anzahl der Stuetzpunkte')
ylabel('Absoluter Fehler')
title('Termauswertung (:) und Extrapolation (-)')


function A = richardson(n, fun)
h = pi./n;
i = length(h);
A = zeros(i);
A(:,1) = feval(fun, n)';
for k = 2:i
  A(k:i,k) = A(k:i, k-1) + (A(k:i, k-1) - A(k-1:i-1, k-1))
  ./ (h(1:i-k+1)'./h(k:i)' - 1);
end
return


function erg = T(n)
erg(1) = 6;
for i=2:length(n)
    erg(i) = 2*sqrt(2*n(i-1)^2 - n(i-1)*sqrt(4*n(i-1)^2 - erg(i-1)^2));
end
return


function erg = T2(n)
erg = 2*n .* sin(pi./n);
return
```

Lösung A.2.12: (i) Für $m \neq n$ ist:

$$\int_{-\pi}^{\pi} \cos(mx)\cos(nx)\,dx = \underbrace{\cos(mx)\frac{1}{n}\sin(nx)\Big|_{-\pi}^{\pi}}_{=0} + \frac{m}{n}\int_{-\pi}^{\pi} \sin(mx)\sin(nx)\,dx$$

$$= \underbrace{-\frac{m}{n}\sin(mx)\frac{1}{n}\cos(nx)\Big|_{-\pi}^{\pi}}_{=0} + \frac{m^2}{n^2}\int_{-\pi}^{\pi} \cos(mx)\cos(nx)\,dx$$

bzw.

$$\left(1 - \frac{m^2}{n^2}\right)\int_{-\pi}^{\pi} \cos(mx)\cos(nx)\,dx = 0.$$

Analog ergibt sich in diesem Fall

$$\int_{-\pi}^{\pi} \cos(kx)\sin(mx)\,dx = 0 = \int_{-\pi}^{\pi} \sin(mx)\sin(nx)\,dx$$

Für $n = m$ gilt:

$$\int_{-\pi}^{\pi} \sin(mx)^2\,dx = \int_{-\pi}^{\pi} \cos(mx)^2\,dx = \int_{-\pi}^{\pi} (1 - \sin(mx)^2)\,dx$$

und folglich

$$\int_{-\pi}^{\pi} \sin(mx)^2\,dx = \int_{-\pi}^{\pi} \cos(mx)^2\,dx = \pi.$$

Die Funktionen $\{\varphi_0, \varphi_k, \psi_k, k = 1, \ldots, n\}$ sind also linear unabhängig und bilden somit wegen $\dim(T_n) = 2k + 1$ eine Orthonormalbasis von T_n.

(ii) Die beste Approximation $g \in T_n$ zu $f(x) = x$ ist gegeben durch

$$g(x) = (f, \varphi_0)\varphi_0(x) + \sum_{k=1}^{n} (f, \varphi_k)\varphi_k(x) + \sum_{k=1}^{n} (f, \psi_k)\psi_k(x).$$

Durch Nachrechnen sieht man daß

$$(x, 1) = 0, \qquad (x, \psi_k) = \int_{-\pi}^{\pi} x\cos(kx)\,dx = 0, \quad k = 1, \ldots, n,$$
$$(x, \varphi_k) = \int_{-\pi}^{\pi} x\sin(kx)\,dx = -\frac{1}{k} x\cos(kx)\Big|_{-\pi}^{\pi} = \frac{2\pi}{k}(-1)^{k+1}, \quad k = 1, \ldots, n.$$

Die beste Approximation ist also

$$g(x) = 2\sum_{k=1}^{n} \frac{(-1)^{k+1}}{k} \sin(kx).$$

Lösung A.2.13: Wir zeigen zunächst die Orthogonalität der Funtionen $\psi_k(x) := (2k)!/k!\,\varphi_k$. Dabei wird verwendet, dass für $0 \le i < k$ gilt:

$$\frac{d^i}{dx^i}(x^2 - 1)^k\Big|_{-1}^{1} = 0, \qquad (*) \quad \frac{d^{i+k}}{dx^{i+k}}(x^2 - 1)^i \equiv 0.$$

Durch partielle Integration folgt dann für $i < k$:

$$\begin{aligned}
\int_{-1}^{1} \psi_k(x)\psi_i(x)\,dx &= \int_{-1}^{1} \frac{d^k}{dx^k}(x^2-1)^k \frac{d^i}{dx^i}(x^2-1)^i\,dx \\
&= \frac{d^{k-1}}{dx^{k-1}}(x^2-1)^k \frac{d^i}{dx^i}(x^2-1)^i\Big|_{-1}^{1} - \int_{-1}^{1} \frac{d^{k-1}}{dx^{k-1}}(x^2-1)^k \frac{d^{i+1}}{dx^i}(x^2-1)^i\,dx \\
&\;\;\vdots \\
&= (-1)^k \int_{-1}^{1} (x^2-1)^k \frac{d^{i+k}}{dx^{i+k}}(x^2-1)^i\,dx = 0.
\end{aligned}$$

Dies ist die erste Behauptung. Im Fall $i = k$ gilt

$$\begin{aligned}\int_{-1}^{1} |\psi_k(x)|^2\,dx &= (-1)^k \int_{-1}^{1} (x^2-1)^k \,\frac{d^{2k}}{dx^{2k}}(x^2-1)^k\,dx \\ &= (2k)! \int_{-1}^{1} (1-x^2)^k\,dx =: (2k)!\, I_k.\end{aligned}$$

Für das Integral I_k, $k \geq 1$, folgt weiter durch partielle Integration:

$$\begin{aligned}I_k &= \int_{-1}^{1} (1-x^2)^{k-1}\,dx - \frac{1}{2}\int_{-1}^{1} x(1-x^2)^{k-1}2x\,dx \\ &= I_{k-1} + \frac{1}{2k}x(1-x^2)^k\Big|_{-1}^{1} - \frac{1}{2k}\int_{-1}^{1}(1-x^2)^k\,dx = I_{k-1} - \frac{1}{2k}I_k\end{aligned}$$

bzw.

$$\begin{aligned}I_k &= \frac{2k}{2k+1}I_{k-1} = \ldots = \frac{2k(2k-2)\cdot\ldots\cdot 2}{(2k+1)(2k-1)(2k-3)\cdot\ldots\cdot 3}\,I_0 \\ &= \frac{2^k k! 2^k k!}{(2k+1)2k(2k-1)(2k-2)(2k-3)\cdot\ldots\cdot 4\cdot 3\cdot 2}\,2 = \frac{2^{2k+1}k!^2}{(2k+1)!}.\end{aligned}$$

Dies impliziert die zweite Behauptung:

$$\begin{aligned}\|\varphi_k\| &= \frac{k!}{(2k)!}\|\psi_k\| = \frac{k!}{(2k)!}\sqrt{(2k)!}\,\sqrt{I_k} \\ &= \frac{k!}{(2k)!}\sqrt{(2k)!}\,\sqrt{\frac{2^{2k+1}k!^2}{(2k+1)!}} = \frac{k!}{(2k)!}\sqrt{(2k)!}\,\sqrt{\frac{2^{2k+1}k!^2}{(2k)!(2k+1)}} = \frac{k!^2}{(2k)!}\sqrt{\frac{2^{2k+1}}{2k+1}}.\end{aligned}$$

Zum Nachweis der dritten Behauptung schreiben wir mit Hilfe der obigen Beziehung $(*)$:

$$\begin{aligned}\frac{d^k}{dx^k}(x^2-1)^k\Big|_{x=1} &= \frac{d^{k-1}}{dx^{k-1}}\big(2kx(x^2-1)^{k-1}\big)\Big|_{x=1} \\ &= \frac{d^{k-2}}{dx^{k-2}}\big(2^2x^2k(k-1)(x^2-1)^{k-2}\big)\Big|_{x=1} + \frac{d^{k-2}}{dx^{k-2}}\big(2k(x^2-1)^{k-1}\big)\Big|_{x=1} \\ &= \frac{d^{k-3}}{dx^{k-3}}\big(2^3x^3k(k-1)(k-2)(x^2-1)^{k-3}\big)\Big|_{x=1} + \ldots\Big|_{x=1} \\ &\;\;\vdots \\ &= 2^k x^k k!\Big|_{x=1} + \ldots\Big|_{x=1} = 2^k k!\end{aligned}$$

und damit

$$\varphi_k(1) = \frac{k!}{(2k)!}\psi_k(1) = \frac{k!^2}{(2k)!^2}2^k.$$

Lösung A.2.14: a) Mit Hilfe der Dreiecksungleichung folgt

$$\|x\| = \|x - y + y\| \le \|x - y\| + \|y\|, \qquad \|y\| = \|y - x + x\| \le \|x - y\| + \|x\|$$

und somit $\big|\|x\| - \|y\|\big| \le \|x - y\|$.

b) Die in (a) gezeigte Ungleichung besagt, dass die Funktion $N(\cdot) = \|\cdot\|$ sogar Lipschitzstetig ist. Zu jedem $\varepsilon \in \mathbb{R}_+$ gilt mit $\delta_\varepsilon := \varepsilon \in \mathbb{R}_+$:

$$\forall x, y \in E,\ \|x - y\| < \delta_\varepsilon \quad \Rightarrow \quad \big|\|x\| - \|y\|\big| \le \|x - y\| < \varepsilon.$$

c) Sei $\{e^1, \ldots, e^n\}$ eine Basis von E. Wir betrachte auf E die Norm

$$\|x\|_\infty := \max_{i=1,\ldots,n} |\alpha_i|, \qquad x = \sum\nolimits_{i=1}^n \alpha_i e^i \in E,$$

und werden zeigen, dass jede andere Norm $\|\cdot\|$ auf E zu dieser äquivalent ist, was die Behauptung impliziert. Die Funktion $f(\alpha_1, \ldots, \alpha_n) := \|x\|$ für $x = \sum_{i=1}^n \alpha_i e^i \in E$ ist wegen (s. Teil a)

$$\begin{aligned} |f(\alpha_1, \ldots, \alpha_n) - f(\beta_1, \ldots, \beta_n)| &= \big|\|x\| - \|y\|\big| \le \|x - y\| \\ &\le \sum_{i=1}^n |\alpha_i - \beta_i| \|e^i\| \le \gamma \|x - y\|_\infty, \quad \gamma := \sum\nolimits_{i=1}^n \|e^i\|. \end{aligned}$$

stetig (bzgl. der Norm $\|\cdot\|_\infty$. Auf der ebenfalls bzgl. $\|\cdot\|$ beschränkten und abgeschlossenen (und damit nach dem Satz von Bolzano-Weierstraß kompakten) Menge $S := \{x \in E,\ \|x\|_\infty = 1\}$ nimmt sie dann ihr Maximum M und ihr Minimum m an:

$$0 \le m \le f(\alpha, \ldots, \alpha_n) \le M, \quad x = \sum\nolimits_{i=1}^n \alpha_i e^i \in S.$$

Dabei ist $m > 0$, da $m = f(\alpha_1, \ldots, \alpha_n) = \|x_{\min}\| = 0$ nur für $x_{\min} = 0 \notin S$ möglich ist. Damit folgt schließlich für beliebiges $x \in E$:

$$0 < m \le \Big\| \frac{x}{\|x\|_\infty} \Big\| = \frac{\|x\|}{\|x\|_\infty} = \Big\| \frac{x}{\|x\|_\infty} \Big\| \le M.$$

Lösung A.2.15: Die beste Approximation $p_n \in P_n$ zur Funktion $\sqrt{x} \in C[,1]$ ist charakterisiert durch

$$\int_0^1 (\sqrt{x} - p(x))\varphi(x)\, dx = 0 \quad \forall \varphi \in P_n.$$

(i) Für $p_0 \in P_0$ gilt mit dem Ansatz $p \equiv a_0$:

$$\int_0^1 (\sqrt{x} - a_0)\, dx = 0 \quad \Rightarrow \quad a_0 = \tfrac{2}{3}.$$

(ii) Für $p_1 \in P_1$ gilt mit dem Ansatz $p_1(x) = a_0 + a_1 x$:

$$\int_0^1 (\sqrt{x} - a_0 - a_1 x) x^i \, dx = 0, \quad i = 0, 1, \quad \Rightarrow \quad \tfrac{2}{3} = a_0 + \tfrac{1}{2} a_1, \quad \tfrac{2}{5} = \tfrac{1}{2} a_0 + \tfrac{1}{3} a_1.$$

Dies impliziert $p_1(x) = \frac{4}{15} + \frac{4}{5} x$.

(iii) Für $p_2 \in P_2$ gilt mit dem Ansatz $p_2(x) = a_0 + a_1 x + a_2 x^2$:

$$\int_0^1 (\sqrt{x} - a_0 - a_1 x - a_2 x^2) x^i \, dx = 0, \quad i = 0, 1, 2,$$

bzw. das lineare Gleichungssystem vor und nach Vorwärtselimination:

$$\begin{array}{rcl} a_0 + \frac{1}{2} a_1 + \frac{1}{3} a_2 = \frac{2}{3}, & & a_0 + \frac{1}{2} a_1 + \frac{1}{3} a_2 = \frac{2}{3}, \\ \frac{1}{2} a_0 + \frac{1}{3} a_1 + \frac{1}{12} a_2 = \frac{2}{5}, & \Leftrightarrow & \frac{1}{12} a_1 + \frac{1}{12} a_2 = \frac{1}{15}, \\ \frac{1}{3} a_0 + \frac{1}{4} a_1 + \frac{1}{180} a_2 = \frac{2}{7}. & & \frac{1}{180} a_2 = -\frac{1}{315}. \end{array}$$

Die Lösung ist $(\frac{54}{315}, \frac{432}{315}, -\frac{180}{315})^T$. Die gesuchte beste Approximation ist also

$$p_2(x) = \tfrac{54}{315} + \tfrac{432}{315} x - \tfrac{180}{315} x^2.$$

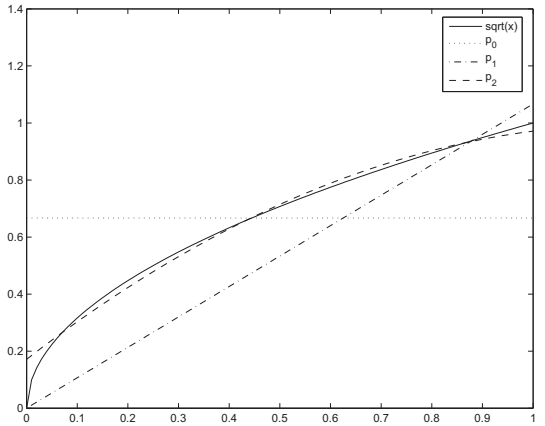


Abbildung A.6: *Gauß-Approximationen in* P_0, P_1, P_2 *der Funktion* $f(x) = \sqrt{x}$ *auf dem Intervall* $[0, 1]$.

Lösung A.2.16:

MATLAB-Programm:

```
clear;
x = linspace(-1,1,101);
k = [0,1,2,3,4];
```

```
for i = 1:length(k)
    LL(i,:) = min(max(factorial(2*k(i))/(2^k(i)*factorial(k(i))^2)
    *L(x,k(i)),-1),1);
    TT(i,:) = T(x,k(i));
end
figure(1);
plot(x,LL,'k')
title('Gauss-Legendre-Polynome fuer k=0,..,4')
print -deps2 legendre1.eps
figure(2);
plot(x,TT,'k')
title('Tschebyscheff-Polynome fuer k=0,..,4')
print -deps2 tschebyscheff1.eps
k = [5,6,7,8,9,10];
for i = 1:length(k)
    LL(i,:) = min(max(factorial(2*k(i))/(2^k(i)*factorial(k(i))^2)
    *L(x,k(i)),-1),1);
    TT(i,:) = T(x,k(i));
end
figure(3);
plot(x,LL,'k')
title('Gauss-Legendre-Polynome fuer k=5,..,10')
print -deps2 legendre2.eps
figure(4);
plot(x,TT,'k')
title('Tschebyscheff-Polynome fuer k=5,..,10')
print -deps2 tschebyscheff2.eps

function erg = L(x,k)
if k == 0
    erg = ones(1,length(x));
elseif k == 1
    erg = x;
else
    erg = x .* L(x,k-1);
    erg = erg - (k-1)^2/(4*(k-1)^2-1) .* L(x,k-2);
end
return

function erg = T(x,k)
if k == 0
    erg = ones(1,length(x));
elseif k == 1
    erg = x;
else
```

```
    erg = 2 * (x .* T(x,k-1));
    erg = erg - T(x,k-2);
end
return
```

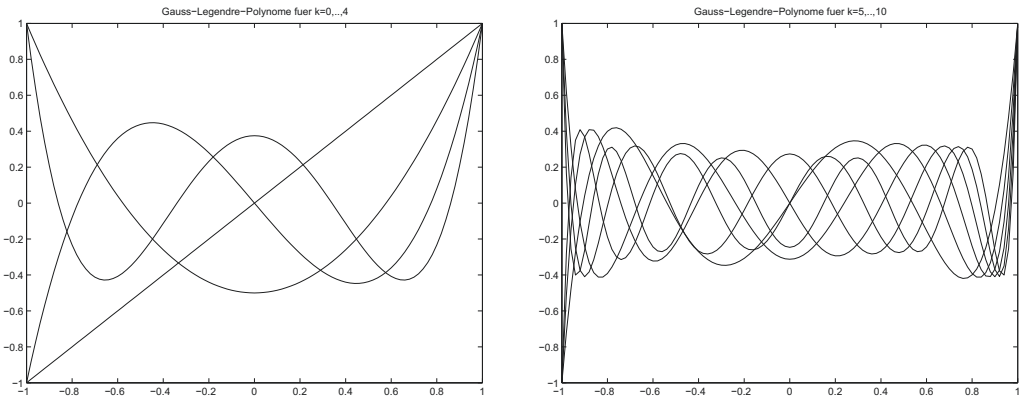


Abbildung A.7: *Gauß-Legendre-Polynome für* $k = 0, \ldots, 4$ *(links) und* $k = 5, \ldots, 10$ *(rechts)*

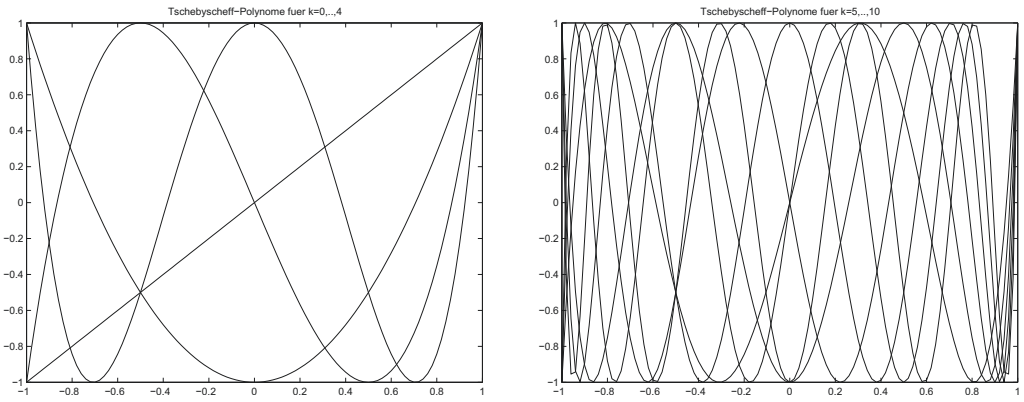


Abbildung A.8: *Tschebyscheff-Polynome für* $k = 0, \ldots, 4$ *(links) und* $k = 5, \ldots, 10$ *(rechts)*

A.3 Kapitel 3

Lösung A.3.1: Zur Anwendung der Fehlerabschätzung für die Newton-Cotes-Formeln sind die entsprechenden Ableitungen des Integranden $f(x)$ abzuschätzen. Es ist

$$f^{(k)}(x) = \frac{d^k}{dx^k}\frac{1}{1+2x} = \frac{(-1)^k 2^k k!}{(1+2x)^{k+1}}, \qquad \max_{x\in[0,1]} |f^{(k)}(x)| = 2^k k!.$$

Die summierte Trapezregel lautet mit Schrittweite $h = (b-a)/N$ und zugehöriger Fehlerdarstellung:

$$I_1^\Sigma(f) = \frac{h}{2}\Big\{ f(a) + 2\sum_{i=1}^{N-1} f(x_i) + f(b)\Big\}$$

$$R_1(f) := I(f) - I_1^\Sigma(f) = -\frac{b-a}{12} h^2 f''(\xi), \quad \xi \in [a,b].$$

Für den vorliegenden Fall ergibt dies

$$|R_1(f)| \le \frac{h^2}{12} 8 \le 10^{-8}\,! \quad \Rightarrow \quad h \le 1.23 \cdot 10^{-4} \;\approx\; \mathbf{8131} \text{ Funktionsauswertungen.}$$

Die summierte Simpson-Regel lautet mit Schrittweite $h = (b-a)/N$ und zugehöriger Fehlerdarstellung:

$$I_2^\Sigma(f) = \frac{h}{2}\Big\{ f(a) + 2\sum_{i=1}^{N-1} f(x_i) + 4\sum_{i=0}^{N-1} f\big(\tfrac{1}{2}(x_i + x_{i+1})\big) + f(b)\Big\}$$

$$I(f) - I_2^\Sigma(f) = -\frac{b-a}{2880} h^2 f^{(iv)}(\xi), \quad \xi \in [a,b].$$

Für den vorliegenden Fall ergibt dies

$$|R_1(f)| \le \frac{h^4}{2880} 384 \le 10^{-8}\,! \quad \Rightarrow \quad h \le 1.66 \cdot 10^{-2} \;\approx\; \mathbf{121} \text{ Funktionsauswertungen.}$$

Lösung A.3.2: Es wird eine Gauß-Formel zum Integrationsgewicht $\omega(x) = \sqrt{|x|}$ konstruiert. Der Theorie folgend werden zur Erzielung der Ordnung $m = 4$ (d. h. exakte Integration für kubische Polynome) die zuehörigen orthogonalen Polynome bis zum Grad $k = 2$ zum Skalarprodukt

$$(f,g)_\omega = \int_{-1}^{1} f(x) g(x) \sqrt{|x|}\, dx$$

bestimmt. Aus Symmetriegründen gilt $(x,1)_\omega = (x^2, x)_\omega = 0$ und dmait:

$$
\begin{aligned}
&p_0(x) \equiv 1;\\
&p_1(x) = x - \gamma, \quad (p_1, p_0)_\omega = (p_1, 1)_\omega = -\gamma(1,1)_\omega = 0\,! \;\Rightarrow\; \gamma = 0,\ p_1(x) = x;\\
&p_2(x) = x^2 - \alpha x - \beta, \quad (p_2, p_0)_\omega = (p_2, 1)_\omega = (x^2 - \beta, 1)_\omega = 0\,! \;\Rightarrow\; \beta = -\tfrac{3}{7};\\
&\qquad (p_2, p_1)_\omega = (p_2, x)_\omega = -\alpha(x,x)_\omega = 0\,! \;\Rightarrow\; \alpha = 0, \quad p_2(x) = x^2 - \tfrac{3}{7}.
\end{aligned}
$$

Mit den Nullstellen $\lambda_1 = -\sqrt{3/7}$ und $\lambda_2 = \sqrt{3/7}$ von p_2 lautet die gesuchte Quadraturformel:

$$I_1(f) = \alpha_1 f(\lambda_1) + \alpha_2 f(\lambda_2).$$

Die Gewichte α_1 und α_2 ergeben sich nach der allgemeinen Formel

$$
\begin{aligned}
\alpha_1 &= \int_{-1}^{1} \Big(\frac{x-\lambda_2}{\lambda_1-\lambda_2}\Big)^2 \sqrt{|x|}\,dx = \int_{-1}^{1} \frac{x-\lambda_2}{\lambda_1-\lambda_2}\sqrt{|x|}\,dx\\
&= \frac{-1}{2\sqrt{3/7}} \int_{-1}^{1} (-\sqrt{3/7})\sqrt{|x|}\,dx = \frac{1}{2}\int_{-1}^{1}\sqrt{|x|}\,dx = \int_0^1 \sqrt{x}\,dx = \tfrac{2}{3},\\
\alpha_2 &= \ \ldots\ = \tfrac{2}{3}.
\end{aligned}
$$

Wir erhalten somit die folgende Quadraturformel der Ordnung $m = 4$:

$$I_1(f) = \tfrac{2}{3} f(-\sqrt{3/7}) + \tfrac{2}{3} f(\sqrt{3/7}).$$

Lösung A.3.3: Zur Berechnung des Integrals bietet sich die Verwendung einer Gauß-Quadraturformel zum Gewicht $\omega(x) = (1-x^2)^{-1/2}$ an, d. h. einer Gauß-Tschebyscheff-Formel. Die zugehörige Fehlerdarstellung ist:

$$I(f\omega) - I_n(f\omega) = R_n(f\omega) = \frac{2\pi}{2^{2n+2}(2n+2)!} f^{(2n+2)}(\xi), \quad \xi \in (-1,1).$$

Im vorliegenden Fall gilt $|f^{(2n+2)}(\xi)| \le (\pi/2)^{2n+2}$ und folglich

$$
\begin{aligned}
n = 2: &\quad |R_2(f\omega)| \le \frac{2\pi}{2^6 6!}\Big(\frac{\pi}{2}\Big)^6 \approx 2.37 \cdot 10^{-3},\\
n = 3: &\quad |R_3(f\omega)| \le \frac{2\pi}{2^8 8!}\Big(\frac{\pi}{2}\Big)^8 \approx 2.25 \cdot 10^{-5}.
\end{aligned}
$$

Wir verwenden also die Formel für $n = 3$. Ihre Stützstellen und Gewichte sind nach der allgemeinen Formel:

$$\alpha_i = \frac{\pi}{n+1}, \quad \lambda_i = \cos\Big(\frac{\pi}{2}\frac{2i+1}{i+1}\Big), \quad i = 0, \ldots, n.$$

und somit für $n = 3$:

$$\alpha_i = \frac{\pi}{4}, \ \ i = 1, \ldots, 4, \quad \lambda_0 = -\lambda_3 \approx 0.023879, \ \ \lambda_1 = -\lambda_2 \approx 1.48281.$$

Da der Cosinus eine gerade Funktion ist, gilt $\cos(\lambda_0) = \cos(\lambda_3)$ und $\cos(\lambda_1) = \cos(\lambda_2)$. Daher ist

$$I_3(f\omega) = \frac{\pi}{2}\big\{ \cos(\tfrac{1}{2}\pi\lambda_0) + \cos(\tfrac{1}{2}\pi\lambda_1)\big\} \approx 0.119285 + 0.8247030 \approx 1.48281.$$

Bemerkung: Man kommt hier wegen der Symmetrie des Integranden mit zwei Funktionsauswertungen aus, also ebensovielen, wie im Fall $n = 2$ erforderlich wären, erzielt aber eine deutlich bessere Genauigkeit.

Lösung A.3.4: Ausgehend von den folgenden Werten des Sinus

x	$\sin(x)$
0	0.00000000000000000
$\frac{1}{16}\pi$	0.19509032201612825
$\frac{1}{8}\pi$	0.38268343236508978
$\frac{3}{16}\pi$	0.55557023301960218
$\frac{1}{4}\pi$	0.70710678118654746
$\frac{5}{16}\pi$	0.83146961230254524
$\frac{3}{8}\pi$	0.92387953251128674
$\frac{7}{16}\pi$	0.98076528040323043
$\frac{1}{2}\pi$	1.00000000000000000

wird zur Schrittweitenfolge $h_i = 2^{-i-1}\pi$, $i = 0, 1, 2, \ldots,$ mit

$$\Big(\frac{h_{i-k}}{h_i}\Big)^q = \Big(\frac{2^{-i+k-1}\pi}{2^{-i-1}\pi}\Big)^2 = 2^{2k}$$

die folgende Rekursionsformel ausgewertet:

$$\begin{aligned} i = 0, 1, 2, \ldots : &\quad a_{i,0} = a(h_i)\,, \\ i = 1, 2, 3, \ldots,\ k = 1, 2, 3, \ldots, i : &\quad a_{ik} = a_{i,k-1} + \frac{a_{i,k-1} - a_{i-1,k-1}}{2^{2k} - 1}. \end{aligned}$$

Dann gilt dann für festes k (Spaltenindex des Extrapolationstableaus):

$$a(0) - a_{ik} = O(h_{i-k}^{(k+1)q}) \quad (i \to \infty),$$

Es ergibt sich das Extrapolationstableau

i	$a_{i,0}$	$a_{i,1}$	$a_{i.2}$	$a_{i,3}$
0	0.78539816339744828			
1	0.94805944896851990	1.00227987749221037		
2	0.98711580097277518	1.00013458497419361	0.99999156547299250	
3	0.99678517188616955	1.00000829552396775	0.99999987622728603	1.00000000814402079

Zur Fehlerkontrolle werden zusätzlich die Größen $b_{ik} \equiv 2a_{i+1,k} - a_{ik}$ bestimmt, was zu folgendem Abbruchkriterium führt:

$$|a_{ii} - b_{ii}| < \text{TOL} \quad \Rightarrow \quad \text{STOP}.$$

Es ergibt sich

i	$\lvert a_{0,0} - b_{0,0}\rvert$	$\lvert a_{1,1} - b_{1,1}\rvert$	$\lvert a_{2,2} - b_{2,2}\rvert$
	0.32532257114214	0.00429058503603	0.00001662150859

Der akzeptierte Näherungswert ist also $a_{2,2} = 0.99999156547299250$.

Lösung A.3.5 (Praktische Aufgabe): Es ist

$$I(f) = \int_0^1 \frac{4}{x^2+1}\, dx = \pi\,.$$

Die Schrittweitenfolge für die Extrapolation der summierten Trapezregel nach dem Romberg-Verfahren ist

$$h_i = 2^{-i}, \quad i = 0, 1, 2, \ldots, 20.$$

Alternativ wird die theoretisch problematische Folge $h_i = 1/(i+1)$ verwendet. Die extrapolierten Werte $a_{ik} \approx a(0) = I(f)$ werden mit Hilfe der Rekursionsformel berechnet:

$$\begin{aligned} i = 0, 1, 2, \ldots: &\quad a_{i,0} = a(h_i)\,, \\ i = 1, 2, 3, \ldots,\ k = 1, 2, 3, \ldots, i: &\quad a_{ik} = a_{i,k-1} + \frac{a_{i,k-1} - a_{i-1,k-1}}{(h_{i-k}/h_i)^q - 1}. \end{aligned}$$

Dann gilt dann für festes k (Spaltenindex des Extrapolationstableaus):

$$a(0) - a_{ik} = O(h_{i-k}^{(k+1)q}) \quad (i \to \infty),$$

Es wird mit den Parameterwerten $q = 2$ (von der Theorie nahegelegt) und $q = 1$ gerechnet. Zur Fehlerkontrolle werden zusätzlich die folgenden Größen bestimmt:

$$b_{ik} \equiv 2a_{i+1,k} - a_{ik}.$$

Das Abbruchkriterium lautet dann $|a_{ii} - b_{ii}| < \text{TOL} \Rightarrow \text{STOP}$. Das Romberg-Verfahren (h^2-Extrapolation mit Schrittweitenfolge $h_i = 2^{-i}$) ergibt die folgenden Resultate:

i	a_{ii}	i	a_{ii}
0	3.0000000000000000	10	3.1415926535897878
1	3.1333333333333333	11	3.1415926535897887
2	3.1421176470588237	12	3.1415926535897918
3	3.1415857837618737	13	3.1415926535898020
4	3.1415926652777171	14	3.1415926535897767
5	3.1415926536382446	15	3.1415926535897918
6	3.1415926535897203	16	3.1415926535897740
7	3.1415926535897940	17	3.1415926535898406
8	3.1415926535897913	18	3.1415926535898255
9	3.1415926535898024	19	3.1415926535897536

MATLAB-Programm:

```
clear;
clf;
n1 = (1:21);
n2 = 2.^(0:20);
n  = 0:19;
[ext1, krit1] = romberg('f',0,1,n1,1);
figure(1)
semilogy(n,abs(ext1-pi),'k-',n,krit1,'k:')
legend('Fehler', 'Kriterium')
title('Folge 1/i mit linerarer Extrapolation')
print -deps2 plot1.eps
[ext2, krit2] = romberg('f',0,1,n2,1);
figure(2)
semilogy(n,abs(ext2-pi),'k-',n,krit2,'k:')
legend('Fehler', 'Kriterium')
title('Folge 1/2^i mit linerarer Extrapolation')
print -deps2 plot2.eps
[ext3, krit3] = romberg('f',0,1,n1,2);
figure(3)
semilogy(n,abs(ext3-pi),'k-',n,krit3,'k:')
legend('Fehler', 'Kriterium')
title('Folge 1/i mit quadratischer Extrapolation')
print -deps2 plot3.eps
[ext4, krit4] = romberg('f',0,1,n2,2);
figure(4)
semilogy(n,abs(ext4-pi),'k-',n,krit4,'k:')
```

```
legend('Fehler', 'Kriterium')
title('Folge 1/2^i mit quadratischer Extrapolation')
print -deps2 plot4.eps
format long
fid = fopen('ext.txt','wt');
fprintf(fid,'Diagonalfolge bei quadratischer Extrapolation ...
mit Romberg-Folge:\n\n');
fprintf(fid,'%12.16f\n',ext4);
fclose(fid);

function erg = f(x)
erg = 4 ./ (x.^2 + 1);
return

function erg = Trapez(fun, a, b, n)
h = (b-a)/n;
erg = feval(fun,a) + feval(fun,b);
for i=1:n-1;
    erg = erg + 2 * feval(fun,a+i*h);
end
erg = erg / 2 * h;
return

function [ext krit] = romberg(fun, a, b, n, o)
h = (b-a)./n;
A   = zeros(length(n));
for i=1:length(n)
    A(i,1) = Trapez(fun,a,b,n(i));
end
ext(1) = A(1,1);
for k = 2:length(n)
    A(k:i,k) = A(k:i, k-1) + (A(k:i, k-1) - A(k-1:i-1, k-1)) ./ ...
        ((h(1:i-k+1)'./h(k:i)').^o - 1);
   if k < length(n)
       ext(k)  = A(k,k);
       krit(k) = 2 * abs(A(k+1,k) - A(k,k));
       if krit(k) < 1.e-10
%           break;
       end
   end
end
return
```

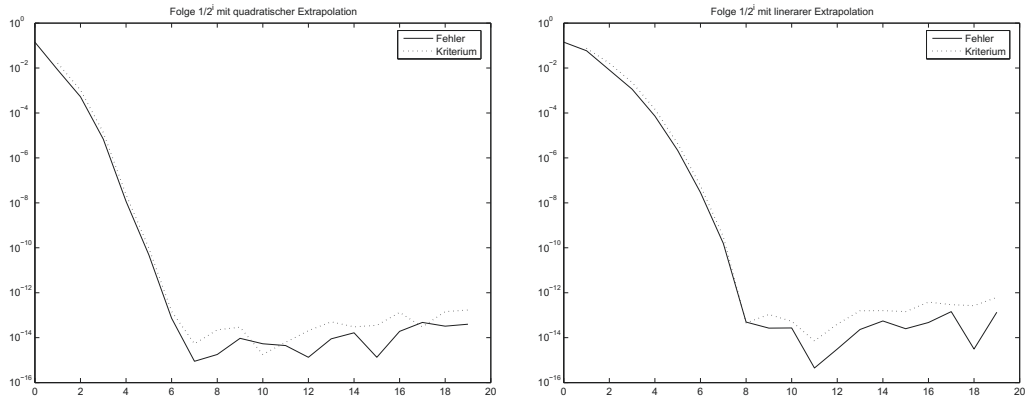


Abbildung A.9: *Extrapolierte Werte a_{ii} zur Schrittweitenfolge $h_i = 2^{-i}$, mit h^2-Extrapolation (links) und mit h-Extrapolation (rechts)*

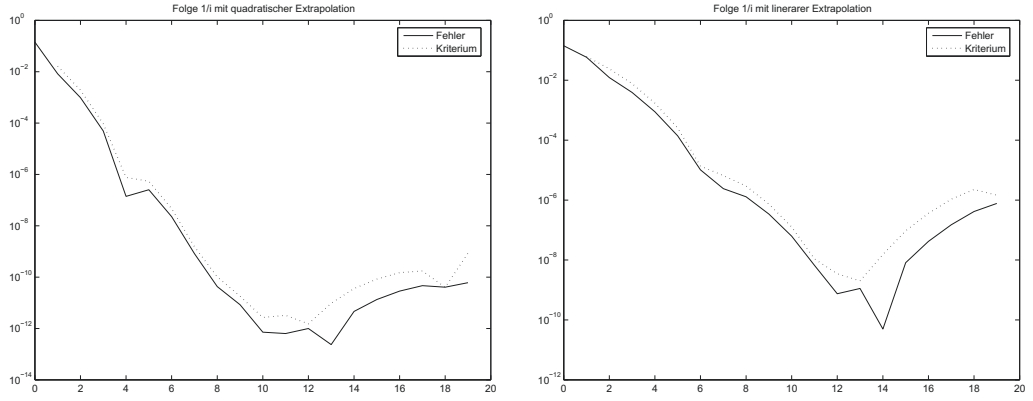


Abbildung A.10: *Extrapolierte Werte a_{ii} zur Schrittweitenfolge $h_i = 1/i$, mit h^2-Extrapolation (links) und mit h-Extrapolation (rechts)*

A.4 Kapitel 4

Lösung A.4.1: (i) Es werden die gewünschten Normeigenschaftzen nachgeprüft:

1. Definitheit:

$$\|A\| \geq 0; \quad \|A\| = 0 \;\Rightarrow\; \|Ax\| = 0 \;\forall x \in \mathbb{K}^n \;\Rightarrow\; A = 0.$$

2. Homogenität:

$$\|\alpha A\| = \sup\big\{\|\alpha Ax\|,\, x \in \mathbb{K}^n, \|x\| = 1\big\} = |\alpha| \sup\big\{\|Ax\|,\, x \in \mathbb{K}^n, \|x\| = 1\big\} = |\alpha|\|A\|.$$

3. Subadditivität:

$$\begin{aligned}\|A + B\| &= \sup\big\{\|(A+B)x\|,\, x \in \mathbb{K}^n, \|x\| = 1\big\} \\ &\leq \sup\big\{\|Ax\| + \|Bx\|,\, x \in \mathbb{K}^n, \|x\| = 1\big\} \\ &\leq \sup\big\{\|Ax\|,\, x \in \mathbb{K}^n, \|x\| = 1\big\} + \sup\big\{\|Bx\|,\, x \in \mathbb{K}^n, \|x\| = 1\big\} = \|A\| + \|B\|.\end{aligned}$$

4. Submultiplikativität:

$$\begin{aligned}\|AB\| &= \sup\left\{\frac{\|ABx\|}{\|x\|},\, x \in \mathbb{K}^n, x \neq 0\right\} \\ &= \sup\left\{\frac{\|ABx\|}{\|Bx\|}\frac{\|Bx\|}{\|x\|},\, x \in \mathbb{K}^n, x \neq 0\,, Bx \neq 0\right\} \\ &\leq \sup\left\{\frac{\|Ay\|}{\|y\|},\, y \in \mathbb{K}^n, y \neq 0\right\} \sup\left\{\frac{\|Bx\|}{\|x\|},\, x \in \mathbb{K}^n, x \neq 0\right\} = \|A\|\,\|B\|.\end{aligned}$$

5. Verträglichkeit:

$$\|Ax\| \leq \frac{\|Ax\|}{\|x\|}\|x\| \leq \|A\|\,\|x\|, \quad x \in \mathbb{K}^n,\, x \neq 0.$$

Also ist $\|\cdot\|$ eine mit der gegebenen Vektornorm verträgliche Matrizennorm.

(ii) Die Quadratsummennorm kann für $n > 1$ wegen

$$\|I\|_{\mathrm{FR}} = \Big(\sum_{i,j=1}^{n} \delta_{ij}^2\Big)^{1/2} = \sqrt{n} \neq 1$$

keine natürliche Matrizennorm sein.

Lösung A.4.2: Die Lösung des Gleichunsgsystems ist $x = (x_1, x_2)^T = (2, 0)^T$ mit $\|x\|_1 = \|x\|_\infty = 2$. Mit der l_p-Vektornorm (hier $p = 1, \infty$) gilt:

$$\frac{\|\delta x\|_p}{\|x\|_p} \leq \frac{\mathrm{cond}_p(A)}{1 - \mathrm{cond}_p(A)\|\delta A\|_p\|A\|_p^{-1}} \left\{\frac{\|\delta b\|_p}{\|b\|_p} + \frac{\|\delta A\|_p}{\|A\|_p}\right\}.$$

mit der zur Vektornorm $\|\cdot\|_p$ gehörenden natürlichen Matrizennorm $\|\cdot\|_p$ und der Konditionszahl

$$\mathrm{cond}_p(A) = \|A\|_p \|A^{-1}\|_p.$$

Die Inverse der Koeffizientenmatrix A ist

$$A^{-1} = \begin{pmatrix} 4/3 & 2/3 \\ 2/3 & -2/3 \end{pmatrix}$$

und somit

$$\begin{aligned} \|A\|_1 = 1,5, \quad \|A^{-1}\|_1 = 2, \quad &\Rightarrow \quad \mathrm{cond}_1(A) = 3, \\ \|A\|_\infty = 1,5, \quad \|A^{-1}\|_\infty = 2 \quad &\Rightarrow \quad \mathrm{cond}_\infty(A) = 3. \end{aligned}$$

Unter der Annahme $|\delta a_{ij}|/|a_{ij}| \le 0,01$ und $|\delta b_i|/|b_i| \le 0,03$ gilt

$$\begin{aligned} \|\delta A\|_1 &= \max_{j=1,2} \sum_{i=1}^{2} |\delta a_{ij}| \le \max_{i,j=1,2} \frac{|\delta a_{ij}|}{|a_{ij}|} \max_{j=1,2} \sum_{i=1}^{2} |a_{ij}| \le 0,01 \, \|A\|_\infty, \\ \|\delta A\|_\infty &= \max_{i=1,2} \sum_{j=1}^{2} |\delta a_{ij}| \le \max_{i,j=1,2} \frac{|\delta a_{ij}|}{|a_{ij}|} \max_{i=1,2} \sum_{j=1}^{2} |a_{ij}| \le 0,01 \, \|A\|_\infty, \\ \|\delta b\|_1 &= \sum_{i=1}^{2} |\delta b_i| \le \max_{i=1,2} \frac{|\delta b_i|}{|b_i|} \sum_{i=1}^{2} |b_i| \le 0,03 \, \|b\|_1, \\ \|\delta b\|_\infty &= \max_{i=1,2} |\delta b_i| \le \max_{i=1,2} \frac{|\delta b_i|}{|b_i|} \max_{i=1} |b_i| \le 0,03 \, \|b\|_\infty, \end{aligned}$$

und folglich für beide Normen $\|\cdot\| = \|\cdot\|_1$ sowie $\|\cdot\| = \|\cdot\|_\infty$:

$$\frac{\|\delta x\|}{\|x\|} \le \frac{3}{1 - 3 \cdot 0,01} \{0,03 + 0,01\} \le \frac{12}{97} \le 0,124.$$

Die Punktmengen im $\mathbb{R}^2$, in denen die Lösungen $x + \delta x$ des gestörten Systems jeweils liegen sind:

$$\begin{aligned} LS_1 &= \{y \in \mathbb{R}^2, \, \|y - x\|_1 = |y_1 - 2| + |y_2| \le 0,248\} \quad \text{(Raute um } x\text{)} \\ LS_\infty &= \{y \in \mathbb{R}^2, \, \|y - x\|_\infty = \max\{|y_1 - 2|, |y_2|\} \le 0,248\} \quad \text{(Rechteck um } x\text{)}. \end{aligned}$$

Lösung A.4.3: a) Lösung des Gleichungssystems:

$$\left[\begin{array}{cccc|c} -\frac{1}{2} & 9 & -2 & 1 & 3 \\ -\frac{3}{2} & 30 & -12 & 0 & 3 \\ 1 & -15 & 0 & -4 & 2 \\ 0 & -6 & 18 & 8 & -4 \end{array}\right] \rightarrow \left[\begin{array}{cccc|c} -\frac{1}{2} & 9 & -2 & 1 & 3 \\ 0 & 3 & -6 & -3 & -6 \\ 0 & 3 & -4 & -2 & 8 \\ 0 & -6 & 18 & 8 & -4 \end{array}\right]$$

$$\rightarrow \left[\begin{array}{cccc|c} -\frac{1}{2} & 9 & -2 & 1 & 3 \\ 0 & 3 & -6 & -3 & -6 \\ 0 & 0 & 2 & 1 & 14 \\ 0 & 0 & 6 & 2 & -16 \end{array}\right] \rightarrow \left[\begin{array}{cccc|c} -\frac{1}{2} & 9 & -2 & 1 & 3 \\ 0 & 3 & -6 & -3 & -6 \\ 0 & 0 & 2 & 1 & 14 \\ 0 & 0 & 0 & -1 & -58 \end{array}\right], \quad x = \begin{bmatrix} 414 \\ 12 \\ -22 \\ 58 \end{bmatrix}.$$

b) LR-Zerlegung $A = LR$:

$$A = \begin{bmatrix} -\frac{1}{2} & 9 & -2 & 1 \\ -\frac{3}{2} & 30 & -12 & 0 \\ 1 & -15 & 0 & -4 \\ 0 & -6 & 18 & 8 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 3 & 1 & 0 & 0 \\ -2 & 1 & 1 & 0 \\ 0 & -2 & 3 & 1 \end{bmatrix} \begin{bmatrix} -\frac{1}{2} & 9 & -2 & 1 \\ 0 & 3 & -6 & -3 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & -1 \end{bmatrix} = LR.$$

$$\det(A) = (-\tfrac{1}{2}) \cdot 3 \cdot 2 \cdot (-1) = 3.$$

c) Berechnung der Inversen A^{-1} und der Konditionszahl:

$$\left[\begin{array}{cccc|cccc} -\frac{1}{2} & 9 & -2 & 1 & 1 & 0 & 0 & 0 \\ -\frac{3}{2} & 30 & -12 & 0 & 0 & 1 & 0 & 0 \\ 1 & -15 & 0 & -4 & 0 & 0 & 1 & 0 \\ 0 & -6 & 18 & 8 & 0 & 0 & 0 & 1 \end{array}\right] \rightarrow \left[\begin{array}{cccc|cccc} -\frac{1}{2} & 9 & -2 & 1 & 1 & 0 & 0 & 0 \\ 0 & 3 & -6 & -3 & -3 & 1 & 0 & 0 \\ 0 & 0 & 2 & 1 & 5 & -1 & 1 & 0 \\ 0 & 0 & 0 & -1 & -21 & 5 & -3 & 1 \end{array}\right]$$

$$\rightarrow \left[\begin{array}{cccc|cccc} -\frac{1}{2} & 0 & 0 & 0 & -72 & 15 & -14 & 2 \\ 0 & 3 & 0 & 0 & 12 & -2 & 3 & 0 \\ 0 & 0 & 2 & 0 & -16 & 4 & -2 & 1 \\ 0 & 0 & 0 & -1 & -21S & 5 & -3 & 1 \end{array}\right] \rightarrow \left[\begin{array}{cccc|cccc} 1 & 0 & 0 & 0 & 144 & -30 & 28 & -4 \\ 0 & 1 & 0 & 0 & 4 & -2/3 & 1 & 0 \\ 0 & 0 & 1 & 0 & -8 & 2 & -1 & 1/2 \\ 0 & 0 & 0 & 1 & 21 & -5 & 3 & -1 \end{array}\right].$$

$$\mathrm{cond}_\infty(A) = \|A\|_\infty \|A^{-1}\|_\infty = 43,5 \cdot 206 = 8961.$$

Lösung A.4.4: Das Resultat der ersten $k-1$ Eliminationsschritte ist eine Blockmatrix $A^{(k-1)}$ der Form

$$A^{(k-1)} = \left[\begin{array}{c|c} 0 & * \\ \hline 0 & \overline{A}^{k-1} \end{array}\right], \qquad \overline{A}^{k-1} \in \mathbb{R}^{(n-k)\times(n-k)} \text{ positiv definit.}$$

Der k-te Eliminationsschritt lautet dann:

$$a_{ij}^{(k)} = a_{ij}^{(k-1)} - \frac{a_{ik}^{(k-1)} a_{kj}^{(k-1)}}{a_{kk}^{(k-1)}}, \quad i,j = k,\ldots,n.$$

(i) Die Hauptdiagonalelemente positiv definiter Matrizen sind positiv: $a_{jj}^{(k-1)} > 0$. Für die Diagonalelemente folgt damit unter Ausnutzung der Symmetrie:

$$a_{ii}^{(k)} = a_{ii}^{(k-1)} - \frac{a_{ij}^{(k-1)} a_{ji}^{(k-1)}}{a_{kk}^{(k-1)}} = a_{ii}^{(k-1)} - \frac{|a_{ij}^{(k-1)}|^2}{a_{kk}^{(k-1)}} \leq a_{ii}^{(k-1)}, \quad i = k,\ldots,n.$$

(ii) Das maximale Element einer positiv definiten Matrix $\overline{A}^{(k-1)}$ liegt auf der Hauptdiagonalen:

$$\max_{k\leq i,j\leq n} |a_{ij}^{(k-1)}| \leq \max_{k\leq i\leq n} |a_{ii}^{(k-1)}|.$$

Die im k-ten Schritt erzeugte teilmatrix $\overline{A}^{(k)}$ ist (nach Vorlesung) wieder positiv definit. Das Resultat (i) impliziert also

$$\max_{k\leq i,j\leq n} |a_{ij}^{(k)}| \leq \max_{k\leq i\leq n} |a_{ii}^{(k)}| \leq \max_{k\leq i\leq n} |a_{ii}^{(k-1)}| \leq \max_{k\leq i,j\leq n} |a_{ij}^{(k-1)}|.$$

Da im k-ten Eliminationsschritt die ersten $k-1$ Zeilen nicht mehr verändert werden, ergibt sich durch Induktion nach $k = 1,\ldots,n$:

$$\max_{1\leq i,j\leq n} |r_{ij}| = \max_{1\leq i,j\leq n} |a_{ij}^{(n-1)}| \leq \max_{1\leq i,j\leq n} |a_{ij}^{(0)}| \leq \max_{1\leq i,j\leq n} |a_{ij}|.$$

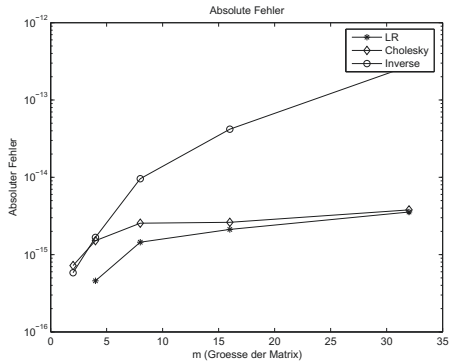
Lösung A.4.5 (Praktische Aufgabe):

MATLAB-Programm:

```
%Berechnung der LR-Zerlegung (ohne Pivotierung)
function [L,R] = LR(A)
[m,n] = size(A);
if (m ~= n)
error('Matrix A ist nicht quadratisch')
end
for k=1:n-1
A(k+1:n,k) = A(k+1:n,k)/A(k,k);
A(k+1:n,k+1:n) = A(k+1:n,k+1:n)-A(k+1:n,k)*A(k,k+1:n);
end
L = eye(n,n) + tril(A,-1);
R = triu(A);

%Hauptprogramm
clear;
for k=1:5
```

```
m=2^k;
%Matrix erstellen
[A]=Laplace_Mat(m);
%LR Zerlegung
[L,R]=LR(A);

%selbstgeschriebene Choleskyzerlegung
[X,Y]=cholesky(L,R);
%selbstgeschriebene Invertierung der Matrix
B=inv_L(X);

%Fehlernormen
AbsErr(k,1)=norm(A-L*R,inf);
AbsErr(k,2)=norm(A-X*Y,inf);
AbsErr(k,3)=norm(eye(m^2)-A*B,inf);
cond_A(k)=norm(A,inf)*norm(B,inf);
end;

%Ausgabe der Fehler
semilogy(2.^(1:5), AbsErr(:,1),'-*k', 2.^(1:5), AbsErr(:,2),'-dk', 2.^(1:5),
     AbsErr(:,3), '-ok');
legend('LR', 'Cholesky', 'Inverse')
title('Absolute Fehler')
xlabel('m (Groesse der Matrix)')
ylabel('Absoluter Fehler')

%Konditionszahl gegen Dimension auftragen
figure;
plot(2.^(1:5), cond_A, 'k');
title('Auftragung Konditionszahl gegen Dimension')
xlabel('m (Groesse der Matrix)')
ylabel('Konditionszahl von A')
clear;

%Funktion zum Aufstellen der Matrix
function [A]=Laplace_Mat(m)
B=4*eye(m);
for i=1:m-1
    B(i+1,i) = -1;
    B(i,i+1) = -1;
end;
A=zeros(m^2);
for i=1:m
    A((i-1)*m+1:i*m,(i-1)*m+1:i*m)=B;
end;
```

```
for i=1:m-1
    A((i-1)*m+1:i*m,i*m+1:(i+1)*m)=-eye(m);
    A(i*m+1:(i+1)*m,(i-1)*m+1:i*m)=-eye(m);
end;
return;

%Funktion, um die Cholesky Zerlegung aus der LR-Zerlegung berechnen
function [X,Y]=cholesky(L,R)
Y=R;
for i=1:length(Y)
    Y(i,:)=Y(i,:)/sqrt(Y(i,i));
end;
X=Y';
return;

%Funktion zur Berechnung der Inversen aus der Choleskyzerlegung
function [B]=inv_L(L)
Z=eye(length(L));
for i=1:length(L)
    Z(i,:)=Z(i,:)/L(i,i);
    for j=i+1:length(L)
        Z(j,:)=Z(j,:)-L(j,i)*Z(i,:);
    end;
end;
B=Z'*Z;
```



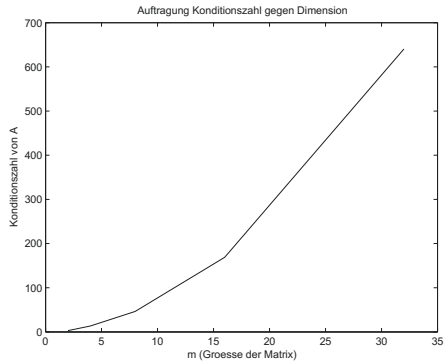


Abbildung A.11: *Residuen-Normen und Kondition in Abhäangigkeit von* m $(h = 1/m)$

Lösung A.4.6: Seien $\lambda_1, \ldots, \lambda_n$ die Eigenwerte der hermiteschen (und positiv semidefiniten) Matrix $\bar{A}^T A$ und $\{w^1, \ldots, w^n\}$ ein zugehöriges ONS von Eigenvektoren: $\bar{A}^T A w = \lambda w^i$. Damit gilt:

$$\|Ax\|_2^2 = (\bar{A}^T Ax, x)_2 = \sum_{i,j=1}^{n} \lambda_i (x, w^i)_2 \overline{(x, w^j)_2} (w^i, w^j)_2 \leq \max_{1 \leq i \leq n} |\lambda_i| \sum_{i=1}^{n} |(x, w^i)_2|^2 = \|x\|_2^2.$$

Es folgt

$$\|A\|_2 = \sup\left\{\frac{\|Ax\|_2}{\|x\|_2}, x \in \mathbb{K}^n \setminus \{0\}\right\} \leq \max_{1\leq i\leq n} \sqrt{|\lambda_i|}.$$

Andererseits gilt (wegen $\lambda_i \geq 0$):

$$\sqrt{|\lambda_i|} = \sqrt{\lambda_i (w^i, w^i)_2} = \sqrt{(\bar{A}^T A w^i, w^i)_2} = \|Aw^i\|_2 \leq \|A\|_2.$$

Lösung A.4.7: (i) Sei

$$\mathcal{L} := \{L \in \mathbb{R}^{n\times n},\ L \text{ reguläre untere Dreiecksmatrix mit } l_{ii} = 1\},$$
$$\mathcal{R} := \{R \in \mathbb{R}^{n\times n},\ L \text{ reguläre obere Dreiecksmatrix}\}.$$

Es sind die folgenden Gruppeneigenschaften bzgl. der Matrizenmultiplikation $\circ$ nachzuweisen:
(G1) Abgeschlossenheit: $L_1, L_2 \in \mathcal{L} \Rightarrow L_1 \circ L_2 \in \mathcal{L}$.
(G2) Assoziativgesetz: $L_1, L_2, L_3 \in \mathcal{L} \Rightarrow L_1 \circ (L_2 \circ L_3) = (L_1 \circ L_2) \circ L_3$.
(G3) Neutrales Element I: $L \in \mathcal{L} \Rightarrow L \circ I = L$.
(G4) Inverse: $L \in \mathcal{L} \Rightarrow \exists L^{-1} \in \mathcal{L} : L \circ L^{-1} = I$.

(G1) folgt durch Nachrechnen. (G2) und (G3) folgen aus den Eigenschaften der Matrizenmultiplikation. (G4) sieht man mit der Inversenbestimmung mit simultaner Elimination:

$$\left[\begin{array}{ccc|ccc} 1 & & 0 & 1 & & 0 \\ & \ddots & & & \ddots & \\ 0 & & 1 & * & & 1 \end{array}\right] \quad \Rightarrow \quad L^{-1} = \left[\begin{array}{ccc} 1 & & 0 \\ & \ddots & \\ * & & 1 \end{array}\right] \in \mathcal{L}.$$

Die Gruppe $\mathcal{L}$ ist i. Allg. *nicht* abelsch, wie das folgende 3×3-Beispiel zeigt:

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix} \neq \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ -1 & -1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ -1 & -1 & 1 \end{bmatrix}.$$

Die Argumentation für $\mathcal{R}$ ist analog. Dabei ist die Gruppe $\mathcal{R}$ i. Allg. ebenfalls *nicht* abelsch, wie das folgenden einfache 2×2-Beispiels zeigt:

$$\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \circ \begin{bmatrix} -1 & 1 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 2 \\ 0 & 1 \end{bmatrix} \neq \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 1 \\ 0 & 1 \end{bmatrix} \circ \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}.$$

(ii) Zum Nachweis der Eindeutigkeit der LR-Zerlegung seien für eine reguläre Matrix $A \in \mathbb{R}^{n\times n}$ zwei LR-Zerlegungen $A = L_1 R_1 = L_2 R_2$ gegeben. Dann ist nach (i) $L_1, L_2 \in \mathcal{L}$ sowie $R_1, R_2 \in \mathcal{R}$ und folglich

$$\underbrace{R_1 R_2^{-1}}_{\in \mathcal{R}} = \underbrace{L_1^{-1} L_2}_{\in \mathcal{L}} = \operatorname{diag}(d_{ii}).$$

Mit L_1 (und L_2) hat auch die Inverse L_1^{-1} Einsen auf der Hauptdiagonale. Also muss $d_{ii} = 1$ sein, was schließlich $R_1 = R_2$ bzw. $L_2 = L_1$ impliziert.

Lösung A.4.8: Der sog. „Cholesky-Algorithmus“ zur Berechnung der Zerlegungsmatrix

$$\tilde{L} = \begin{bmatrix} \tilde{l}_{11} & & \\ \vdots & \ddots & \\ \tilde{l}_{n1} & \cdots & \tilde{l}_{nn} \end{bmatrix}$$

geht direkt von der Beziehung $A = \tilde{L}\tilde{L}^T$ aus, die man als ein System von $n(n+1)/2$ Gleichungen für die Größen $\tilde{l}_{jk}$, $k \le j$, auffassen kann. Dieses System wird sukzessive nach den Spalten von $\tilde{L}$ aufgelöst. Ausmultiplizieren von

$$\begin{bmatrix} \tilde{l}_{11} & & \\ \vdots & \ddots & \\ \tilde{l}_{n1} & \cdots & \tilde{l}_{nn} \end{bmatrix} \begin{bmatrix} \tilde{l}_{11} & \cdots & \tilde{l}_{n1} \\ & \ddots & \vdots \\ & & \tilde{l}_{nn} \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix}$$

ergibt in der ersten Spalte von $\tilde{L}$:

$$\tilde{l}_{11}^2 = a_{11}\,, \quad \tilde{l}_{21}\,\tilde{l}_{11} = a_{21}\,, \quad \ldots\,, \quad \tilde{l}_{n1}\,\tilde{l}_{11} = a_{n1}\,,$$

woraus sich

$$\tilde{l}_{11} = \sqrt{a_{11}}\,, \quad j = 2, \ldots, n: \quad \tilde{l}_{j1} = \frac{a_{j1}}{\tilde{l}_{11}}\,,$$

berechnet. Seien nun für ein $i \in \{2, \cdots, n\}$ die Elemente der 1-ten bis $(i-1)$-ten Spalte $\tilde{l}_{jk}$, $k = 1, \ldots, i-1$, $j = k, \ldots, n$, schon bekannt. Dann erhält man aus

$$\begin{aligned} \tilde{l}_{i1}^2 + \tilde{l}_{i2}^2 + \ldots + \tilde{l}_{i,i-1}^2 + \tilde{l}_{ii}^2 = a_{ii}\,, \quad \tilde{l}_{ii} > 0\,, \\ \tilde{l}_{j1}\tilde{l}_{i1} + \tilde{l}_{j2}\tilde{l}_{i2} + \ldots + \tilde{l}_{j,i-1}\tilde{l}_{i,i-1} + \tilde{l}_{ji}\tilde{l}_{ii} = a_{ji}, \quad j = i+1, \ldots, n, \end{aligned}$$

bzw.

$$\begin{aligned} \tilde{l}_{ii} &= \sqrt{a_{ii} - \tilde{l}_{i1}^2 - \tilde{l}_{i2}^2 - \ldots - \tilde{l}_{i,i-1}^2}\,, \\ \tilde{l}_{ji} &= \tilde{l}_{ii}^{-1}\{a_{ji} - \tilde{l}_{j1}\tilde{l}_{i1} - \tilde{l}_{j2}\tilde{l}_{i2} - \ldots - \tilde{l}_{j,i-1}\tilde{l}_{i,i-1}\}, \quad j = i+1, \ldots, n, \end{aligned}$$

die Elemente der i-ten Spalte. Nach diesem Algorithmus ergibt sich für die gegebene Matrix:

$$L_{\mathrm{Ch}} = \begin{bmatrix} 2,236 & 0 & 0 & 0 \\ -2,236 & 1,414 & 0 & 0 \\ 0 & -1,414 & 4,243 & 0 \\ 0 & 0 & -4,242 & 1,003 \end{bmatrix}.$$

Vorwärteinsetzen und Rückwärtseinsetzen:

$$L_{\text{Ch}}y = b \quad \Rightarrow \quad y = \begin{bmatrix} 2,236 \\ -1,415 \\ 4,242 \\ 0.9916 \end{bmatrix}, \qquad L_{\text{Ch}}^T x = y \quad \Rightarrow \quad x = \begin{bmatrix} 1,987 \\ 0,9873 \\ 1,988 \\ 0,9886 \end{bmatrix}.$$

Nachkorrektur der Näherungslösung $x^{(0)}$ (Defekt 8-stellig berechnet):

$$d = \begin{bmatrix} 0,0015 \\ -0,0001 \\ 0,0094 \\ 0,0006 \end{bmatrix}$$

Löse defektgleichung $Ak = d$ mit Hilfe der bereits erstellten Cholesky-Zerlegung:

$$L_{\text{Ch}}y = d \quad \Rightarrow \quad y = \begin{bmatrix} 0,0007 \\ 0,0010 \\ 0,0025 \\ 0,0112 \end{bmatrix}, \qquad L_{\text{Ch}}^T k = y \quad \Rightarrow \quad k = \begin{bmatrix} 0,0128 \\ 0,0125 \\ 0,0118 \\ 0,0112 \end{bmatrix}.$$

Korrekturschritt:

$$x^{(1)} = x^{(0)} + k = \begin{bmatrix} 1,987 \\ 0,9873 \\ 1,988 \\ 0,9886 \end{bmatrix} + \begin{bmatrix} 0,0128 \\ 0,0125 \\ 0,0118 \\ 0,0112 \end{bmatrix} = \begin{bmatrix} 2,000 \\ 0,9998 \\ 2,000 \\ 0,9998 \end{bmatrix}.$$

Lösung A.4.9: Sei A eine Bandmatrix mit $m_l = m_r =: m$. Man mache sich von dieser Situation eine Skizze.
i) Der k-te Eliminationsschritt

$$a_{ij}^{(k)} = a_{ij}^{(k-1)} - \frac{a_{ik}^{(k-1)}}{a_{kk}^{(k-1)}} a_{kj}^{(k-1)}, \quad b_i^{(k)} = b_i^{(k-1)} - \frac{a_{ik}^{(k-1)}}{a_{kk}^{(k-1)}} b_k^{(k-1)}, \quad i,j = k+1,\ldots,k+m.$$

erfordert im Wesentlichen m Divisionen und m^2 Multiplikationen und Additionen; also alles zusammen

$$N_{\text{Band-Gauß}} = nm^2 + O(nm) \quad \text{a. Op.}$$

für die $n-1$ Schritte der Vorwärtselimination zur Berechnung der Matrix R und simultan dazu der Matrix L. Für die dünn besetzte Modellmatrix ist $N_{\text{Band-Gauß}} = 10^8 + O(10^6)$ a. Op. im Gegensatz zu $N_{\text{Band-Gauß}} = \frac{1}{3}10^{12} + O(10^8)$ a. Op. für eine entsprechende voll besetzte Matrix.

ii) Ist A noch symmetrische (und positiv definit), so erhält man die Cholesky-Zerlegung aus der LR-Zerlegung durch

$$A = \tilde{L}\tilde{L}^T, \qquad \tilde{L} = LD^{1/2},\ D = \mathrm{diag}(r_{ii}).$$

Wegen der Symmetrie aller entstehenden reduzierten Teilmatrizen brauchen nur die Elemente auf und oberhalb der Hauptdiagonalen berechnet zu werden. Dies reduziert den Aufwand auf $N_{\text{Band-Gauß}} = \frac{1}{2}nm^2 + O(nm)$ a. Op. , d. h. für die Modellmatrix auf $N_{\text{Band-Gauß}} = \frac{1}{2}10^8 + O(10^6)$ a. Op. .

Lösung A.4.10 (Praktische Aufgabe):

MATLAB-Programm:

```
clear
m = 2:2:20;
disp('Laplace:')
for i = 1:length(m)
  Am = A(m(i));
  L1 = mycholesky(Am);
  L2 = chol(Am)';
  disp(['m = ', num2str(m(i)), ': myres = ', num2str(norm(full(Am-L1*L1'),
  inf)), ...
      ', res = ', num2str(norm(full(Am-L2*L2'),inf))]);
  err1(i) = norm(full(Am-L1*L1'),inf);
  err2(i) = norm(full(Am-L2*L2'),inf);
end
clf
semilogy(m, err1, 'k-', 'LineWidth', 2)
hold on
semilogy(m, err2, 'k--', 'LineWidth', 2)
xlabel('m')
ylabel('||A - LL^T||')
title('Laplace Matrix')
legend('meins', 'Matlab')
print -deps2 laplace.eps
m = 2:2:20;
disp('Hilbert:')
for i = 1:length(m)
    Am = hilb(m(i));
    L1 = mycholesky(Am);
    err1(i) = norm(Am-L1*L1', inf);
    try
        L2 = chol(Am)';
        err2(i) = norm(Am-L2*L2', inf);
        disp(['m = ', num2str(m(i)), ': myres = ', num2str(norm(Am-L1*L1',
```

```
        inf)), ...
            ', res = ', num2str(norm(Am-L2*L2', inf))]);
    catch
        err2(i) = 0;
        disp(['m = ', num2str(m(i)), ': myres = ', num2str(norm(Am-L1*L1',
         inf)), ...
            ', Matlabinterne Cholesky-Zerlegung versagt!'])
    end
end
hold off
semilogy(m, err1, 'k-', 'LineWidth', 2)
hold on
semilogy(m, err2, 'k--', 'LineWidth', 2)
xlabel('m')
ylabel('||A - LL^T||')
title('Hilbert Matrix')
legend('meins', 'Matlab')
print -deps2 hilbert.eps

function erg = A(m)
erg = zeros(m^2);
Bm = B(m);
Id = -eye(m);
for i = 1:m
  erg((i-1)*m+1:i*m,(i-1)*m+1:i*m) = Bm;
  if (i < m)
    erg((i-1)*m+1:i*m,i*m+1:(i+1)*m) = Id;
    erg(i*m+1:(i+1)*m,(i-1)*m+1:i*m) = Id;
  end
end
return

function erg = B(m)
erg = 4*eye(m);
erg(2:m,1:m-1) = erg(2:m,1:m-1) - eye(m-1);
erg(1:m-1,2:m) = erg(1:m-1,2:m) - eye(m-1);
return

function erg = B(m)
erg = 4*eye(m);
erg(2:m,1:m-1) = erg(2:m,1:m-1) - eye(m-1);
erg(1:m-1,2:m) = erg(1:m-1,2:m) - eye(m-1);
return

function L = mycholesky(A)
```

```
n = length(A);
L = zeros(n);
% Berechne erste Spalte
L(1,1) = sqrt(A(1,1));
for i = 1:n
  L(i,1) = A(i,1) / L(1,1);
end
% Berechne restliche Spalten
for i = 2:n
  L(i,i) = sqrt(A(i,i) - sum(L(i,1:i).^2));
  for j = i+1:n
    L(j,i) = (A(j,i) - L(j,1:i)*L(i,1:i)')/L(i,i);
  end
end
return
```

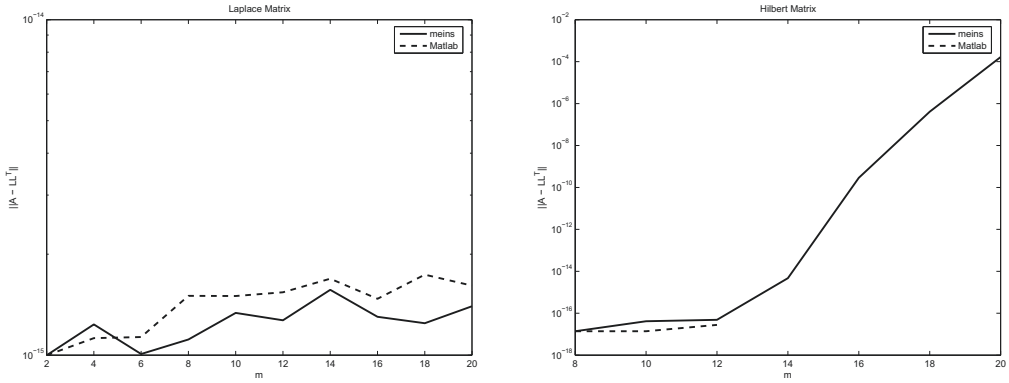


Abbildung A.12: *Residuennorm in Abhqangigkeit von n für die Berechnung der Cholesky-Zerlegung der „Laplace-Matrix“ (oben) und der „Hilbert-Matrix“ (unten)*

Lösung A.4.11: a) Mit Hilfe der Gauß-Elimination wird der Rang der Matrix A mit dem der Matrix $[A|b]$ verglichen:

$$\left[\begin{array}{ccc|c} 1 & 3 & -4 & 1 \\ 3 & 9 & -2 & 1 \\ 4 & 12 & -6 & 1 \\ 2 & 6 & 2 & 1 \end{array}\right] \quad \rightarrow \quad \left[\begin{array}{ccc|c} 1 & 3 & -4 & 1 \\ 0 & 0 & 10 & -2 \\ 0 & 0 & 10 & -3 \\ 0 & 0 & 10 & -1 \end{array}\right]$$

Wegen $\operatorname{Rang} A = 2 \neq 3 = \operatorname{Rang}[A|b]$ ist das Gleichungssystem nicht lösbar. Insbesondere hat A nicht den maximal möglichen Rang 3.

b) Das Normalgleichungssystem ist

$$\begin{bmatrix} 1 & 3 & 4 & 2 \\ 3 & 9 & 12 & 6 \\ -4 & -2 & -6 & 2 \end{bmatrix} \begin{bmatrix} 1 & 3 & -4 \\ 3 & 9 & -2 \\ 4 & 12 & -6 \\ 2 & 6 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 & 3 & 4 & 2 \\ 3 & 9 & 12 & 6 \\ -4 & -2 & -6 & 2 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}.$$

bzw.

$$\begin{bmatrix} 30 & 90 & -30 \\ 90 & 270 & -90 \\ -30 & -90 & 60 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 10 \\ 30 \\ -10 \end{bmatrix}.$$

Die Matrix $A^TA \in \mathbb{R}^{3\times 3}$ hat wegen $\operatorname{Rang} A = 2 < 3$ einen eindimensionalen Nullraum und ist damit insbesondere nicht regulär. Elimination ergibt

$$\left[\begin{array}{ccc|c} 30 & 90 & -30 & 10 \\ 90 & 270 & -90 & 30 \\ -30 & -90 & 60 & -10 \end{array}\right] \rightarrow \left[\begin{array}{ccc|c} 30 & 90 & -30 & 10 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 30 & 0 \end{array}\right] \rightarrow \left[\begin{array}{ccc|c} 3 & 9 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right]$$

und somit die allgemeine Lösung $x = (\frac{1}{3} - 3t, t, 0)^T$, $t \in \mathbb{R}$.

c) Das Normalgleichungsssystem ist also lösbar, doch ist die Lösung nicht eindeutig.

d) Da A nicht maximalen Rang hat, ist die Matrix $A^TA \in \mathbb{R}^{3\times 3}$ nicht injektiv und folglich nur positiv semi-definit. Gegenbeispiel: $x = (-3, 1, 0)^T$.

Lösung A.4.12: a) Das kurze Argument geht von dem transformierten System $\tilde{A}x = DAx = Db = \tilde{b}$ aus und bildet die zugehörige Normalgleichung:

$$\tilde{A}^T\tilde{A}x = \tilde{A}^T\tilde{b} \quad \Leftrightarrow \quad (DA)^TDAx = (DA)^TDb \quad \Leftrightarrow \quad A^TD^2Ax = A^TD^2b.$$

b) Das längere Argument wertet die notwenidge Bedingung für ein Minimum der Defektnorm $\|D(Ax-b)\|_2$ aus:

$$\begin{aligned} 0 &= \frac{\partial}{\partial x_i}\|D(Ax-b)\|_2^2 = \frac{\partial}{\partial x_i}\sum_{j=1}^{n}\Big|d_{jj}\Big(\sum_{k=1}^{n}a_{jk}x_k - b_j\Big)\Big|^2 \\ &= 2\sum_{j=1}^{n} d_{jj}\Big(\sum_{k=1}^{n}a_{jk}x_k - b_j\Big)d_{jj}a_{ji} = 2\sum_{j=1}^{n} d_{jj}a_{ji}d_{jj}\Big(\sum_{k=1}^{n}a_{jk}x_k - b_j\Big) \\ &= 2\sum_{j=1}^{n} a_{ij}^Td_{jj}^2\Big(\sum_{k=1}^{n}a_{jk}x_k - b_j\Big) = 2(A^TD^2(Ax-b))_i, \quad i = 1,\ldots,n. \end{aligned}$$

Dies impliziert gerade die modifizierte Normalgleichung

$$A^T D^2 A x = A^T D^2 b.$$

Lösung A.4.13: 1. Schritt:

$$\tilde{v}_1 = a_1 - ||a_1||_2 e_1 = \begin{bmatrix} 0 \\ -2 \\ 0 \\ 0 \end{bmatrix} - 2 \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} -2 \\ -2 \\ 0 \\ 0 \end{bmatrix} \quad \Rightarrow \quad v_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} -1 \\ -1 \\ 0 \\ 0 \end{bmatrix}.$$

$$S_1 = I - 2v_1 v_1^T = \begin{bmatrix} 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad S_1 A = \begin{bmatrix} 2 & -1 & 0 & 0 \\ 0 & 0 & 0 & -2 \\ 0 & -2 & 1 & 0 \\ 0 & 0 & -2 & 1 \end{bmatrix}.$$

2. Schritt:

$$\tilde{v}_2 = \begin{bmatrix} 0 \\ a_2 - \|a_2\|_2 e_2 \end{bmatrix} = \begin{bmatrix} 0 \\ -2 \\ -2 \\ 0 \end{bmatrix} \Rightarrow v_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 0 \\ -1 \\ -1 \\ 0 \end{bmatrix}$$

$$S_2 = I - 2v_2 v_2^T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad S_2 S_1 A = \begin{bmatrix} 2 & -1 & 0 & 0 \\ 0 & 2 & -1 & 0 \\ 0 & 0 & 0 & 2 \\ 0 & 0 & -2 & 1 \end{bmatrix}.$$

3. Schritt:

$$\tilde{v}_3 = \begin{bmatrix} 0 \\ 0 \\ a_3 - \|a_3\|_2 e_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -2 \\ -2 \end{bmatrix} \Rightarrow v_3 = \frac{1}{\sqrt{2}} \begin{bmatrix} 0 \\ 0 \\ -1 \\ -1 \end{bmatrix}, \quad S_3 = I - 2v_3 v_3^T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 \end{bmatrix}.$$

Das Ergebnis ist:

$$R = S_3 S_2 S_1 A = \begin{bmatrix} 2 & -1 & 0 & 0 \\ 0 & 2 & -1 & 0 \\ 0 & 0 & 2 & -1 \\ 0 & 0 & 0 & -2 \end{bmatrix}, \qquad Q = S_1 S_2 S_3 = \begin{bmatrix} 0 & 0 & 0 & -1 \\ -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \end{bmatrix}.$$

Lösung A.4.14: Es ergibt sich das überbestimmte Gleichungssystem für die Unbekannten $1/p$ und e/p:

$$Ax = \begin{bmatrix} 1 & -0,63 \\ 1 & -0,39 \\ 1 & -0,12 \\ 1 & 0,31 \\ 1 & 0,59 \end{bmatrix} \begin{bmatrix} 1/p \\ e/p \end{bmatrix} = \begin{bmatrix} 0,1 \\ 0,2 \\ 0,4 \\ 0,77 \\ 1 \end{bmatrix}$$

Das Normalgleichungssystem $A^T Ax = A^T b$ (zweistellige Rechnung)

$$\begin{bmatrix} 5 & -0,2 \\ -0,2 & 1 \end{bmatrix} \begin{bmatrix} 1/p \\ e/p \end{bmatrix} = \begin{bmatrix} 2,5 \\ 0,64 \end{bmatrix}$$

hat die Lösung $1/p = 0.53,\ e/p = 0.74$ bzw. $e = 1,4,\ p = 1,9$. Die Kometenbahn ist also eine Hyperbel.

Lösung A.4.15 (Praktische Aufgabe):

MATLAB-Programm:

```
function [Q,R] = myQR(A)
[m,n] = size(A);
Q = eye(m);
for k = 1:n
   Tmp = Q'*A;
   B = Tmp(k:m,k:n);
   Qtilde = Q1(B);
   Tmp = eye(m);
   Tmp(k:m,k:m) = Qtilde;
   Q = Q * Tmp;
end
R = Q'*A;
return

function erg = Q1(A)
n = length(A);
a1 = A(:,1);
s = sign(A(1,1));
if s==0
    s = 1; end
alpha = -s * norm(a1);
v = a1;
v(1) = v(1) - alpha;
erg = eye(n) - 2*v*v' / (v'*v);
return
```

```
%Teil 1
clear;
A = [ 0  0  0 2; -2  1  0 0; 0 -2  1 0; 0  0 -2 1]
[Q,R] = myQR(A)
%Teil 2
n = [3:15].^2;
for i=1:length(n)
    A = hilb(n(i));
    %QR-Zerlegung
    [Q,R] = myQR(A);
    err1(i) = norm(R'*R-A*A,inf);
    %Cholesky-Zerlegung von A*A ueber eigene LR-Zerlegung
    [LL,RR] = LR(A*A);
    LLL = CHOL(RR)';
    err2(i) = norm(LLL*LLL'-A*A,inf);
end
clf;
semilogy(n,err1,'k-',n,err2,'k--')
xlabel('Dimension')
ylabel('Defektnorm')
title('Cholesky-Zerlegung der Hilbertmatrix')
legend('mit QR-Zerlegung von A', 'mit LR-Zerlegung von A*A',
'Location','NorthWest')
print -depsc2 plot.eps

Die Cholesky- und LR-Zerlegungen sind von einer frueheren Aufgabe
uebernommen.
```

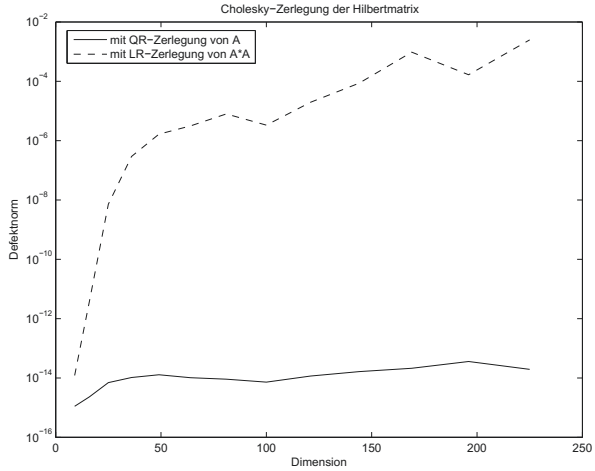


Abbildung A.13: *Residuennorm in Abhïangigkeit von n*

A.5 Kapitel 5

Lösung A.5.1: a) Intervallschachtelung: Ausgehend vom Startintervall $[a_0, b_0] = [2, 4]$ lautet der Intervallschachtelungsalgorithmus:

$$x_t = \tfrac{1}{2}(a_t + b_t), \quad a_{t+1} := \begin{cases} a_t, & f(a_t)f(x_t) < 0 \\ x_t, & \text{sonst} \end{cases}, \quad b_{t+1} = \begin{cases} x_t, & f(a_t)f(xc_t) < 0 \\ b_t, & \text{sonst} \end{cases}.$$

mit dem Abbruchkriterium:

$$b_t - a_t < 5 \cdot 10^{-6} \quad \text{oder} \quad f(a_t) = 0 \ \text{ oder } \ f(b_t) = 0.$$

t	a_t	b_t
1	3,0000000	3,5000000
2	3,0000000	3,2500000
3	3,1250000	3,2500000
4	3,1250000	3,1875000
5	3,1250000	3,1562500
6	3,1406250	3,1562500
7	3,1406250	3,1484375
8	3,1406250	3,1445312
9	3,1406250	3,1425781
10	3,1406250	3,1416015
11	3,1411132	3,1416015
12	3,1413574	3,1416015
13	3,1414784	3,1416015
14	3,1415405	3,1416015
15	3,1415710	3,1416015
16	3,1415863	3,1416015
17	3,1415863	3,1415939
18	3,1415901	3,1415939

b) Fixpunktiteration: Ausgehend vom Startwert $x_0 = 4$ lautet die Fixpunktiteration:

$$x_t = x_{t-1} + f(x_{t-1})$$

mit dem Abbruchkriterium

$$|x_t - x_{t-1}| < 5 \cdot 10^{-6}.$$

t	x_t
1	$3,2431975$
2	$3,4117873$
3	$3,1415926$

c) Newton-Verfahren: Ausgehend vom Startwert $x_0 = 4$ lautet das Newton-Verfahren:

$$x_t = x_{t-1} - \frac{f(x_{t-1}}{f'(x_{t-1}} = x_{t-1} - \tan(x_{t-1})$$

mit dem Abbruchkriterium

$$|f(x_t)| < 5 \cdot 10^{-6}.$$

t	x_t
1	$2,8421787$
2	$3,1508729$
3	$3,1415923$

Die Iterationen (b) und (c) konvergieren kubisch. Zur Erklärung notieren wir:

$$\begin{aligned} g_b'(x) &= 1 + \cos(x) \quad \Rightarrow \quad g_b'(\pi) = 1 + \cos(\pi) = 0. \\ g_b''(x) &= -\sin(x) \quad \Rightarrow \quad g_b''(\pi) = 0. \\ g_c''(x) &= 1 - (\cos(x))^{-2} \quad \Rightarrow \quad g_c''(\pi) = 0. \end{aligned}$$

Lösung A.5.2: Die Ableitung der Fixpunktabbildung g im Fixpunkt lässt sich abschätzen wie folgt:

$$\begin{aligned} &a) \quad g(x) = -\ln(x) \quad \Rightarrow \quad |g'(z)| = \left|\frac{1}{z}\right| \geq \frac{1}{0.6} > 1. \\ &b) \quad g(x) = e^{-x} \quad \Rightarrow \quad |g'(z)| = |e^{-z}| \leq 4e^{-0,5} \approx 0,61 < 1. \\ &c) \quad g(x) = \tfrac{1}{2}(x + e^{-x}) \quad \Rightarrow \quad |g'(z)| = \tfrac{1}{2}|1 - e^{-z}| \leq \tfrac{1}{2}|1 - e^{-0,6}| \approx 0,23 < 1. \end{aligned}$$

Die Iteration (a) divergiert. Die Iterationen (b) und (c) können verwendet werden, wobei (c) die kleinere (lineare) Konvergenzrate besitzt.

Ein Vergleich von (b) und (c) legt die folgende allgemeine Form einer Iterationsvorschrift nahe:

$$(1 + \beta)x_{t+1} = \beta x_t + e^{-x_t}, \quad g(x) = \frac{1}{1+\beta}(\beta x - e^{-x}).$$

Dafür gilt

$$|g'(z)| = \frac{1}{1+\beta}|\beta - e^{-z}| = \frac{1}{1+\beta}|\beta - z|.$$

Die „richtige“ Wahl ist also $\beta = z$.

1. Wähle $\beta = 0,5\,, \beta = 0,6$ oder β als Intervallmittelpunkt $\beta = 0,55$. Dies ergibt

$$|g'(z)| \le 0,071, \quad |g'(z)| \le 0,032, \quad |g'(z)| \le 0,036.$$

2. Wähle β so, dass $g'(0,5) = -g'(0,6)$. Dies ergibt

$$\beta \approx 0,5777, \quad |g'(z)| \le 0,019.$$

Lösung A.5.3: Für $x_t \le \sqrt{a}$ folgt $x_{t+1} \ge x_t$, und für $x_t \ge \sqrt{a}$ folgt $x_{t+1} \le x_1$. Also ergibt sich monotone Konvergenz $x_t \to z \ge 0$ und damit die Beziehung

$$z = \frac{z^3 + 3az}{3z^2 + a} \quad \Rightarrow \quad z = \sqrt{a}.$$

Für die Iterierten gilt

$$x_{t+1} - \sqrt{a} = \frac{x_t^3 + 3ax_t}{3x_t^2 + a} - \sqrt{a} = \frac{(x_t - \sqrt{a})^3}{3x_t^2 + a}$$

und folglich

$$\frac{x_{t+1} - \sqrt{a}}{(x_t - \sqrt{a})^3} = \frac{1}{3x_t^2 + a} \;\to\; \frac{1}{4a} \quad (t \to \infty).$$

Die Konvergenz dieser Fixpunktiteration ist also kubisch.

Lösung A.5.4: Wegen der angenommenen Stetigkeit von f' gilt

$$x_t \to z \quad \Rightarrow \quad f'(x_t) \to f'(z) \quad (t \to \infty).$$

Dies impliziert die Beziehung

$$\frac{x_{t+1} - z}{x_t - z} = \frac{x_t - f(x_t)'^{-1} f(x_t) - z}{x_t - z} = 1 - \underbrace{f(x_t)'^{-1}}_{\to f(z)'^{-1}} \underbrace{\frac{f(x_t) - f(z)}{x_t - z}}_{\to f'(z)} \;\to\; 0 \quad (t \to \infty).$$

Das Newton-Verfahren konvergiert also super-linear.

Lösung A.5.5 (Praktische Aufgabe): Mit Hilfe der Rekursionsformeln

$$p_{k+1}^{\mathrm{L}}(x) = x p_k^{\mathrm{L}}(x) - \frac{k^2}{4k^2 - 1} p_{k-1}^{\mathrm{L}}(x), \qquad p_{k+1}^{\mathrm{T}}(x) = 2x p_k^{\mathrm{T}}(x) - p_{k-1}^{\mathrm{T}}(x)$$

bestimmt man ausgehend von $p_0^{\mathrm{L}}(x) \equiv p_0^{\mathrm{T}}(x) \equiv 1$ und $p_1^{\mathrm{L}}(x) = p_1^{\mathrm{T}}(x) = x$ die Legendre- und Tschebyscheff-Polynome mit der Normierung $L_k(1) = T_k(1) = 1$ zu

$$L_4(x) = \frac{1}{8}\big(35x^4 - 30x^2 + 3\big), \qquad L_5(x) = \frac{1}{8}\big(63x^5 - 70x^3 + 15x\big),$$

und

$$T_4(x) = 8x^4 - 8x^2 + 1, \qquad T_5(x) = 16x^5 - 20x^3 + 5x.$$

MATLAB-Programm:

```
clear
format long
x0 = Nullstellen('L4', 'DL4', 1);
disp(' ')
disp('Die Nullstellen des Legendre-Polynoms vom Grad 4 sind')
x0
x0 = Nullstellen('T4', 'DT4', 1);
disp(' ')
disp('Die Nullstellen des Tschebyscheff-Polynoms vom Grad 4 sind')
x0
x0 = Nullstellen('L5', 'DL5', 1);
disp(' ')
disp('Die Nullstellen des Legendre-Polynoms vom Grad 5 sind')
x0
x0 = Nullstellen('T5', 'DT5', 1);
disp(' ')
disp('Die Nullstellen des Tschebyscheff-Polynoms vom Grad 5 sind')
x0

function erg = newton(fun, dfun, start, varargin)
tol = 1.e-8;
x = start;
% if (nargin == 4)
%     y = feval(fun, x, varargin{1});
%     dy = feval(dfun, x, varargin{1});
% else
%     y = feval(fun, x);
%     dy = feval(dfun, x);
% end
if (nargin == 4)
    y = ReduzierteFunktion(fun, x, varargin{1});
    dy = ReduzierteAbleitung(dfun, fun, x, varargin{1});
else
    y = feval(fun, x);
    dy = feval(dfun, x);
end
if (abs(dy*x) > 1.e-5)
    err = abs(y/(dy*x));
else
    err = abs(y);
end
n = 1;
disp(sprintf('n = 1:  x = %0.5f, \t y = %0.5g,  \t\t error = %0.5g',
x, y, err))
```

```
while ((err > tol) & (n<100))
    n = n+1;
    x = x - y/dy;
    if (nargin == 4)
        y = ReduzierteFunktion(fun, x, varargin{1});
        dy = ReduzierteAbleitung(dfun, fun, x, varargin{1});
    else
        y = feval(fun, x);
        dy = feval(dfun, x);
    end
    if (abs(dy*x) > 1.e-5)
        err = abs(y/(dy*x));
    else
        err = abs(y);
    end
    disp(sprintf('n = %d:  x = %0.5f, \t y = %0.5e, \t error = %0.5g',
    n, x, y, err));
end
erg = x;
return

function erg = Nullstellen(fun, dfun, x0)
% Suche alle positiven Nullstellen der Funktion fun, nehme an, alle
% Nullstellen laegen links von x0.
erg(1) = newton(fun, dfun, x0);
n = 1;
while (min(erg) > 1.e-6 & n<200)
    n = n + 1;
    disp(' ')
    erg(n) = newton(fun, dfun, erg(n-1)*1.001, erg);
end
% erg = erg(1:length(erg)-1);
if (min(erg) > -1.e-8)
    erg = [erg, -erg(length(erg)-1:-1:1)];
else
    erg = erg(1:length(erg)-1);
    erg = [erg, -erg(length(erg):-1:1)];
end
return
%
function erg = ReduzierteAbleitung(dfun, fun, x, varargin)
erg = feval(dfun, x);
if (nargin == 4)
    f = ReduzierteFunktion(fun, x, varargin{1});
    lf = LinearFaktoren(x, varargin{1});
```

```
    dlf = DLinearFaktoren(x, varargin{1});
    erg = erg - f * dlf;
    if (lf == 0)
        error('Muell!!')
    else
        erg = erg / lf;
    end
end
return

function erg = ReduzierteFunktion(fun, x, varargin)
erg = feval(fun, x);
if (nargin == 3)
    lf = LinearFaktoren(x, varargin{1});
    if (lf == 0)
        error('Muell!!')
    else
        erg = erg / lf;
    end
end
return

function erg = T4(x)
erg = 8*x.^4 - 8*x.^2 + 1;
return

function erg = T5(x)
erg = 16*x.^5 - 20*x.^3 + 5*x;
return

function erg = DL4(x)
erg = .5*(35*x.^3 - 15*x);
return

function erg = DL5(x)
erg = .125*(315*x.^4 - 210*x.^2 + 15);
return

function erg = DLinearFaktoren(x, a)
erg = 0;
for i=1:length(a)
    erg = erg + LinearFaktoren(x, [a(1:i-1), a(i+1:length(a))]);
end
return
```

```
function erg = DT4(x)
erg = 32*x.^3 - 16*x;
return

function erg = DT5(x)
erg = 80*x.^4 - 60*x.^2 + 5;
return

function erg = L4(x)
erg = .125*(35*x.^4 - 30*x.^2 + 3);
return

function erg = L5(x)
erg = .125*(63*x.^5 - 70*x.^3 + 15*x);
return

function erg = LinearFaktoren(x, a)
erg = 1;
for i=1:length(a)
    erg = erg * (x-a(i));
end
return
%-----------------------------------------------
```

Ergebnisse:

Konvergenzhistorie fuer das 4-te Legendre-Polynom:

```
n = 1:  x = 1.00000,   y = 1,             error = 0.1
n = 2:  x = 0.90000,   y = 2.07938e-01,   error = 0.038459
n = 3:  x = 0.86539,   y = 2.03342e-02,   error = 0.0048437
n = 4:  x = 0.86120,   y = 2.78402e-04,   error = 6.8512e-05
n = 5:  x = 0.86114,   y = 5.47175e-08,   error = 1.3472e-08
n = 6:  x = 0.86114,   y = 2.22045e-15,   error = 5.4668e-16

n = 1:  x = 0.86200,   y = 4.7302,        error = 0.34859
n = 2:  x = 0.56151,   y = 1.24300e+00,   error = 0.28151
n = 3:  x = 0.40344,   y = 2.61006e-01,   error = 0.13852
n = 4:  x = 0.34756,   y = 2.75394e-02,   error = 0.021424
n = 5:  x = 0.34011,   y = 4.59984e-04,   error = 0.00037828
n = 6:  x = 0.33998,   y = 1.36244e-07,   error = 1.1215e-07
n = 7:  x = 0.33998,   y = 1.20363e-14,   error = 9.9081e-15

n = 1:  x = 0.34032,   y = 3.5759,         error = 1.2763
n = 2:  x = -0.09404,   y = 8.25411e-01,   error = 1.9805
```

```
n = 3:  x = -0.28027,   y = 1.51741e-01,   error = 0.19319
n = 4:  x = -0.33442,   y = 1.28259e-02,   error = 0.016469
n = 5:  x = -0.33992,   y = 1.32710e-04,   error = 0.00017119
n = 6:  x = -0.33998,   y = 1.48150e-08,   error = 1.9112e-08
n = 7:  x = -0.33998,   y = 6.79689e-17,   error = 8.7682e-17

Die Nullstellen des 4-ten Legendre-Polynoms sind (bei Beachtung
der Symmetrie):

x_i =  0.86113631159405   0.33998104358486  -0.33998104358486
      -0.86113631159405
%

Konvergenzhistorie fuer das 5-te Legendre-Polynom:

n = 1:  x = 1.00000,   y = 1,             error = 0.066667
n = 2:  x = 0.93333,   y = 2.13360e-01,   error = 0.025722
n = 3:  x = 0.90933,   y = 2.19645e-02,   error = 0.0034064
n = 4:  x = 0.90623,   y = 3.37416e-04,   error = 5.4167e-05
n = 5:  x = 0.90618,   y = 8.38967e-08,   error = 1.3476e-08
n = 6:  x = 0.90618,   y = 4.66294e-15,   error = 7.4897e-16

n = 1:  x = 0.90709,   y = 6.9019,        error = 0.21793
n = 2:  x = 0.70940,   y = 1.92514e+00,   error = 0.1624
n = 3:  x = 0.59420,   y = 4.43154e-01,   error = 0.079471
n = 4:  x = 0.54698,   y = 5.77915e-02,   error = 0.015109
n = 5:  x = 0.53871,   y = 1.59562e-03,   error = 0.00044821
n = 6:  x = 0.53847,   y = 1.33836e-06,   error = 3.7674e-07
n = 7:  x = 0.53847,   y = 9.44432e-13,   error = 2.6586e-13

n = 1:  x = 0.53901,   y = 6.6096,        error = 0.53384
n = 2:  x = 0.25126,   y = 1.80867e+00,   error = 0.65136
n = 3:  x = 0.08760,   y = 4.29209e-01,   error = 0.81429
n = 4:  x = 0.01627,   y = 6.55586e-02,   error = 0.95514
n = 5:  x = 0.00073,   y = 2.81020e-03,   error = 0.99785
n = 6:  x = 0.00000,   y = 6.03845e-06,   error = 6.0384e-06
n = 7:  x = 0.00000,   y = 2.80934e-11,   error = 2.8093e-11

Die Nullstellen des Legendre-Polynoms vom Grad 5 sind (bei Beachtung
der Symmetrie):

x_i =  0.90617984593866   0.53846931010583   0.00000000000731
      -0.53846931010583   -0.90617984593866
%
```

```
Konvergenzhistorie fuer das 4-te Tschebyscheff-Polynom:

n = 1:  x = 1.00000,   y = 1,             error = 0.0625
n = 2:  x = 0.93750,   y = 1.48560e-01,   error = 0.01394
n = 3:  x = 0.92443,   y = 5.77259e-03,   error = 0.00059534
n = 4:  x = 0.92388,   y = 9.99629e-06,   error = 1.0351e-06
n = 5:  x = 0.92388,   y = 3.01545e-11,   error = 3.1226e-12

n = 1:  x = 0.92480,   y = 10.483,        error = 0.34323
n = 2:  x = 0.60738,   y = 2.72519e+00,   error = 0.26931
n = 3:  x = 0.44381,   y = 5.52756e-01,   error = 0.12312
n = 4:  x = 0.38917,   y = 5.25644e-02,   error = 0.01644
n = 5:  x = 0.38277,   y = 6.82750e-04,   error = 0.00022289
n = 6:  x = 0.38268,   y = 1.20653e-07,   error = 3.941e-08
n = 7:  x = 0.38268,   y = 4.10285e-15,   error = 1.3402e-15

n = 1:  x = 0.38307,   y = 8.0063,         error = 1.2605
n = 2:  x = -0.09978,   y = 1.86512e+00,   error = 2.1107
n = 3:  x = -0.31039,   y = 3.54836e-01,   error = 0.20837
n = 4:  x = -0.37506,   y = 3.34642e-02,   error = 0.020043
n = 5:  x = -0.38258,   y = 4.52100e-04,   error = 0.00027284
n = 6:  x = -0.38268,   y = 8.71635e-08,   error = 5.2608e-08
n = 7:  x = -0.38268,   y = 3.33067e-15,   error = 2.0102e-15

Die Nullstellen des 4-ten Tschebyscheff-Polynoms  sind (bei Beachtung
der Symmetrie)

x_i=  0.92387953251417   0.38268343236509  -0.38268343236509
     -0.92387953251417
%

Konvergenzhistorie fuer das 5-te Tschebyscheff-Polynom:

n = 1:  x = 1.00000,   y = 1,             error = 0.04
n = 2:  x = 0.96000,   y = 1.51243e-01,   error = 0.0089252
n = 3:  x = 0.95143,   y = 6.08389e-03,   error = 0.00039373
n = 4:  x = 0.95106,   y = 1.13201e-05,   error = 7.3562e-07
n = 5:  x = 0.95106,   y = 3.94360e-11,   error = 2.5627e-12

n = 1:  x = 0.95201,   y = 16.257,        error = 0.21131
n = 2:  x = 0.75084,   y = 4.46251e+00,   error = 0.15135
n = 3:  x = 0.63719,   y = 9.80032e-01,   error = 0.067486
n = 4:  x = 0.59419,   y = 1.11245e-01,   error = 0.010567
n = 5:  x = 0.58791,   y = 2.17452e-03,   error = 0.00021723
n = 6:  x = 0.58779,   y = 8.88658e-07,   error = 8.8866e-08
```

```
n = 7:  x = 0.58779,   y = 1.47919e-13,   error = 1.4792e-14

n = 1:  x = 0.58837,   y = 17.045,        error = 0.53122
n = 2:  x = 0.27582,   y = 4.67577e+00,   error = 0.64759
n = 3:  x = 0.09720,   y = 1.11671e+00,   error = 0.80996
n = 4:  x = 0.01847,   y = 1.73721e-01,   error = 0.95282
n = 5:  x = 0.00087,   y = 7.81455e-03,   error = 0.99761
n = 6:  x = 0.00000,   y = 1.86364e-05,   error = 0.99999
n = 7:  x = 0.00000,   y = 1.06891e-10,   error = 1.0689e-10

Die Nullstellen des 5-ten Tschebyscheff-Polynoms sind (bei Beachtung
der Symmetrie)

x_i =  0.95105651629759   0.58778525229248   0.00000000001195
      -0.58778525229248  -0.95105651629759
```

Lösung A.5.6: a) Wir haben

$$X_t = g(X_{t-1}), \qquad g(X) := X(I - AC) + C$$

und folglich

$$\|g(X) - g(Y)\| = \|(X - Y)(I - AC)\| \le \|X - Y\| \, \|I - AC\|.$$

Also ist g eine Kontraktion unter der Bedingung $\|I - AC\| =: q < 1$. Die zugehörige Fixpunktiteration konvergiert dann für jeden Startwert X_0 Ihr Limes genügt der Beziehung $Z = Z(I - AC) + C$ bzw. $ZAC = C$, was äquivalent mit $ZA = I$ bzw. $Z = A^{-1}$ ist. Die Fixpunktiteration konvergiert also für jeden Startwert $X_0 \in \mathbb{R}^{n\times n}$ gegen die Inverse von A mit der Fehlerabschätzung

$$\|X_t - A^{-1}\| \le q^t \|X_0 - A^{-1}\|, \quad t \in \mathbb{N}.$$

b) Wir haben

$$X_t = g(X_{t-1}), \qquad g(X) := X(2I - AX)$$

Ein Fixpunkt Z von g erfüllt $Z = Z(2I - AZ)$ bzw. $Z = ZAZ$. Wenn Z regulär ist, folgt $Z = A^{-1}$. Dies ist eine wesentliche Annahme, denn $Z = 0$ ist stets Fixpunkt von g. Zum Nachweis der Konvergenz setzen wir $Z = A^{-1}$ und finden

$$\begin{aligned} X_t - Z &= 2X_{t-1} - X_{t-1}AX_{t-1} - Z \\ &= -X_{t-1}AX_{t-.1} + \underbrace{ZA}_{=I} Y_{t-1} + X_{t-1}\underbrace{AZ}_{=I} - \underbrace{ZA}_{=I} Z \\ &= -(X_{t-1} - Z)A(X_{t-1} - Z). \end{aligned}$$

Dies impliziert

$$\|X_t - Z\| \le \|A\| \, \|X_{t-1} - Z\|^2.$$

Unter der Bedingung

$$\|X_0 - Z\| < \frac{1}{\|A\|}$$

liegt also quadratische Konvergenz vor:

$$\|A\|\,\|X_t - Z\| \le \big(\|A\|\,\|X_{t-1} - Z\|\big)^2 \le \ \ldots\ \le \big(\|A\|\,\|X_0 - Z\|\big)^{2^t} \to 0 \quad (t \to \infty).$$

Diese Iteration ist gerade das Newton-Verfahren zur Inversenbestimmung.

Bemerkung: Aus dem Kriterium in (a) ergibt sich für den Startwert $X_0 = C$:

$$1 > \|I - AX_0\| = \|A\|\,\|A^{-1} - X_0\| \quad \text{bzw.} \quad \|A^{-1} - X_0\| < \frac{1}{\|A\|},$$

d. h. auch das Kriterium in (b) ist erfüllt.

Lösung A.5.7: Die Funktion $f : \mathbb{R}^2 \to \mathbb{R}^2$ sei definiert durch

$$f_1(x_1, x_2) := x_1^2 + x_2^2 - 2, \quad f_2(x_1, x_2) := x_1^2 - x_2^2 - 1.$$

Ihre Nullstellen sind $(z_1, z_2)^T = \big(\pm\sqrt{3/2}, \pm\sqrt{1/2}\big)^T$.

a) Zum Aufstellen der Newton-Iteration berechnen wir

$$f'(x_1, x_2) = \begin{bmatrix} 2x_1 & 2x_2 \\ 2x_1 & -2x_2 \end{bmatrix}, \qquad f'(x_2, x_2)^{-1} = \frac{-1}{8x_1x_2} \begin{bmatrix} -2x_2 & -2x_2 \\ -2x_1 & 2x_1 \end{bmatrix}$$

und erhalten die folgenden Iterierten:

$$\begin{bmatrix} x_1^{(1)} \\ x_2^{(1)} \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} + \frac{1}{8} \begin{bmatrix} -2 & -2 \\ -2 & 2 \end{bmatrix} \begin{bmatrix} 0 \\ -1 \end{bmatrix} = \begin{bmatrix} 1.25 \\ 0.75 \end{bmatrix}$$

$$\begin{bmatrix} x_1^{(2)} \\ x_2^{(2)} \end{bmatrix} = \begin{bmatrix} 1.25 \\ 0.75 \end{bmatrix} + \frac{1}{7.5} \begin{bmatrix} -1.5 & -1.5 \\ -2.5 & 2.5 \end{bmatrix} \begin{bmatrix} 0.125 \\ 0 \end{bmatrix} = \begin{bmatrix} 1.225 \\ 0.7083 \end{bmatrix}.$$

Es ist $\|x^{(2)} - x^{(1)}\|_2 < 2 \cdot 10^{-3}$.

b) Die Iteration lautet

$$x^{(t+1)} = x^{(t)} - Cf(x^{(t)}) =: g(x^{(t)}).$$

Die zugehörige Jacobi-Matrix ist

$$g'(x_1, x_2) = I - \begin{bmatrix} c & c \\ c & -c \end{bmatrix} \begin{bmatrix} 2x_1 & 2x_2 \\ 2x_1 & -2x_2 \end{bmatrix} = \begin{bmatrix} 1 - 4cx_1 & 0 \\ 0 & 1 - 4cx_2 \end{bmatrix},$$

und ihre Norm

$$\|g'(x)\|_\infty = \max\{|1 - 4cx_1|, |1 - 4cx_2|\}.$$

Sei K die $\|\cdot\|_\infty$-Kugel um die Nullstelle $z = \left(\sqrt{3/2}, \sqrt{1/2}\right)^T$ mit $x^{(0)} \in \partial K$. Für $x \in K$ gilt

$$1 \le x_1 \le 1.5, \quad 0.4 \le x_2 \le 1.$$

Wir wählen $c = 1/4$ und erhalten

$$\max_{x\in K} \|g'(x)\|_\infty \le \max\{0.5, 0.6\} = 0.6 =: q.$$

Damit folgt dann

$$\|x^{(t)} - z||_\infty \le \frac{q^t}{1-q}\|x^{(0)} - z\|_\infty = \frac{(0.6)^t}{0.4} \cdot 0.25 < 2 \cdot 10^{-3}$$

für $t \ge 12.$.

Lösung A.5.8: a) Es ist eine Nullstelle der folgenden Funktion zu bestimmen:

$$f(x, \lambda) = \begin{bmatrix} Ax - \lambda x \\ \|x||_2^2 - 1 \end{bmatrix}.$$

Die zugehörige $(n+1) \times (n+1)$-Jacobi-Matrix ist:

$$f'(x, \lambda) = \begin{bmatrix} A - \lambda I & -x \\ 2x^T & 0 \end{bmatrix}$$

Die Newton-Iteration lautet dann

$$\begin{bmatrix} A - \lambda^{(t)} I & -x^{(t)} \\ 2x^{(t)T} & 0 \end{bmatrix} \begin{bmatrix} \delta x^{(t)} \\ \delta\lambda^{(t)} \end{bmatrix} = - \begin{bmatrix} Ax - \lambda x \\ \|x||_2^2 - 1 \end{bmatrix}, \quad \begin{bmatrix} x^{(t+1)} \\ \lambda^{(t+1)} \end{bmatrix} = \begin{bmatrix} x^{(t)} \\ \lambda^{(t)} \end{bmatrix} + \begin{bmatrix} \delta x^{(t)} \\ \delta\lambda^{(t)} \end{bmatrix}$$

b) Startwert $(x^{(0)}, \lambda^{(0)})^T = (0, 1.5, 3.5)^T$. Dann ist

$$f(x^{(0)}, \lambda^{(0)}) = \begin{bmatrix} 0 \\ 0.75 \\ 1.25 \end{bmatrix}, \qquad f'(x^{(0)}, \lambda^{(0)}) = \begin{bmatrix} 0.5 & 0 & 0 \\ -1 & 0.5 & -1.5 \\ 0 & 3 & 0 \end{bmatrix}.$$

Der erste Newton-Iterationsschritt lautet also in der Schreibweise der Gauß-Elimination:

$$\left[\begin{array}{ccc|c} 0.5 & 0 & 0 & 0 \\ -1 & 0.5 & -1.5 & -0.75 \\ 0 & 3 & 0 & -1.25 \end{array}\right] \to \left[\begin{array}{ccc|c} 0.5 & 0 & 0 & 0 \\ 0 & 0.5 & -1.5 & -0.75 \\ 0 & 0 & 9 & 3.25 \end{array}\right]$$

mit der Lösung $\delta x_1^{(0)} = 0\,,\ \delta x_2^{(0)} = -0.4176\,,\ \delta\lambda^{(0)} = 0.3611\,,$ bzw.

$$x_1^{(1)} = x_1^{(0)} + \delta x_1^{(0)} = 0, \quad x_2^{(1)} = x_2^{(0)} + \delta x_2^{(0)} = 1.0833, \quad \lambda^{(1)} = \lambda^{(0)} + \delta\lambda^{(0)} = 3.8611.$$

Für den zweiten Schritt notieren wir

$$f(x^{(0)},\lambda^{(0)}) = \begin{bmatrix} 0 \\ 0.1505 \\ 0.1736 \end{bmatrix}, \qquad f'(x^{(0)},\lambda^{(0)}) = \begin{bmatrix} 0.1389 & 0 & 0 \\ -1 & 0.1389 & -1.0833 \\ 0 & 2.1667 & 0 \end{bmatrix}.$$

Der zweite Newton-Iterationsschritt lauten dann wieder in der Schreibweise des Gauß-Elimination:

$$\left[\begin{array}{ccc|c} 0.1389 & 0 & 0 & 0 \\ -1 & 0.1389 & -1.0883 & -0.1505 \\ 0 & 2.1667 & 0 & -0.1736 \end{array}\right] \to \left[\begin{array}{ccc|c} 0.1389 & 0 & 0 & 0 \\ 0 & 0.1389 & -1.0833 & -0.1505 \\ 0 & 0 & 16.9 & 2.1736 \end{array}\right]$$

mit der Lösung $\delta x_1^{(1)} = 0$, $\delta x_2^{(1)} = 0.0801$, $\delta\lambda^{(1)} = 0.1286$, bzw.

$$x_1^{(2)} = x_1^{(1)} + \delta x_1^{(1)} = 0, \quad x_2^{(2)} = x_2^{(1)} + \delta x_2^{(1)} = 1.0032, \quad \lambda^{(2)} = \lambda^{(1)} + \delta\lambda^{(1)} = 3.9897.$$

Die Matrix hat den zweifachen Eigenwert $\lambda = 4$. Ein zugehöriger normierter Eigenvektor ist $x = (0,1)^T$. Die Fehler der Iteration sind also

$$\begin{aligned} &|\lambda^{(0)} - \lambda| \approx 0.5, \quad |\lambda^{(1)} - \lambda| \approx 0.1389, \quad |\lambda^{(2)} - \lambda| \approx 0.0103 \\ &|x_1^{(0)} - x_1| \approx 0, \quad |x_1^{(1)} - x_1| \approx 0, \quad |x_1^{(0)} - x_1| \approx 0 \\ &|x_2^{(0)} - x_1| \approx 0.5, \quad |x_2^{(1)} - x_1| \approx 0.0833, \quad |x_2^{(2)} - x_1| \approx 0.0032. \end{aligned}$$

Das Newton-Verfahren konvergiert offenbar mindestens quadratisch, obwohl die Jacobi-Matrix in der Nullstelle singulär ist:

$$f'(x,\lambda) = \begin{bmatrix} 0 & 0 & 0 \\ -1 & 0 & -1 \\ 0 & 2 & 0 \end{bmatrix}.$$

Lösung A.5.9: (i) Mit der maximalen Spaltensummen gilt

$$\|B\|_1 = \max_{j=1,2,3} \sum_{i=1}^{3} |a_{ij}| = 0.9 < 1.$$

Also ist $\mathrm{spr}(B) \le \|B\|_1 < 1$, was Konvergenz der Fixpunktiteration impliziert. (Man beachte, dass $\|B\|_\infty = 1.4 > 1$.)

(ii) Seien λ_i die Eigenwerte von B. Es ist

$$\prod_{i=1}^{3} \lambda_i = \det(B) = -1.$$

Hieraus folgt, dass für mindestens einen der Eigenwerte $|\lambda| \geq 1$ sein muss. I. Allg. liegt also keine Konvergenz vor, denn ist $x^0 - x$ gerade Eigenvektor zu diesem Eigenwert, so gilt

$$\|x^t - x\| = \|B^t(x^0 - x)|| = \|\lambda^t(x^0 - x)|| = |\lambda|^t\|x^0 - x\| \not\to 0 \quad (t \to \infty).$$

Lösung A.5.10 (Praktische Aufgabe): Die betrachteten Iterationen sind:

$$\begin{aligned} &a) \quad X_t = X_{t-1}(I - AC) + C, \quad t = 1, 2, \ldots, \quad C \in \mathbb{R}^{n\times n} \text{ regulär}, \\ &b) \quad X_t = X_{t-1}(2I - AX_{t-1}), \quad t = 1, 2, \ldots, \end{aligned}$$

Nach Lösungh A.5.6 konvergiert die Iteration (a) mit einer (regulären) Matrix C, für die $\|I - AC\| < 1$ ist, für jeden Startwert X_0, während bei Iteration (b) Konvergenz nur für Startwerte mit

$$\|X_0 - A^{-1}\|_\infty < \frac{1}{\|A\|_\infty} = \frac{1}{4}$$

garantiert ist. Für die Testrechnungen sollen die folgenden Startwerte verwendet werden:

$$a) \quad C = \tfrac{1}{8}I,\ X_0 = C, \qquad b) \quad X_0 = \tfrac{1}{8}I.$$

MATLAB-Programm:

```
clear
for k=2:6
    n = 2^k;
    A = GetA(k);
    X0 = .125 * eye(n);
    disp(' ')
    disp(['k = ', int2str(k)])
    disp('Erste Iterationsvorschrift (a) mit C = ID / 8:')
    C = X0;
    tic
    data2 = Iteration(A, X0, C);
    t = toc;
    disp(['Benoetigte Zeit: ', num2str(t), ',  # Iterationen: ', int2str(length(data2

    disp(' ')
    disp('Zweite Iterationsvorschrift (b):')
    tic;
    data1 = Iteration(A, X0);
    t = toc;
    disp(['Benoetigte Zeit: ', num2str(t), ',  # Iterationen: ', int2str(length(data1
end

function matrix = GetA(k)
n = 2^k;
```

```
matrix = 2*eye(n);
matrix(2:n,1:n-1) = matrix(2:n,1:n-1) - eye(n-1);
matrix(1:n-1,2:n) = matrix(1:n-1,2:n) - eye(n-1);
return

function data = Iteration(A, X0, varargin)
X = X0;
n = length(X);
fehler = norm(A*X - eye(n), inf);
data = fehler;
k = 0;
while (fehler > 1.e-8  &  k < 1000000)
    k = k+1;
    if (nargin == 2)
        X = IterationsSchritt(A, X);
    elseif (nargin == 3)
        X = IterationsSchritt(A, X, varargin{1});
    end
    fehler = norm(A*X - eye(n), inf);
    data = [data, fehler];
end
return

function X = IterationsSchritt(A, X0, varargin)
X = X0;
n = length(X);
if (nargin == 3)
    % Fhre erste (a) Iterationsvorschrift aus:
    C = varargin{1};
    X = X*(eye(n) - A*C) + C;
elseif (nargin == 2)
    % Fhre zweite (b) Iterationsvorschrift aus:
    X = X*(2*eye(n) - A*X);
end
return
```

Ergebnisse:

```
k = 2
Erste Iterationsvorschrift (a) mit C = ID/8:
Benoetigte Zeit: 0.11103,  # Iterationen: 380

Zweite Iterationsvorschrift (b):
Benoetigte Zeit: 0.025549,  # Iterationen: 10
```

```
k = 3
Erste Iterationsvorschrift (a) mit C = ID/8:
Benoetigte Zeit: 0.35854,  # Iterationen: 1227

Zweite Iterationsvorschrift (b):
Benoetigte Zeit: 0.00235,  # Iterationen: 12

k = 4
Erste Iterationsvorschrift (a) mit C = ID/8:
Benoetigte Zeit: 1.3563,  # Iterationen: 4374

Zweite Iterationsvorschrift (b):
Benoetigte Zeit: 0.003194,  # Iterationen: 14

k = 5
Erste Iterationsvorschrift (a) mit C = ID/8:
Benoetigte Zeit: 8.5067,  # Iterationen: 16475

Zweite Iterationsvorschrift (b):
Benoetigte Zeit: 0.033671,  # Iterationen: 16

k = 6
Erste Iterationsvorschrift (a) mit C = ID/8:
Benoetigte Zeit: 146.1633,  # Iterationen: 63914

Zweite Iterationsvorschrift (b):
Benoetigte Zeit: 0.029058,  # Iterationen: 17
```

A.6 Kapitel 6

Lösung A.6.1: a) Die Jacobi- und Gauß-Seidel-Matrizen von A_1 sind

$$J = -D^{-1}(L+R) = \begin{bmatrix} 0 & 0.5 & -1 \\ -0.5 & 0 & 1 \\ -1 & -1 & 0 \end{bmatrix}, \quad H_1 = -(D+L)^{-1}R = \begin{bmatrix} 0 & 0.5 & -1 \\ 0 & -0.25 & 1.5 \\ 0 & -0.25 & -0.5 \end{bmatrix}$$

Mit den Eigenwerten λ_i von J gilt $\lambda_1\lambda_2\lambda_3 = \det(J) = -1$ und folglich $\mathrm{spr}(J) \geq 1$. Also ist das Jacobi-Verfahren i. Allg. nicht konvergent. Die Matrix H_1 hat das charakteristische Polynom $\chi(\lambda) = -\lambda(\lambda^2 + \frac{3}{4}\lambda + \frac{1}{2})$ und folglich die Eigenwerte $\lambda_1 = 0,\ \lambda_{2/3} = \pm 1/\sqrt{2}$. Also ist $\mathrm{spr}(H_1) < 1$ und folglich das Gauß-Seidel-Verfahren konvergent.

b) Die Matrix A_2 erfüllt das schwache Zeilenkriterioum und ist irreduzibel. Also sind sowohl Jacobi- als auch Gauß-Seidel-Verfahren konvergent.

Lösung A.6.2: Für die allgemeine Fixpunktiteration $x^{t+1} = Bx^t g + c$ gilt im Falle der Konvergenz gegen ein z die asymptotische Fehlerabschätzung

$$\|x^t - z\| \leq \mathrm{spr}(B)^t \|x^0 - z\|.$$

Hieraus ergibt sich die Anzahl der zur Reduzierung des Anfangsfehlers um den Faktor ε erforderlichen Iterationsschritte zu

$$t \approx \frac{\log_{10}(\varepsilon)}{\log_{10}(\mathrm{spr}(B))} + 1.$$

Für Jacobi- und Gauß-Seidel-Matrix gilt

$$J = \begin{bmatrix} 0 & 1/3 \\ 1/3 & 0 \end{bmatrix}, \qquad H_1 = \begin{bmatrix} 0 & -1/3 \\ 0 & 1/9 \end{bmatrix}$$

und somit $\mathrm{spr}(J) = 1/3$ und $\mathrm{spr}(H_1) = 1/9$. Die notwendige Iterationszahlen für das Jacobi- und das Gauß-Seidel-Verfahren ist also (wie von der Theorie vorausgesagt):

$$t_J \approx \frac{6}{\log_{10}(3)} + 1 \approx 13, \qquad t_{H_1} \approx \frac{6}{\log_{10}(9)} + 1 \approx 7 \approx \tfrac{1}{2} t_J.$$

Lösung A.6.3: Zunächst wird die Iterationsmatrix bestimmt. Mit

$$\begin{bmatrix} 1 & 0 \\ -\omega a & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & 0 \\ \omega a & 1 \end{bmatrix}$$

lautet die Iteration

$$x^t = \begin{bmatrix} 1 & 0 \\ \omega a & 1 \end{bmatrix} \begin{bmatrix} 1-\omega & \omega a \\ 0 & 1-\omega \end{bmatrix} x^{t-1} + \omega \begin{bmatrix} 1 & 0 \\ \omega a & 1 \end{bmatrix} b.$$

und folglich

$$B_\omega = \begin{bmatrix} 1-\omega & \omega a \\ \omega a(1-\omega) & \omega^2 a^2 + 1 - \omega \end{bmatrix}.$$

Also ist $\det(B_\omega - \lambda I) = -\lambda\omega^2 a^2 + (1 - \omega - \lambda)^2$.

a) Für $\omega = 1$ gilt

$$\det(B_1 - \lambda I) = -\lambda a^2 + \lambda^2 \quad \Rightarrow \quad \mathrm{spr}(B_1) = a^2,$$

d. h.: Konvergenz liegt vor für $|a| < 1$.

b) Im Fall $a = \frac{1}{2}$ ist

$$\lambda_{1,2} = 1 - \omega + \tfrac{1}{8}\omega^2 \pm \tfrac{1}{2}\omega\sqrt{1 - \omega + \tfrac{1}{16}\omega^2}.$$

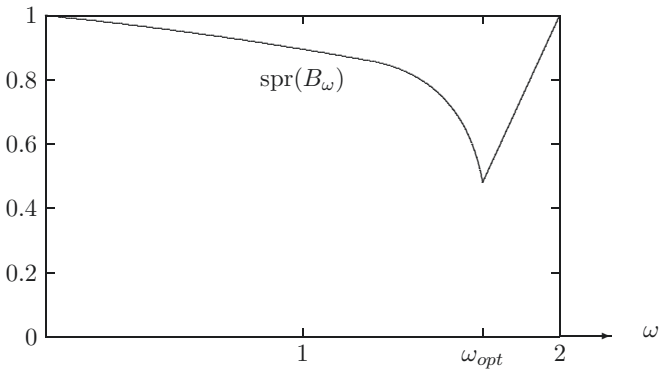
Die beiden Nullstellen sind reell für $1-\omega+\frac{1}{16}\omega^2 \geq 0$, bzw. $\omega \leq 8-4\sqrt{3} = 1.07179677\ldots$, und sonst komplex. Also:

$$\mathrm{spr}(B_\omega) = \begin{cases} 1-\omega+\frac{1}{8}\omega^2+\frac{1}{2}\omega\sqrt{1-\omega+\frac{1}{16}\omega^2}, & 0 \leq \omega \leq 8-4\sqrt{3}, \\ \omega-1, & 8-4\sqrt{3} < \omega \leq 2. \end{cases}$$

Auswertung dieser Formel ergibt

ω	0.8	0.9	1.0	1.1	1.2	1.3	1.4
$\mathrm{spr}(B_\omega)$	0.476	0.376	0.25	0.1	0.2	0.3	0.4

Der Graph der Funktion $\rho(\omega) := \mathrm{spr}(B_\omega)$, $0 \leq \omega <\leq$, beginnt mit $\rho(0) = 1$, weist im Minimum $\omega_{\mathrm{opt}} := 8-4\sqrt{3}$ eine scharfe Spitze nach unten auf und steigt dann linear bis $\rho(2) = 1$ an.



Schematische Darstellung der Abhängigkiet des Spektralradius $\mathrm{spr}(H_\omega)$ *von* $\omega \in [0,2]$

Lösung A.6.4: Wir rekapitulieen die beiden Definitionen von „irreduzibel“:

a) Eine Matrix $A \in \mathbb{R}^{n\times n}\, m$ heißt „irreduzibel“, wenn es keine Partitionierung J, K von $\{1,\ldots,n\} =: \mathbb{N}_n$ mit $J \cup K = \mathbb{N}_n$ und $J \cap K = \emptyset$ gibt, so dass $a_{jk} = 0$ ist für alle $j \in J$ und alle $k \in K$.

b) Eine Matrix $A \in \mathbb{R}^{n\times n}$ heißt „irreduzibel“, wenn es zu jedem Indexpaar $j, k \in \mathbb{N}_n$ eine Teilmenge $\{i_1,\ldots,i_m\} \in \mathbb{N}_n$ gibt, so dass $a_{j,i_1} \neq 0, a_{i_1,i_2} \neq 0, \ldots, a_{i_{m-1},i_m} \neq 0, a_{i_m,k} \neq 0$.

(i) (a) $\Rightarrow$ (b): Sei A irreduzibel im Sinne von (a). Es sei $i \in \mathbb{N}_n$ irgendein Index. Es bezeichne J die Menge aller Indizes $l \in \mathbb{N}_n$, so dass $a_{i,i_1},\ldots,a_{i_m,l}$ alle $\neq 0$ sind für geeignete $\{i_1,\ldots,i_m\} \in \mathbb{N}_n$. Wir setzen $K := \mathbb{N}_n \setminus J$ und haben zu zeigen, dass $J = \mathbb{N}_n$ bzw. $K = \emptyset$. Falls $a_{il} = 0, l \neq i$, so ist offensichtlich $J = \{i\}$, und wir haben eine Partition im Sinne von (a), so dass für alle $j \in J, k \in K$ gilt $a_{jk} = 0$. Also enthält J noch weitere Elemente. Zu $p \in J$, $p \neq i$, gibt es $\{i_1,\ldots,i_m\} \in \mathbb{N}_n$ mit $a_{i,i_1},\ldots,a_{i_m,p} \neq 0$.

Für beliebiges $q \in K$ müsste wegen $J \cap K = \emptyset$ dann $a_{pq} = 0$ sein. Andererseits wäre A reduzibel gemäß der negation von (a). Also muss K leer sein.

(i) (b) $\Rightarrow$ (a): Sei A irreduzibel im Sinne von (b). Wir nehmen an, $\{J, K\}$ wäre eine zugehörige Partition im Sinne von (b). Dann gibt es zu beliebigen Indixpaaren $\{j, k\} \in J \times K$ eine Teilmenge von Indizes $\{i_1, \ldots, i_m\} \in \mathbb{N}_n$, so dass $a_{j,i_1} \neq 0, \ldots, a_{i_m,k} \neq 0$. Wegen $a_{j,i_1} \neq 0$ ist dann $i_1 \in J$. Induktiv ergibt sich, dass alle $i_2, \ldots, i_m$ zu J gehören, und schließlich wegen $a_{i_m,k} \neq 0$ auch k im Widerspruch zur Annahme über J und K. A ist also auch im Sinne von (a) irreduzibel.

Lösung A.6.5 (Praktische Aufgabe): Das lineare Gleichungssystem $Ax = b$ mit der Modellmatrix aus der Vorlesung und der rechten Seite $b = h^2(1, \ldots, 1)^T$ wird mit dem Jacobi-, dem Gauß-Seidel- und dem „optimalen“ SOR-Verfahren gelöst. Der Startvektor ist $x^0 = 0$ und das Abbruchkriterium

$$\frac{\|b - Ax^t\|_\infty}{\|x^t\|_\infty} \leq TOL := 10^{-8}.$$

Mit der kanonischen additiven Aufspaltung $A = L + D + R$ haben die drei Iterationsverfahren die folgende Form:

- Jacobi-Verfahren:

$$x^{t+1} = Jx^t + c = -D^{-1}(L + R)x^t + D^{-1}b;$$

- Gauß-Seidel-Verfahren:

$$x^{t+1} = H_1x^t + c = -(D + L)^{-1}Rx^t + (D + L)^{-1}b;$$

- SOR-Verfahren mit $\omega = \omega_{\text{opt}} = 2\big(1 + \sqrt{1 - \text{spr}(J)^2}\big)^{-1}$ und $\text{spr}(J) = \cos(h\pi) < 1$:

$$x^{t+1} = H_\omega x^t + c = -(D + \omega L)^{-1}[(1 - \omega)D - \omega R]x^t + \omega(D + \omega L)^{-1}b.$$

MATLAB-Programm:

```
clear
tol = 1.e-8;
for k = 1:1
    disp(['k = ', int2str(k)])
    m = 2^k;
    A = GetA(m);
    b = GetB(m);
    start = zeros(m^2, 1);
    omega = 1;
    tic
    [dataj, xj] = loese(A, b, start, 'jacobi', tol);
    tj = toc;
```

```
    disp(['Jacobi:       ', int2str(length(dataj)), ' Schritte in ', ...
        num2str(tj), ' Sekunden, res = ', num2str(dataj(length(dataj)))])
    tic
    [datag, xg] = loese(A, b, start, 'gaussseidel', tol);
    tg = toc;
    disp(['Gauss-Seidel: ', int2str(length(datag)), ' Schritte in ', ...
        num2str(tg), ' Sekunden, res = ', num2str(datag(length(datag)))])
    omega = omegaopt(m);
    tic
    [datas, xs] = loese(A, b, start, 'sor', tol);
    ts = toc;
    disp(['SOR:          ', int2str(length(datas)), ' Schritte in ', ...
        num2str(ts), ' Sekunden, res = ', num2str(datas(length(datas)))])
    disp(' ')
end

function A = GetA(m)
n = m^2;
B = 4*speye(m);
B(2:m,1:m-1) = B(2:m,1:m-1) - speye(m-1);
B(1:m-1,2:m) = B(1:m-1,2:m) - speye(m-1);
for i = 1:m-1
   A((i-1)*m+1:i*m, (i-1)*m+1:i*m) = B;
   A((i-1)*m+1:i*m, i*m+1:(i+1)*m) = -speye(m);
   A(i*m+1:(i+1)*m, (i-1)*m+1:i*m) = -speye(m);
end;
A((m-1)*m+1:m^2, (m-1)*m+1:m^2) = B;
return

function b = GetB(m)
n = m^2;
b = ones(n,1) / (m+1)^2;
return

function [data, x] = loese(A, b, start, vorkonditionierer, tol)
x = start;
r = b - A*x;
res = norm(r, inf);
data = res;
m = round(sqrt(length(A)));
omega = omegaopt(m);
k = 0;
while (res > tol & k < 100000)
    h = feval(vorkonditionierer, A, r, omega)
    x = x + h;
```

```
    r = b - A*x;
    res = norm(r, inf);
    data = [data, res];
end
return

function h = jacobi(A, b, omega)
n = length(A);
D = speye(n);
for i = 1:n
    D(i,i) = A(i,i);
end
h = zeros(n,1);
h = b ./ diag(D);
return

function h = gaussseidel(A, b, omega)
n = length(A);
DL = sparse(n,n);
for i = 1:n
    DL(i:n,i) = A(i:n,i);
end
h = zeros(n,1);
for i = 1:n
    h(i) = (b(i) - DL(i,1:i-1)*h(1:i-1)) / DL(i,i);
end
return

function h = sor(A, b, omega)
n = length(A);
DL = speye(n);
for i = 1:n
    DL(i,i) = A(i,i)/omega;
    DL(i+1:n,i) = A(i+1:n,i);
e
h = zeros(n,1);
for i = 1:n
    h(i) = (b(i) - DL(i,1:i-1)*h(1:i-1)) / DL(i,i);
end
return

function erg = omegaopt(m)
rho = cos(pi/(m+1));
erg = 2 / (1 + sqrt(1-rho^2));
return
```

Konvergenzresultate mit Jacobi-, Gauß-Seidel- und optimalem SOR-Verfahren für die Modellmatrix wachsender Dimension n (MATLAB-Implementierung)

		Jacobi		Gauß-Seidel		SOR(ω_{opt})	
m	$n = m^2$	# Iter.	Sek.	# Iter.	Sek.	# Iter.	Sek.
3	9	87	0.01	46	0.03	19	0.02
7	49	276	0.06	140	0.28	34	0.08
15	225	924	1.58	464	5.47	59	0.72
31	961	3001	66.83	1602	171.35	106	12.7
63	3969	—	—	—	—	—	—

Konvergenzresultate mit Jacobi-, Gauß-Seidel- und optimalem SOR-Verfahren für die Modellmatrix wachsender Dimension n (C++ Implementierung)

		Jacobi		Gauß-Seidel		SOR(ω_{opt})	
m	$n = m^2$	# Iter.	Sek.	# Iter.	Sek.	# Iter.	Sek.
3	9	63	0	32	0	15	0
7	49	273	0	136	0	34	0
15	225	1110	0.02	555	0.02	69	0
31	961	4458	0.47	2229	0.32	139	0.03
63	3969	17849	6.89	8925	4.79	279	0.18

A.7 Kapitel 7

Lösung A.7.1: a) Sei $\{e^1, \ldots, e^n\}$ eine kartesische Basis des $\mathbb{K}^n$. Für Vektoren

$$x = \sum_{i=1}^{n} x_i e^i, \quad y = \sum_{j=1}^{n} y_j e^j$$

gilt dann

$$(x, y) = \sum_{i,j=1}^{n} x_i y_j (e^i, e^j) = (Ax, y)_2$$

mit der Matrix $A := \big((e^i, e^j)\big)_{i,j=1}^{n}$. Die Matrix A ist offensichtlich symmetrisch und auch positiv definit:

$$(Ax, x)_2 = \sum_{i,j=1}^{n} x_i x_j (e^i, e^j) = (x, x) > 0, \quad x \neq 0.$$

b) Sei $A \in \mathbb{C}^{n\times n}$ positiv definit, d. h.: $\bar{x}^T Ax \in \mathbb{R}_+,\ x \in \mathbb{C}^n$. Dann ist A notwendig hermitesch, da für beliebige $x, y \in \mathbb{C}$ gilt:

$$\begin{aligned}(\overline{x+y})^T A(x+y) \in \mathbb{R} \quad &\Longrightarrow \quad \bar{x}^T Ax + \bar{y}^T Ay + (\bar{x}^T Ay + \bar{y}^T Ax) \in \mathbb{R},\\ (\overline{x+iy})^T A(x+iy) \in \mathbb{R} \quad &\Longrightarrow \quad \bar{x}^T Ax + \bar{y}^T Ay + i\,(\bar{x}^T Ay - \bar{y}^T Ax) \in \mathbb{R}.\end{aligned}$$

Für $x = e_i$ und $y = e_j$ ergibt sich also $a_{ij} + \bar{a}_{ji} \in \mathbb{R}$ und $i(a_{ij} - \bar{a}_{ji}) \in \mathbb{R}$, d. h.:

$$\begin{aligned}&\mathrm{Re}(a_{ji} + a_{ij}) + i\mathrm{Im}(a_{ji} + a_{ij}) \in \mathbb{R},\\ &i\mathrm{Re}(a_{ji} - a_{ij}) + \mathrm{Im}(a_{ji} - a_{ij}) \in \mathbb{R}.\end{aligned}$$

Also ist $a_{ij} = \mathrm{Re}\, a_{ij} + i\mathrm{Im}\, a_{ij} = \mathrm{Re}\, a_{ji} - i\mathrm{Im}\, a_{ji} = \mathrm{Re}\, \bar{a}_{ji} + i\mathrm{Im}\, \bar{a}_{ji} = \bar{a}_{ji}$.
Bem.: Das obige Argument verwendet im Wesentlichen nur, dass $\bar{x}^T Ax \in \mathbb{R},\ x \in \mathbb{C}^n$ und nicht die eigentliche Definitheit von A.

Lösung A.7.2: Wir verwenden ein sog. „Deformationsargument“. Für $t \in [0,1]$ wird die folgende Matrixfunktion definiert:

$$A(t) = (1-t)\mathrm{diag}_{i=1,\ldots,n}(a_{ii}) + t\,A.$$

Offensichtlich ist $A(0)$ eine Diagonalmatrix mit Eigenwerten $\lambda_i(0) = a_{ii}$. Die Entwicklung des i-ten Eigenwerts $\lambda_i(t)$ ist eine stetige Funktion in t. Dies folgt aus der Tatsache, dass die Nullstelle eines Polnoms $p_\alpha(t)$ (in diesem Fall das charakteristische Polynom der Matrix $A(t)$ lokal beliebig oft differenzierbar nach den Koeffizienten des Polynoms ist - eine direkte Folgerung aus dem Satz über implizite Funktionen angewendet auf $p(\alpha, t) = p_\alpha(t)$ (unter besonderer Behandlung mehrfacher Nullstellen). Ferner haben die Gerschgorin-Kreise von $A(t),\ 0 \le t \le 1$, alle dieselben Mittelpunkte a_{ii}, nur die Radien wachsen monoton für wachsendes t. Daher impliziert der Einschließungsatz von Gerschgorin, dass das Bild der Funktion $t \to \lambda_i(t)$ vollständig in der Vereinigung aller Gerschgorin-Kreise von $A(1)$ liegt und folglich wegen seiner Stetigkeit auch vollständig in der zusammenhängenden Komponente, welche a_{ii} enthält.

Lösung A.7.3: (i) Seien λ_1 und λ_2 zwei Eigenwerte und zugehörigen Eigenvektorenh v^1 und v^2. Es gilt:

$$0 = (v^1, Av^2) - (v^1, Av^2) = (v^1, Av^2) - (Av^1, v^2) = (\lambda_2 - \lambda_1)(v^1, v^2).$$

Also muss im Falle $\lambda_1 \neq \lambda_2$ notwendig $(v^1, v^2) = 0$ sein.

Diese Aussage ist auch richtig für „normale“ (komplexe) Matrizen und ist ein Spezialfall des sog. „Spektralsatzes“ für „normale“Operatoren im Hilbert-Räumen.

(ii) Sei v ein Eigenvektor zu dem maximalen Eigenwert $\lambda_{\max}(A)$ von A. Es gilt dann:

$$\max_{x\in\mathbb{K}^n\setminus\{0\}} \frac{(Ax,x)_2}{\|x\|_2^2} \ge \frac{(Av,v)_2}{\|v\|_2^2} = \lambda_{\max}(A),$$

Umgekehrt existiert für beliebiges $x \in \mathbb{K}^n \setminus \{0\}$ eine Entwicklung $x = \sum_i x_i\, v^i$ mit einer ONB $\{v^i\}$ von Eigenvektoren, so dass:

$$\frac{(Ax,x)_2}{\|x\|_2^2} = \frac{(A\sum_i x_i v^i, \sum_i x_i v^i)_2}{\|x\|_2^2} = \frac{\sum_i \lambda_i x_i^2}{\sum_i x_i^2} \leq \lambda_{\max}(A).$$

Die entsprechende Aussage für $\lambda_{\min}(A)$ ergibt sich mit Hilfe eines analogen Arguments. Die Beziehung $\lambda_{\min}(A) \leq \lambda_{\max}(A)$ ist offensichtlich.

Lösung A.7.4: Es gilt $z^0 = \sum_{i=1}^n \alpha_i w^i$ und $z^t = \|A^t z^0\|_2^{-1} A^t z^0$ und folglich:

$$\begin{aligned}
\lambda^t = (Az^t, z^t)_2 &= \frac{(A^{t+1}z^0, A^t z^0)_2}{\|A^t z^0\|_2^2} = \frac{\sum_{i=1}^n |\alpha_i|^2\, \lambda_i^{2t+1}}{\sum_{i=1}^n |\alpha_i|^2\, \lambda_i^{2t}} \\
&= \frac{(\lambda_n)^{2t+1}\left\{|\alpha_n|^2 + \sum_{i=1}^{n-1} |\alpha_i|^2 \left(\frac{\lambda_i}{\lambda_n}\right)^{2t+1}\right\}}{(\lambda_n)^{2t}\left\{|\alpha_n|^2 + \sum_{i=1}^{n-1} |\alpha_i|^2 \left(\frac{\lambda_i}{\lambda_n}\right)^{2t}\right\}} \\
&= \lambda_n \frac{|\alpha_n|^2 + \sum_{i=1}^{n-1} |\alpha_i|^2 \left(\frac{\lambda_i}{\lambda_n}\right)^{2t} + \sum_{i=1}^{n-1} |\alpha_i|^2 \left(\frac{\lambda_i}{\lambda_n}\right)^{2t}\left(\frac{\lambda_i}{\lambda_n} - 1\right)}{|\alpha_n|^2 + \sum_{i=1}^{n-1} |\alpha_i|^2 \left(\frac{\lambda_i}{\lambda_n}\right)^{2t}} \\
&= \lambda_n + \lambda_n \frac{\sum_{i=1}^{n-1} |\alpha_i|^2 \left(\frac{\lambda_i}{\lambda_n}\right)^{2t}\left(\frac{\lambda_i}{\lambda_n} - 1\right)}{|\alpha_n|^2 + \sum_{i=1}^{n-1} |\alpha_i|^2 \left(\frac{\lambda_i}{\lambda_n}\right)^{2t}} =: \lambda_n + \lambda_n E_t.
\end{aligned}$$

Der Fehlerterm rechts kann wie folgt abgeschätz werden:

$$|E_t| \leq \left(\frac{\lambda_{n-1}}{\lambda_n}\right)^{2t} \frac{\sum_{i=1}^{n-1} |\alpha_i|^2}{|\alpha_n|^2} = \left(\frac{\lambda_{n-1}}{\lambda_n}\right)^{2t} \frac{\|z^0\|_2^2}{|\alpha_n|^2}.$$

Also ist:

$$|\lambda^t - \lambda_n| \leq |\lambda_n| \frac{\|z^0\|_2^2}{|\alpha_n|^2} \left(\frac{\lambda_{n-1}}{\lambda_n}\right)^{2t}.$$

Lösung A.7.5: Sei $A = \tilde{Q}\tilde{R}$ eine beliebige QR-Zerlegung von A. Wir definieren eine unitäre Matrix $H := \operatorname{diag}(h_i) \in \mathbb{C}^{n\times n}$ durch $h_i := \frac{\bar{r}_{ii}}{|r_{ii}|}$ und setzen $R = H\tilde{R}$, $Q = \tilde{Q}\bar{H}$.

Nun ist $\bar{A}^T A = \bar{R}^T \bar{Q}^T Q R = \bar{R}^T R$ die Cholesky-Zerlegung der reellen, symmetrischen und positiv definiten Matrix $\bar{A}^T A$. Da die Cholesky-Zerlegung (mit positiver Diagonale) infeutig bestimmt ist, folgt R is unique und damit auch $Q = AR^{-1}$.

Lösung A.7.6: Es genügt, die folgenden beiden Aussagen über die QR-Iteration zu zeigen; die Behauptung folgt dann durch Induktion:

1. Sei A eine Hessenberg-Matrix und $A = QR$ ihre QR-Zerlegung. Dann ist $\tilde{A} = RQ$ ebenfalls eine Hessenberg-Matrix.
2. Sei A eine symmetrische Matrix und $A = QR$ ihre QR-Zerlegung. Dann ist $\tilde{A} = RQ$ ebenfalls symmetrisch.

Die QR-Zerlegung einer Hessenberg-Matrix A kann geschrieben werden in der Form

$$G_{n-1}\, G_{n-2} \cdots G_1 A = R$$

mit

$$G_i = \begin{pmatrix} I_{i-1} & & 0 \\ & \tilde{G}_i & \\ 0 & & I_{n-i-1} \end{pmatrix},$$

und aeiner orthogonalen Komponente $\tilde{G}_i \in \mathbb{R}^{2\times 2}$, welche die linke untere Nebendiagonalelement des Blocks

$$\tilde{G}_i \begin{pmatrix} *_{i,i} & *_{i,i+1} \\ a_{i+1,i} & a_{i+1,i+1} \end{pmatrix} = \begin{pmatrix} * & * \\ 0 & * \end{pmatrix}.$$

eliminiert. Neben der Elimination des Elements $a_{i+1,i}$ agiert die orthogonale Matrix G_i nur auf dem rechten oberen Teil der Zwischenmatrix. Daher ist R eine obere Dreiecksmatrix und es gilt

$$\tilde{A} = RQ = RG_1^T G_2^T \cdots G_{n-1}^T.$$

Analog ergibt sich durch Induktion, dass Multiplikation mit G_i^T von rechts höchstens ein nichttriviales Element in der unteren Nebendiagonale an der Position $*_{i+1,i}$ erzeugt. Also ist $\tilde{A}$ in der Tat eine Hessenberg-Matrix.

Sei nun A eine symmetrische Matrix. Es gilt $QR = A = A^T = R^T Q^T$ und folglich $R = Q^T R^T Q^T$. Wir sehen also, dass

$$\tilde{A} = RQ = Q^T R^T Q^T Q = (RQ)^T = \tilde{A}^T.$$

A.8 Kapitel 8

Lösung A.8.1: Eine lineare Optimierungsaufgabe in kanonischer Form hat die Gestalt: Suche $x \in \mathbb{R}^n$ mit

$$Q(x) = c^T \cdot x \longrightarrow \min!,$$

$$x \geq 0,\ Ax = b,$$

mit vorgegebenen $c \in \mathbb{R}^n,\ b \in \mathbb{R}^m$ und $A \in \mathbb{R}^{m\times n}$.

a) 1. Version: Aufspalten von x_3 in einen positiven und negativen Anteil $x_3^+ - x_3^-$ mit $x_3^+ \geq 0,\ x_3^- \geq 0$ und Einführen dreier Schlupfvariablen $x_4,\ x_5$ und x_6 werden die Ungleichungen in Gleichungen überführt:

$$Q(x) = x_1 + x_2 + x_3^+ - x_3^- \to \min!, \quad x \geq 0,$$

$$\begin{array}{rcrcrcrcrcl} x_1 & + & 2x_2 & & & & + & x_4 & & & = 5 \\ & & x_2 & + & x_3^+ & - & x_3^- & + & x_5 & & = 0 \\ & & 3x_2 & - & 4x_3^+ & + & 4x_3^- & & & + x_6 & = 1 \end{array}$$

2. Version: Die Ungleichungen $x_2 \geq 0$ und $x_2 + x_3 \leq 0$ implizieren $x_3 \leq 0$. Durch Substitution $x_3 \to -x_3$ erhält man die Bedingung $x \geq 0$. Durch Einführen von Schlupfvariablen x_4, x_5 und x_6 erhält man äquivalentes System letztendlich

$$Q(x) = x_1 + x_2 - x_3 \to \min!, \quad x \geq 0,$$

$$\begin{array}{rcrcrcrcl} x_1 & + & 2x_2 & & + & x_4 & & & = 5 \\ & & x_2 & - & x_3 & + & x_5 & & = 0 \\ & & 3x_2 & + & 4x_3 & & & + x_6 & = 1 \end{array}$$

b) Aufspalten von x_i in $x_i^+ - x_i^-$ mit der Bedingung $x_i^\pm \geq 0$, sowie Einführen zweier Schlupfvariablen für die Ungleichungen liefert:

$$Q(x) = x_1^+ + x_1^- + x_2^+ + x_2^- + x_3^+ + x_3^- \to \min!, \quad x \geq 0,$$

$$\begin{array}{rcrcrcrcrcl} x_1^+ & - & x_1^- & + & x_2^+ & - & x_2^- & + & x_4 & = & 1, \\ 2x_1^+ & - & 2x_1^- & & + & x_3^+ & - & x_3^- & + & x_5 & = 3. \end{array}$$

Lösung A.8.2: Lösung ist der Schnittpunkt der Geraden (Man zeichne eine Skizze.)

$$\begin{aligned} -2x_1 + x_2 &= -2, \\ -x_1 - x_2 &= -5. \end{aligned}$$

Dieser ist $(x_1^*, x_2^*) = (7/3, 8/3)$ mit $Q_{\min} = 22/3$.

Die Aufgabe ist für $Q(x) \to \max!$ offensichtlich nicht lösbar.

Lösung A.8.3: a) Gauß-Jordan-Algorithmus:

$$\begin{array}{cccc} x_1 & x_2 & x_3 & \\ 1 & 0 & -1 & y_1 \\ \mathbf{2} & 2 & 0 & y_2 \\ 1 & 2 & 1 & y_3 \end{array} \quad \to \quad \begin{array}{cccc} y_2 & x_2 & x_3 & \\ 1/2 & -1 & -1 & y_1 \\ 1/2 & -1 & 0 & x_1 \\ 1/2 & \mathbf{1} & 1 & y_3 \end{array} \quad \to \quad \begin{array}{cccc} y_2 & y_3 & x_3 & \\ 1 & -1 & 0 & y_1 \\ 1 & -1 & 1 & x_1 \\ -1/2 & 1 & -1 & x_2 \end{array}$$

Die Matrix A hat Rang 2, da der Algorithmus nach dem zweiten Schritt abbricht. Aus der ersten Zeile erschließt man, dass das Gleichungssystem $Ax = y$ genau dann lösbar ist, wenn $y_2 + y_3 = y_1$ gilt. Aus den beiden übrigen Zeilen ergibt sich, dass dann $(y_2 - y_3, 1/2y_2 + y_3, 0)$ eine Lösung ist und der Kern von $(1, -1, 1)$ erzeugt wird.

Lösung A.8.4: Durch Einführen zweier Schlupfvariablen x_4 und x_5 überführt man das Optimierungsproblem in Normalform:

$$Q(x) = c^T \cdot x \to \min!, \quad x \geq 0, \quad Ax = b, \qquad \text{mit}$$
$$c = \begin{pmatrix} -1 & -3 & -1 & 0 & 0 \end{pmatrix}, \qquad A = \begin{pmatrix} 5 & 3 & 0 & 1 & 0 \\ 1 & 2 & 4 & 0 & 1 \end{pmatrix}, \qquad b = \begin{pmatrix} 3 \\ 4 \end{pmatrix}.$$

Dank der beiden Schlupfvariablen ist das Raten einer Startecke einfach: $x^0 = (0, 0, 0, 3, 4)^T$ mit Basis $\hat{B}(x^0) = \{a_4, a_5\}$. (Es ist $x^0 \geq 0$ und $Ax = b$. Weiterhin ist $\hat{B}(x^0)$ linear unabhängig.) Das Starttableau ist

	x_1	x_2	x_3	
x_4	-5	-3	0	$x_4^0 = 3$
x_5	-1	-2	-4	$x_5^0 = 4$
	$\gamma_1 = -1$	$\gamma_2 = -3$	$\gamma_3 = -1$	$c^T \cdot x^0 = 0$

Es liegt Fall 2b vor. Wir wählen $q = 2$ und nach der Auswahlregel (R) $p = 4$:

	x_1	x_4	x_3	
x_2	$-5/3$	$-1/3$	0	$x_2^1 = 1$
x_5	$7/3$	$2/3$	-4	$x_5^1 = -2$
	$\gamma_1 = 4$	$\gamma_2 = 1$	$\gamma_3 = -1$	$c^T \cdot x^1 = -3$

Es liegt Fall 2b vor. Wir wählen $q = 3$ (da γ_3 negativ) und nach Auswahlregel (R) $p = 5$.

	x_1	x_4	x_5	
x_2		$*$		$x_2^2 = 1$
x_3				$x_3^2 = 1/2$
	$\gamma_1 = 41/12$	$\gamma_2 = 5/6$	$\gamma_3 = 1/4$	$c^T \cdot x^1 = -7/2$

Hier bricht nun der Simplexalgorithmus ab. Als Eckenlösung ergibt sich also $x^3 = (0, 1, 1/2, 0, 0)^T$ mit $Q(x^3) = -7/2$.

Lösung A.8.5: Das lineare Programm liegt in Normalform vor:

$$Q(x) = c^T \cdot x \to \min!, \quad x \geq 0, \quad Ax = b,$$

mit

$$c = (4, 1, 1)^T, \quad A = \begin{pmatrix} 2 & 1 & 2 \\ 3 & 3 & 1 \end{pmatrix}, \quad b = \begin{pmatrix} 4 \\ 3 \end{pmatrix}$$

Da keine Startecke ersichtlich ist löst man zunächst das folgende Hilfsproblem: Durch Einführen zweier zusätzlicher Freiheitsgrade $v = (x_4, x_5)$ wird die Starteckensuche in ein äquivalentes Optimierungsproblem überführt:

$$x_4 + x_5 \to \min!, \quad x \geq 0, \quad \begin{pmatrix} A & I_2 \end{pmatrix} \begin{pmatrix} x \\ v \end{pmatrix} = b.$$

Dank der beiden Schlupfvariablen ist das Raten einer Startecke einfach: $x^0 = (0, 0, 0, 4, 3)^T$ mit Basis $\hat{B}(x^0) = \{a_4, a_5\}$. (Es ist $x^0 \geq 0$ und $Ax = b$. Weiterhin ist $\hat{B}(x^0)$ linear unabhängig.) Das Starttableau ist

	x_1	x_2	x_3	
x_4	-2	-1	-2	$x_4^0 = 4$
x_5	$[\mathbf{-3}]$	-3	-1	$x_5^0 = 3$
	$\gamma_1 = -5$	$\gamma_2 = -4$	$\gamma_3 = -3$	$\tilde{c}^T \cdot x^0 = 7$

Es liegt Fall 2b vor. Wir wählen $q = 1$ und nach der Auswahlregel (R) $p = 5$:

	x_5	x_4	x_3	
x_4	$2/3$	1	$[\mathbf{-4/3}]$	$x_4^1 = 2$
x_1	$-1/3$	-1	$-1/3$	$x_1^1 = 1$
	$\gamma_1 = 5/3$	$\gamma_2 = 1$	$\gamma_3 = -4/3$	$\tilde{c}^T \cdot x^1 = 2$

Es liegt Fall 2b vor. Wir wählen $q = 3$ (da γ_3 negativ) und nach Auswahlregel (R) $p = 4$.

	x_5	x_2	x_4	
x_3	$1/2$	$3/4$	$-3/4$	$x_3^2 = 3/2$
x_1	$-1/2$	$-5/4$	$1/4$	$x_1^2 = 1/2$
	$\gamma_1 = 1$	$\gamma_2 = 0$	$\gamma_3 = 1$	$\tilde{c}^T \cdot x^1 = 0$

Eine Startecke des ursprünglichen Problems ist demnach

$$x^0 = (1/2, 0, 3/2).$$

Das zugehörige Starttableau ergibt sich direkt aus obigem finalen Tableau durch Streichen der Spalten für x_4 und x_5:

	x_2	
x_3	$3/4$	$x_3^0 = 3/2$
x_1	$[-\mathbf{5/4}]$	$x_1^0 = 1/2$
	$\gamma = -13/4$	$c^T \cdot x^1 = 7/2$

Fall 2b mit $q = 2$ und nach Auswahlregel (R) $p = 1$:

	x_1	
x_3	$-3/5$	$x_3^1 = 9/5$
x_2	$-4/5$	$x_2^1 = 2/5$
	$\gamma = 13/5$	$c^T \cdot x^1 = 11/5$

Die Optimierungsaufgabe wird durch die Eckenlösung $x^* = (0, 2/5, 9/5)$ mit dem Wert $Q(x^*) = -11/5$ gelöst.

Lösung A.8.6: Überführt auf Normalform lautet die Optimierungsaufgabe:
Suche $x \in \mathbb{R}^7$:

$$c^T \cdot x \to \min!, \quad x \geq 0, \quad Ax = b$$

mit

$$c = (-3/4, 20, -1/2, 6, 0, 0, 0)^T$$

$$A = \begin{pmatrix} 1/4 & -8 & -1 & 9 & 1 & 0 & 0 \\ 1/2 & -12 & -1/2 & 3 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix}, \quad b = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}.$$

Eine Ausgangsecke ist $x^0 = (0_4, b)^T \in \mathbb{R}^7$.

a) Löse mit Simplexverfahren mittels Regel (R) + minimales $p \in I^0$. Das q sei so bestimmt, dass γ_q minimal ist.

	x_1	x_2	x_3	x_4	
x_5	$[-\mathbf{1/4}]$	8	1	-9	$x_5^0 = 0$
x_6	$-1/2$	12	$1/2$	-3	$x_6^0 = 0$
x_7	0	0	-1	0	$x_7^0 = 1$
γ	$-3/4$	20	$-1/2$	6	$c^T \cdot x^0 = 0$

(I) Auswahl: $q = 1$, $p = 5$

	x_5	x_2	x_3	x_4	
x_1	-4	32	4	-36	$x_1^1 = 0$
x_6	2	$[-\mathbf{4}]$	$-3/2$	15	$x_6^1 = 0$
x_7	0	0	-1	0	$x_7^1 = 1$
γ	3	-4	$-7/2$	33	$c^T \cdot x^0 = 0$

(II) Auswahl: $q = 2$, $p = 6$

	x_5	x_6	x_3	x_4	
x_1	12	-8	$[\mathbf{-8}]$	84	$x_1^2 = 0$
x_2	1/2	$-1/4$	$-3/8$	15/4	$x_2^2 = 0$
x_7	0	0	-1	0	$x_7^2 = 1$
γ	1	1	-2	18	$c^T \cdot x^0 = 0$

(III) Auswahl: $q = 3$, $p = 1$

	x_5	x_6	x_1	x_4	
x_3	3/2	-1	$-1/8$	21/2	$x_3^3 = 0$
x_2	$-1/16$	1/8	3/64	$[\mathbf{-3/16}]$	$x_2^3 = 0$
x_7	$-3/2$	1	1/8	$-21/2$	$x_7^3 = 1$
γ	-2	3	1/4	-3	$c^T \cdot x^0 = 0$

(IV) Auswahl: $q = 4$, $p = 2$

	x_5	x_6	x_1	x_2	
x_3	$[\mathbf{-2}]$	6	5/2	-56	$x_3^3 = 0$
x_4	$-1/3$	2/3	1/4	$-16/3$	$x_4^3 = 0$
x_7	2	-6	$-5/2$	56	$x_7^3 = 1$
γ	-1	1	$-1/2$	16	$c^T \cdot x^0 = 0$

(V) Auswahl: $q = 4$, $p = 2$

	x_3	x_6	x_1	x_2	
x_5	$-1/2$	3	5/4	-28	$x_5^3 = 0$
x_4	1/6	$[\mathbf{-1/3}]$	1/6	4	$x_4^3 = 0$
x_7	-1	0	0	0	$x_7^3 = 1$
γ	1/2	-2	$-7/4$	44	$c^T \cdot x^0 = 0$

(VI) Auswahl: $q = 6$, $p = 4$

	x_3	x_4	x_1	x_2	
x_5	1	-9	$-1/4$	8	$x_5^0 = 0$
x_6	1/2	-3	$-1/2$	12	$x_6^0 = 0$
x_7	-1	0	0	0	$x_7^0 = 1$
γ	$-1/2$	6	$-3/4$	20	$c^T \cdot x^0 = 0$

Endlosschleife!

b) Löse mit Simplexverfahren mittels Regel ($\tilde{R}$). Das q sei so bestimmt, dass γ_q minimal ist.

	x_1	x_2	x_3	x_4	
x_5	$-1/4$	8	1	-9	$x_5^0 = 0$
x_6	$[\mathbf{-1/2}]$	12	1/2	-3	$x_6^0 = 0$
x_7	0	0	-1	0	$x_7^0 = 1$
γ	$-3/4$	20	$-1/2$	6	$c^T \cdot x^0 = 0$

(I) Auswahl: $q = 1$, $r = 6$

	x_6	x_2	x_3	x_4	
x_5	1/2	2	3/4	$-15/2$	$x_5^1 = 0$
x_1	-2	24	1	-6	$x_1^1 = 0$
x_7	0	0	$[\mathbf{-1}]$	0	$x_7^1 = 1$
γ	3/2	2	$-5/4$	21/2	$c^T \cdot x^0 = 0$

(II) Auswahl: $q = 3$, $7 = 7$

	x_3	x_4	x_1	x_2	
x_5	$1/2$	2	$-3/4$	$-15/2$	$x_5^2 = 3/4$
x_1	-2	24	-1	-6	$x_1^2 = 1$
x_3	0	0	-1	0	$x_3^2 = 1$
γ	$3/2$	2	$5/4$	$21/2$	$c^T \cdot x^0 = \mathbf{-5/4}$

Optimales Tableau!

Wir haben die Eckenlösung $x^2 = (1, 0, 1, 0, 3/4, 0, 0)^T$ mit $Q(x^2) = -5/4$ gefunden.