

Grundlagen der Numerischen Mathematik

Heinrich Voß

Technische Universität Hamburg–Harburg

Arbeitsbereich Mathematik

2004

Inhaltsverzeichnis

| | | |
|----------|---|----------|
| 1 | Einleitung | 1 |
| 1.1 | Zahldarstellung | 4 |
| 1.2 | Rundungsfehler und Gleitpunktrechnung | 5 |
| 2 | Interpolation | 8 |
| 2.1 | Problemstellung | 8 |
| 2.2 | Polynominterpolation | 10 |
| 2.2.1 | Lagrangesche Interpolationsformel | 10 |
| 2.2.2 | Die Newtonsche Interpolationsformel | 14 |
| 2.2.3 | Fehlerbetrachtungen | 18 |
| 2.2.4 | Hermite Interpolation | 24 |
| 2.3 | Spline Interpolation | 26 |
| 2.3.1 | Stückweise lineare Funktionen | 27 |
| 2.3.2 | Kubische Hermite Splines | 29 |
| 2.3.3 | Kubische Splines | 31 |
| 2.4 | Bezierkurven | 40 |

| | | |
|----------|---|------------|
| 3 | Numerische Integration | 45 |
| 3.1 | Konstruktion von Quadraturformeln | 45 |
| 3.2 | Fehler von Quadraturformeln | 51 |
| 3.3 | Romberg Verfahren | 56 |
| 3.4 | Quadraturformeln von Gauß | 63 |
| 3.5 | Adaptive Quadratur | 77 |
| 3.6 | Numerische Differentiation | 87 |
| | | |
| 4 | Lineare Systeme | 93 |
| 4.1 | Zerlegung regulärer Matrizen | 93 |
| 4.2 | Modifikationen des Gaußschen Verfahrens | 99 |
| 4.3 | Bandmatrizen | 108 |
| 4.4 | Störungen linearer Systeme | 111 |
| 4.5 | Lineare Systeme mit spezieller Struktur | 119 |
| 4.5.1 | Vandermonde Systeme | 120 |
| 4.5.2 | Schnelle Fourier Transformation | 123 |
| 4.5.3 | Toeplitz Matrizen | 125 |
| 4.6 | Software für Lineare Gleichungssysteme | 130 |
| | | |
| 5 | Lineare Ausgleichsprobleme | 132 |
| 5.1 | Normalgleichungen | 132 |
| 5.2 | Orthogonale Zerlegung von Matrizen | 133 |
| 5.3 | Singulärwertzerlegung | 142 |
| 5.4 | Pseudoinverse | 149 |
| 5.5 | Störung von Ausgleichsproblemen | 153 |
| 5.6 | Regularisierung | 155 |

| | |
|---|------------|
| 6 Nichtsymmetrische Eigenwertaufgaben | 161 |
| 6.1 Vorbetrachtungen | 161 |
| 6.2 Störungssätze | 166 |
| 6.3 Potenzmethode | 174 |
| 6.3.1 Potenzmethode; von Mises Iteration | 175 |
| 6.3.2 Inverse Iteration | 178 |
| 6.3.3 Deflation | 181 |
| 6.3.4 Unterraum Iteration | 183 |
| 6.4 Der QR Algorithmus | 185 |
| 6.4.1 Beschleunigung des QR Algorithmus, explizite Shifts | 192 |
| 6.4.2 Implizite Shifts | 198 |
| 6.5 Der QZ Algorithmus | 203 |
| 7 Symmetrische Eigenwertaufgaben | 210 |
| 7.1 Charakterisierung von Eigenwerten | 210 |
| 7.2 QR Algorithmus | 218 |
| 7.3 Bisektion | 221 |
| 7.4 Rayleigh Quotienten Iteration | 224 |
| 7.5 Divide-and-Conquer Verfahren | 226 |
| 7.6 Jacobi Verfahren | 235 |
| 7.7 Berechnung der Singulärwertzerlegung | 239 |
| 7.8 Allgemeine Eigenwertaufgaben | 246 |

| | |
|---|------------|
| 8 Nichtlineare Gleichungssysteme | 248 |
| 8.1 Fixpunktsatz für kontrahierende Abbildungen | 248 |
| 8.2 Nullstellen reeller Funktionen | 256 |
| 8.2.1 Newton Verfahren | 256 |
| 8.2.2 Einschließende Verfahren | 266 |
| 8.3 Newton Verfahren für Systeme | 272 |
| 8.4 Bairstow Verfahren | 281 |
| 8.5 Newton ähnliche Verfahren | 284 |
| 8.6 Quasi-Newton Verfahren | 289 |
| 8.7 Homotopieverfahren | 299 |
| 8.8 Nichtlineare Ausgleichsprobleme | 305 |
| Literaturverzeichnis | 310 |

Kapitel 1

Einleitung

Aufgabe der Numerischen Mathematik ist, Algorithmen (d.h. Rechenvorschriften) für die näherungsweise numerische Lösung mathematischer Probleme der Naturwissenschaften, Technik, Ökonomie u.s.w. bereitzustellen und zu diskutieren.

Gesichtspunkte bei der Bewertung eines Algorithmus (und beim Vergleich von Algorithmen) sind der Aufwand (z.B. Zahl der Operationen), Speicherplatzbedarf und eine Fehleranalyse.

Man unterscheidet drei Typen von Fehlern nach ihren Quellen:

1. **Datenfehler:** Die Eingangsdaten einer Aufgabe können fehlerhaft sein, wenn sie etwa aus vorhergehenden Rechnungen, physikalischen Messungen oder empirischen Untersuchungen stammen.
2. **Verfahrensfehler:** Dies sind Fehler, die dadurch entstehen, dass man ein Problem diskretisiert. (z.B. eine Differentialgleichung durch eine Differenzgleichung ersetzt) oder ein Iterationsverfahren nach endlich vielen Schritten abbricht.
3. **Rundungsfehler:** Bei der Ausführung der Rechenoperationen auf einer Rechenanlage entstehen Fehler, da das Ergebnis (aber auch schon alle Zwischenergebnisse) nur im Rahmen eines begrenzten Zahlbereichs (sog. Maschinenzahlen) dargestellt werden kann, also gerundet werden muss.

Die Frage, wie sich Datenfehler auf die Lösung einer Aufgabe auswirken, nennt man das Konditionsproblem der Aufgabe.

Bewirken kleine Eingangsfehler auch nur kleine Ergebnisfehler, so nennt man das Problem gut konditioniert, anderenfalls schlecht konditioniert. Die Kondition eines Problems hängt nicht nur von der Aufgabenstellung (z.B. "Lösung eines linearen Gleichungssystems"), sondern auch von den Eingangsdaten ab (z.B. Koeffizienten der Matrix).

Beispiel 1.1. Das lineare Gleichungssystem

$$\begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad a \in \mathbb{R} \quad (1.1)$$

besitzt die Lösung

$$x_0 = 1 \quad , \quad y_0 = 0 \quad .$$

Das gestörte Gleichungssystem

$$\begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 1 \\ \delta \end{pmatrix}$$

hat die Lösung

$$x_\delta = 1 - \delta a \quad , \quad y_\delta = \delta \quad .$$

Damit gilt

$$\begin{pmatrix} x_0 \\ y_0 \end{pmatrix} - \begin{pmatrix} x_\delta \\ y_\delta \end{pmatrix} = \delta \begin{pmatrix} -a \\ 1 \end{pmatrix} \quad .$$

Änderungen der rechten Seite $(1 \ 0)^T$ in der zweiten Komponente werden also mit dem Faktor $\sqrt{1+a^2}$ (bzgl. der Euklidischen Norm) verstärkt. Damit ist das Problem für kleine $|a|$ gut und für große $|a|$ schlecht konditioniert. \square

Ein numerisches Verfahren heißt gut konditioniert (numerisch stabil), wenn die gelieferte Lösung eines gegebenen Problems die exakte Lösung eines Problems ist, das aus dem ursprünglichen Problem durch geringe Änderung der Eingangsdaten hervorgeht. Anderenfalls heißt das numerische Verfahren schlecht konditioniert (oder numerisch instabil).

Beispiel 1.2. Das Integral

$$\int_0^1 \frac{x^{20}}{10+x} dx$$

kann auf folgende Weise berechnet werden:

Für

$$y_n := \int_0^1 \frac{x^n}{10+x} dx$$

Tabelle 1.1: Rekursion

| n | vorwärts | rückwärts |
|-----|---------------------------|--------------------------|
| 0 | 9.53101798043249E - 0002 | 9.53101798043249E - 0002 |
| 1 | 4.68982019567514E - 0002 | 4.68982019567514E - 0002 |
| 2 | 3.10179804324860E - 0002 | 3.10179804324860E - 0002 |
| 3 | 2.31535290084733E - 0002 | 2.31535290084733E - 0002 |
| 4 | 1.84647099152673E - 0002 | 1.84647099152671E - 0002 |
| 5 | 1.53529008473272E - 0002 | 1.53529008473289E - 0002 |
| 6 | 1.31376581933944E - 0002 | 1.31376581933773E - 0002 |
| 7 | 1.14805609231986E - 0002 | 1.14805609233700E - 0002 |
| 8 | 1.01943907680135E - 0002 | 1.01943907663000E - 0002 |
| 9 | 9.16720343097581E - 0003 | 9.16720344811137E - 0003 |
| 10 | 8.32796569024192E - 0003 | 8.32796551888631E - 0003 |
| 11 | 7.62943400667172E - 0003 | 7.62943572022779E - 0003 |
| 12 | 7.03899326661614E - 0003 | 7.03897613105546E - 0003 |
| 13 | 6.53314425691553E - 0003 | 6.53331561252233E - 0003 |
| 14 | 6.09712885941609E - 0003 | 6.09541530334817E - 0003 |
| 15 | 5.69537807250574E - 0003 | 5.71251363318499E - 0003 |
| 16 | 5.54621927494255E - 0003 | 5.37486366815006E - 0003 |
| 17 | 3.36133666233920E - 0003 | 5.07489273026414E - 0003 |
| 18 | 2.19421889321635E - 0002 | 4.80662825291418E - 0003 |
| 19 | -1.66790310374267E - 0001 | 4.56529641822660E - 0003 |
| 20 | 1.71790310374267E + 0000 | 4.34703581773402E - 0003 |
| 21 | | 4.14868944170746E - 0003 |
| 22 | | 3.96765103747089E - 0003 |
| 23 | | 3.80175049485636E - 0003 |
| 24 | | 3.64916171810310E - 0003 |
| 25 | | 3.50838281896903E - 0003 |
| 26 | | 3.37771027184820E - 0003 |
| 27 | | 3.25993431855501E - 0003 |
| 28 | | 3.11494252873563E - 0003 |
| 29 | | 3.33333333333333E - 0003 |
| 30 | | 0.00000000000000E + 0000 |

gilt

$$y_n + 10y_{n-1} = \int_0^1 \frac{x^n + 10x^{n-1}}{x + 10} dx = \int_0^1 x^{n-1} dx = \frac{1}{n}, \quad (1.2)$$

$$y_0 = \int_0^1 \frac{dx}{10 + x} = \ln(1.1).$$

Wertet man die Differenzenformel $y_n = \frac{1}{n} - 10y_{n-1}$ für $n = 1, \dots, 20$ aus, so erhält man die Werte der Tabelle 1.1.

Obwohl das Problem, das Integral zu berechnen, gut konditioniert ist, erhält man ein unbrauchbares Resultat. Das Verfahren ist also instabil.

Löst man (1.2) nach y_{n-1} auf:

$$y_{n-1} = 0.1 \left(\frac{1}{n} - y_n \right)$$

und startet man mit der groben Näherung $y_{30} = 0$, so erhält man y_{20}, \dots, y_0 mit einer Genauigkeit von wenigstens 10 gültigen Stellen. \square

1.1 Zahlendarstellung

Üblicherweise stellt man Zahlen im Dezimalsystem dar, d.h. eine reelle Zahl x wird durch die Koeffizienten α_i der Dezimaldarstellung von x festgelegt:

$$x = \pm (\alpha_n \cdot 10^n + \alpha_{n-1} \cdot 10^{n-1} + \dots + \alpha_0 \cdot 10^0 + \alpha_{-1} \cdot 10^{-1} + \dots)$$

$$\text{mit } \alpha_i \in \{0, 1, \dots, 9\} .$$

Abgekürzt schreibt man

$$\alpha_n \alpha_{n-1} \dots \alpha_0 . \alpha_{-1} \alpha_{-2} \dots .$$

Aus technischen Gründen arbeiten digitale Rechenanlagen im Dualsystem (Basis: 2) oder Hexadezimalsystem (Basis: 16). Wir bleiben bei der Basis 10.

Für die interne Darstellung einer Zahl in einem Rechner steht nur eine feste Anzahl t (=Wortlänge) von Dezimalstellen zur Verfügung. Diese Wortlänge wird auf zwei Arten zur Darstellung einer Zahl benutzt:

Bei der **Festpunktdarstellung** sind n_1 und n_2 , die Zahl der Stellen vor und nach dem Dezimalpunkt, festgelegt:

Beispiel 1.3. $t = 8, n_1 = 3, n_2 = 5$

$$\begin{array}{l} 30.411 \rightarrow 030 \\ 0.0023 \rightarrow 000 \end{array} \left| \begin{array}{l} 41100 \\ 00230 \end{array} \right. \quad \square$$

Wegen des verhältnismäßig kleinen Bereichs darstellbarer Zahlen wird mit Festpunktzahlen nur dann gearbeitet, wenn keine zu großen Unterschiede in der Größenordnung der auftretenden Zahlen bestehen (kaufmännisch-organisatorischer Bereich: Stückzahlen: $n_2 = 0$, Preise: $n_2 = 2$).

Schreibt man x in der **Gleitpunktdarstellung**, so liegt die Mantissenlänge $t = n_1 + n_2$ fest ; die Lage des Dezimalpunktes wird durch einen Exponenten markiert:

$$\begin{aligned} x &= \pm \left(\alpha_{n_1-1} \cdot 10^{n_1-1} + \alpha_{n_1-2} \cdot 10^{n_1-2} + \dots + \alpha_0 \cdot 10^0 + \alpha_{-1} \cdot 10^{-1} + \dots + \alpha_{-n_2} \cdot 10^{-n_2} \right) . \\ &= \pm \left(\alpha_{n_1-1} \cdot 10^{-1} + \alpha_{n_1-2} \cdot 10^{-2} + \dots + \alpha_{-n_2} \cdot 10^{-(n_1+n_2)} \right) \cdot 10^{n_1} \\ &= \pm 0 \cdot \underbrace{\alpha_{n_1-1} \alpha_{n_1-2} \dots \alpha_{-n_2}}_{\text{Mantisse}} \cdot 10^{n_1}, \quad n_1 : \text{Exponent} \end{aligned}$$

Beispiel 1.4. $t = 4$

$$0.0023 \rightarrow 0.0023_{10}0 \quad \text{oder} \quad 0.2300_{10} - 2. \quad \square$$

Die Gleitpunktdarstellung einer Zahl ist i.a. nicht eindeutig. Sie heißt normalisiert, falls $x = 0$ oder für die erste Ziffer $\alpha_1 \neq 0$ gilt. Wir betrachten von nun an nur noch normalisierte Gleitpunktzahlen.

1.2 Rundungsfehler und Gleitpunktrechnung

Die Menge A der in einer Maschine darstellbaren Zahlen ist endlich (endliche Mantissenlänge t , und für die Darstellung des Exponenten stehen auch nur $e < \infty$ viele Stellen zur Verfügung).

Für ein gegebenes $x \in \mathbb{R}$ bezeichnen wir mit $\text{fl}(x) \in A$ eine Maschinenzahl, durch die x am besten approximiert wird, d.h.

$$|\text{fl}(x) - x| \leq |x - a| \quad \text{für alle } a \in A .$$

Diese Vorschrift ist noch nicht eindeutig (wird 0.5 auf- oder abgerundet?). Wir setzen fest:

Sei $x \in \mathbb{R}$ mit der normalisierten Gleitpunktdarstellung

$$x = \pm 0.\alpha_1\alpha_2 \dots \alpha_t \alpha_{t+1} \dots \cdot 10^n ,$$

dann wird x durch die folgende Vorschrift gerundet:

$$\text{fl}(x) = \begin{cases} \pm 0.\alpha_1\alpha_2 \dots \alpha_t \cdot 10^n & , \text{ falls } 0 \leq \alpha_{t+1} \leq 4 \\ \pm (0.\alpha_1\alpha_2 \dots \alpha_t + 10^{-t}) \cdot 10^n & , \text{ falls } 5 \leq \alpha_{t+1} \leq 9 . \end{cases}$$

Für den absoluten Fehler gilt

$$|x - \text{fl}(x)| \leq \frac{1}{2} \cdot 10^{n-t}$$

und für den relativen Fehler

$$\left| \frac{x - \text{fl}(x)}{x} \right| \leq \frac{1}{2} \cdot 10^{n-t} 10^{-n+1} = 5 \cdot 10^{-t} \quad (\alpha_1 \neq 0!).$$

Mit der Abkürzung $\varepsilon ps = 5 \cdot 10^{-t}$ (Maschinengenauigkeit) gilt also

$$\text{fl}(x) = (1 + \varepsilon)x, \quad |\varepsilon| \leq \varepsilon ps. \quad (1.3)$$

$\text{fl}(x)$ ist nicht stets eine Maschinenzahl, da nicht beliebig große oder kleine Zahlen dargestellt werden können:

Beispiel 1.5. $(t = 4, e = 2)$

$$\text{fl}(0.99997_{10}99) = 0.1000_{10}100 \notin A \quad (\text{Exponentenüberlauf})$$

$$\text{fl}(0.01234_{10} - 99) = 0.1234_{10} - 100 \notin A \quad (\text{Exponentenunterlauf}). \quad \square$$

Setzt man im zweiten Fall $\text{fl}(0.01234_{10} - 99) = 0$ oder $0.0123_{10} - 99$, so gilt zwar $\text{fl}(\dots) \in A$, aber es ist nicht mehr (1.3) erfüllt. Da bei den heutigen Anlagen e genügend groß ist, tritt Exponentenüberlauf oder -unterlauf nur sehr selten auf. Wir nehmen daher für das Weitere $e = \infty$ an, so dass bei der Rundung (1.3) gilt.

Offensichtlich gilt

$$x, y \in A \quad \not\Rightarrow \quad x \pm y, x \cdot y, x/y \in A.$$

Statt der Operationen $+, -, \cdot, /$ sind daher auf dem Rechner als Ersatz die **Gleitpunktoperationen** $+^*, -^*, \cdot^*, /^*$ realisiert, die man mit Hilfe von fl so beschreiben kann: $(x, y \in A)$

$$x +^* y := \text{fl}(x + y), \quad x -^* y := \text{fl}(x - y), \quad y \cdot^* x := \text{fl}(x \cdot y), \quad x /^* y := \text{fl}(x/y)$$

(In der Maschine wird die Operation exakt ausgeführt, danach wird gerundet). Wegen (1.3) gilt

$$x \circ^* y = (x \circ y)(1 + \varepsilon), \quad |\varepsilon| \leq \varepsilon ps,$$

für jede der Operationen $\circ \in \{+, -, \cdot, /\}$. (Der relative Fehler hat also die Größenordnung der Maschinengenauigkeit).

Waren aber x und y keine Maschinenzahlen, so wird zunächst gerundet und dann $\text{fl}(x) \circ^* \text{fl}(y)$ berechnet.

Hierfür gilt wegen $\text{fl}(x) = (1 + \varepsilon_x)x$, $\text{fl}(y) = (1 + \varepsilon_y)y$

$$\frac{\text{fl}(x) + \text{fl}(y) - (x + y)}{x + y} = \frac{x}{x + y}\varepsilon_x + \frac{y}{x + y}\varepsilon_y .$$

Haben also bei der Addition die Summanden entgegengesetztes Vorzeichen, gleichen Exponenten und gleich führende Ziffern der Mantisse, so ist $x + y$ klein gegen x und gegen y und der relative Fehler wird verstärkt. (Man spricht dann von Auslöschung).

Beispiel 1.6. $t = 6$, $x = 1234.567$, $y = -1234.60$.

Es gilt

$$\left| \frac{(\text{fl}(x) + \text{fl}(y)) - (x + y)}{x + y} \right| = \left| \frac{-0.03 - (-0.033)}{-0.033} \right| = \frac{1}{11} ,$$

aber

$$\varepsilon_x = \left| \frac{\text{fl}(x) - x}{x} \right| = \frac{0.003}{1234.567} \approx 2.5 \cdot 10^{-6}, \varepsilon_y = 0 . \quad \square$$

Die Operationen \cdot und $/$ sind wegen

$$\frac{\text{fl}(x) \cdot \text{fl}(y) - x \cdot y}{x \cdot y} = \varepsilon_x + \varepsilon_y + \varepsilon_x \cdot \varepsilon_y \approx \varepsilon_x + \varepsilon_y$$

für die Fehlerfortpflanzung in einer Rechnung unkritisch.

Kapitel 2

Interpolation

2.1 Problemstellung

Wir betrachten in diesem Kapitel das

Interpolationsproblem:

Gegeben seien eine Funktion

$$\Phi(x; a_1, \dots, a_n) : \mathbb{R} \supset I \rightarrow \mathbb{R},$$

die auf einem Intervall I erklärt ist und von n Parametern a_1, \dots, a_n abhängt, paarweise verschiedene **Knoten** (oder **Stützstellen**) $x_1, \dots, x_m \in I$, Vielfachheiten $r_1, \dots, r_m \in \mathbb{N}$ mit $\sum_{i=1}^m r_i = n$ und Werte $y_{ij} \in \mathbb{R}$, $i = 1, \dots, m$, $j = 0, \dots, r_i - 1$.

Bestimme die Parameter a_1, \dots, a_n so, dass die Interpolationsbedingungen

$$\Phi^{(j)}(x_i; a_1, \dots, a_n) = y_{ij}, \quad i = 1, \dots, m, j = 0, \dots, r_i - 1,$$

erfüllt sind.

Ist Φ linear in den Parametern a_i , d.h.

$$\Phi(x; a_1, \dots, a_n) = \sum_{i=1}^n a_i \phi_i(x),$$

so heißt das Interpolationsproblem **linear**.

Gilt $r_j = 1$ für alle j , so spricht man von **Lagrange Interpolation**, anderenfalls von **Hermite Interpolation**.

Bemerkung 2.1. Bei der Lagrange Interpolation werden nur Funktionswerte, aber keine Ableitungen vorgeschrieben. Man beachte, dass bei der Hermite Interpolation alle Ableitungen der Ordnung $0, \dots, r_i - 1$ vorgeschrieben werden. Lässt man Lücken bei den vorgeschriebenen Ableitungen zu, so spricht man von einem **Hermite Birkhoff Interpolationsproblem**. \square

Beispiel 2.2. 1. Polynominterpolation

$$\Phi(x; a_0, \dots, a_n) = \sum_{j=0}^n a_j x^j$$

2. trigonometrische Interpolation

$$\Phi(x; a_0, \dots, a_{2n}) = a_0 + \sum_{j=1}^n (a_{2j-1} \sin(jx) + a_{2j} \cos(jx))$$

3. rationale Interpolation

$$\Phi(x; a_0, \dots, a_n, b_0, \dots, b_p) = \frac{\sum_{i=0}^n a_i x^i}{\sum_{j=0}^p b_j x^j}.$$

4. Spline Interpolation

$$\Phi(x; a_1, \dots, a_n) = \sum_{i=1}^n a_i \phi_i(x),$$

wobei die ϕ_i auf Teilintervallen von I mit Polynomen übereinstimmen.

Wir werden uns nur mit der Polynominterpolation und der Interpolation mit Splines beschäftigen. Die trigonometrische und die rationale Interpolation werden ausführlich in Braess [13], Stoer [101] und Schwarz [93] behandelt.

Man benötigt die Interpolation zur

1. Bestimmung von Zwischenwerten aus Funktionstabellen (was seit der Verbreitung von elektronischen Taschenrechnern an Bedeutung verloren hat),
2. Herleitung von Formeln zur numerischen Integration,
3. Konvergenzbeschleunigung durch Extrapolation,
4. Numerische Behandlung von gewöhnlichen Differentialgleichungen.

2.2 Polynominterpolation

Wir betrachten in diesem Abschnitt die Interpolation mit Polynomen. Dabei behandeln wir vor allem das Lagrangesche Interpolationsproblem.

Gegeben seien $n+1$ verschiedene Knoten $x_j \in \mathbb{R}$, $j = 0, \dots, n$, und $n+1$ nicht notwendig verschiedene Werte $y_j \in \mathbb{R}$.

Gesucht ist ein Polynom p vom Höchstgrade n , so dass gilt

$$p(x_j) = y_j \quad \text{für } j = 0, \dots, n.$$

Nur im letzten Unterabschnitt gehen wir kurz auf einen Spezialfall der Hermite Interpolation ein.

Bemerkung 2.3. Die Beweise werden zeigen, dass die Existenz- und Eindeutigkeitsresultate und die Algorithmen zur Berechnung des Interpolationspolynoms ohne Änderungen für komplexe Knoten x_j und komplexe Daten y_j richtig bleiben. Nur die Fehlerabschätzungen beziehen sich ausschließlich auf reelle Probleme. \square

Wir bezeichnen mit Π_n die Menge der Polynome von Höchstgrad n (mit reellen oder komplexen Koeffizienten).

2.2.1 Lagrangesche Interpolationsformel

Satz 2.4. (Existenz und Eindeutigkeit)

Zu beliebigen $n+1$ Daten $(x_j, y_j) \in \mathbb{R}^2$, $j = 0, \dots, n$, mit $x_j \neq x_k$ für $j \neq k$ gibt es genau ein Polynom $p \in \Pi_n$ mit

$$p(x_j) = y_j, \quad j = 0, \dots, n. \quad (2.1)$$

Beweis: Eindeutigkeit: Angenommen es gibt zwei Polynome $p_1, p_2 \in \Pi_n$ mit $p_k(x_j) = y_j$, $j = 0, \dots, n$, $k = 1, 2$. Dann besitzt das Polynom $p := p_1 - p_2 \in \Pi_n$ die $n+1$ Nullstellen x_0, \dots, x_n , und daher folgt aus dem Fundamentalsatz der Algebra $p \equiv 0$.

Existenz: Die Existenz zeigen wir konstruktiv. Es sei

$$\ell_j(x) = \prod_{\substack{i=0 \\ i \neq j}}^n (x - x_i) \Bigg/ \prod_{\substack{i=0 \\ i \neq j}}^n (x_j - x_i), \quad j = 0, \dots, n. \quad (2.2)$$

Dann gilt $\ell_j \in \Pi_n$ und $\ell_j(x_k) = \delta_{jk}$ für $j, k = 0, \dots, n$, und daher erfüllt

$$p(x) := \sum_{j=0}^n y_j \ell_j(x) \in \Pi_n \quad (2.3)$$

die Interpolationsbedingungen (2.1) . ■

Definition 2.5. Die Darstellung (2.2), (2.3) des Interpolationsspolynoms heißt **Lagrangesche Interpolationsformel**.

Bemerkung 2.6. Prinzipiell kann man bei dem Ansatz $p(x) = \sum_{j=0}^n a_j x^j$ die Koeffizienten a_j aus dem linearen Gleichungssystem

$$\sum_{j=0}^n a_j x_k^j = y_k, \quad k = 0, \dots, n \quad (2.4)$$

berechnen. Dies erfordert mit dem in Abschnitt 4.5 gegebenen Algorithmus für Vandermondesche Systeme $2.5n^2$ flops. Weniger Aufwand erfordern die folgenden Algorithmen. □

Bemerkung 2.7. MATLAB stellt die Funktionen `p=polyfit(x,y,n)` zur Verfügung, durch die die Koeffizienten p des Ausgleichspolynoms vom Grad n zu den Daten (x, y) bestimmt werden. Ist $\dim x = n + 1$, so erhält man das Interpolationsspolynom. Mit `y=polyval(p,x)` kann man das Polynom dann an der Stelle x (oder den Stellen $x(j)$, falls x ein Vektor ist) auswerten. □

Verwendet man die Lagrangesche Interpolationsformel (2.3) naiv, so benötigt man zur Auswertung von p an einer festen Stelle \hat{x} zur Bereitstellung der $\ell_j(\hat{x})$ aus (2.2) jeweils $4n$ flops (berechne $(\hat{x} - x_i)/(x_j - x_i)$ für $i = 0, \dots, n, i \neq j$, und das Produkt dieser Quotienten) und zur Auswertung von (2.3) weitere $2n$ flops, zusammen also $4n^2 + O(n)$ flops.

Wir leiten nun eine Rechentechnik her, mit der dieser Aufwand wesentlich reduziert werden kann. Dazu schreiben wir (2.3) um in

$$p(\hat{x}) = \left(\sum_{j=0}^n y_j \cdot \frac{1}{\hat{x} - x_j} \prod_{\substack{i=0 \\ i \neq j}}^n \frac{1}{x_j - x_i} \right) \cdot \prod_{i=0}^n (\hat{x} - x_i). \quad (2.5)$$

Hierin sind die Stützkoeffizienten

$$\lambda_j := \prod_{\substack{i=0 \\ i \neq j}}^n \frac{1}{x_j - x_i}, \quad j = 0, \dots, n, \quad (2.6)$$

nur von den Knoten x_0, \dots, x_n abhängig und unabhängig von der Stelle \hat{x} , an der das Interpolationspolynom p ausgewertet werden soll.

Der Faktor

$$\gamma := \prod_{i=0}^n (\hat{x} - x_i)$$

hängt zwar von \hat{x} ab, ist aber unabhängig von den zu interpolierenden Daten y_j .

Für das Interpolationspolynom $q \in \Pi_n$ mit $q(x_j) = 1$ für $j = 0, \dots, n$ gilt wegen der Eindeutigkeit sicher $q(x) \equiv 1$, und damit insbesondere

$$q(\hat{x}) = 1 = \left(\sum_{j=0}^n \frac{\lambda_j}{\hat{x} - x_j} \right) \cdot \prod_{i=0}^n (\hat{x} - x_i),$$

d.h.

$$\gamma = 1 \left/ \sum_{j=0}^n \frac{\lambda_j}{\hat{x} - x_j} \right.$$

Damit erhält die Lagrangesche Interpolationsformel die Gestalt

$$p(x) = \sum_{j=0}^n y_j \cdot \frac{\lambda_j}{x - x_j} \left/ \sum_{j=0}^n \frac{\lambda_j}{x - x_j} \right. . \quad (2.7)$$

Sind also die Stützkoeffizienten λ_j bekannt (mit (2.6) benötigt man für ihre Berechnung offenbar $2n^2 + O(n)$ flops) so kann man das Interpolationspolynom p an jeder gewünschten Stelle \hat{x} auswerten durch

$$\mu_j := \frac{\lambda_j}{\hat{x} - x_j}, \quad j = 0, \dots, n,$$

und

$$p(\hat{x}) = \sum_{j=0}^n y_j \mu_j \left/ \sum_{j=0}^n \mu_j \right. .$$

Jede Auswertung erfordert also zusätzliche $5n$ flops.

Den Aufwand zur Berechnung der Stützkoeffizienten λ_j kann man verringern, indem man sie rekursiv berechnet.

Es seien die Stützkoeffizienten $\lambda_j^{(n-1)}$ für das Interpolationsproblem mit den n paarweise verschiedenen Knoten x_0, \dots, x_{n-1} bekannt, und es sei $x_n \neq x_j$, $j = 0, \dots, n-1$, ein zusätzlicher Knoten. Dann gilt sicher für die Stützkoeffizienten $\lambda_j^{(n)}$ des Interpolationsproblems mit den Knoten x_0, \dots, x_{n-1}, x_n

$$\lambda_j^{(n)} = \frac{\lambda_j^{(n-1)}}{x_j - x_n}, \quad j = 0, \dots, n-1.$$

Den noch fehlenden Stützkoeffizienten $\lambda_n^{(n)}$ erhalten wir aus

Lemma 2.8. *Es seien $\lambda_j^{(n)}$, $j = 0, \dots, n$, die Stützkoeffizienten zu den (paarweise verschiedenen) Knoten x_j , $j = 0, \dots, n$. Dann gilt für $n \geq 1$*

$$\sum_{j=0}^n \lambda_j^{(n)} = 0.$$

Beweis: Wir betrachten wieder das Interpolationspolynom

$$q(x) =: \sum_{k=0}^n \alpha_k x^k$$

mit den Daten $y_j = 1$ für $j = 0, \dots, n$. Dann gilt

$$q(x) = \sum_{j=0}^n \lambda_j^{(n)} \prod_{\substack{i=0 \\ i \neq j}}^n (x - x_i),$$

d.h. für den führenden Koeffizienten

$$\alpha_n = \sum_{j=0}^n \lambda_j^{(n)}.$$

Da andererseits $q(x) \equiv 1$ ist, folgt für $n \geq 1$ die Behauptung

$$0 = \alpha_n = \sum_{j=0}^n \lambda_j^{(n)}. \quad \blacksquare$$

Unsere Überlegungen zeigen, dass man die $\lambda_j^{(n)} =: l[j]$ mit dem folgenden Programmteil berechnen kann:

Algorithmus 2.9.

```

l(0) = 1;
for k = 1 : n
  l(k) = 0;
  for i = 0 : k-1
    l(i) = l(i)/(x(i) - x(k));
    l(k) = l(k) - l(i);
  end
end
end

```

Mit diesem Algorithmus erhält man alle Stützkoeffizienten mit

$$\sum_{k=1}^n 3k = \frac{3}{2}n(n+1)$$

flops.

2.2.2 Die Newtonsche Interpolationsformel

Wir werden nun eine Form des Interpolationspolynoms herleiten, bei der ebenfalls $1.5n(n+1)$ flops zur Vorbereitung (entsprechend der Bereitstellung der Stützkoeffizienten) benötigt werden, die Auswertung an einer festen Stelle dann aber nur noch $3n$ flops erfordert. Wir behandeln dazu zunächst die folgende Frage:

Das Polynom $p_n(x) \in \Pi_n$ interpoliere die Daten $(x_j, y_j) \in \mathbb{R}^2, j = 0, \dots, n$. Wir nehmen ein weiteres Zahlenpaar $(x_{n+1}, y_{n+1}) \in \mathbb{R}^2, x_{n+1} \neq x_j, j = 0, \dots, n$, hinzu. Kann man dann das Interpolationspolynom $p_{n+1}(x) \in \Pi_{n+1}$ zu den Daten $(x_j, y_j), j = 0, \dots, n+1$, schreiben als

$$p_{n+1}(x) = p_n(x) + f(x)$$

mit einer leicht berechenbaren Funktion $f(x)$?

Wegen $p_n(x) \in \Pi_n, p_{n+1}(x) \in \Pi_{n+1}$ gilt $f(x) = p_{n+1}(x) - p_n(x) \in \Pi_{n+1}$, und wegen

$$y_j = p_{n+1}(x_j) = p_n(x_j) + f(x_j) = y_j + f(x_j), \quad j = 0, \dots, n,$$

gilt $f(x_j) = 0, j = 0, \dots, n$. Daher hat $f(x)$ mit einem $a \in \mathbb{R}$ die Gestalt

$$f(x) = a \prod_{j=0}^n (x - x_j).$$

a kann man aus der Interpolationsbedingung

$$y_{n+1} = p_{n+1}(x_{n+1}) = p_n(x_{n+1}) + a \prod_{j=0}^n (x_{n+1} - x_j)$$

ermitteln:

$$a = \frac{y_{n+1} - p_n(x_{n+1})}{(x_{n+1} - x_0) \cdot \dots \cdot (x_{n+1} - x_n)}.$$

Wir wollen nun ein Verfahren zur Berechnung der Zahl a , des führenden Koeffizienten des Interpolationspolynoms, herleiten. Grundlage dafür ist

Satz 2.10. (Aitken Lemma)

Es sei zu $(x_j, y_j) \in \mathbb{R}^2, j = 0, \dots, n, x_i \neq x_j$ für $i \neq j$, das Interpolationspolynom gesucht.

Seien $p_{[0]} \in \Pi_{n-1}$ durch die Interpolationsbedingungen $p_{[0]}(x_j) = y_j, j = 0, \dots, n-1$, festgelegt, und sei $p_{[n]} \in \Pi_{n-1}$ definiert durch $p_{[n]}(x_j) = y_j, j = 1, \dots, n$. Dann gilt

$$p(x) = \frac{p_{[0]}(x)(x - x_n) - p_{[n]}(x)(x - x_0)}{x_0 - x_n}.$$

Beweis: Das angegebene Polynom erfüllt offenbar die Interpolationsbedingungen $p(x_j) = y_j, j = 0, \dots, n$. ■

Für $0 \leq i \leq j \leq n$ sei nun $p_{ij} \in \Pi_{j-i}$ das Interpolationspolynom, das $p_{ij}(x_k) = y_k, k = i, \dots, j$, erfüllt. Dann folgt aus dem Aitken Lemma, dass die p_{ij} rekursiv berechnet werden können durch

$$\begin{aligned}
 p_{ij}(x) &= \frac{p_{i,j-1}(x)(x - x_j) - p_{i+1,j}(x)(x - x_i)}{x_i - x_j} \\
 &= p_{i+1,j}(x) + (p_{i+1,j}(x) - p_{i,j-1}(x)) \frac{x - x_j}{x_j - x_i}
 \end{aligned}
 \tag{2.8}$$

Diese Rekursionsformel kann verwendet werden, um den Wert des Interpolationspolynoms an einer festen Stelle x (nicht das Polynom selbst) zu berechnen:

Algorithmus 2.11. (Algorithmus von Neville und Aitken)

```

for j = 0 : n
    t(j) = y(j);
    xj = x - x(j);
    if j > 0
        for i = j-1 : -1 : 0
            t(i) = t(i+1) + (t(i+1) - t(i))*xj / (x(j) - x(i));
        end
    end
end
p = t(0);
    
```

Bemerkung 2.12. Mit dem Algorithmus von Neville und Aitken wird das linke Tableau aufgestellt, das die Werte P_{ij} der interpolierenden Polynome p_{ij} an der festen Stelle x enthält. Das rechte Tableau enthält die Reihenfolge, in der die P_{ij} berechnet werden.

| | | | | | | | | | | |
|-------|----------------|----------|----------|----------|-----------------|--|---|---|---|----|
| x_0 | $y_0 = P_{00}$ | P_{01} | P_{02} | P_{03} | $P_{04} = p(x)$ | | 1 | | | |
| x_1 | $y_1 = P_{11}$ | P_{12} | P_{13} | P_{14} | | | 2 | 3 | | |
| x_2 | $y_2 = P_{22}$ | P_{23} | P_{24} | | | | 4 | 5 | 6 | 10 |
| x_3 | $y_3 = P_{33}$ | P_{34} | | | | | 7 | 8 | | |
| x_4 | $y_4 = P_{44}$ | | | | | | | | | |

□

Bemerkung 2.13. Zur Auswertung des Interpolationspolynoms p an einer festen Stelle benötigt man mit dem Algorithmus von Neville und Aitken offenbar $\frac{5}{2}n^2 + O(n)$ flops. \square

Man erhält aus der Rekursionsformel (2.8) eine weitere Darstellung des Interpolationspolynoms, die Newtonsche Interpolationsformel. Da a in

$$p_n(x) = p_{n-1}(x) + a \prod_{j=0}^{n-1} (x - x_j)$$

der Koeffizient bei der höchsten Potenz x^n in $p_n(x)$ ist, liest man aus (2.8) sofort ab:

Satz 2.14. (Newtonsche Interpolationsformel) *Das Interpolationspolynom $p \in \Pi_n$ aus (2.1) hat die Gestalt*

$$p(x) = \sum_{j=0}^n [x_0, \dots, x_j] \prod_{k=0}^{j-1} (x - x_k), \quad (2.9)$$

wobei die **dividierten Differenzen** $[x_0, \dots, x_j]$ rekursiv definiert sind durch

$$\begin{aligned} [x_j] &:= y_j, \\ [x_k, \dots, x_j] &:= \frac{[x_{k+1}, \dots, x_j] - [x_k, \dots, x_{j-1}]}{x_j - x_k}, \quad j > k \geq 0. \end{aligned} \quad (2.10)$$

Die dividierten Differenzen $c_j := [x_0, \dots, x_j]$ kann man mit folgendem Algorithmus aus den Wertepaaren (x_j, y_j) berechnen:

Algorithmus 2.15. (Dividierte Differenzen)

```

for k = 0 : n
  t(k) = y(k);
  if k > 0
    for i = k-1 : -1 : 0
      t(i) = (t(i+1) - t(i)) / (x(k) - x(i));
    end
  end
  c(k) = t(0);
end

```

Danach kann man das Interpolationspolynom an jeder gewünschten Stelle x_0 mit einem Horner-ähnlichen Schema auswerten:

Algorithmus 2.16.

```

p = c(n);
for i = n-1 : -1 : 0
    p = p * (x0 - x(i)) + c(i);
end

```

Bemerkung 2.17. Die $t(k)$ in dem Algorithmus enthalten die folgenden dividier-ten Differenzen, die in derselben Reihenfolge wie im Algorithmus von Neville und Aitken berechnet werden:

$$\begin{array}{llll}
 t(0) = y_0 & & & \\
 t(1) = y_1 & t(0) = [x_0, x_1] & t(0) = [x_0, x_1, x_2] & \\
 t(2) = y_2 & t(1) = [x_1, x_2] & t(1) = [x_1, x_2, x_3] & t(0) = [x_0, x_1, x_2, x_3] \\
 t(3) = y_3 & t(2) = [x_2, x_3] & &
 \end{array}$$

□

Bemerkung 2.18. Man benötigt (wie für die Berechnung der Stützkoeffizienten bei der Lagrangeschen Interpolationsformel) $\frac{3}{2}n(n+1)$ flops zur Berechnung aller c_j , zur Auswertung von p an einer festen Stelle \hat{x} zusätzlich $3n$ flops.

Die Newtonsche Interpolationsformel liefert also die effizienteste Methode zur Auswertung des Interpolationspolynoms. Selbst wenn man nur an einem Funktionswert interessiert ist, ist der Aufwand geringer als mit dem Algorithmus von Neville und Aitken. Wegen seiner einfachen Gestalt wird der Algorithmus von Neville und Aitken dennoch in der Praxis verwendet. □

Bemerkung 2.19. Natürlich erhält man für jede Anordnung der Knoten x_i (bei Rundungsfehlerfreier Rechnung) dasselbe Interpolationspolynom $p_n(x)$. Will man $p_n(x)$ nur an einer Stelle x (oder in deren Nähe) auswerten, so kann man den Rundungsfehlereinfluss klein halten, indem man die Knoten so nummeriert, dass gilt

$$|x - x_i| \leq |x - x_{i+1}|, \quad i = 0, \dots, n-1.$$

□

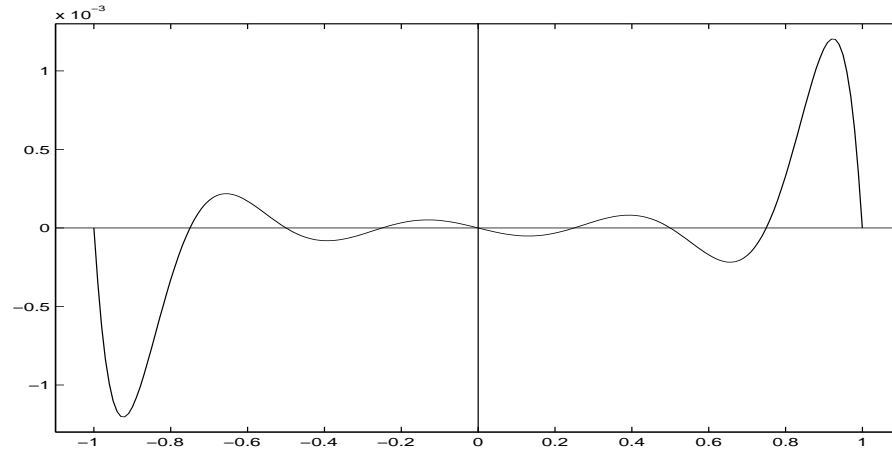


Abbildung 2.1: Fehlerkurve (äquidistante Knoten)

2.2.3 Fehlerbetrachtungen

Wir behandeln nun die Frage, wie gut eine gegebene, auf einem reellen Intervall definierte Funktion f durch das Interpolationspolynom zu vorgegebenen Knoten x_i in I approximiert wird (es sei also $y_i = f(x_i)$).

Beispiel 2.20. Gegeben sei auf dem Intervall $I = [-1, 1]$

$$f(x) = \sin \pi x$$

und $p_8(x) \in \Pi_8$, das Interpolationspolynom zu den Knoten

$$x_i = -1 + i \cdot 0.25, \quad i = 0, \dots, 8.$$

Dann erhält man die Fehlerkurve aus Abbildung 2.1. Das Interpolationspolynom liefert also am Rande des Intervalls eine ziemlich schlechte Approximation für f , obwohl f sehr gutartig ist (beliebig oft differenzierbar). \square

Das gezeichnete Verhalten ist typisch für die Polynominterpolation mit äquidistanten Knoten bei größerem n . Eine gewisse Erklärung hierfür gibt

Satz 2.21. Sei $f \in C^{n+1}[a, b]$, seien $x_j \in [a, b]$, $j = 0, \dots, n$, paarweise verschiedene Knoten, und sei $p \in \Pi_n$ das durch $p(x_j) = f(x_j)$, $j = 0, \dots, n$, bestimmte Interpolationspolynom. Dann gilt:

Zu jedem $x \in [a, b]$ existiert $\xi = \xi(x)$ aus dem kleinsten Intervall $I(x, x_0, \dots, x_n)$, das alle x_j und x enthält, so dass

$$f(x) - p(x) = \frac{\omega(x)}{(n+1)!} f^{(n+1)}(\xi), \quad (2.11)$$

gilt mit

$$\omega(x) = \prod_{i=0}^n (x - x_i) . \quad (2.12)$$

Bemerkung 2.22. ω hat für äquidistante Knoten x_j die Gestalt von $f - p_8$ der Skizze aus Abbildung 2.1. \square

Beweis: Für $x = x_j$, $j = 0, \dots, n$, ist die Aussage trivial.

Für festes $x \neq x_j$ betrachten wir die Funktion

$$F(z) := f(z) - p(z) - \alpha\omega(z) \quad (2.13)$$

und bestimmen $\alpha \in \mathbb{R}$ so, dass $F(x) = 0$ gilt.

Dann besitzt F in $I(x, x_0, \dots, x_n)$ wenigstens $n + 2$ Nullstellen. Nach dem Satz von Rolle besitzt F' dort wenigstens $n + 1$ Nullstellen, F'' wenigstens n Nullstellen, ... Schließlich hat $F^{(n+1)}$ mindestens eine Nullstelle $\xi \in I(x, x_0, \dots, x_n)$.

Wegen $p \in \Pi_n$ erhält man aus (2.13)

$$F^{(n+1)}(\xi) = f^{(n+1)}(\xi) - 0 - \alpha \cdot (n + 1)! = 0 .$$

Hiermit folgt wegen $F(x) = 0$ aus (2.13) die Behauptung. \blacksquare

Bemerkung 2.23. Aus Satz 2.21. erhält man die folgende Abschätzung für die Güte der Approximation einer Funktion durch das Interpolationspolynom

$$\|f - p\|_\infty := \max_{x \in [a, b]} |f(x) - p(x)| \leq \frac{K(f)}{(n + 1)!} \|\omega\|_\infty , \quad (2.14)$$

wobei

$$K(f) = \|f^{(n+1)}\|_\infty . \quad \square$$

Bemerkung 2.24. Man erhält ferner aus Satz 2.21. eine Darstellung der dividier- ten Differenzen.

Nimmt man im Newtonschen Interpolationspolynom in Satz 2.14. x als weitere Stützstelle mit dem Wert $f(x)$ hinzu, so erhält man für das in Satz 2.14. definierte Polynom p

$$f(x) - p(x) = [x_n, x_{n-1}, \dots, x_0, x] \prod_{i=0}^n (x - x_i) . \quad (2.15)$$

Durch Vergleich mit der Abschätzung (2.11) folgt

$$[x_n, \dots, x_0, x] = \frac{1}{(n+1)!} f^{(n+1)}(\xi),$$

und damit allgemein für die n -te dividierte Differenz

$$[x_0, \dots, x_n] = \frac{1}{n!} f^{(n)}(\xi) \quad \text{für ein } \xi \in I(x_0, \dots, x_n).$$

□

Bei äquidistanten Knoten

$$x_i := a + \frac{b-a}{n}i, \quad i = 0, \dots, n, \quad (2.16)$$

zeigt ω aus (2.11) einen Verlauf wie $f - p_8$ in Beispiel 2.20. Dies legt die Idee nahe, die Knoten am Rande des Intervalls dichter zu legen.

Als "beste" Wahl (vgl. Satz 2.25.) erweisen sich die **Chebyshev Knoten**

$$x_i := \frac{b-a}{2} \cos\left(\frac{2i+1}{2n+2}\pi\right) + \frac{a+b}{2}, \quad i = 0, \dots, n. \quad (2.17)$$

Zur näheren Untersuchung nehmen wir $a = -1$ und $b = 1$ an und betrachten die Funktionen

$$T_n(x) := \cos(n \cdot \arccos x), \quad -1 \leq x \leq 1, \quad n \in \mathbb{N}_0. \quad (2.18)$$

T_n ist ein Polynom vom Grade n , das n -te **Chebyshev Polynom**, denn aus

$$\cos((n+1)z) = 2 \cos z \cos(nz) - \cos((n-1)z)$$

und aus (2.18) liest man sofort ab, dass die Chebyshev Polynome die folgende Rekursionsformel erfüllen:

$$\begin{aligned} T_0(x) &\equiv 1, T_1(x) = x, \\ T_{n+1}(x) &= 2xT_n(x) - T_{n-1}(x). \end{aligned} \quad (2.19)$$

Aus dieser Darstellung folgt, dass der Koeffizient bei x^{n+1} in T_{n+1} gleich 2^n ist.

(2.18) liefert, dass T_{n+1} in $[-1, 1]$ genau die Nullstellen

$$x_i = \cos\left(\frac{2i+1}{2n+2}\pi\right), \quad i = 0, \dots, n,$$

besitzt. Es gilt also mit diesen x_i

$$\omega(x) = \prod_{i=0}^n (x - x_i) = 2^{-n} T_{n+1}(x),$$

und aus der Darstellung (2.18) folgt

$$\|\omega\|_\infty = 2^{-n}.$$

Ferner erhält man aus (2.18), dass T_{n+1} in $[-1, 1]$ genau $(n+2)$ Extrema besitzt, in denen abwechselnd die Werte $+1$ und -1 angenommen werden.

Satz 2.25. *Es seien $x_0, \dots, x_n \in [a, b]$ paarweise verschiedene Knoten, und sei hiermit die Funktion ω wie in (2.12) definiert.*

Unter allen Knotenwahlen ist

$$\|\omega\|_\infty := \max_{x \in [a, b]} |\omega(x)|$$

für die Chebyshev Knoten (2.17) minimal.

Bemerkung 2.26. Wegen Satz 2.25. ist die Abschätzung (2.14) für die Chebyshev-Knoten optimal. Hieraus kann man die Empfehlung ableiten, zur Polynominterpolation in der Regel die Chebyshev Knoten zu verwenden. \square

Beweis: Sei

$$\omega_0(x) := \prod_{i=0}^n (x - x_i)$$

mit den Chebyshev Knoten $x_i, i = 0, \dots, n$.

Wir nehmen an, dass es bessere Knoten $\xi_i, i = 0, \dots, n$, gibt, d.h. dass für

$$\omega(x) := \prod_{i=0}^n (x - \xi_i)$$

$$\max_{x \in [a, b]} |\omega(x)| < \max_{x \in [a, b]} |\omega_0(x)| \quad (2.20)$$

gelte

Da ω und ω_0 beide den führenden Koeffizienten 1 besitzen, ist

$$p := \omega - \omega_0 \in \Pi_n .$$

Es seien $\eta_j, j = 0, \dots, n+1$, die der Größe nach geordneten Extremwerte von ω_0 in $[a, b]$ ($\eta_0 = a, \eta_{n+1} = b, \eta_1, \dots, \eta_n$ relative Extrema). Dann gilt

$$\omega_0(\eta_j) + \omega_0(\eta_{j+1}) = 0, \quad j = 0, \dots, n,$$

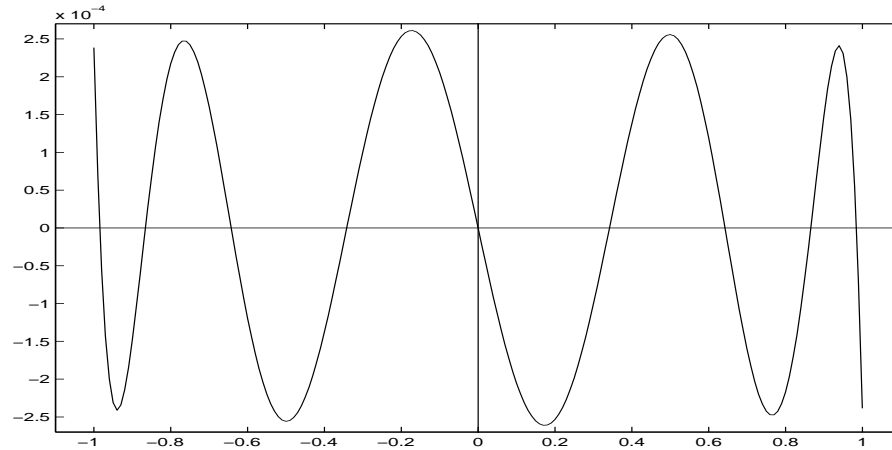


Abbildung 2.2 : Fehlerkurve (Chebyshev Knoten)

und

$$|\omega_0(\eta_j)| = \|\omega_0\|_\infty, \quad j = 0, \dots, n+1.$$

Wegen (2.20) hat p in den η_j wechselnde Vorzeichen, und daher liegt zwischen η_j und η_{j+1} , $j = 0, \dots, n$ nach dem Zwischenwertsatz eine Nullstelle von p , d.h. $p \in \Pi_n$ hat $n+1$ Nullstellen im Widerspruch zu $p \not\equiv 0$. ■

Für Chebyshev Knoten hat die Fehlerkurve des Interpolationspolynoms aus Beispiel 2.20. die ausgeglichene Gestalt aus Abbildung 2.2.

Für $f \in C[a, b]$ definieren wir

$$E_n(f) := \inf_{p \in \Pi_n} \|f - p\|_\infty, \quad (2.21)$$

als ein Maß dafür, wie gut die stetige Funktion f durch ein Polynom vom Höchstgrad n approximiert werden kann, und fragen, um wieviel schlechter die Approximation von f durch ein Interpolationspolynom zu vorgegebenen Knoten ist als dieser beste erreichbare Wert in Π_n . (Das Infimum in (2.21) wird übrigens angenommen, d.h. zu jedem $f \in C[a, b]$ gibt es ein $\hat{p} \in \Pi_n$ mit $\|f - \hat{p}\|_\infty = E_n(f)$).

Satz 2.27. Sei $f \in C[a, b]$ und $p_n \in \Pi_n$ mit

$$f(x_i) = p_n(x_i), \quad i = 0, \dots, n.$$

Dann gilt mit den Polynomen $\ell_j = \ell_j^n$ aus (2.2)

$$\|f - p_n\|_\infty \leq \left(1 + \sum_{j=0}^n \|\ell_j^n\|_\infty\right) \cdot E_n(f). \quad (2.22)$$

Beweis: Sicher gilt

$$p(x) = \sum_{j=0}^n p(x_j) \ell_j^n(x) \quad \text{für alle } p \in \Pi_n.$$

Sei $q_k \in \Pi_n$ eine Minimalfolge, d.h. es gelte $\|f - q_k\|_\infty \rightarrow E_n(f)$ für $k \rightarrow \infty$.

Dann gilt

$$\begin{aligned} p_n(x) - q_k(x) &= \sum_{j=0}^n (p_n(x_j) - q_k(x_j)) \ell_j^n(x) \\ &= \sum_{j=0}^n (f(x_j) - q_k(x_j)) \ell_j^n(x), \end{aligned}$$

und daher

$$\begin{aligned} \|f - p_n\|_\infty &\leq \|f - q_k\|_\infty + \|q_k - p_n\|_\infty \\ &\leq \|f - q_k\|_\infty + \sum_{j=0}^n \|f - q_k\|_\infty \|\ell_j^n\|_\infty \\ &= \left(1 + \sum_{j=0}^n \|\ell_j^n\|_\infty\right) \|f - q_k\|_\infty. \end{aligned}$$

Lässt man in dieser Ungleichung $k \rightarrow \infty$ gehen, so erhält man die Behauptung. ■

Für die Chebyshev Knoten (2.17) wurden die in (2.22) auftretenden “Überschätzungsfaktoren”

$$c_n := 1 + \sum_{j=0}^n \|\ell_j^n\|_\infty$$

von Powell (1967) berechnet. Es gilt

$$1.96 + \frac{2}{\pi} \ln(n+1) \leq c_n \leq 2 + \frac{2}{\pi} \ln(n+1)$$

(insbesondere also $c_n \leq 4$ für $n \leq 20$, $c_n \leq 5$ für $n \leq 100$), und man erhält durch Polynominterpolation an den Chebyshev Knoten gute Approximationen, wenn f genügend glatt ist.

Obwohl diese Überschätzungsfaktoren nur langsam anwachsen und obwohl man jede stetige Funktion durch Polynome beliebig gut approximieren kann (Satz von Weierstraß), kann man dennoch keine beliebig guten Approximationen durch Interpolation erwarten. Es gilt der überraschende

Satz 2.28. (Faber) *Zu jeder Folge von Zerlegungen*

$$\Delta_n : a \leq x_0^n < x_1^n < \dots < x_n^n \leq b$$

von $[a, b]$ mit

$$\max_{i=1, \dots, n} (x_i^n - x_{i-1}^n) \rightarrow 0 \quad \text{für } n \rightarrow \infty$$

gibt es ein $f \in C[a, b]$, so dass die Folge der zugehörigen Interpolationspolynome $p_n \in \Pi_n$, die f an den Knoten x_j^n interpolieren, nicht gleichmäßig gegen f konvergiert.

Beweis: s. Werner und Schaback [119], pp. 150 – 151.

2.2.4 Hermite Interpolation

Wir beschränken uns auf den Fall, dass alle Knoten die Vielfachheit 2 besitzen, betrachten also nur das folgende Problem:

Sei $f \in C^1[a, b]$, und es seien die Knoten $x_0, \dots, x_n \in [a, b]$ paarweise verschieden.

Bestimme $p \in \Pi_{2n+1}$ mit

$$\left. \begin{aligned} p(x_j) &= f(x_j) =: f_j, \\ p'(x_j) &= f'(x_j) =: f'_j \end{aligned} \right\} \quad j = 0, \dots, n. \quad (2.23)$$

Zur Bestimmung von p machen wir ähnlich wie bei der Lagrangeschen Interpolation den Ansatz

$$p(x) = \sum_{j=0}^n f_j \phi_j(x) + \sum_{j=0}^n f'_j \psi_j(x), \quad (2.24)$$

wobei $\phi_j, \psi_j \in \Pi_{2n+1}$ Hermitesche Interpolationspolynome sind, die die Bedingungen

$$\left. \begin{aligned} \phi_j(x_i) &= \delta_{ij}, & \phi'_j(x_i) &= 0 \\ \psi_j(x_i) &= 0, & \psi'_j(x_i) &= \delta_{ij} \end{aligned} \right\} \quad i, j = 0, 1, \dots, n \quad (2.25)$$

erfüllen.

Existieren solche ϕ_j und ψ_j , so ist p aus (2.24) das gesuchte Hermitesche Interpolationspolynom.

Man rechnet leicht nach, dass

$$\left. \begin{aligned} \phi_j(x) &:= \left(1 - 2\ell'_j(x_j)(x - x_j)\right) \ell_j^2(x) \\ \psi_j(x) &:= (x - x_j) \ell_j^2(x), \end{aligned} \right\} \quad (2.26)$$

wobei die ℓ_j wie in Satz 2.4. definiert sind, die Bedingungen (2.25) erfüllen.

Daher existiert ein Polynom $p \in \Pi_{2n+1}$ mit (2.23) .

Dieses ist eindeutig, denn angenommen $\tilde{p} \in \Pi_{2n+1}$ ist ein weiteres Polynom, das die Interpolationsbedingungen (2.23) erfüllt, so besitzt $p - \tilde{p} \in \Pi_{2n+1}$ die $n+1$ doppelten Nullstellen x_0, \dots, x_n , ist also identisch 0.

Damit ist gezeigt

Satz 2.29. *Das Hermite-Interpolationspolynom $p \in \Pi_{2n+1}$ mit den Eigenschaften ((2.23)) existiert und ist eindeutig bestimmt.*

Die Darstellung (2.24),(2.26) des Interpolationspolynoms p entspricht der Lagrangeschen Interpolationsformel in Satz 2.4.

Ähnlich wie für die Lagrange Interpolation kann man auch eine Newtonsche Interpolationsformel herleiten. Dazu müssen nur die dividierten Differenzen für mehrfache Knoten erklärt werden. Wir setzen

$$[x_0, x_0] := \lim_{x \rightarrow x_0} [x_0, x] = \lim_{x \rightarrow x_0} \frac{f(x_0) - f(x)}{x_0 - x} = f'(x_0).$$

Höhere dividierte Differenzen erhält man dann wie in (2.10), wobei mehrfache Knoten entsprechend ihrer Vielfachheit unmittelbar nacheinander behandelt werden müssen. Näheres findet man in Meinardus und März [78], pp. 57 ff.

Für den Fehler gilt (entsprechend Satz 2.21.):

Satz 2.30. *Sei $f \in C^{2n+2}[a, b]$, seien die Knoten $x_0, \dots, x_n \in [a, b]$ paarweise verschieden, und sei $p \in \Pi_{2n+1}$ das Hermitesche Interpolationspolynom, das den Bedingungen (2.23) genügt.*

Dann gibt es zu jedem $x \in [a, b]$ ein $\xi \in I(x, x_0, \dots, x_n)$ mit

$$f(x) - p(x) = \frac{\omega^2(x)}{(2n+2)!} f^{(2n+2)}(\xi). \quad (2.27)$$

Beweis: Für $x = x_i$ ist (2.27) trivial.

Für $x \neq x_i$ für alle $i = 0, \dots, n$ sei

$$h(z) := (f(x) - p(x))\omega^2(z) - (f(z) - p(z))\omega^2(x).$$

Dann besitzt $h \in C^{2n+2}[a, b]$ in jedem x_i eine doppelte Nullstelle und in x eine einfache Nullstelle.

$(2n + 2)$ -fache Anwendung des Satzes von Rolle liefert, dass $h^{(2n+2)}$ eine Nullstelle $\xi \in I(x, x_0, \dots, x_n)$ besitzt, und aus

$$h^{(2n+2)}(z) = (2n + 2)!(f(x) - p(x)) - f^{(2n+2)}(z)\omega^2(x)$$

folgt für $z = \xi$ die Behauptung. ■

2.3 Spline Interpolation

Die in den vorhergehenden Abschnitten behandelte Polynominterpolation ist zwar leicht durchführbar, hat aber den Nachteil, dass bei Verfeinerung der Zerlegung keine Konvergenz zu erwarten ist (vgl. Satz 2.28.). Bessere Konvergenzeigenschaften haben die nun zu besprechenden Spline-Funktionen.

Definition 2.31. Sei

$$\Delta : a := x_0 < x_1 < \dots < x_n =: b \tag{2.28}$$

eine Zerlegung des Intervalls $[a, b]$.

Dann bezeichnen wir mit $S(\Delta, p, q)$, $p, q \in \mathbb{N}_0$, $0 \leq q < p$, die Menge aller Funktionen $s \in C^q[a, b]$, die auf jedem Teilintervall $[x_{i-1}, x_i]$, $i = 1, \dots, n$, mit einem Polynom vom Höchstgrad p übereinstimmen.

Jedes $s \in S(\Delta, p, q)$ heißt (Polynom-)Spline vom Grade p der Differenzierbarkeitsklasse q zur Zerlegung Δ .

Am häufigsten treten in den Anwendungen (auf Randwertaufgaben) die Räume $S(\Delta, 3, 2)$ (**kubische Splines**) und $S(\Delta, 3, 1)$ (**kubische Hermite Splines**) auf.

Daneben untersucht man für spezielle Aufgabenstellungen, bei denen Pole oder verschiedenes Wachstum in verschiedenen Teilen des Intervalls erwartet wird (Grenzschichtprobleme), nichtlineare Splines (rationale oder exponentielle Splines).

Wir behandeln nur $S(\Delta, 3, 2)$ und $S(\Delta, 3, 1)$, sowie zur Motivation $S(\Delta, 1, 0)$.

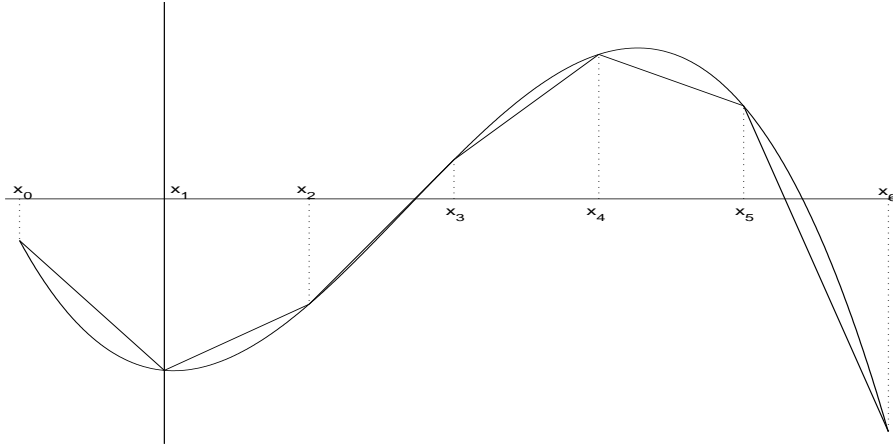


Abbildung 2.3: Stückweise lineare Interpolation

2.3.1 Stückweise lineare Funktionen

Es sei $\Delta : a = x_0 < x_1 < \dots < x_n = b$ eine gegebene Zerlegung des Intervalls $[a, b]$ und

$$S(\Delta, 1, 0) = \left\{ s \in C[a, b] : s|_{[x_{i-1}, x_i]} \in \Pi_1, i = 1, \dots, n \right\}$$

die Menge der stückweise linearen Funktionen auf $[a, b]$ zu dieser Zerlegung.

Für gegebene Interpolationsdaten $y_0, \dots, y_n \in \mathbb{R}$ gibt es offenbar genau einen interpolierenden Spline $s \in S(\Delta, 1, 0)$ mit $s(x_i) = y_i$, $i = 0, \dots, n$, und dieses s erhält man, indem man in jedem Teilintervall $[x_{i-1}, x_i]$ die Daten (x_{i-1}, y_{i-1}) und (x_i, y_i) nach Abschnitt 2.2 durch eine lineare Funktion interpoliert.

Man erhält in den Teilintervallen

$$s(x) = \frac{1}{x_i - x_{i-1}} (y_i(x - x_{i-1}) + y_{i-1}(x_i - x)), \quad x \in [x_{i-1}, x_i]. \quad (2.29)$$

Gilt $y_i = f(x_i)$ für eine Funktion $f \in C^2[a, b]$, so erhalten wir aus Satz 2.21. sofort für $x \in [x_{i-1}, x_i]$ den Fehler

$$f(x) - s(x) = \frac{1}{2}(x - x_{i-1})(x - x_i)f''(\xi_i), \quad \xi_i \in [x_{i-1}, x_i],$$

und daher

$$|f(x) - s(x)| \leq \frac{1}{8}(x_i - x_{i-1})^2 |f''(\xi_i)|.$$

Mit $|\Delta| := \max_{i=1, \dots, n} (x_i - x_{i-1})$ folgt

$$\|f - s\|_\infty \leq \frac{1}{8} |\Delta|^2 \|f''\|_\infty. \quad (2.30)$$

Ist also Δ_n irgendeine Zerlegungsfolge von $[a, b]$ mit

$$\lim_{n \rightarrow \infty} |\Delta_n| = 0$$

und $f \in C^2[a, b]$, so konvergiert die zugehörige Folge der interpolierenden Splines $s_n \in S(\Delta_n, 1, 0)$ gleichmäßig gegen f , und die Konvergenzgeschwindigkeit wird durch (2.30) beschrieben.

Für $f \in C^0[a, b]$ erhält man für $x \in [x_{i-1}, x_i]$ aus (2.29)

$$\begin{aligned} |f(x) - s(x)| &= \frac{1}{x_i - x_{i-1}} \left| (x - x_{i-1})(f(x) - f(x_i)) + (x_i - x)(f(x) - f(x_{i-1})) \right| \\ &\leq \frac{1}{x_i - x_{i-1}} \left((x - x_{i-1})|f(x) - f(x_i)| + (x_i - x)|f(x) - f(x_{i-1})| \right) \\ &\leq \max(|f(x) - f(x_i)|, |f(x) - f(x_{i-1})|), \end{aligned}$$

und daher folgt

$$\|f - s\|_\infty \leq \omega(f, |\Delta|),$$

wobei

$$\omega(f, h) := \sup \{|f(x) - f(y)| : x, y \in [a, b], |x - y| \leq h\}$$

den **Stetigkeitsmodul** von f zur Schrittweite h bezeichnet.

Ist Δ_n eine Zerlegungsfolge von $[a, b]$ mit $\lim_{n \rightarrow \infty} |\Delta_n| = 0$, so konvergieren auch für nur stetiges f die interpolierenden linearen Splines gleichmäßig gegen f .

Ähnlich wie bei der Lagrangeschen Interpolation können wir s darstellen als

$$s(x) = \sum_{i=0}^n f_i \phi_i(x), \quad x \in [a, b]$$

mit den Basisfunktionen

$$\phi_i(x) := \begin{cases} (x - x_{i-1}) / (x_i - x_{i-1}) & , \quad x \in [x_{i-1}, x_i] \\ (x_{i+1} - x) / (x_{i+1} - x_i) & , \quad x \in [x_i, x_{i+1}] \\ 0 & , \quad x \notin [x_{i-1}, x_{i+1}] \end{cases}, \quad i = 0, \dots, n,$$

wobei $x_{-1} < a$ und $x_{n+1} > b$ beliebig gewählt sind.

Die angegebenen ϕ_i heißen **Dachfunktionen** (engl.: hat functions). Sie besitzen einen lokalen Träger $[x_{i-1}, x_{i+1}]$. Will man also s an einer Stelle $x \in (x_{i-1}, x_i)$ auswerten, so hat man wegen $\phi_j(x) = 0$ für $j \notin \{i, i-1\}$ nur $\phi_i(x)$ und $\phi_{i-1}(x)$ zu berechnen (anders als bei den $\ell_j(x)$ in Satz 2.4.).

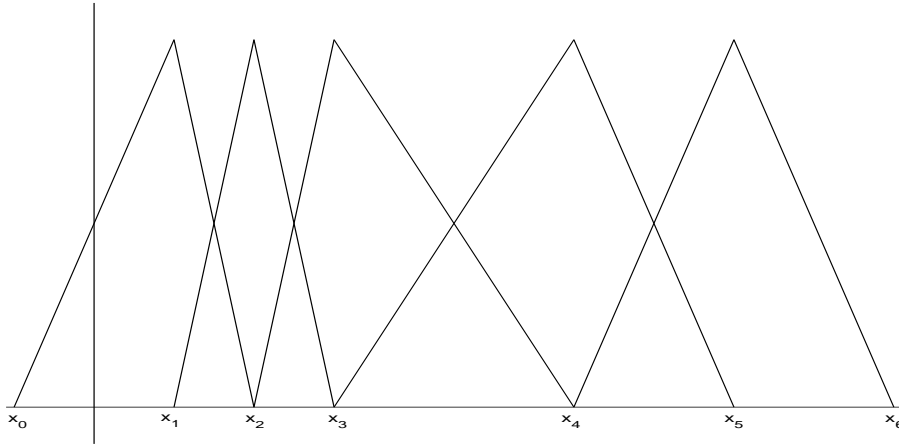


Abbildung 2.4: Dachfunktionen

2.3.2 Kubische Hermite Splines

Nach den Vorüberlegungen im letzten Unterabschnitt ist klar, wie man die Interpolationsaufgabe für kubische Hermite Splines zu stellen hat und welche Konvergenzeigenschaften man erwarten kann.

Es seien $y_0, \dots, y_n, y'_0, \dots, y'_n \in \mathbb{R}$ gegeben. Man bestimme $s \in S(\Delta, 3, 1)$ so, dass die Interpolationsbedingungen

$$s(x_i) = y_i, \quad s'(x_i) = y'_i, \quad i = 0, \dots, n, \quad (2.31)$$

erfüllt sind.

Dieses Problem können wir wie im letzten Unterabschnitt intervallweise behandeln. In jedem Teilintervall $[x_{i-1}, x_i]$ lösen wir die Hermitesche Interpolationsaufgabe (2.23) mit einem kubischen Polynom und den beiden Knoten x_{i-1} und x_i . Die aus diesen Interpolierenden zusammengesetzte Funktion s ist dann sicher in $S(\Delta, 3, 1)$ und erfüllt alle Interpolationsbedingungen.

Die Approximationseigenschaften ergeben sich aus Satz 2.30.

Ist $f \in C^4[a, b]$, so gilt für $x \in [x_{i-1}, x_i]$

$$f(x) - s(x) = \frac{1}{4!} ((x - x_i)(x - x_{i-1}))^2 |f^{(4)}(\xi)|, \quad \xi \in [x_{i-1}, x_i].$$

Durch Ausrechnen des Maximums erhält man

$$|f(x) - s(x)| \leq \frac{1}{4!} \cdot \frac{1}{16} (x_i - x_{i-1})^4 |f^{(4)}(\xi)|,$$

und daher

$$\|f - s\|_\infty \leq \frac{1}{384} \cdot |\Delta|^4 \cdot \|f^{(4)}\|_\infty. \quad (2.32)$$

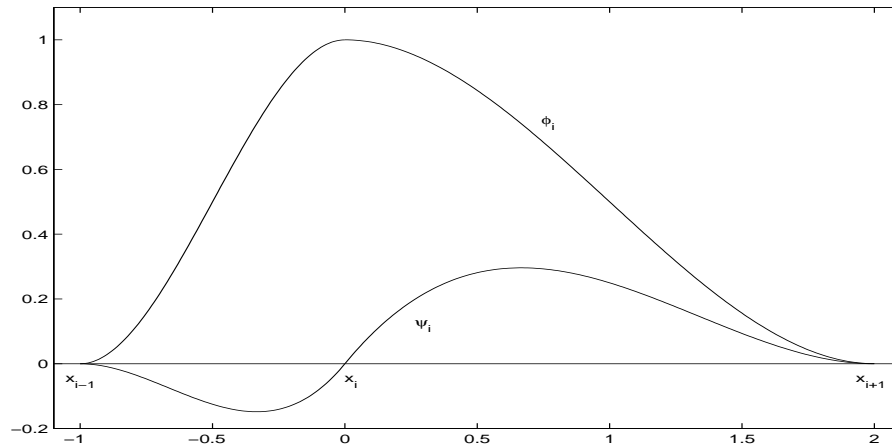


Abbildung 2.5: Kubische Hermite Splines (Basisfunktionen)

Wir erhalten also für $f \in C^4[a, b]$ gleichmäßige Konvergenz für jede Zerlegungsfolge Δ_n mit $|\Delta_n| \rightarrow 0$, die nun aber von der Ordnung 4 ist.

Auch zu den kubischen Hermite Splines existiert eine lokale Basis. Man rechnet leicht nach, dass für $i = 0, \dots, n$ (mit beliebig gewählten $x_{-1} < a$ und $x_{n+1} > b$),

$$\phi_i(x) := \begin{cases} (x - x_{i-1})^2(3x_i - x_{i-1} - 2x) / (x_i - x_{i-1})^3 & , x \in [x_{i-1}, x_i] \\ (x_{i+1} - x)^2(x_{i+1} - 3x_i + 2x) / (x_{i+1} - x_i)^3 & , x \in [x_i, x_{i+1}] \\ 0 & , x \notin [x_{i-1}, x_{i+1}] \end{cases}$$

$$\psi_i(x) := \begin{cases} (x - x_{i-1})^2(x - x_i) / (x_i - x_{i-1})^2 & , x \in [x_{i-1}, x_i] \\ (x - x_{i+1})^2(x - x_i) / (x_{i+1} - x_i)^2 & , x \in [x_i, x_{i+1}] \\ 0 & , x \notin [x_{i-1}, x_{i+1}] \end{cases}$$

kubische Hermite Splines sind, die die speziellen Interpolationsbedingungen

$$\left. \begin{aligned} \phi_i(x_j) &= \delta_{ij} & , \quad \phi_i'(x_j) &= 0 \\ \psi_i(x_j) &= 0 & , \quad \psi_i'(x_j) &= \delta_{ij} \end{aligned} \right\} i, j = 0, \dots, n,$$

erfüllen.

Der interpolierende kubische Hermite Spline ist daher gegeben durch

$$s(x) = \sum_{i=0}^n (f_i \phi_i(x) + f_i' \psi_i(x)) .$$

Jedes ϕ_i und ψ_i hat den (lokalen) Träger $[x_{i-1}, x_{i+1}]$, so dass man für die Berechnung von $s(x)$ für $x \in (x_{i-1}, x_i)$ nur die vier Funktionen $\phi_{i-1}(x)$, $\psi_{i-1}(x)$, $\phi_i(x)$ und $\psi_i(x)$ auszuwerten hat.

2.3.3 Kubische Splines

Bei den kubischen Splines $s \in S(\Delta, 3, 2)$ sind die Verhältnisse nicht ganz so einfach wie in den Fällen $S(\Delta, 3, 1)$ und $S(\Delta, 1, 0)$, da man die Interpolationsaufgabe nicht in jedem Teilintervall getrennt ausführen kann.

$s \in S(\Delta, 3, 2)$ ist in jedem Teilintervall $[x_{i-1}, x_i]$ ein Polynom dritten Grades, also ist s durch $4n$ Parameter bestimmt. Durch die Forderung $s \in C^2[a, b]$ werden in jedem inneren Knoten x_i , $i = 1, \dots, n-1$, drei Bedingungen gegeben:

$$s^{(j)}(x_i - 0) = s^{(j)}(x_i + 0), \quad j = 0, 1, 2.$$

Daher besitzt ein kubischer Spline noch (wenigstens) $n + 3$ Freiheitsgrade, und wir können nicht erwarten, dass s durch die $n + 1$ Interpolationsbedingungen

$$s(x_i) = y_i, \quad i = 0, \dots, n,$$

festgelegt ist, sondern es müssen noch 2 “Randbedingungen” hinzugenommen werden. Dies kann (je nach Aufgabenstellung) auf verschiedene Weise geschehen.

Satz 2.32. *Es sei Δ eine Zerlegung von $[a, b]$, und es seien $y_0, \dots, y_n \in \mathbb{R}$ gegeben.*

Dann gibt es für jede der vier folgenden Randbedingungen genau einen interpolierenden kubischen Spline $s \in S(\Delta, 3, 2)$ mit

$$s(x_j) = y_j, \quad j = 0, \dots, n. \tag{2.33}$$

$$(i) \quad s'(x_0) = y'_0, \quad s'(x_n) = y'_n \quad (y'_0, y'_n \in \mathbb{R} \text{ gegeben})$$

$$(ii) \quad s'(x_0) = s'(x_n), \quad s''(x_0) = s''(x_n).$$

$$(iii) \quad s''(x_0) = s''(x_n) = 0.$$

$$(iv) \quad s''(x_0) = y''_0, \quad s''(x_n) = y''_n \quad (y''_0, y''_n \in \mathbb{R} \text{ gegeben})$$

Bemerkung 2.33. Wird die Interpolation mit kubischen Splines verwendet, um eine Funktion $f : [a, b] \rightarrow \mathbb{R}$ zu approximieren, so wird man die Randbedingungen (i) verwenden, wenn die Ableitung von f in den Randpunkten a und b des Intervalls bekannt sind, die Randbedingung (ii), wenn die Funktion f periodisch mit der Periode $b - a$ ist, und die Randbedingung (iv), wenn zusätzlich zu den Lagrangeschen Interpolationsdaten am Rand des Intervalls die zweiten Ableitungen bekannt sind.

□

Bemerkung 2.34. Erfüllt $s \in S(\Delta, 3, 2)$ die Randbedingung (iii), so kann man s auf $\{x : x < x_0\}$ bzw. $\{x : x > x_n\}$ zweimal stetig differenzierbar als lineare Funktion fortsetzen. Kubische Splines, die man auf diese Weise erhält, heißen **natürliche Splines**. \square

Bemerkung 2.35. In de Boor [22] werden vier weitere Randbedingungen diskutiert, unter denen die Existenz und Eindeutigkeit des interpolierenden Splines gesichert ist. Unter ihnen besonders wichtig ist die sog. **not-a-knot Bedingung**. Diese wird verwendet, wenn nichts über das Verhalten der interpolierenden Funktion an den Rändern des Intervalls (außer den Funktionswerten) bekannt ist. Man wählt dann als Knoten für den Spline die Punkte $x_0 < x_2 < x_3 < \dots < x_{n-3} < x_{n-2} < x_n$. Ein Spline zu diesen $n - 2$ Intervallen hat $n + 1$ Freiheitsgrade und ist durch die $n + 1$ Interpolationsbedingungen $s(x_j) = y_j, j = 0, \dots, n$ eindeutig bestimmt. \square

Beweis: (von Satz 2.32.)

Der Beweis ist konstruktiv, er liefert also ein Verfahren zur Berechnung der jeweiligen interpolierenden Splines.

Sei $h_{j+1} := x_{j+1} - x_j, j = 0, \dots, n - 1$, und $m_j := s''(x_j), j = 0, \dots, n$, die zweite Ableitung der gesuchten Splinefunktion an der Stelle x_j .

Wir wollen zeigen, dass der Vektor $\mathbf{m} := (m_0, \dots, m_n)^T$ für jede der vier Randbedingungen eindeutige Lösung eines linearen Gleichungssystems ist und dass man aus den m_j den Spline $s(x)$ an jeder Stelle $x \in [a, b]$ leicht errechnen kann.

s ist in jedem Teilintervall $[x_j, x_{j+1}]$ ein kubisches Polynom. Also ist s'' stückweise linear, und man kann s'' mit Hilfe der m_j so beschreiben (vgl. die Überlegungen in Abschnitt 2.3.1 für $S(\Delta, 1, 0)$).

$$s''(x) = \frac{1}{h_{j+1}} (m_j(x_{j+1} - x) + m_{j+1}(x - x_j)), \quad x \in [x_j, x_{j+1}].$$

Durch Integration erhält man für $x \in [x_j, x_{j+1}], j = 0, \dots, n - 1$,

$$s'(x) = -m_j \frac{(x_{j+1} - x)^2}{2h_{j+1}} + m_{j+1} \frac{(x - x_j)^2}{2h_{j+1}} + \alpha_j \quad (2.34)$$

und

$$s(x) = m_j \frac{(x_{j+1} - x)^3}{6h_{j+1}} + m_{j+1} \frac{(x - x_j)^3}{6h_{j+1}} + \alpha_j(x - x_j) + \beta_j. \quad (2.35)$$

Die Integrationskonstanten α_j und β_j erhält man aus den Interpolationsbedingungen

$$\begin{aligned} s(x_j) &= m_j \frac{h_{j+1}^2}{6} + \beta_j = y_j \\ s(x_{j+1}) &= m_{j+1} \frac{h_{j+1}^2}{6} + \alpha_j h_{j+1} + \beta_j = y_{j+1}, \end{aligned}$$

d.h.

$$\begin{aligned} \beta_j &= y_j - m_j \frac{h_{j+1}^2}{6} \\ \alpha_j &= \frac{y_{j+1} - y_j}{h_{j+1}} - \frac{h_{j+1}}{6} (m_{j+1} - m_j). \end{aligned} \quad (2.36)$$

Setzt man α_j und β_j aus (2.36) in (2.35) ein, so erhält man die gewünschte Darstellung von s in Abhängigkeit von den m_j .

$n - 1$ Bestimmungsgleichungen für die m_j erhält man, indem man die Stetigkeit von s' ausnutzt: Setzt man α_j aus (2.36) in (2.34) ein, so erhält man

$$\begin{aligned} s'(x) &= -m_j \frac{(x_{j+1} - x)^2}{2h_{j+1}} + m_{j+1} \frac{(x - x_j)^2}{2h_{j+1}} \\ &+ \frac{y_{j+1} - y_j}{h_{j+1}} - \frac{h_{j+1}}{6} (m_{j+1} - m_j), \quad x \in [x_j, x_{j+1}]. \end{aligned} \quad (2.37)$$

Also liefert die Forderung $s'(x_j - 0) = s'(x_j + 0)$

$$\frac{y_j - y_{j-1}}{h_j} + \frac{h_j}{3} m_j + \frac{h_j}{6} m_{j-1} = \frac{y_{j+1} - y_j}{h_{j+1}} - \frac{h_{j+1}}{3} m_j - \frac{h_{j+1}}{6} m_{j+1},$$

d.h. für $j = 1, \dots, n - 1$

$$\frac{h_j}{6} m_{j-1} + \frac{h_j + h_{j+1}}{3} m_j + \frac{h_{j+1}}{6} m_{j+1} = \frac{y_{j+1} - y_j}{h_{j+1}} - \frac{y_j - y_{j-1}}{h_j}. \quad (2.38)$$

Die restlichen beiden Bedingungen für die m_j erhält man aus den Randbedingungen (i), (ii), (iii) oder (iv).

Für die natürlichen Splines hat man in (2.37)

$$s''(a) = m_0 = 0 = m_n = s''(b)$$

zu setzen, im Fall (iv) die inhomogenen Gleichungen

$$m_0 = y_0'' \quad \text{und} \quad m_n = y_n''.$$

Im Fall (i) hat man wegen (2.37) das System (2.38) zu ergänzen durch

$$\begin{aligned} \frac{h_1}{3} m_0 + \frac{h_1}{6} m_1 &= \frac{y_1 - y_0}{h_1} - y_0', \\ \frac{h_n}{6} m_{n-1} + \frac{h_n}{3} m_n &= y_n' - \frac{y_n - y_{n-1}}{h_n}. \end{aligned} \quad (2.39)$$

Im Falle periodischer Randbedingungen gilt $m_0 = m_n$, und wegen $s'(a) = s'(b)$ erhält man aus (2.37)

$$\frac{h_1}{6}m_1 + \frac{h_n}{6}m_{n-1} + \frac{h_n + h_1}{3}m_0 = \frac{y_1 - y_0}{h_1} - \frac{y_n - y_{n-1}}{h_n}. \quad (2.40)$$

In jedem Fall ergeben sich also die m_j als Lösung eines linearen Gleichungssystems

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \quad (2.41)$$

wobei für alle Zeilen der Matrix \mathbf{A} gilt:

$$|a_{ii}| > \sum_{j \neq i} |a_{ij}|.$$

Die Koeffizientenmatrix ist also strikt diagonaldominant. Nach dem Satz von Gerschgorin ist sie dann regulär, und daher ist das Gleichungssystem (2.41) für jede rechte Seite \mathbf{b} eindeutig lösbar. ■

Bemerkung 2.36. Eine andere Möglichkeit, den interpolierenden Spline s zu berechnen, wird in de Boor [22] dargestellt. Es werden dort die Unbekannten $\eta_j := s'(x_j)$, $j = 0, \dots, n$, eingeführt.

Für $\boldsymbol{\eta} := (\eta_0, \dots, \eta_n) \in \mathbb{R}^{n+1}$ sei $s(x)$ der kubische Hermite Spline, der definiert ist durch $s(x_j) = y_j$, $s'(x_j) = \eta_j$, $j = 0, \dots, n$, d.h. mit den Basisfunktionen $\phi_i(x)$ und $\psi_i(x)$ aus Abschnitt 2.3.2 gilt für $x \in [x_{j-1}, x_j]$

$$s(x) = y_{j-1}\phi_{j-1}(x) + y_j\phi_j(x) + \eta_{j-1}\psi_{j-1}(x) + \eta_j\psi_j(x).$$

Es ist $s \in C^1[a, b]$, und die Forderung $s''(x_j - 0) = s''(x_j + 0)$, $j = 1, \dots, n - 1$, liefert das lineare Gleichungssystem

$$\frac{1}{h_j}\eta_{j-1} + 2\left(\frac{1}{h_j} + \frac{1}{h_{j+1}}\right)\eta_j + \frac{1}{h_{j+1}}\eta_{j+1} = \frac{3}{h_{j+1}^2}(y_{j+1} - y_j) + \frac{3}{h_j^2}(y_j - y_{j-1}),$$

$j = 1, \dots, n - 1$, das für die Randbedingungen (i) (Vorgabe der Ableitungen am Rande) ergänzt wird um die Gleichungen

$$s'(x_0) = y'_0, \quad s'(x_n) = y'_n,$$

für die Randbedingungen (ii) (Periodizität) um

$$\begin{aligned} \eta_0 &= s'(x_0 + 0) = s'(x_n - 0) = \eta_n, \\ 2\left(\frac{1}{h_1} + \frac{1}{h_n}\right)\eta_0 + \frac{1}{h_1}\eta_1 + \frac{1}{h_n}\eta_{n-1} &= \frac{3}{h_1^2}(y_1 - y_0) + \frac{3}{h_n^2}(y_n - y_{n-1}), \end{aligned}$$

und für den natürlichen Spline um

$$\frac{2}{h_1}\eta_0 + \frac{1}{h_1}\eta_1 = \frac{3}{h_1^2}(y_1 - y_0), \quad \frac{2}{h_n}\eta_n + \frac{1}{h_n}\eta_{n-1} = \frac{3}{h_n^2}(y_n - y_{n-1}).$$

In jedem Fall erhält man (wie im Beweis von Satz 2.32.) ein lineares Gleichungssystem mit diagonaldominanter Koeffizientenmatrix.

Beide Zugänge erfordern denselben Rechenaufwand. Wir haben den Zugang über die zweiten Ableitungen gewählt, da wir diese Darstellung in dem folgenden Satz bei der Herleitung der Fehlerabschätzung benötigen. \square

Die Approximationseigenschaft der interpolierenden kubischen Splines wird beschrieben durch

Satz 2.37. *Sei $f \in C^3[a, b]$ und sei $s \in S(\Delta, 3, 2)$ der interpolierende Spline, der eine der Randbedingungen (i) oder (ii) oder $s''(a) = f''(a)$, $s''(b) = f''(b)$ erfüllt.*

Dann gilt mit den Konstanten $C_0 = \frac{9}{8}$ und $C_1 = C_2 = \frac{9}{4}$

$$\|s^{(\nu)} - f^{(\nu)}\|_{\infty} \leq C_{\nu} |\Delta|^{3-\nu} \omega(f''', |\Delta|), \quad \nu = 0, 1, 2, \quad (2.42)$$

wobei

$$\omega(g, h) := \sup \{|g(x) - g(y)| : x, y \in [a, b], |x - y| \leq h\}$$

den Stetigkeitsmodul von g bezeichnet.

Genügt f''' einer Lipschitzbedingung

$$|f'''(x) - f'''(y)| \leq L|x - y| \quad \text{für alle } x, y \in [a, b],$$

so gilt

$$\|s^{(\nu)} - f^{(\nu)}\|_{\infty} \leq L \cdot C_{\nu} |\Delta|^{4-\nu}, \quad \nu = 0, 1, 2. \quad (2.43)$$

Beweis: Wir beweisen den Satz nur für die Randbedingung

$$s'(a) = f'(a), \quad s'(b) = f'(b).$$

die anderen Randbedingungen erfordern nur eine leichte Modifikation des Beweises.

Wir bezeichnen wieder mit $m_j := s''(x_j)$ die zweiten Ableitungen von s in den Knoten. Dann gilt (vgl. (2.38) und (2.39))

$$\frac{h_1}{3}m_0 + \frac{h_1}{6}m_1 = \frac{f(x_1) - f(x_0)}{h_1} - f'(x_0), \quad (2.44)$$

$$\frac{h_j}{6}m_{j-1} + \frac{h_j + h_{j+1}}{3}m_j + \frac{h_{j+1}}{6}m_{j+1} = \frac{f(x_{j+1}) - f(x_j)}{h_{j+1}} - \frac{f(x_j) - f(x_{j-1}))}{h_j}. \quad (2.45)$$

$$\frac{h_n}{6}m_{n-1} + \frac{h_n}{3}m_n = f'(x_n) - \frac{f(x_n) - f(x_{n-1}))}{h_n}. \quad (2.46)$$

Es seien $z_j := m_j - f''(x_j)$, $j = 0, \dots, n$, und

$$\rho := \max_{j=0, \dots, n} |z_j|.$$

Das Maximum werde für $k \in \{0, \dots, n\}$ angenommen, d.h. $\rho = |z_k|$.

Wir nehmen zunächst an, dass $k \in \{1, \dots, n-1\}$ gilt. Dann erhält man aus (2.45)

$$\frac{h_k}{6}z_{k-1} + \frac{h_k + h_{k+1}}{3}z_k + \frac{h_{k+1}}{6}z_{k+1} = \zeta_k \quad (2.47)$$

mit

$$\begin{aligned} \zeta_k &:= \frac{f(x_{k+1}) - f(x_k)}{h_{k+1}} - \frac{f(x_k) - f(x_{k-1}))}{h_k} - \frac{h_k}{6}f''(x_{k-1}) \\ &\quad - \frac{h_k + h_{k+1}}{3}f''(x_k) - \frac{h_{k+1}}{6}f''(x_{k+1}). \end{aligned} \quad (2.48)$$

Nach dem Taylorschen Satz gilt mit $\xi_1, \xi_2 \in (x_k, x_{k+1})$

$$\begin{aligned} &\frac{1}{h_{k+1}} \left(f(x_{k+1}) - f(x_k) - \frac{1}{3}h_{k+1}^2 f''(x_k) - \frac{1}{6}h_{k+1}^2 f''(x_{k+1}) \right) \\ &= \frac{1}{h_{k+1}} \left(f(x_k) + h_{k+1}f'(x_k) + \frac{1}{2}h_{k+1}^2 f''(x_k) + \frac{1}{6}h_{k+1}^3 f'''(\xi_1) \right. \\ &\quad \left. - f(x_k) - \frac{1}{3}h_{k+1}^2 f''(x_k) - \frac{1}{6}h_{k+1}^2 f''(x_{k+1}) \right) \\ &= f'(x_k) + \frac{1}{6}h_{k+1}^2 \left(\frac{f''(x_k) - f''(x_{k+1}))}{h_{k+1}} + f'''(\xi_1) \right) \\ &= f'(x_k) + \frac{1}{6}h_{k+1}^2 (f'''(\xi_1) - f'''(\xi_2)), \end{aligned}$$

und genauso mit $\xi_3, \xi_4 \in (x_{k-1}, x_k)$

$$\begin{aligned} &\frac{1}{h_k} \left(f(x_{k-1}) - f(x_k) - \frac{1}{3}h_k^2 f''(x_k) - \frac{1}{6}h_k^2 f''(x_{k-1}) \right) \\ &= -f'(x_k) + \frac{1}{6}h_k^2 (f'''(\xi_3) - f'''(\xi_4)). \end{aligned}$$

Zusammen folgt

$$|\zeta_k| \leq \frac{h_k^2 + h_{k+1}^2}{6} \omega(f''', |\Delta|).$$

Aus (2.47) erhält man

$$|\zeta_k| \geq -\frac{h_k}{6}|z_{k-1}| + \frac{h_k + h_{k+1}}{3}|z_k| - \frac{h_{k+1}}{6}|z_{k+1}| \geq \frac{h_k + h_{k+1}}{6} \rho,$$

und wegen $h_k^2 + h_{k+1}^2 \leq (h_k + h_{k+1})^2$ gilt daher

$$\rho \leq (h_k + h_{k+1}) \omega(f''', |\Delta|) \leq 2|\Delta| \omega(f''', |\Delta|). \quad (2.49)$$

Die äußere Ungleichung in (2.49) gilt auch für den Fall $k = 0$ oder $k = n$, denn z.B. folgt für $\rho = |z_0|$ aus (2.44) wie eben aus dem Satz von Taylor mit $\xi_1, \xi_2 \in (x_0, x_1)$

$$\begin{aligned} \zeta_0 &:= \frac{h_1}{3} z_0 + \frac{h_1}{6} z_1 = \frac{f(x_1) - f(x_0)}{h_1} - f'(x_0) - \frac{h_1}{3} f''(x_0) - \frac{h_1}{6} f''(x_1) \\ &= \frac{h_1^2}{6} (f'''(\xi_1) - f'''(\xi_2)), \end{aligned}$$

d.h.

$$|\zeta_0| \leq \frac{h_1^2}{6} \omega(f''', |\Delta|),$$

und wegen

$$|\zeta_0| \geq \frac{h_1}{6} \rho$$

gilt auch in diesem Fall (2.49).

Aus der Ungleichung (2.49) folgt für alle $j = 0, \dots, n$

$$|m_j - f''(x_j)| = |z_j| \leq 2|\Delta| \omega(f''', |\Delta|). \quad (2.50)$$

Sei $x \in [x_{j-1}, x_j]$. Dann gilt wegen der Linearität von s'' in $[x_{j-1}, x_j]$ mit Zwischenpunkten $\sigma_j, \tau_j \in (x_{j-1}, x_j)$

$$\begin{aligned} s''(x) - f''(x) &= \frac{x - x_{j-1}}{h_j} (z_j + f''(x_j) - f''(x)) + \frac{x_j - x}{h_j} (z_{j-1} + f''(x_{j-1}) - f''(x)) \\ &= \frac{x - x_{j-1}}{h_j} z_j + \frac{x_j - x}{h_j} z_{j-1} + \frac{(x_j - x)(x - x_{j-1})}{h_j} (f'''(\sigma_j) - f'''(\tau_j)). \end{aligned}$$

Zusammen mit (2.50) folgt also

$$|s''(x) - f''(x)| \leq 2|\Delta| \omega(f''', |\Delta|) + \frac{1}{4} |\Delta| \omega(f''', |\Delta|) = \frac{9}{4} |\Delta| \omega(f''', |\Delta|), \quad (2.51)$$

und dies ist die behauptete Abschätzung (2.42) für den Fall $\nu = 2$.

Nach dem Satz von Rolle existieren auf Grund der Interpolationsbedingungen $s(x_j) = f(x_j)$, $j = 0, \dots, n$, für $i = 1, \dots, n$ Zahlen $\xi_i \in (x_{i-1}, x_i)$ mit $s'(\xi_i) = f'(\xi_i)$, und zu jedem $x \in [a, b]$ gibt es ein derartiges ξ_i mit $|x - \xi_i| \leq |\Delta|$. Daher folgt die Abschätzung (2.42) für den Fall $\nu = 1$ aus

$$s'(x) - f'(x) = \int_{\xi_i}^x (s''(t) - f''(t)) dt.$$

Da es schließlich zu jedem $x \in [a, b]$ ein x_i mit $|x - x_i| \leq \frac{|\Delta|}{2}$ gilt, erhält man (2.42) für $\nu = 0$ aus

$$s(x) - f(x) = \int_{x_i}^x (s'(t) - f'(t)) dt.$$

Ist f''' Lipschitz stetig, so folgt offenbar (2.43) aus (2.42) ■

Bemerkung 2.38. Der hier angegebene, sehr elementare Beweis geht auf **J.W. Schmidt** (ZAMM 58 (1978)) zurück. Die Konstanten C_ν in den Abschätzungen sind nicht optimal. Tatsächlich gelten z.B. für $f \in C^4[a, b]$ und die vollständige Spline Interpolation s (Randbedingung (i)) die Abschätzungen (vgl. **Hall & Meyer** (J.Appr.Theory 16 (1976)))

$$\begin{aligned} \|f - s\|_\infty &\leq \frac{5}{384} |\Delta|^4 \|f^{(4)}\|_\infty, \\ \|f' - s'\|_\infty &\leq \frac{1}{24} |\Delta|^3 \|f^{(4)}\|_\infty, \\ \|f'' - s''\|_\infty &\leq \frac{3}{8} |\Delta|^2 \|f^{(4)}\|_\infty, \end{aligned}$$

und die hierin auftretenden Konstanten können nicht verbessert werden. □

Bemerkung 2.39. Satz 2.37. zeigt, dass Spline Interpolationen geeignet sind, um Ableitungen von Funktionen, von denen nur diskrete Werte bekannt sind, zu approximieren. □

Auch der Raum der kubischen Splines $S(\Delta, 3, 2)$ besitzt eine lokale Basis.

Seien $x_{-3} < x_{-2} < x_{-1} < a$ und $b < x_{n+1} < x_{n+2} < x_{n+3}$ zusätzliche Knoten. Dann überzeugt man sich leicht (aber mit Hilfe einer längeren Rechnung), dass die Funktionen

$$\phi_i(x) = \begin{cases} (x - x_{i-2})_+^3 \prod_{j=i-1}^{i+2} (x_j - x_{i-2}) + (x - x_{i-1})_+^3 \prod_{\substack{j=i-2 \\ j \neq i-1}}^{i+2} (x_j - x_{i-1}) & , \quad x \leq x_i \\ (x_{i+2} - x)_+^3 \prod_{j=i-2}^{i+1} (x_{i+2} - x_j) + (x_{i+1} - x)_+^3 \prod_{\substack{j=i-2 \\ j \neq i+1}}^{i+2} (x_{i+1} - x_j) & , \quad x \geq x_i \end{cases}$$

für $i = -1, \dots, n+1$, eine Basis von $S(\Delta, 3, 2)$ bilden. Dabei bezeichnet $(x)_+$ die Funktion

$$(x)_+ := \begin{cases} x & \text{für } x \geq 0 \\ 0 & \text{für } x \leq 0. \end{cases}$$

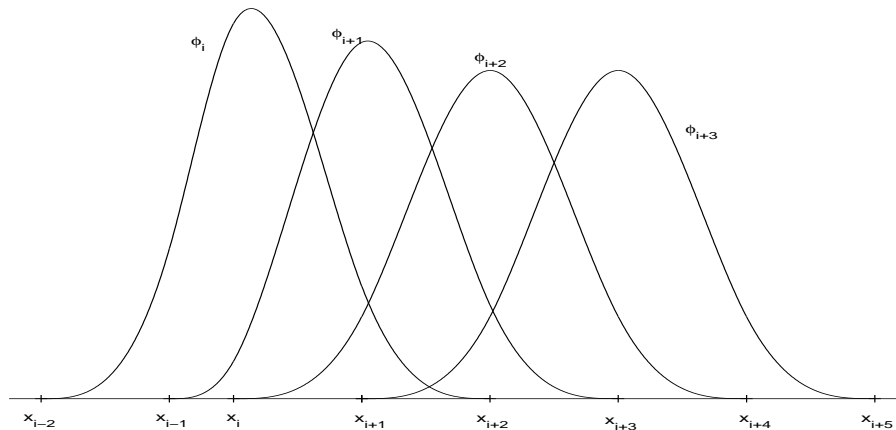


Abbildung 2.6: B-Splines

Die Basisfunktionen ϕ_j heißen **B-Splines**. Abbildung 2.6 enthält die Graphen einiger B-Splines.

Jeder B-Spline ist auf 4 Teilintervallen nichttrivial. Man kann zeigen, dass es keine Basis von $S(\Delta, 3, 2)$ mit kleinerem Träger gibt.

Man kann mit diesen ϕ_i den Ansatz

$$s(x) = \sum_{i=-1}^{n+1} c_i \phi_i(x)$$

machen, und zur Lösung des Interpolationsproblems das lineare Gleichungssystem

$$s(x_i) = y_i + \text{Randbedingungen}$$

behandeln. Dieses besitzt ähnlich wie das für die M_i wieder eine tridiagonale, diagonaldominante Koeffizientenmatrix, ist also problemlos lösbar.

Nicht benutzen sollte man jedoch für numerische Zwecke die folgende Basis von $S(\Delta, 3, 2)$, die bisweilen in Büchern angegeben ist:

$$1, x, x^2, x^3, (x - x_i)_+^3, \quad i = 1, \dots, n - 1,$$

wegen der schlechten Kondition des entstehenden linearen Gleichungssystems.

MATLAB stellt die Funktion `yy=spline(x,y,xx)` zur Verfügung. Besitzen die Vektoren x und y gleiche Länge, so wird der interpolierende kubische Spline mit not-a-knot Randbedingungen bestimmt. Ist die Länge von y um 2 größer als die Länge n von x , so werden die erste und letzte Komponente von y als Steigungen von s in $x(1)$ und $x(n)$ gedeutet. Der Vektor yy enthält in der j -ten Komponente $yy(j) = s(x(j))$.

Zusätzlich wird von MATLAB die Toolbox SPLINES angeboten. Hierin werden auch andere Randbedingungen für $S(\Delta, p, p-1)$, Glättung von Daten durch Ausgleich und Tensorprodukte von Splines zur Darstellung von Flächen angeboten.

2.4 Bezierkurven

Wir betrachten in diesem Abschnitt die Designaufgabe: Gegeben die "Idee von einer Kurve" (z.B. "α" für ein Textverarbeitungssystem). Bestimme (z.B. interaktiv an einem Rechner) eine Realisierung $\mathbf{x} : [0, 1] \rightarrow \mathbb{R}^n$ (meistens $n = 2$ oder $n = 3$), die leicht auswertbar ist und die durch Parameter festgelegt ist, die auf anschauliche Weise die Gestalt der Kurve beeinflussen.

Eine erste Möglichkeit ist, $m + 1$ Punkte $\mathbf{x}^i = (x_1^i, \dots, x_n^i)^T$ zu wählen, die nacheinander durchlaufen werden, diesen Punkten "Zeiten" $t_i \in [0, 1]$, $t_i < t_{i+1}$, zuzuordnen und komponentenweise durch Polynome zu interpolieren.

Bestimme also $p_j \in \Pi_m$ mit $p_j(t_i) = x_j^i$, $j = 1, \dots, n$, $i = 0, \dots, m$, und wähle als Realisierung $\mathbf{x}(t) = (p_1(t), \dots, p_n(t))^T$, $t \in [0, 1]$.

Diese Vorgehensweise hat zwei Nachteile:

1. Die Realisierung durch Interpolation ist sehr stark abhängig von der Wahl der t_i .

Als Beispiel wählen wir $n = 2$, $m = 2$, $\mathbf{x}^0 = (-1, 0)^T$, $\mathbf{x}^1 = (0, 1)^T$, $\mathbf{x}^2 = (1, 0)^T$, $t_0 = 0$ und $t_2 = 1$. Die Skizze auf der linken Seite in Abbildung 2.7 zeigt die interpolierenden Kurven für $t_1 = 0.5, 0.6, 0.7, 0.8$.

2. Die interpolierende Kurve hängt (für große m) sehr empfindlich von den Punkten \mathbf{x}^i ab.

Als Beispiel wählen wir $n = 2$, $m = 12$, $\mathbf{x}^i = \left(-1 + \frac{i}{6}, 1 - \left(x_1^i\right)^2\right)^T$, $t_i = \frac{i}{12}$, $i = 0, \dots, 12$.

Die Skizze auf der rechten Seite von Abbildung 2.7 enthält die interpolierende Kurve zu diesen Daten und zu Daten mit einer relativen Störung von 1%

Der folgende Ansatz wurde (unabhängig) von de Casteljau (1959) und Bezier (1962) entwickelt.

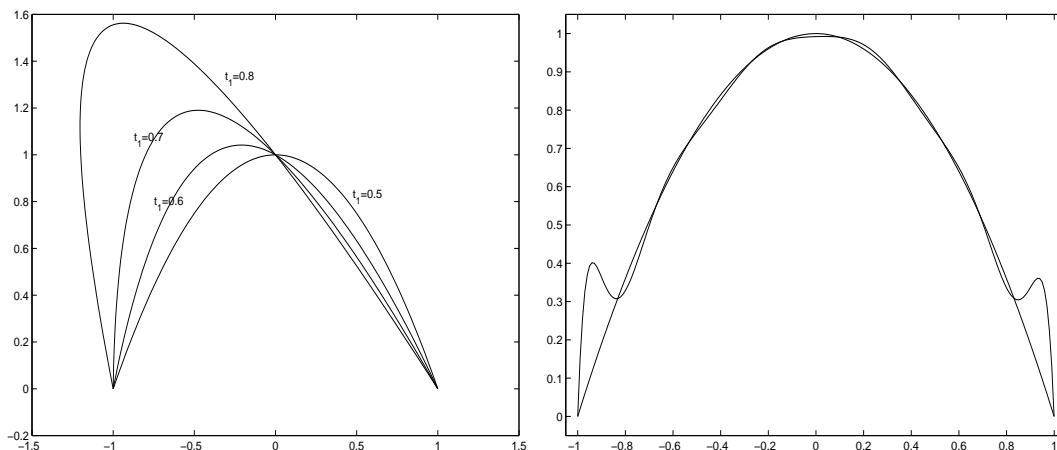


Abbildung 2.7: Kurvendarstellung durch Polynominterpolation

Definition 2.40. *Die Polynome*

$$B_i^m(t) := \binom{m}{i} t^i (1-t)^{m-i}, \quad t \in [0, 1], \quad i = 0, \dots, m,$$

heißen die **Bernstein Polynome** vom Grade m .

Definition 2.41. *Es seien die Punkte $\mathbf{x}^i \in \mathbb{R}^n$, $i = 0, \dots, m$, gegeben. Dann heißt*

$$\mathbf{F}(t) := \sum_{i=0}^m \mathbf{x}^i B_i^m(t).$$

die **Bezierkurve** vom Grade m zu den **Bezierpunkten** (auch **Kontrollpunkten**) \mathbf{x}^i .

Das durch die \mathbf{x}^i bestimmte Polynom heißt das zu F gehörende **Bezierpolygon** (oder **Kontrollpolygon**).

Abbildung 2.8 enthält die Bezierkurven vom Grade 5 mit den Kontrollpunkten $\mathbf{x}^0 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$, $\mathbf{x}^1 = \begin{pmatrix} 1 \\ 3 \end{pmatrix}$, $\mathbf{x}^2 = \begin{pmatrix} 3 \\ 3 \end{pmatrix}$, $\mathbf{x}^4 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$, $\mathbf{x}^5 = \begin{pmatrix} 0 \\ 4 \end{pmatrix}$ und $\mathbf{x}^3 = \begin{pmatrix} 4 \\ 0 \end{pmatrix}$ (bzw. $\mathbf{x}^3 = \begin{pmatrix} 9 \\ 0 \end{pmatrix}$ für die nicht durchgezogene Kurve).

Der folgende Satz enthält einige einfache Eigenschaften der Bernstein Polynome.

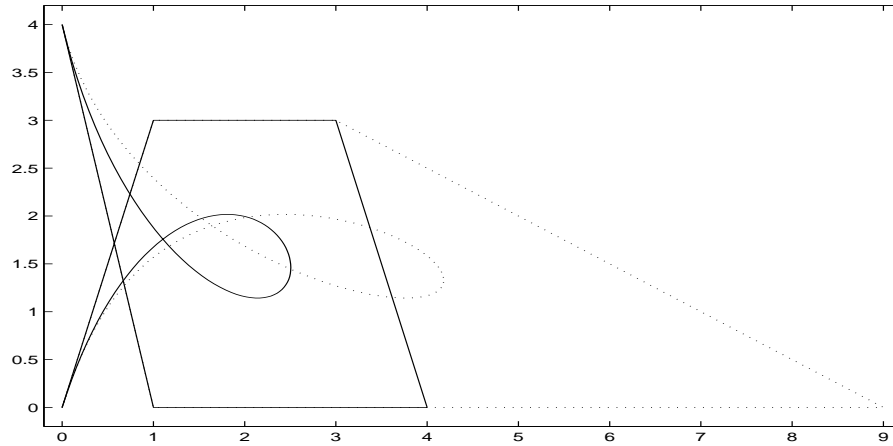


Abbildung 2.8: Bezierkurve

Satz 2.42. *Es gilt*

- (i) $B_i^m(t) > 0$ für alle $t \in (0, 1)$
- (ii) $\sum_{i=0}^m B_i^m(t) \equiv 1$,
- (iii) $B_i^m(t)$ hat eine i -fache Nullstelle in $t = 0$ und eine $(m - i)$ -fache Nullstelle in $t = 1$,
- (iv) Es gilt für alle m und $i = 0, \dots, m$

$$B_i^{m+1}(t) = (1 - t)B_i^m(t) + tB_{i-1}^m(t), \quad (2.52)$$

wobei $B_{-1}^m(t) \equiv 0$ gesetzt ist.

Beweis: Die Eigenschaften (i) und (iii) liest man unmittelbar aus der Darstellung der Bernstein Polynome ab.

(ii) folgt aus dem binomischen Satz, denn

$$\sum_{i=0}^m B_i^m(t) = \sum_{i=0}^m \binom{m}{i} t^i (1 - t)^{m-i} = (t + (1 - t))^m \equiv 1.$$

(iv) gilt wegen

$$\begin{aligned} B_i^{m+1}(t) &= \binom{m+1}{i} t^i (1 - t)^{m+1-i} \\ &= \left(\binom{m}{i} + \binom{m}{i-1} \right) t^i (1 - t)^{m+1-i} \\ &= (1 - t) \binom{m}{i} t^i (1 - t)^{m-i} + t \binom{m}{i-1} t^{i-1} (1 - t)^{m-(i-1)} \\ &= (1 - t) B_i^m(t) + t B_{i-1}^m(t). \end{aligned}$$

Bemerkung 2.43. Wegen (iii) sind die Bernstein Polynome linear unabhängig, denn aus

$$\phi(t) := \sum_{i=0}^m \alpha_i B_i^m(t) \equiv 0$$

folgt

$$\phi(0) = \sum_{i=0}^m \alpha_i B_i^m(0) = \alpha_0 = 0,$$

und daher

$$\phi(t) := \sum_{i=1}^m \alpha_i B_i^m(t) \equiv 0.$$

Genauso erhält man nun aus $\phi'(t) \equiv 0$, dass $\alpha_1 = 0$ gilt, und dann mit Hilfe der weiteren Ableitungen $\alpha_2 = \alpha_3 = \dots = \alpha_m = 0$.

B_i^m , $i = 0, \dots, m$, bilden also eine Basis des Polynomraumes Π_m . □

Bemerkung 2.44. Für die Bezierkurve $\mathbf{F}(t)$ gilt wegen (iii) $\mathbf{F}(0) = \mathbf{x}^0$ und $\mathbf{F}(1) = \mathbf{x}^m$. Der erste und letzte Kontrollpunkt liegen also auf der Kurve. Für die übrigen Bezierpunkte gilt dies i.a. nicht. □

Bemerkung 2.45. Wegen (i) und (ii) ist $\mathbf{F}(t)$ für jedes t eine Konvexkombination der \mathbf{x}^i , $i = 0, \dots, m$. Die Bezierkurve verläuft also ganz in der konvexen Hülle der Bezierpunkte. □

Die Rekursionsformel (2.52) liefert eine einfache Methode zur Auswertung des Bezierpolynoms an einer festen Stelle, den **Algorithmus von de Casteljau**.

Satz 2.46. Sei $\hat{t} \in [0, 1]$ fest.

Wir setzen $\mathbf{x}_0^i := \mathbf{x}^i$, $i = 0, \dots, m$, und berechnen

$$\mathbf{x}_{k+1}^i := (1 - \hat{t})\mathbf{x}_k^{i-1} + \hat{t}\mathbf{x}_k^i, \quad k = 0, \dots, m-1, \quad i = k+1, \dots, m.$$

Dann gilt

$$\mathbf{x}_m^m = \mathbf{F}(\hat{t}).$$

Beweis: Für $m = 1$ gilt

$$\mathbf{x}_1^1 = (1 - \hat{t})\mathbf{x}_0^0 + \hat{t}\mathbf{x}_0^1 = \mathbf{x}^0 B_0^1(\hat{t}) + \mathbf{x}^1 B_1^1(\hat{t}) = \mathbf{F}(\hat{t}).$$

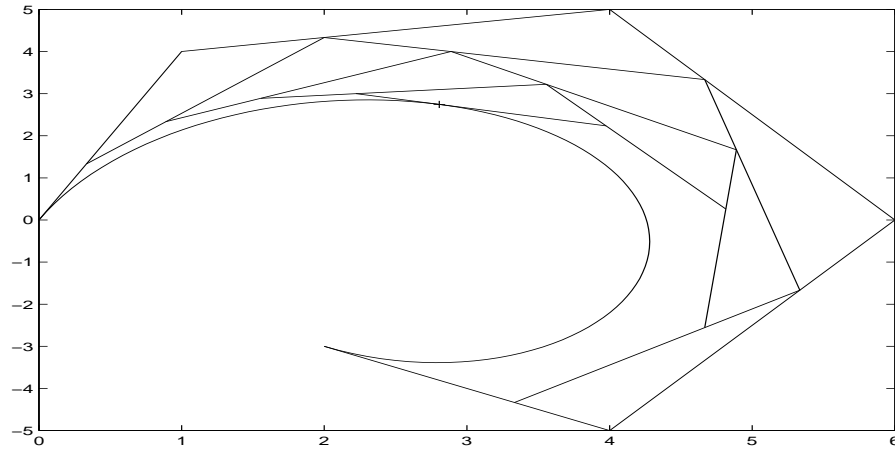


Abbildung 2.9: Konstruktion nach Satz 2.46.

Ist $m \geq 2$ und die Behauptung für $m - 1$ schon bewiesen, so folgt

$$\begin{aligned}
 \mathbf{x}_m^m &= (1 - \hat{t})\mathbf{x}_{m-1}^{m-1} + \hat{t}\mathbf{x}_{m-1}^m \\
 &= (1 - \hat{t}) \sum_{i=0}^{m-1} \mathbf{x}^i B_i^{m-1}(\hat{t}) + \hat{t} \sum_{i=0}^{m-1} \mathbf{x}^{i+1} B_i^{m-1}(\hat{t}) \\
 &= (1 - \hat{t})^m \mathbf{x}^0 + \sum_{i=1}^{m-1} \mathbf{x}^i \left((1 - \hat{t}) B_i^{m-1}(\hat{t}) + \hat{t} B_{i-1}^{m-1}(\hat{t}) \right) + \hat{t}^m \mathbf{x}^m \\
 &= \sum_{i=0}^m \mathbf{x}^i B_i^m(\hat{t}) = \mathbf{F}(\hat{t}). \quad \blacksquare
 \end{aligned}$$

Geometrisch entspricht der Algorithmus von de Casteljau der Konstruktion in Abbildung 2.9, die den Fall $\hat{t} = \frac{1}{3}$ enthält.

Kapitel 3

Numerische Integration

In vielen Fällen ist es nicht möglich, ein gegebenes Integral $\int_a^b f(x) dx$ in geschlossener Form auszuwerten; z.B. ist für das in der Statistik häufig auftretende Integral

$$F(x) = \frac{1}{\sqrt{2\pi}} \int_0^x e^{-t^2/2} dt$$

keine elementare Stammfunktion angebar. In diesem Fall ist man auf numerische Verfahren zur Integration, sog. Quadraturverfahren, angewiesen. Wir wollen in diesem Abschnitt einige wichtige Verfahren zur näherungsweise Berechnung bestimmter Integrale besprechen.

3.1 Konstruktion von Quadraturformeln

Wir betrachten das bestimmte Integral

$$\int_a^b f(x) dx$$

mit einer gegebenen integrierbaren Funktion $f : [a, b] \rightarrow \mathbb{R}$.

Mit der Variablentransformation $x = a + t(b - a)$ erhält man

$$\int_a^b f(x) dx = (b - a) \int_0^1 f(a + t(b - a)) dt,$$

so dass man sich ohne Beschränkung der Allgemeinheit auf das Intervall $[a, b] = [0, 1]$ beschränken kann.

Eine naheliegende Idee zur Konstruktion von Quadraturformeln ist, $n + 1$ verschiedene Knoten $x_0, x_1, \dots, x_n \in [0, 1]$ zu wählen, das Interpolationspolynom p von f zu diesen Knoten zu bestimmen und als Näherung für das Integral von f das Integral über das Interpolationspolynom p zu wählen.

$$\int_0^1 f(x) dx \approx \int_0^1 p(x) dx =: Q(f).$$

Mit

$$\ell_j(x) := \prod_{\substack{i=0 \\ i \neq j}}^n (x - x_i) \bigg/ \prod_{\substack{i=0 \\ i \neq j}}^n (x_j - x_i), \quad j = 0, \dots, n,$$

gilt nach der Lagrangeschen Interpolationsformel

$$p(x) = \sum_{j=0}^n f(x_j) \ell_j(x),$$

und daher erhält man

$$Q(f) = \int_0^1 \sum_{j=0}^n f(x_j) \ell_j(x) dx = \sum_{j=0}^n f(x_j) \int_0^1 \ell_j(x) dx =: \sum_{j=0}^n \alpha_j f(x_j).$$

Dabei hängen die Gewichte

$$\alpha_j := \int_0^1 \ell_j(x) dx$$

nur von den gewählten Knoten x_0, \dots, x_n ab und sind unabhängig vom aktuellen Integranden f . Sie können also ein für alle mal berechnet werden und in Tafeln oder Dateien bereitgestellt werden.

Beispiel 3.1. Für $n = 0$ und $x_0 = 0.5$ gilt $\ell_0(x) \equiv 1$ und $\alpha_0 = \int_0^1 \ell_0(x) dx = 1$. Die entstehende Quadraturformel

$$\int_0^1 f(x) dx \approx f(0.5) =: R(f),$$

bzw. für das allgemeine Intervall

$$\int_a^b f(x) dx \approx (b - a) f\left(\frac{a + b}{2}\right) =: R(f),$$

heißt **Rechteckregel** oder **Mittelpunktregel**. □

Beispiel 3.2. Für $n = 1$, $x_0 = 0$ und $x_1 = 1$ ist $\ell_0(x) = 1 - x$ und $\ell_1(x) = x$. Durch Integration erhält man $\alpha_0 = \alpha_1 = 0.5$ und damit die **Trapezregel**

$$\int_0^1 f(x) dx \approx \frac{1}{2}(f(0) + f(1)) =: T(f)$$

bzw. für das allgemeine Intervall

$$\int_a^b f(x) dx \approx (b - a) \frac{f(a) + f(b)}{2} =: T(f). \quad \square$$

Beispiel 3.3. Für $n = 2$, $x_0 = 0$, $x_1 = 0.5$ und $x_2 = 1$ erhält man wie in den beiden vorhergehenden Beispielen die **Simpson Regel** (in der deutschsprachigen Literatur auch **Keplersche Fassregel**)

$$\int_0^1 f(x) dx \approx \frac{1}{6} \cdot (f(0) + 4f(0.5) + f(1)) =: S(f)$$

bzw.

$$\int_a^b f(x) dx \approx \frac{b - a}{6} \cdot (f(a) + 4f(\frac{a+b}{2}) + f(b)) =: S(f). \quad \square$$

Beispiel 3.4. Für $n = 4$ und $x_j = a + j(b - a)/4$, $j = 0, 1, \dots, 4$, erhält man die **Milne Regel**

$$\int_a^b f(x) dx \approx \frac{b - a}{90} (7f(x_0) + 32f(x_1) + 12f(x_2) + 32f(x_3) + 7f(x_4)) \quad \square$$

Es ist naheliegend (und dies ist auch die historisch älteste Wahl), die Knoten äquidistant im Intervall $[a, b]$ zu wählen. Berücksichtigt man dabei die Intervallenden, wählt man also

$$x_j = a + j \cdot \frac{b - a}{n}, \quad j = 0, \dots, n,$$

so erhält man die **abgeschlossenen Newton–Cotes Formeln**

$$\int_0^1 f(x) dx \approx \sum_{j=0}^n \alpha_j^{(n)} f\left(\frac{j}{n}\right)$$

bzw.

$$\int_a^b f(x) dx \approx (b - a) \sum_{j=0}^n \alpha_j^{(n)} f\left(a + j \frac{b - a}{n}\right) =: ANC_n(f).$$

Tabelle 3.1: abgeschlossene Newton–Cotes Formeln

| n | $\alpha_j^{(n)}$ | | | | | Name |
|-----|------------------|-------|-------|-------|------|---------------|
| 1 | 1/2 | 1/2 | | | | Trapezregel |
| 2 | 1/6 | 4/6 | 1/6 | | | Simpson Regel |
| 3 | 1/8 | 3/8 | 3/8 | 1/8 | | 3/8 Regel |
| 4 | 7/90 | 32/90 | 12/90 | 32/90 | 7/90 | Milne Regel |

Tabelle 3.2: offene Newton–Cotes Formeln

| n | $\beta_j^{(n)}$ | | | | |
|-----|-----------------|--------|-------|--------|-------|
| 0 | 1 | | | | |
| 1 | 1/2 | 1/2 | | | |
| 2 | 2/3 | -1/3 | 2/3 | | |
| 3 | 11/24 | 1/24 | 1/24 | 11/24 | |
| 4 | 11/20 | -14/20 | 26/20 | -14/20 | 11/20 |

Berücksichtigt man die Intervallenden nicht, wählt man also

$$x_j = a + (j + 1) \cdot \frac{b - a}{n + 2}, \quad j = 0, \dots, n,$$

so erhält man die **offenen Newton–Cotes Formeln**

$$\int_0^1 f(x) dx \approx \sum_{j=0}^n \beta_j^{(n)} f\left(\frac{j+1}{n+2}\right)$$

bzw.

$$\int_a^b f(x) dx \approx (b - a) \sum_{j=0}^n \beta_j^{(n)} f\left(a + (j + 1) \frac{b - a}{n + 2}\right) =: ONC_n(f).$$

Bemerkung 3.5. Die Mittelpunkregel ist die offene Newton–Cotes Regel für den Fall $n = 0$. Die Trapezregel, die Simpson und die Milne Regel sind die abgeschlossenen Newton–Cotes Formeln für die Fälle $n = 1$, $n = 2$ und $n = 4$. \square

Tabelle 3.1 enthält die wichtigsten abgeschlossenen Newton–Cotes Formeln, Tabelle 3.2 die ersten offenen Newton–Cotes Formeln.

Bemerkung 3.6. Offene Formeln (d.h. solche Formeln, bei denen die Intervallenden keine Knoten sind,) verwendet man, wenn die Auswertung des Integranden an den Randpunkten schwierig ist (z.B. der Grenzwert eines Quotienten, für den Zähler und Nenner gegen 0 gehen, oder gar eine Singularität von f am Rand vorliegt).

Offene Newton–Cotes Formeln werden heute (mit Ausnahme der Mittelpunktregel) kaum noch verwendet, da sie wesentlich schlechtere Fehlerordnungen haben als die Gauß Formeln (vgl. Abschnitt 3.4), die ebenfalls offen sind. \square

Die Gewichte der Newton–Cotes Formeln wachsen rasch an. Für die abgeschlossenen Formeln treten für $n \geq 8$ wechselnde Vorzeichen auf (für die offenen Formeln sogar schon für $n \geq 2$). Diese Formeln sind also anfällig für Rundungsfehler. Man benutzt die Newton–Cotes Formeln daher nur für kleine n auf Teilintervallen von $[a, b]$ und summiert auf. Man erhält dann die **summierten Newton–Cotes Formeln** oder **zusammengesetzten Newton–Cotes Formeln**.

Beispiele hierfür sind mit $h := \frac{b-a}{m}$ und $x_j := a + j \cdot h, j = 0, \dots, m$, die **summierte Rechteckregel**

$$\int_a^b f(x) dx \approx h \sum_{j=1}^m f(x_j - h/2) =: R_h(f), \quad (3.1)$$

die **summierte Trapezregel**

$$\int_a^b f(x) dx \approx h \left(\frac{1}{2} f(a) + \sum_{i=1}^{m-1} f(x_i) + \frac{1}{2} f(b) \right) =: T_h(f), \quad (3.2)$$

und für $m = 2k$ die **summierte Simpson Regel** (beachten Sie, dass die Simpson Regel jeweils auf zwei benachbarte Teilintervalle der Gesamtlänge $2h$ angewandt wird)

$$\begin{aligned} \int_a^b f(x) dx &\approx \frac{2h}{6} \left(f(x_0) + 4f(x_1) + 2f(x_2) + 4f(x_3) + \dots + 4f(x_{2k-1}) + f(x_{2k}) \right) \\ &= \frac{2h}{6} \left(f(a) + 4 \sum_{i=1}^k f(x_{2i-1}) + 2 \sum_{i=1}^{k-1} f(x_{2i}) + f(b) \right) =: S_h(f). \end{aligned}$$

Abbildung 3.1 enthält links eine graphische Darstellung der summierten Mittelpunktregel und rechts der summierten Trapezregel. In Abbildung 3.2 ist die summierte Simpson Regel dargestellt.

In Abschnitt 2.2 haben wir gesehen, dass der Fehler des Interpolationspolynoms bei äquidistanten Knoten am Rande des Intervalls stark wächst und dass das Fehlerverhalten wesentlich günstiger ist, wenn man an den Chebyshev Knoten interpoliert. Nimmt man die Endpunkte des Intervalls hinzu, und wählt man als Knoten

$$x_i = \frac{a+b}{2} - \frac{b-a}{2} \cos\left(\frac{i}{n} \pi\right), \quad i = 0, 1, \dots, n,$$

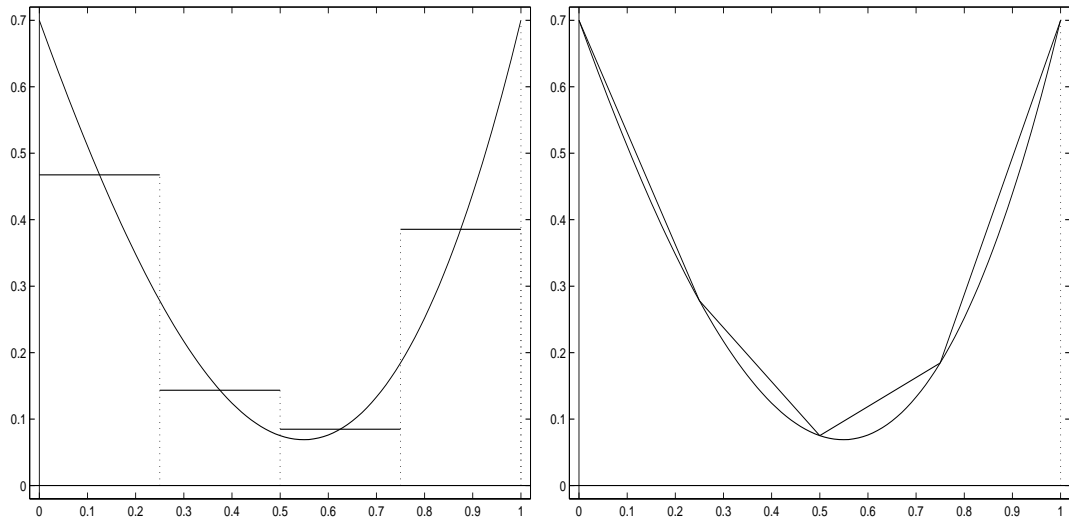


Abbildung 3.1 : Summierte Mittelpunkt- / Trapezregel

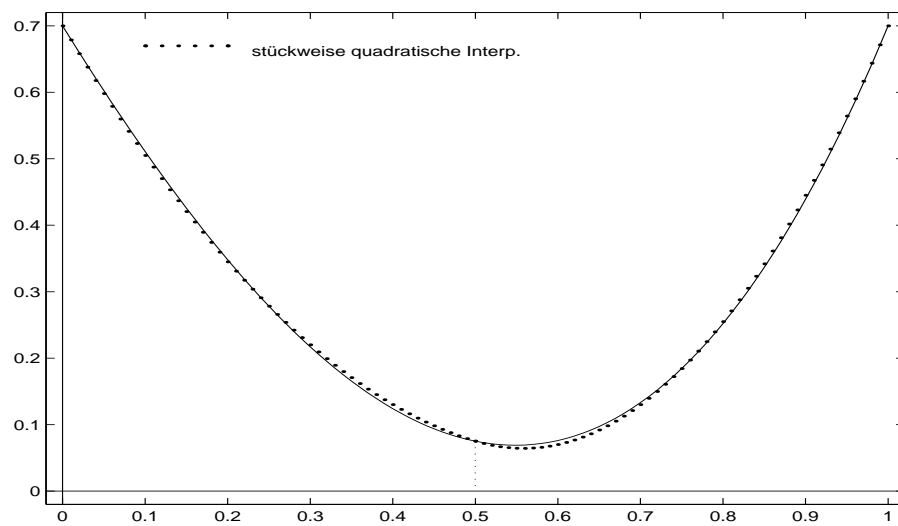


Abbildung 3.2: Summierte Simpson Regel

Tabelle 3.3: Clenshaw–Curtis Formeln

| n | $\gamma_j^{(n)}$ | | | | |
|-----|------------------|------|-------|------|------|
| 1 | 1/2 | 1/2 | | | |
| 2 | 1/6 | 4/6 | 1/6 | | |
| 3 | 1/18 | 8/18 | 8/18 | 1/18 | |
| 4 | 1/30 | 8/30 | 12/30 | 8/30 | 1/30 |

so erhält man die **Clenshaw–Curtis Formeln**.

Die Gewichte der ersten Clenshaw–Curtis Formeln

$$CC_n(f) = (b - a) \sum_{j=0}^n \gamma_j^{(n)} f(x_j)$$

enthält Tabelle 3.3.

3.2 Fehler von Quadraturformeln

Wir untersuchen nun den Fehler von Quadraturformeln. Wir betrachten das Integral

$$I := \int_0^1 f(x) dx$$

und eine zugehörige Quadraturformel mit den Knoten $x_0, \dots, x_n \in [0, 1]$ und den Gewichten $w_0, \dots, w_n \in \mathbb{R}$

$$Q(f) := \sum_{i=0}^n w_i f(x_i)$$

für das Referenzintervall $[0, 1]$.

Definition 3.7. Die Quadraturformel $Q(f)$ hat die **Fehlerordnung** m , wenn für den Fehler

$$E(f) := \int_0^1 f(x) dx - \sum_{i=0}^n w_i f(x_i)$$

gilt

(i) $E(p) = 0$ für alle Polynome $p \in \Pi_{m-1}$

(ii) $E(p) \neq 0$ für ein $p \in \Pi_m$.

Bemerkung 3.8. Wegen der Linearität des Fehlers ist klar, dass Q genau die Fehlerordnung m hat, wenn $E(x^j) = 0$ für $j = 0, \dots, m - 1$ und $E(x^m) \neq 0$ gilt. \square

Bemerkung 3.9. Die Konstruktion liefert, dass die Newton–Cotes Formeln und die Clenshaw–Curtis Formeln mit n Knoten wenigstens die Fehlerordnung $n + 1$ haben.

Für die Trapezregel ist dies die genaue Fehlerordnung, denn

$$T(x^2) = \frac{1}{2} \neq \int_0^1 x^2 dx = \frac{1}{3}.$$

Für die Simpson Regel gilt

$$S(x^3) = \frac{1}{6} (0 + 4 \cdot \frac{1}{8} + 1) = \frac{1}{4} = \int_0^1 x^3 dx, \quad S(x^4) = \frac{5}{24} \neq \int_0^1 x^4 dx = \frac{1}{5},$$

so dass die Simpson Regel sogar die Ordnung 4 hat. \square

Satz 3.10. Es sei Q eine Quadraturformel der Fehlerordnung $m \geq 1$. Dann gibt es eine Funktion $K : [0, 1] \rightarrow \mathbb{R}$, so dass der Fehler von Q die Darstellung

$$E(f) = \int_0^1 f(x) dx - Q(f) = \int_0^1 K(x) f^{(m)}(x) dx \quad (3.3)$$

für alle $f \in C^m[0, 1]$ hat.

Definition 3.11. Die Funktion K in der Fehlerdarstellung (3.3) heißt der **Peano Kern** der Quadraturformel Q .

Beweis: Nach dem Taylorschen Satz gilt

$$f(x) = \sum_{k=0}^{m-1} \frac{f^{(k)}(0)}{k!} x^k + \frac{1}{(m-1)!} \int_0^x f^{(m)}(t) (x-t)^{m-1} dt =: p(x) + r(x),$$

und wegen $p \in \Pi_{m-1}$ folgt mit der Funktion

$$(t)_+^k := \begin{cases} t^k, & \text{falls } t \geq 0, \\ 0, & \text{falls } t < 0, \end{cases} \quad k \in \mathbb{N}_0,$$

$$\begin{aligned} E(f) &= E(p) + E(r) = E(r) \\ &= \frac{1}{(m-1)!} \left\{ \int_0^1 \int_0^x f^{(m)}(t) (x-t)^{m-1} dt dx - \sum_{i=0}^n w_i \int_0^{x_i} f^{(m)}(t) (x_i-t)^{m-1} dt \right\} \\ &= \frac{1}{(m-1)!} \left\{ \int_0^1 \int_t^1 f^{(m)}(t) (x-t)^{m-1} dx dt - \sum_{i=0}^n w_i \int_0^1 f^{(m)}(t) (x_i-t)_+^{m-1} dt \right\} \\ &= \frac{1}{(m-1)!} \left\{ \int_0^1 \left(\frac{1}{m} (1-t)^m - \sum_{i=0}^n w_i (x_i-t)_+^{m-1} \right) f^{(m)}(t) dt \right\}, \end{aligned}$$

d.h. (3.3) mit dem Peano Kern

$$K(x) = \frac{1}{(m-1)!} \left(\frac{1}{m} (1-x)^m - \sum_{i=0}^n w_i (x_i - x)_+^{m-1} \right). \quad (3.4)$$

■

Beispiel 3.12. Für die Trapezregel gilt $x_0 = 0$, $x_1 = 1$, $w_0 = w_1 = 0.5$, $m = 2$, und daher

$$K_T(x) = \frac{1}{2} (1-x)^2 - \frac{1}{2} (1-x) = -\frac{1}{2} x(1-x). \quad \square$$

Beispiel 3.13. Für die Simpson Regel gilt $x_0 = 0$, $x_1 = 0.5$, $x_2 = 1$, $w_0 = w_2 = \frac{1}{6}$, $w_1 = \frac{2}{3}$ und $m = 4$, und daher

$$K_S(x) = \frac{1}{3!} \left(\frac{1}{4} (1-x)^4 - \frac{1}{6} (1-x)^3 - \frac{2}{3} \left(\frac{1}{2} - x \right)_+^3 \right). \quad \square$$

Aus Satz 3.10. erhält man die folgende Abschätzung für den Fehler

$$|E(f)| \leq \|f^{(m)}\|_\infty \int_0^1 |K(x)| dx =: \tilde{c}_m \|f^{(m)}\|_\infty. \quad (3.5)$$

In vielen Fällen wechselt der Peano Kern $K(x)$ das Vorzeichen auf dem Intervall $[0, 1]$ nicht. Dann folgt aus (3.3) mit dem Mittelwertsatz der Integralrechnung mit einem $\xi \in (0, 1)$

$$E(f) = f^{(m)}(\xi) \int_0^1 K(x) dx =: c_m f^{(m)}(\xi) \quad (3.6)$$

für eine Quadraturformel der Ordnung m .

Definition 3.14. c_m heißt die **Fehlerkonstante** des Verfahrens Q .

Für die Trapezregel gilt $K_T(x) = -\frac{1}{2} x(1-x) \leq 0$ für alle $x \in [0, 1]$. Da T die Ordnung 2 hat, gilt

$$E_T(f) = -\frac{1}{2} f''(\xi) \int_0^1 x(1-x) dx = -\frac{1}{12} f''(\xi),$$

und die Fehlerkonstante ist $c_2 = -\frac{1}{12}$.

Eine elementare Rechnung zeigt, dass auch für die Simpson Regel

$$K_S(x) = \frac{1}{3!} \left(\frac{1}{4} (1-x)^4 - \frac{1}{6} (1-x)^3 - \frac{2}{3} \left(\frac{1}{2} - x \right)_+^3 \right) \leq 0$$

für alle $x \in [0, 1]$ gilt. Durch Integration von K von 0 bis 1 erhält man für den Fehler wegen $m = 4$

$$E_S(f) = -\frac{f^{(4)}(\xi)}{2880}$$

für ein $\xi \in (0, 1)$, d.h. die Fehlerkonstante ist $c_4 = -\frac{1}{2880}$.

Man kann die Fehlerkonstante der Simpson Regel und auch anderer Formeln ohne Integration (sogar ohne Kenntnis) des Peano Kerns bestimmen. Ist bekannt, dass der Peano Kern einer Formel der Ordnung m sein Vorzeichen in $[0, 1]$ nicht wechselt, der Fehler also eine Darstellung (3.6) hat, so hat man nur $E(x^m)$ zu berechnen.

Es ist nämlich $\frac{d^m}{dx^m}(x^m) = m!$ und daher

$$E(x^m) = m! \cdot c_m,$$

und hieraus erhält man c_m .

Im Fall der Simpson Regel ist

$$E(x^4) = \int_0^1 x^4 dx - \frac{1}{6} \left((0)^4 + 4 \cdot (1/2)^4 + 1^4 \right) = \frac{1}{5} - \frac{1}{6} \cdot \frac{5}{4} = -\frac{1}{120},$$

und daher gilt

$$c_4 = \frac{1}{4!} \cdot \left(-\frac{1}{120} \right) = -\frac{1}{2880}.$$

Wir betrachten nun das Integral von f über ein Intervall der Länge h :

$$\int_{\alpha}^{\alpha+h} f(x) dx.$$

Mit der Variablentransformation $x =: \alpha + ht$ geht dieses über in

$$\int_{\alpha}^{\alpha+h} f(x) dx = h \int_0^1 g(t) dt, \quad g(t) := f(\alpha + ht) = f(x),$$

das wir mit der Quadraturformel

$$Q(g) = \sum_{i=0}^n w_i g(x_i)$$

behandeln, d.h.

$$Q_{[\alpha, \alpha+h]}(f) := h \sum_{i=0}^n w_i f(\alpha + hx_i).$$

Für den Fehler gilt

$$\begin{aligned} E_{[\alpha, \alpha+h]}(f) &:= \int_{\alpha}^{\alpha+h} f(x) dx - Q_{[\alpha, \alpha+h]}(f) \\ &= h \left(\int_0^1 g(t) dt - Q(g) \right) = h E(g). \end{aligned}$$

Besitzt der Fehler $E(g)$ eine Darstellung (3.5), so folgt wegen

$$\begin{aligned} \frac{d^m g}{dt^m} &= \frac{d^m f}{dx^m} \cdot \left(\frac{dx}{dt} \right)^m = h^m \frac{d^m f}{dx^m} \\ |E_{[\alpha, \alpha+h]}(f)| &\leq h^{m+1} \cdot \tilde{c}_m \cdot \max_{\alpha \leq x \leq \alpha+h} |f^{(m)}(x)|. \end{aligned} \quad (3.7)$$

Gilt eine Darstellung (3.6), so erhält man genauso

$$E_{[\alpha, \alpha+h]}(f) = h^{m+1} \cdot c_m \cdot f^{(m)}(\eta), \quad \eta \in [\alpha, \alpha + h]. \quad (3.8)$$

Wir haben bereits erwähnt, dass die Genauigkeit einer Näherung für ein Integral nicht durch die Erhöhung der Ordnung der benutzten Quadraturformel verbessert wird, sondern dass das Intervall $[a, b]$ in n Teilintervalle der Länge h zerlegt wird, und dass in jedem Teilintervall eine Quadraturformel kleiner Ordnung verwendet wird. Für die summierte Quadraturformel

$$Q_h(f) := \sum_{i=1}^n Q_{[a+(i-1)h, a+ih]}(f)$$

erhält man aus (3.7) (und genauso aus (3.8))

$$\begin{aligned} \left| \int_a^b f(x) dx - Q_h(f) \right| &= \left| \sum_{i=1}^n \left(\int_{a+(i-1)h}^{a+ih} f(x) dx - Q_{[a+(i-1)h, a+ih]}(f) \right) \right| \\ &\leq \sum_{i=1}^n |E_{[a+(i-1)h, a+ih]}(f)| \\ &\leq \sum_{i=1}^n h^{m+1} \cdot \tilde{c}_m \cdot \max\{|f^{(m)}(x)| : a + (i-1)h \leq x \leq a + ih\} \\ &\leq n h \cdot h^m \cdot \tilde{c}_m \cdot \|f^{(m)}\|_{\infty} = h^m (b-a) \tilde{c}_m \cdot \|f^{(m)}\|_{\infty}. \end{aligned}$$

Man verliert also für die summierte Formel eine Potenz in h . Insbesondere erhält für die summierte Trapezregel den Fehler

$$\left| \int_a^b f(x) dx - T_h(f) \right| \leq \frac{h^2}{12} (b-a) \cdot \|f''\|_{\infty} \quad (3.9)$$

und für die summierte Simpson Regel

$$\left| \int_a^b f(x) dx - S_h(f) \right| \leq \frac{h^4}{2880} (b-a) \cdot \|f^{(4)}\|_{\infty}.$$

3.3 Romberg Verfahren

In diesem Abschnitt werden wir eine asymptotische Entwicklung des Fehlers der summierten Trapezregel

$$T_h(f) = \frac{h}{2} \left(f(a) + 2 \sum_{i=1}^{n-1} f(x_i) + f(b) \right), \quad h := \frac{b-a}{n}, \quad x_i := a + ih, \quad (3.10)$$

herleiten. Hiermit werden wir durch Extrapolation zu besseren Quadraturformeln gelangen.

Satz 3.15. (Euler Maclaurinsche Summenformel)

Sei $g \in C^{2m+2}[0, n]$, $n \in \mathbb{N}$ und

$$T_1(g) := \frac{1}{2}g(0) + \sum_{j=1}^{n-1} g(j) + \frac{1}{2}g(n)$$

die summierte Trapezformel von g mit der Schrittweite 1. Dann gibt es Konstanten $C_1, \dots, C_m \in \mathbb{R}$ und eine beschränkte Funktion $\phi_{2m+2} : \mathbb{R} \rightarrow \mathbb{R}$, so dass

$$\int_0^n g(t) dt = T_1(g) - \sum_{j=1}^m C_j \left(g^{(2j-1)}(n) - g^{(2j-1)}(0) \right) + R_m(g) \quad (3.11)$$

gilt mit

$$R_m(g) = \int_0^n (\phi_{2m+2}(t) - \phi_{2m+2}(0)) g^{(2m+2)}(t) dt. \quad (3.12)$$

Beweis: Mit der periodischen Funktion

$$\phi_1(t) := t - \frac{1}{2}, \quad 0 \leq t < 1, \quad \phi_1(t+1) = \phi_1(t), \quad t \in \mathbb{R}$$

erhält man durch partielle Integration für $j = 1, \dots, n$

$$\begin{aligned} \int_{j-1}^j g(t) dt &= [\phi_1(t)g(t)]_{j-1}^j - \int_{j-1}^j \phi_1(t)g'(t) dt \\ &= \frac{1}{2}(g(j) + g(j-1)) - \int_{j-1}^j \phi_1(t)g'(t) dt, \end{aligned}$$

und durch Summation

$$\int_0^n g(t) dt = T_1(g) - \int_0^n \phi_1(t)g'(t) dt. \quad (3.13)$$

Um das Integral der rechten Seite weiter durch partielle Integration umzuformen, benötigen wir Funktionen $\phi_j : \mathbb{R} \rightarrow \mathbb{R}$, $j \geq 2$, mit $\phi'_j = \phi_{j-1}$.

ϕ_1 besitzt die Fourierreihe

$$\phi_1(t) \sim -2 \sum_{k=1}^{\infty} \frac{\sin(2\pi kt)}{2\pi k}.$$

Durch wiederholte gliedweise Integration erhält man die Funktionen

$$\begin{aligned} \phi_{2j}(t) &:= 2 \cdot (-1)^{j+1} \sum_{k=1}^{\infty} \frac{\cos(2\pi kt)}{(2\pi k)^{2j}}, \quad j \geq 1, \\ \phi_{2j+1}(t) &:= 2 \cdot (-1)^{j+1} \sum_{k=1}^{\infty} \frac{\sin(2\pi kt)}{(2\pi k)^{2j+1}}, \quad j \geq 0. \end{aligned}$$

Nach dem Majorantenkriterium konvergieren die Reihen ϕ_j für $j \geq 2$ gleichmäßig in \mathbb{R} . Daher gilt $\phi'_j = \phi_{j-1}$ für $j \geq 3$. Da die Fourierreihe von ϕ_1 für jedes abgeschlossene Intervall, das keine ganzzahligen Punkte enthält, gleichmäßig gegen ϕ_1 konvergiert, gilt auch $\phi'_2(t) = \phi_1(t)$ für alle $t \notin \mathbb{Z}$. Damit folgt aus (3.13)

$$\begin{aligned} \int_0^n g(t) dt - T_1(g) &= - \int_0^n \phi_1(t) g'(t) dt \\ &= - [\phi_2(t) g'(t)]_0^n + \int_0^n \phi_2(t) g''(t) dt = \dots \\ &= \sum_{j=2}^{2m+1} (-1)^{j+1} [\phi_j(t) g^{(j-1)}(t)]_0^n - \int_0^n \phi_{2m+1}(t) g^{(2m+1)}(t) dt, \end{aligned}$$

und nochmalige partielle Integration liefert

$$\int_0^n \phi_{2m+1}(t) g^{(2m+1)}(t) dt = - \int_0^n (\phi_{2m+2}(t) - \phi_{2m+2}(0)) g^{(2m+2)}(t) dt.$$

Offensichtlich gilt

$$\begin{aligned} \phi_{2j+1}(0) &= \phi_{2j+1}(n) = 0, \quad j \in \mathbb{N}, \\ \phi_{2j}(0) &= \phi_{2j}(n) = (-1)^{j+1} \sum_{k=1}^{\infty} \frac{2}{(2\pi k)^{2j}}, \quad j \geq 1. \end{aligned}$$

Daher folgt die Entwicklung (3.11) mit $C_j = \phi_{2j}(0)$. ■

Bemerkung 3.16. Die Zahlen

$$B_j := (-1)^{j+1} (2j)! \phi_{2j}(0), \quad j \in \mathbb{N},$$

heißen **Bernoullische Zahlen**. Mit ihnen lautet die Euler Maclaurin Entwicklung (und dies ist die übliche Darstellung)

$$\int_0^n g(t) dt - T_1(g) = \sum_{j=1}^m \frac{(-1)^j B_j}{(2j)!} \left(g^{(2j-1)}(n) - g^{(2j-1)}(0) \right) + R_m(g).$$

Als Folgerung erhalten wir die gewünschte asymptotische Entwicklung des Fehlers der summierten Trapezregel.

Korollar 3.17. Ist $f \in C^{2m+2}[a, b]$ und $h := \frac{b-a}{n}$, $n \in \mathbb{N}$, so gilt

$$T_h(f) = \int_a^b f(x) dx + \sum_{j=1}^m c_j h^{2j} + \psi_{m+1}(h) h^{2m+2} \quad (3.14)$$

wobei $|\psi_{m+1}(h)| \leq M$ mit einer von h unabhängigen Konstante M gilt.

Beweis: Mit $x = a + th$ und $g(t) := f(a + th)$ gilt

$$\int_a^b f(x) dx = h \int_0^n g(t) dt \quad \text{und} \quad T_h(f) = hT_1(g),$$

und daher folgt mit $g^{(j)}(t) = h^j f^{(j)}(a + th)$ aus Satz 3.15.

$$T_h(f) = \int_a^b f(x) dx + \sum_{j=1}^m C_j \left(f^{(2j-1)}(b) - f^{(2j-1)}(a) \right) h^{2j} - hR_m(g),$$

wobei

$$\begin{aligned} hR_m(g) &= h \int_0^n (\phi_{2m+2}(t) - \phi_{2m+2}(0)) g^{(2m+2)}(t) dt \\ &= h^{2m+2} \int_a^b \left(\phi_{2m+2} \left(\frac{x-a}{h} \right) - \phi_{2m+2}(0) \right) f^{(2m+2)}(x) dx \end{aligned}$$

gilt. Da ϕ_{2m+2} (als stetige und 1-periodische Funktion) beschränkt ist, ist auch

$$\psi_{m+1}(h) := - \int_a^b \left(\phi_{2m+2} \left(\frac{x-a}{h} \right) - \phi_{2m+2}(0) \right) f^{(2m+2)}(x) dx$$

beschränkt. ■

Bemerkung 3.18. Ist $f : \mathbb{R} \rightarrow \mathbb{R}$ periodisch mit der Periode T , so ist es bei der Berechnung von $\int_0^T f(x) dx$ wegen $\int_0^T f(x) dx = \int_a^{a+T} f(x) dx$ für alle $a \in \mathbb{R}$ nicht sinnvoll, gewisse Stützstellen stärker zu gewichten als andere (wie dies bei der Simpson

Regel und den noch zu besprechenden Gaußschen Quadraturformeln der Fall ist). Dies legt die Verwendung der Trapezregel nahe, die hier die Gestalt hat

$$T_h(f) = h \sum_{i=0}^{n-1} f(ih), \quad h := \frac{T}{n}. \quad (3.15)$$

Aus (3.11) und (3.14) entnimmt man in diesem Fall für $f \in C^{2m+2}(\mathbb{R})$

$$\left| T_h(f) - \int_0^T f(x) dx \right| = |\psi_{m+1}(h)| h^{2m+2},$$

und wegen Korollar 3.17. gilt $|\psi_{m+1}| \leq M$. Für die summierte Trapezregel geht der Fehler in diesem Fall also sogar wie h^{2m+2} gegen 0. \square

Wir verwenden nun Korollar 3.17., um durch Extrapolation zu sehr rasch konvergenten Integrationsmethoden zu gelangen. Vernachlässigt man bei der Entwicklung von $T_h(f)$ in (3.14) das Restglied, so lässt sich $T_h(f)$ als ein Polynom in h^2 auffassen, das für $h = 0$ den Wert $c_0 := \int_a^b f(x) dx$ liefert. Dies legt das folgende Verfahren zur Bestimmung von c_0 nahe.

Bestimme zu verschiedenen Schrittweiten $h_0, \dots, h_m > 0$ die zugehörigen Trapezsummen $T_{h_j}(f)$, $j = 0, \dots, m$. Dann gibt es ein eindeutiges Polynom $p \in \Pi_m$ (in dem Argument h^2) mit

$$p(h_j^2) = T_{h_j}(f), \quad j = 0, \dots, m.$$

$p(0)$ dient als verbesserte Näherung von $\int_a^b f(x) dx$.

Da nicht p als Funktion von h^2 gesucht ist, sondern nur der Wert $p(0)$, bietet sich der Algorithmus von Neville und Aitken für diese Aufgabe an.

Sei $h_0 := b - a$, $h_j := h_0/n_j$, $n_j < n_{j+1}$, $j = 1, \dots, m$, und $T_{j,j} := T_{h_j}(f)$ die Trapezsumme zur Schrittweite h_j .

Ferner sei $p_{ij}(h)$ für $0 \leq i \leq j \leq m$ dasjenige Polynom vom Grade $j - i$ in h^2 , für das gilt

$$p_{ij}(h_k) = T_{kk}, \quad k = i, \dots, j.$$

Dann gilt für die extrapolierten Werte $T_{ij} := p_{ij}(0)$ nach dem Algorithmus von Neville und Aitken

$$T_{ij} = T_{i+1,j} + \frac{T_{i+1,j} - T_{i,j-1}}{\left(\frac{h_i}{h_j}\right)^2 - 1}, \quad 0 \leq i < j \leq m. \quad (3.16)$$

Dieses Verfahren wurde erstmals von Romberg [90] 1955 vorgeschlagen für die speziellen Schrittweiten $h_i = (b - a) \cdot 2^{-i}$. Es vereinfacht sich dann zu

$$T_{ij} = \frac{4^{j-i}T_{i+1,j} - T_{i,j-1}}{4^{j-i} - 1}$$

Man erhält das folgende Schema von Näherungswerten für $\int_a^b f(x) dx$

$$\begin{array}{cccccc} T_{00} & & & & & \\ T_{11} & T_{01} & & & & \\ T_{22} & T_{12} & T_{02} & & & \\ T_{33} & T_{23} & T_{13} & T_{03} & & \\ T_{44} & T_{34} & T_{24} & T_{14} & T_{04} & \end{array}$$

In der ersten Spalte stehen die Näherungen nach der zusammengesetzten Trapezregel. Man rechnet leicht nach, dass in der zweiten bzw. dritten Spalte die Näherungen nach der zusammengesetzten Simpson Regel bzw. Milne Regel stehen. Die vierte Spalte entspricht keiner zusammengesetzten Newton–Cotes Formel.

Bei der praktischen Durchführung beachte man, dass bei der Berechnung von $T_{i+1,i+1}$ auf die schon bei T_{ii} berechneten Funktionswerte zurückgegriffen werden kann. Es gilt

$$\begin{aligned} T_{i+1,i+1} &= T_{h_i/2}(f) = \frac{h_i}{4} \left(f(a) + 2 \sum_{j=1}^{2n_i-1} f\left(a + j \frac{h_i}{2}\right) + f(b) \right) \\ &= \frac{h_i}{4} \left(f(a) + 2 \sum_{j=1}^{n_i-1} f(a + j h_i) + f(b) \right) + \frac{h_i}{2} \sum_{j=1}^{n_i} f\left(a + \frac{2j-1}{2} h_i\right) \\ &= \frac{1}{2} \left(T_{ii} + h_i \sum_{j=1}^{n_i} f\left(a + \frac{2j-1}{2} h_i\right) \right). \end{aligned}$$

Nachteil der Romberg-Folge $h_i = h_0 \cdot 2^{-i}$ ist, dass die Zahl der Funktionsauswertungen zur Bestimmung von T_{ii} exponentiell steigt.

Von Bulirsch [14] (1964) wurde daher eine andere Folge vorgeschlagen, die **Bulirsch Folge**

$$h_{2i-1} = 2^{-i} h_0, \quad h_{2i} = \frac{1}{3} 2^{-i+1} h_0, \quad i \in \mathbb{N}.$$

Die Bulirsch Folge hat den Vorteil, dass man bei der Berechnung von $T_{h_i}(f)$ wieder ähnlich wie bei der Romberg Folge den Wert $T_{h_{i-2}}$ verwenden kann.

Beispiel 3.19. Die folgenden Tableaus enthalten die Fehler $T_{ij} - \int_0^1 f(x) dx$ für das Beispiel

$$\int_0^1 e^x \sin 5x dx \quad (= -0.05623058659667)$$

Romberg (33 Funktionsauswertungen)

| | | | | | | | |
|----------------|-----------|----------|-----------|-----------|------------|-----------|--|
| 1 | -1.2E 0 | | | | | | |
| $\frac{1}{2}$ | -1.0E - 1 | 2.8E - 1 | | | | | |
| $\frac{1}{4}$ | -2.1E - 2 | 6.3E - 3 | -1.2E - 2 | | | | |
| $\frac{1}{8}$ | -5.0E - 3 | 3.2E - 4 | -8.3E - 5 | 1.1E - 4 | | | |
| $\frac{1}{16}$ | -1.2E - 3 | 1.9E - 5 | -1.1E - 6 | 2.3E - 7 | -1.8E - 7 | | |
| $\frac{1}{32}$ | -3.1E - 4 | 1.2E - 6 | -1.6E - 8 | 7.6E - 10 | -1.4E - 10 | 4.4E - 11 | |

Bulirsch (20 Funktionsauswertungen)

| | | | | | | | |
|----------------|-----------|----------|-----------|----------|-----------|-----------|-----------|
| 1 | -1.2E 0 | | | | | | |
| $\frac{1}{2}$ | -1.0E - 1 | 2.8E - 1 | | | | | |
| $\frac{1}{3}$ | -3.9E - 2 | 1.2E - 1 | -2.2E - 2 | | | | |
| $\frac{1}{4}$ | -2.1E - 2 | 2.5E - 3 | -6.3E - 4 | 7.8E - 4 | | | |
| $\frac{1}{6}$ | -8.9E - 3 | 5.7E - 4 | -5.9E - 5 | 1.2E - 5 | -9.5E - 6 | | |
| $\frac{1}{8}$ | -5.0E - 3 | 1.4E - 4 | -7.7E - 6 | 6.7E - 7 | -1.2E - 7 | 3.2E - 8 | |
| $\frac{1}{12}$ | -2.2E - 3 | 3.4E - 5 | -8.2E - 7 | 3.9E - 8 | -2.9E - 9 | 4.0E - 10 | 1.8E - 10 |

□

Im Folgenden soll für den Fehler $T_{mm} - \int_a^b f(x) dx$ eine Abschätzung hergeleitet werden. Sei $T_m(h) := p(h^2)$. Dann gilt nach der Lagrangeschen Interpolationsformel

$$T_m(h) = \sum_{i=0}^m T_{h_i}(f) \prod_{\substack{j=0 \\ j \neq i}}^m \frac{h^2 - h_j^2}{h_i^2 - h_j^2}.$$

Wegen

$$p(h^2) = \sum_{i=0}^m p(h_i^2) \prod_{\substack{j=0 \\ j \neq i}}^m \frac{h^2 - h_j^2}{h_i^2 - h_j^2}$$

für alle Polynome in h^2 vom Höchstgrad m erhält man speziell für $p(h^2) = h^{2k}$, $k = 0, \dots, m$, an der Stelle $h = 0$

$$\sum_{i=0}^m h_i^{2k} \prod_{\substack{j=0 \\ j \neq i}}^m \frac{h_j^2}{h_j^2 - h_i^2} =: \sum_{i=0}^m h_i^{2k} d_{mi} = \begin{cases} 1 & \text{für } k = 0 \\ 0 & \text{für } k = 1, \dots, m. \end{cases} \quad (3.17)$$

Für $p(h^2) = h^{2m+2}$ gilt

$$h^{2m+2} = \sum_{i=0}^m h_i^{2m+2} \prod_{\substack{j=0 \\ j \neq i}}^m \frac{h^2 - h_j^2}{h_i^2 - h_j^2} + \prod_{j=0}^m (h^2 - h_j^2) ,$$

denn auf beiden Seiten stehen Polynome vom Grade $m + 1$ (im Argument h^2), die in h_i , $i = 0, \dots, m$, übereinstimmen und deren Koeffizient bei h^{2m+2} gleich 1 ist.

Für $h = 0$ folgt

$$\sum_{i=0}^m h_i^{2m+2} d_{mi} = (-1)^m \prod_{j=0}^m h_j^2 . \quad (3.18)$$

Nun gilt

$$T_{mm} = T_m(0) = \sum_{i=0}^m d_{mi} T_{h_i}(f) , \quad (3.19)$$

sowie

$$T_h(f) = \sum_{i=0}^m c_i h^{2i} + h^{2m+2} \psi_{m+1}(h)$$

mit $c_0 = \int_a^b f(x) dx$ und

$$\psi_{m+1}(h) = - \int_a^b \left\{ \phi_{2m+2} \left(\frac{x-a}{h} \right) - \phi_{2m+2}(0) \right\} f^{(2m+2)}(x) dx .$$

Wegen (3.17) und (3.19) folgt

$$\begin{aligned} T_{mm} &= \sum_{i=0}^m d_{mi} \left(\sum_{k=0}^m c_k h_i^{2k} + h_i^{2m+2} \psi_{m+1}(h_i) \right) \\ &= \sum_{k=0}^m c_k \sum_{i=0}^m h_i^{2k} d_{mi} + \sum_{i=0}^m d_{mi} h_i^{2m+2} \psi_{m+1}(h_i) \\ &= \int_a^b f(x) dx + \int_a^b f^{(2m+2)}(x) K(x) dx \end{aligned}$$

mit dem Kern

$$K(x) = - \sum_{i=0}^m d_{mi} h_i^{2m+2} \left(\phi_{2m+2} \left(\frac{x-a}{h_i} \right) - \phi_{2m+2}(0) \right)$$

Da $K(x)$ für die Romberg-Folge und für die Bulirsch Folge konstantes Vorzeichen auf $[a, b]$ besitzt, liefert der Mittelwertsatz der Integralrechnung, dass es ein $\xi \in [a, b]$ gibt mit

$$T_{mm} - \int_a^b f(x) dx = f^{(2m+2)}(\xi) \int_a^b K(x) dx . \quad (3.20)$$

Es ist

$$\begin{aligned} & \int_a^b \left(\phi_{2m+2} \left(\frac{x-a}{h_i} \right) - \phi_{2m+2}(0) \right) dx \\ &= -\phi_{2m+2}(0)(b-a) + h_i \int_0^{(b-a)/h_i} \phi_{2m+2}(t) dt = \frac{(-1)^{m+1}}{(2m+2)!} B_{m+1}(b-a), \end{aligned}$$

und aus (3.20) erhält man unter Benutzung von (3.18)

$$\begin{aligned} T_{mm} - \int_a^b f(x) dx &= -f^{(2m+2)}(\xi) \sum_{i=0}^m d_{mi} h_i^{2m+2} \int_a^b \left(\phi_{2m+2} \left(\frac{x-a}{h_i} \right) - \phi_{2m+2}(0) \right) dx \\ &= (b-a) \frac{B_{m+1}}{(2m+2)!} f^{(2m+2)}(\xi) \prod_{i=0}^m h_i^2 \end{aligned}$$

Interpoliert man $T_h(f)$ an den Stellen $h_{i-k}, h_{i-k+1}, \dots, h_i$, so erhält man auf dieselbe Weise

$$T_{ik} - \int_a^b f(x) dx = (b-a) \frac{B_{k+1}}{(2k+2)!} f^{(2k+2)}(\xi) \prod_{j=0}^k h_{i-j}^2 \quad (3.21)$$

Hieraus liest man unmittelbar die Konvergenzordnungen der Spalten des Extrapolationsschemas ab:

$$T_{ik} - \int_a^b f(x) dx \rightarrow 0 \quad \text{wie} \quad \prod_{j=0}^k h_{i-j}^2$$

3.4 Quadraturformeln von Gauß

Wir haben bisher in Abschnitt 3.1 die Knoten der Quadraturformeln (äquidistant oder als Nullstellen der Chebyshev Polynome) vorgegeben. Die Fehlerordnung war dann (wenigstens) gleich der Anzahl der Knoten (im Falle der Simpson Regel bei 3 Knoten 4). Wir fragen nun, wie weit wir durch Wahl der Knoten und der Gewichte die Fehlerordnung erhöhen können.

Beispiel 3.20. Wir betrachten die Quadraturformel $G_1(f) := w_1 f(x_1)$ mit einem Knoten x_1 für das Integral $\int_0^1 f(x) dx$ und bestimmen $x_1 \in [0, 1]$ und $w_1 \in \mathbb{R}$ so, dass Polynome möglichst hohen Grades exakt integriert werden:

$$\begin{aligned} \int_0^1 x^0 dx = 1 &= w_1 x_1^0 = w_1 \quad \Rightarrow \quad w_1 = 1, \\ \int_0^1 x^1 dx = 0.5 &= w_1 x_1^1 = x_1 \quad \Rightarrow \quad x_1 = 0.5. \end{aligned}$$

Durch diese beiden Gleichungen ist also die Quadraturformel

$$G_1(f) = f(0.5)$$

bereits festgelegt. Man erhält die Mittelpunkregel. Wegen

$$\int_0^1 x^2 dx = \frac{1}{3} \neq 1 \cdot (0.5)^2$$

hat sie die Fehlerordnung 2. □

Beispiel 3.21. Für die Quadraturformel

$$G_2(f) = w_1 f(x_1) + w_2 f(x_2)$$

mit zwei Knoten $x_1, x_2 \in [0, 1]$ erhält man die Bestimmungsgleichungen

$$\begin{aligned} \int_0^1 x^0 dx &= 1 = w_1 + w_2 \\ \int_0^1 x^1 dx &= \frac{1}{2} = w_1 x_1 + w_2 x_2 \\ \int_0^1 x^2 dx &= \frac{1}{3} = w_1 x_1^2 + w_2 x_2^2 \\ \int_0^1 x^3 dx &= \frac{1}{4} = w_1 x_1^3 + w_2 x_2^3 \end{aligned}$$

mit der (bis auf Vertauschung von x_1 und x_2) eindeutigen Lösung

$$w_1 = w_2 = \frac{1}{2}, \quad x_1 = \frac{1}{2} \left(1 - \frac{1}{\sqrt{3}}\right), \quad x_2 = \frac{1}{2} \left(1 + \frac{1}{\sqrt{3}}\right).$$

Wegen

$$\int_0^1 x^4 dx = \frac{1}{5} \neq w_1 x_1^4 + w_2 x_2^4 = \frac{7}{36}$$

hat die gefundene Formel G_2 die Fehlerordnung 4. □

Prinzipiell kann man so fortfahren und Quadraturformeln immer höherer Ordnung konstruieren. Man erhält dann nichtlineare Gleichungssysteme, die immer unübersichtlicher werden. Wir gehen einen anderen Weg.

In Verallgemeinerung unserer bisherigen Überlegungen betrachten wir gleich

$$I(f) = \int_a^b w(x) f(x) dx$$

mit einer positiven Gewichtsfunktion $w \in C[a, b]$.

Wir werden zeigen, dass es zu jedem $n \in \mathbb{N}$ und jeder positiven Gewichtsfunktion $w \in C[a, b]$ eindeutig bestimmte Gewichte $w_i > 0$ und Knoten $x_i \in (a, b)$, $i = 1, \dots, n$, gibt, so dass die Quadraturformel

$$G_n(f) := \sum_{i=1}^n w_i f(x_i) \quad (3.22)$$

die Fehlerordnung $2n$ besitzt (also für alle Polynome vom Höchstgrad $2n - 1$ exakt ist, nicht aber für x^{2n}) und dass durch keine Wahl von Knoten und Gewichten eine höhere Fehlerordnung erreichbar ist.

Dass mit n Knoten nicht die Fehlerordnung $2n + 1$ erreicht werden kann, sieht man so ein. Besitzt (3.22) die Fehlerordnung $2n + 1$, so wird insbesondere das Polynom

$$p(x) := \prod_{j=1}^n (x - x_j)^2 \in \Pi_{2n}$$

exakt integriert. Wegen $p \geq 0$ und $p \not\equiv 0$ gilt aber

$$\int_a^b w(x)p(x) dx > 0, \quad \text{während} \quad G_n(p) = 0 \quad \text{ist.}$$

Wir fragen nun nach einer notwendigen Bedingung dafür, dass die Quadraturformel (3.22) die Ordnung $2n$ besitzt. Wir bezeichnen mit $p_n \in \Pi_n$ das Polynom

$$p_n(x) := \prod_{j=1}^n (x - x_j).$$

Ist $p \in \Pi_{2n-1}$ beliebig, so gibt es Polynome $q, r \in \Pi_{n-1}$ mit $p(x) = p_n(x)q(x) + r(x)$, und aus der Exaktheit von (3.22) für p folgt

$$\begin{aligned} \int_a^b w(x)p(x) dx &= \int_a^b w(x)(p_n(x)q(x) + r(x)) dx = \sum_{j=1}^n w_j(p_n(x_j)q(x_j) + r(x_j)) \\ &= \sum_{j=1}^n w_j r(x_j) = \int_a^b w(x)r(x) dx, \end{aligned}$$

d.h.

$$\int_a^b w(x)p_n(x)q(x) dx = 0.$$

Durchläuft p die Menge Π_{2n-1} , so durchläuft q in der obigen Zerlegung von p die Menge aller Polynome Π_{n-1} vom Höchstgrad $n - 1$. Notwendig für die Exaktheit

der Quadraturformel G_n aus (3.22) ist also, dass p_n orthogonal zu Π_{n-1} bzgl. des inneren Produktes

$$\langle f, g \rangle_w := \int_a^b w(x) f(x) g(x) dx \quad , \quad f, g \in C[a, b]$$

ist. Um Quadraturformeln maximaler Ordnung zu bestimmen untersuchen wir daher orthogonale Polynome bzgl. des inneren Produktes $\langle \cdot, \cdot \rangle_w$ und ihre Nullstellen.

Wir bezeichnen mit

$$\bar{\Pi}_j := \left\{ p \in \Pi_j : p(x) = x^j + \sum_{i=0}^{j-1} a_i x^i \right\}$$

die Menge der normierten Polynome.

Lemma 3.22. *Es gibt eindeutig bestimmte Polynome $p_j \in \bar{\Pi}_j$, $j \in \mathbb{N}_0$, mit*

$$\langle p_j, p_k \rangle_w = 0 \quad \text{für } j \neq k.$$

Beweis: (durch Orthogonalisierung nach E. Schmidt)

$p_0 \in \bar{\Pi}_0$ ist durch die Normierungsbedingung eindeutig bestimmt als $p_0 \equiv 1$.

Es seien p_0, \dots, p_i , $i \geq 0$ bereits bestimmt, und es sei $q_{i+1} \in \bar{\Pi}_{i+1}$. Dann existieren eindeutige $\alpha_0, \dots, \alpha_i \in \mathbb{R}$ mit

$$q_{i+1}(x) = x^{i+1} + \sum_{j=0}^i \alpha_j p_j(x),$$

denn p_0, \dots, p_i sind linear unabhängig und spannen Π_i auf.

Es folgt

$$\begin{aligned} \langle q_{i+1}, p_k \rangle_w &= \langle x^{i+1}, p_k \rangle_w + \sum_{j=0}^i \alpha_j \langle p_j, p_k \rangle_w \\ &= \langle x^{i+1}, p_k \rangle_w + \alpha_k \langle p_k, p_k \rangle_w, \quad k = 0, \dots, i. \end{aligned}$$

Wegen

$$\langle p_k, p_k \rangle_w = \int_a^b w(x) p_k^2(x) dx > 0$$

gilt also $\langle q_{i+1}, p_k \rangle = 0$ genau dann, wenn gilt

$$\alpha_k = -\frac{\langle x^{i+1}, p_k \rangle_w}{\langle p_k, p_k \rangle_w}. \quad \blacksquare$$

Da p_0, \dots, p_i den Raum Π_i aufspannen, folgt sofort

Korollar 3.23. Für alle $i \in \mathbb{N}$ steht p_i senkrecht auf dem Raum Π_{i-1} , d.h.

$$\langle p_i, p \rangle_w = 0 \quad \text{für alle } p \in \Pi_{i-1}.$$

Lemma 3.24. Für jedes $n \in \mathbb{N}$ besitzt p_n n reelle, einfache Nullstellen, die alle in (a, b) liegen.

Beweis: Es seien $x_1, \dots, x_k \in (a, b)$ die Stellen, in denen p_n sein Vorzeichen wechselt. Die x_i sind dann insbesondere Nullstellen von p_n und wegen

$$\int_a^b w(x)p_n(x) dx = 0$$

gibt es ein x_j mit dieser Eigenschaft.

Wir zeigen, dass $k = n$ ist. Da p_n höchstens n Nullstellen besitzt, sind die x_j dann einfache (und die einzigen) Nullstellen von p_n .

Sicher ist $k \leq n$. Ist $k < n$, so gilt für

$$q(x) := \prod_{j=1}^k (x - x_j) \in \Pi_k$$

wegen Korollar 3.23. $\langle p_n, q \rangle_w = 0$. Andererseits hat aber $p_n \cdot q$ konstantes Vorzeichen in (a, b) und ist nicht identisch 0. Also gilt $\langle p_n, q \rangle_w \neq 0$. ■

Lemma 3.25. Seien $t_i \in \mathbb{R}$, $i = 1, \dots, n$, mit $t_i \neq t_j$ für $i \neq j$. Dann ist die Matrix

$$\mathbf{A} = (p_i(t_j))_{\substack{i=0, \dots, n-1 \\ j=1, \dots, n}}$$

regulär.

Beweis: Ist \mathbf{A} singulär, so existiert $\mathbf{c} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$ mit $\mathbf{c}^T \mathbf{A} = \mathbf{0}^T$. Also besitzt das Polynom

$$q(x) := \sum_{i=0}^{n-1} c_i p_i(x) \in \Pi_{n-1}$$

n Nullstellen t_1, \dots, t_n . Daher gilt $q(x) \equiv 0$, und die lineare Unabhängigkeit von p_0, \dots, p_{n-1} liefert den Widerspruch $c_0 = c_1 = \dots = c_{n-1} = 0$. ■

Nach diesen Vorbereitungen können wir nun den Hauptsatz beweisen.

Satz 3.26. (i) Seien $a < x_1 < \dots < x_n < b$ die Nullstellen von p_n und $\mathbf{w} = (w_1, \dots, w_n)^T$ die Lösung des linearen Gleichungssystems

$$\sum_{j=1}^n p_i(x_j) w_j = \delta_{i0} \langle p_0, p_0 \rangle_{\mathbf{w}}. \quad (3.23)$$

Dann gilt $w_i > 0$ für $i = 1, \dots, n$ und

$$\int_a^b w(x) p(x) dx = \sum_{i=1}^n w_i p(x_i) \quad \text{für alle } p \in \Pi_{2n-1}. \quad (3.24)$$

(ii) Es gibt keine Knoten $x_i \in (a, b)$ und Gewichte $w_i \in \mathbb{R}$, $i = 1, \dots, n$, so dass (3.24) für alle $p \in \Pi_{2n}$ gilt.

(iii) Gilt (3.24) für gewisse Gewichte $w_i \in \mathbb{R}$ und Knoten $x_i \in [a, b]$, $x_1 \leq x_2 \leq \dots \leq x_n$, so sind die x_i die Nullstellen von p_n , und $\mathbf{w} := (w_1, \dots, w_n)$ löst das Gleichungssystem (3.23).

Beweis: (i): Nach Lemma 3.24. besitzt p_n n einfache Nullstellen $x_1 < \dots < x_n \in (a, b)$, und nach Lemma 3.25. ist die Koeffizientenmatrix von (3.23) regulär. (3.23) besitzt also genau eine Lösung $\mathbf{w} = (w_1, \dots, w_n)^T$.

Es sei $p \in \Pi_{2n-1}$. Dann kann man p schreiben als

$$p(x) = p_n(x)q(x) + r(x), \quad \text{mit } q, r \in \Pi_{n-1}.$$

Mit geeigneten $\alpha_k \in \mathbb{R}$ gilt

$$r(x) = \sum_{k=0}^{n-1} \alpha_k p_k(x).$$

Hiermit ist

$$\begin{aligned} \int_a^b w(x) p(x) dx &= \underbrace{\int_a^b w(x) p_n(x) q(x) dx}_{=0} + \int_a^b w(x) r(x) dx \\ &= \langle r, p_0 \rangle_{\mathbf{w}} = \alpha_0 \langle p_0, p_0 \rangle_{\mathbf{w}}. \end{aligned}$$

Andererseits ist $p_n(x_j) = 0$ für $j = 1, \dots, n$, und

$$\sum_{j=1}^n p_i(x_j) w_j = \delta_{i0} \langle p_0, p_0 \rangle_{\mathbf{w}}.$$

Daher gilt

$$\begin{aligned} \sum_{j=1}^n w_j p(x_j) &= \sum_{j=1}^n w_j p_n(x_j) q(x_j) + \sum_{j=1}^n w_j r(x_j) \\ &= \sum_{j=1}^n w_j \sum_{i=0}^{n-1} \alpha_i p_i(x_j) = \sum_{i=0}^{n-1} \alpha_i \sum_{j=1}^n w_j p_i(x_j) = \alpha_0 \langle p_0, p_0 \rangle_{\mathbf{w}}, \end{aligned}$$

d.h. (3.24) ist erfüllt.

Setzt man speziell

$$\bar{p}_i(x) = \prod_{\substack{j=1 \\ j \neq i}}^n (x - x_j)^2 \in \Pi_{2n-2}, \quad i = 1, \dots, n,$$

in (3.24) ein, so erhält man

$$0 < \int_a^b w(x) \bar{p}_i(x) dx = \sum_{j=1}^n w_j \bar{p}_i(x_j) = w_i \bar{p}_i(x_i),$$

d.h. $w_i > 0$.

(ii) haben wir uns schon überlegt.

(iii) Es seien die Gewichte w_i und die Knoten \bar{x}_i , $i = 1, \dots, n$ mit $\bar{x}_1 \leq \bar{x}_2 \leq \dots \leq \bar{x}_n$ so beschaffen, dass (3.24) für alle $p \in \Pi_{2n-1}$ gilt. Dann sind wegen (ii) die \bar{x}_i von einander verschieden.

Für $p = p_k$, $k = 0, \dots, n-1$, erhält man

$$\sum_{i=1}^n w_i p_k(\bar{x}_i) = \int_a^b w(x) p_k(x) dx = \langle p_k, p_0 \rangle_w = \delta_{k0} \langle p_0, p_0 \rangle_w,$$

d.h. die w_i erfüllen (3.23) mit \bar{x}_i an Stelle von x_i .

Einsetzen von $p(x) := p_n(x)p_k(x)$, $k = 0, \dots, n-1$ in (3.24) liefert

$$\sum_{i=1}^n w_i p_n(\bar{x}_i) p_k(\bar{x}_i) = \int_a^b w(x) p_n(x) p_k(x) dx = \langle p_n, p_k \rangle_w = 0, \quad k = 0, \dots, n-1,$$

d.h. $\mathbf{c} := (w_1 p_n(\bar{x}_1), \dots, w_n p_n(\bar{x}_n))^T$ erfüllt das zu (3.23) gehörige homogene Gleichungssystem. Wegen Lemma 3.25. folgt

$$w_i p_n(\bar{x}_i) = 0, \quad i = 1, \dots, n.$$

Wie in (i) erhält man durch Einsetzen von

$$\bar{p}_i := \prod_{j=1}^n (x - \bar{x}_j)^2$$

in (3.24) $w_i > 0$, und daher folgt $p_n(\bar{x}_i) = 0$. ■

Wir betrachten nun den wichtigen Spezialfall $w(x) \equiv 1$ und ohne Beschränkung der Allgemeinheit $[a, b] = [-1, 1]$.

Satz 3.27. Die orthogonalen Polynome p_k , $k \in \mathbb{N}_0$, zu $w(x) \equiv 1$ und $[a, b] = [-1, 1]$ sind gegeben durch

$$p_k(x) = \frac{k!}{(2k)!} \frac{d^k}{dx^k} \left((x^2 - 1)^k \right). \quad (3.25)$$

Bis auf einen Normierungsfaktor, sind die p_k die **Legendre Polynome**.

Beweis: Es genügt, $p_k \in \bar{\Pi}_k$ und $\langle p_k, p \rangle_w = 0$ für alle $p \in \Pi_{k-1}$ zu zeigen.

Es ist $(x^2 - 1)^k = x^{2k} + q$, $q \in \Pi_{2k-1}$. Daher folgt

$$\begin{aligned} \frac{k!}{(2k)!} \frac{d^k}{dx^k} \left((x^2 - 1)^k \right) &= \frac{k!}{(2k)!} \left(2k(2k-1) \dots (k+1)x^k + \frac{d^k}{dx^k} q \right) \\ &= x^k + \frac{k!}{(2k)!} \frac{d^k}{dx^k} q, \end{aligned}$$

und wegen $\frac{k!}{(2k)!} \frac{d^k}{dx^k} q \in \Pi_{k-1}$ gilt $p_k \in \bar{\Pi}_k$.

$(x^2 - 1)^k$ besitzt in $x = \pm 1$ eine k -fache Nullstelle. Es ist also

$$\left. \frac{d^i}{dx^i} \left\{ (x^2 - 1)^k \right\} \right|_{x=\pm 1} = 0 \quad \text{für } 0 \leq i < k,$$

und daher gilt

$$\begin{aligned} &\int_{-1}^1 p(x) \frac{d^k}{dx^k} \left((x^2 - 1)^k \right) dx \\ &= \left[p(x) \frac{d^{k-1}}{dx^{k-1}} \left((x^2 - 1)^k \right) \right]_{-1}^{+1} - \int_{-1}^1 p'(x) \frac{d^{k-1}}{dx^{k-1}} \left((x^2 - 1)^k \right) dx \\ &= \dots = (-1)^k \int_{-1}^1 p^{(k)}(x) (x^2 - 1)^k dx = 0 \quad \text{für alle } p \in \Pi_{k-1}. \end{aligned}$$

■

Wir geben einige Gewichte und Knoten für den Spezialfall aus Satz 3.27. an.

| n | w_i | x_i |
|-----|--|--|
| 1 | $w_1 = 2$ | $x_1 = 0$ |
| 2 | $w_1 = w_2 = 1$ | $x_2 = -x_1 = \frac{1}{\sqrt{3}}$ |
| 3 | $w_1 = w_3 = \frac{5}{9}, w_2 = \frac{8}{9}$ | $x_3 = -x_1 = \sqrt{\frac{3}{5}}, x_2 = 0$ |

Weitere Werte findet man in Abramowitz und Stegun [1], pp. 916 ff, und in Piessens et al. [85], pp. 19 ff. Ferner wird in Schwarz [93], pp. 352 ff., eine stabile Methode zur Bestimmung der Knoten x_i und der Gewichte w_i beschrieben.

Tabelle 3.4: Beispiel 3.28.

| n | $G_n - I$ |
|-----|---------------|
| 1 | $1.04e + 00$ |
| 2 | $-1.97e - 01$ |
| 3 | $1.18e - 02$ |
| 4 | $-3.04e - 04$ |
| 5 | $3.73e - 06$ |

Beispiel 3.28. Für das Integral

$$I := \int_0^1 e^x \sin(5x) dx$$

erhält man mit den Gauß Quadraturformeln G_n die Fehler der Tabelle 3.4. \square

Bemerkung 3.29. Alle Überlegungen bleiben richtig, wenn man nur fordert, dass w auf dem endlichen oder unendlichen Intervall messbar und nichtnegativ ist und dass alle Momente

$$\mu_k := \int_a^b x^k w(x) dx$$

endlich sind und $\mu_0 > 0$. \square

Satz 3.30. Für $f \in C^{2n}[a, b]$ gilt

$$\int_a^b w(x) f(x) dx - \sum_{i=1}^n w_i f(x_i) = \frac{f^{(2n)}(\xi)}{(2n)!} \langle p_n, p_n \rangle_w$$

Beweis: Sei $h \in \Pi_{2n-1}$ das Hermitesche Interpolationspolynom, das f und f' an den Knoten x_1, \dots, x_n interpoliert.

Dann gilt für den Fehler (vgl. Satz 2.30.)

$$r(x) := f(x) - h(x) = \frac{f^{(2n)}(\eta(x))}{(2n)!} \prod_{i=1}^n (x - x_i)^2, \quad \eta(x) \in I(x_1, \dots, x_n, x).$$

Da die x_i die Nullstellen von p_n sind und $p_n \in \bar{\Pi}_n$ gilt, folgt

$$r(x) = \frac{f^{(2n)}(\eta(x))}{(2n)!} p_n^2(x).$$

Aus der Regel von de l'Hospital folgt, dass

$$f^{(2n)}(\eta(x)) = \frac{f(x) - h(x)}{p_n^2(x)} (2n)!$$

stetig auf $[a, b]$ ist. Da weiter $p_n^2(x) \geq 0$ gilt, liefert der Mittelwertsatz der Integralrechnung

$$\int_a^b w(x)r(x) dx = \frac{f^{(2n)}(\xi)}{(2n)!} \int_a^b w(x)p_n^2(x) dx = \frac{f^{(2n)}(\xi)}{(2n)!} \langle p_n, p_n \rangle_w$$

mit einem geeigneten $\xi \in (a, b)$.

Wegen $h \in \Pi_{2n-1}$ und $h(x_i) = f(x_i)$, $i = 1, \dots, n$, folgt schließlich aus Satz 3.26.

$$\begin{aligned} \int_a^b w(x)r(x) dx &= \int_a^b w(x)f(x) dx - \int_a^b w(x)h(x) dx \\ &= \int_a^b w(x)f(x) dx - \sum_{i=1}^n w_i h(x_i) \\ &= \int_a^b w(x)f(x) dx - \sum_{i=1}^n w_i f(x_i). \end{aligned}$$

■

Bemerkung 3.31. Es ist für $w \equiv 1$

$$\langle p_n, p_n \rangle_w = \frac{(b-a)^{2n+1}}{2n+1} \cdot \frac{(n!)^4}{(2n)!^2},$$

das Restglied hat also in diesem Fall die Gestalt

$$\int_a^b f(x) dx - \sum_{j=1}^n w_j f(x_j) = f^{(2n)}(\xi) \cdot \frac{(b-a)^{2n+1}}{2n+1} \frac{(n!)^4}{((2n)!)^2} \quad \square$$

Satz 3.32. Es seien $x_i^{(n)}$, $i = 1, \dots, n$ die Nullstellen des n -ten orthogonalen Polynoms zur Gewichtsfunktion w auf $[a, b]$ und $w_i^{(n)}$ die zugehörigen Gewichte der Gaußschen Quadraturformel. Dann gilt

$$G_n(f) := \sum_{i=1}^n w_i^{(n)} f(x_i^{(n)}) \rightarrow \int_a^b w(x)f(x) dx \quad \text{für alle } f \in C[a, b].$$

Beweis: Nach dem Approximationssatz von Weierstrass gibt es zu $\varepsilon > 0$ ein Polynom p mit $\|f - p\|_\infty < \varepsilon$. Für genügend großes $n \in \mathbb{N}$ ist wegen Satz 3.26.

$$G_n(p) = \int_a^b w(x)p(x) dx$$

und wegen der Positivität der $w_i^{(n)}$ folgt

$$\begin{aligned}
& \left| G_n(f) - \int_a^b w(x)f(x) dx \right| \\
& \leq |G_n(f) - G_n(p)| + \left| G_n(p) - \int_a^b w(x)f(x) dx \right| \\
& = \left| \sum_{i=1}^n w_i^{(n)} (f(x_i^{(n)}) - p(x_i^{(n)})) \right| + \left| \int_a^b w(x)(p(x) - f(x)) dx \right| \\
& \leq \left(\sum_{i=1}^n w_i^{(n)} + \int_a^b w(x) dx \right) \|f - p\|_\infty = 2 \int_a^b w(x) dx \|f - p\|_\infty
\end{aligned}$$

■

Bei einigen Anwendungen, insbesondere der Diskretisierung von Anfangswertaufgaben oder von Integralgleichungen, ist es erforderlich, dass ein Randpunkt des Integrationsintervalls oder beide Randpunkte Knoten der Quadraturformel sind. Wir betrachten (gleich etwas allgemeiner) eine Quadraturformel

$$\int_a^b w(x)f(x) dx \approx \sum_{j=1}^m v_j f(y_j) + \sum_{j=1}^n w_j f(x_j), \quad (3.26)$$

wobei die Knoten y_1, \dots, y_m gegeben sind und die Knoten x_1, \dots, x_n und die Gewichte v_1, \dots, v_m und w_1, \dots, w_n so zu bestimmen sind, dass Polynome von möglichst hohem Grad exakt integriert werden. Da in dieser Formel $2n + m$ Parameter frei sind, kann man hoffen, dass der erreichbare Polynomgrad $2n + m - 1$ ist.

Wir bezeichnen mit r und s die Polynome

$$r(x) := \prod_{j=1}^m (x - y_j) \quad \text{und} \quad s(x) := \prod_{j=1}^n (x - x_j).$$

Satz 3.33. *Die Quadraturformel (3.26) ist genau dann exakt für alle Polynome vom Höchstgrad $2n + m - 1$, wenn gilt:*

(i) (3.26) ist exakt für alle Polynome vom Höchstgrad $m + n - 1$

(ii) $\int_a^b w(x)r(x)s(x)p(x) dx = 0$ für alle $p \in \Pi_{n-1}$.

Beweis: Die Notwendigkeit von (i) ist trivial.

Sei $p \in \Pi_{n-1}$. Dann gilt $q := r \cdot s \cdot p \in \Pi_{2n+m-1}$, und daher

$$\int_a^b w(x)q(x) dx = \sum_{j=1}^m v_j q(y_j) + \sum_{j=1}^n w_j q(x_j) = 0.$$

Sind umgekehrt (i) und (ii) erfüllt und ist $q \in \Pi_{2n+m-1}$, so gibt es $\phi \in \Pi_{n-1}$ und $\psi \in \Pi_{m+n-1}$ mit $q = r \cdot s \cdot \phi + \psi$. Es ist $q(y_j) = \psi(y_j)$, $j = 1, \dots, m$, und $q(x_j) = \psi(x_j)$, $j = 1, \dots, n$. Daher gilt nach (ii)

$$\int_a^b w(x)q(x) dx = \int_a^b w(x)(r(x)s(x)\phi(x) + \psi(x)) dx = \int_a^b w(x)\psi(x) dx,$$

und nach (i) gilt weiter

$$\int_a^b w(x)\psi(x) dx = \sum_{j=1}^m v_j \psi(y_j) + \sum_{j=1}^n w_j \psi(x_j) = \sum_{j=1}^m v_j q(y_j) + \sum_{j=1}^n w_j q(x_j)$$

■

Die freien Knoten x_j der Quadraturformel (3.26) müssen wir wegen der Bedingung (ii) als Nullstellen des n -ten orthogonalen Polynoms auf $[a, b]$ bzgl. der Gewichtsfunktion $w(x)r(x)$ bestimmen. Hilfreich bei der Konstruktion dieses Polynoms ist der folgende Satz.

Satz 3.34. *Es sei $p_n \in \Pi_n$ das n -te orthogonale Polynom auf $[a, b]$ bzgl. der Gewichtsfunktion $w(x)$. Die festen Knoten y_j seien voneinander verschieden, und es sei $r(x) = \prod_{j=1}^m (x - y_j)$ von einem Vorzeichen auf $[a, b]$. Dann erfüllt das n -te orthogonale Polynom q_n auf $[a, b]$ bzgl. der Gewichtsfunktion $w(x)r(x)$ die Gleichung*

$$r(x)q_n(x) = \det \begin{pmatrix} p_n(x) & p_{n+1}(x) & \dots & p_{n+m}(x) \\ p_n(y_1) & p_{n+1}(y_1) & \dots & p_{n+m}(y_1) \\ \dots & \dots & \dots & \dots \\ p_n(y_m) & p_{n+1}(y_m) & \dots & p_{n+m}(y_m) \end{pmatrix} =: \psi(x).$$

Beweis: Offensichtlich gilt $\psi \in \Pi_{n+m}$. Da ψ die Nullstellen y_1, \dots, y_m besitzt, ist ψ durch r teilbar, und daher gilt $\psi = r \cdot q_n$ für ein $q_n \in \Pi_n$.

q_n ist orthogonal zu Π_{n-1} bzgl. des inneren Produktes $\langle \cdot, \cdot \rangle_{wr}$, denn $r q_n$ besitzt eine Darstellung

$$r(x)q_n(x) = \sum_{j=0}^m c_j p_{n+j}(x), \quad c_j \in \mathbb{R},$$

und daher gilt für $q \in \Pi_{n-1}$

$$\int_a^b w(x)r(x)q_n(x)q(x) dx = \int_a^b w(x) (c_0p_n(x) + \dots + c_m p_{n+m}(x)) q(x) dx = 0.$$

Zu zeigen bleibt, dass q_n den Grad n besitzt, d.h. dass der Koeffizient

$$c_m = \det \begin{pmatrix} p_n(y_1) & \dots & p_{n+m-1}(y_1) \\ \dots & \dots & \dots \\ p_n(y_m) & \dots & p_{n+m-1}(y_m) \end{pmatrix}$$

in der Darstellung von $r q_n$ als Linearkombination von p_n, \dots, p_{n+m} nicht verschwindet.

Ist $c_m = 0$, so existiert $\mathbf{d} = (d_1, \dots, d_m)^T \in \mathbb{R}^m \setminus \{\mathbf{0}\}$, so dass das Polynom

$$\phi(x) := \sum_{j=1}^m d_j p_{n+j-1}(x) \in \Pi_{n+m-1}$$

die Nullstellen y_1, \dots, y_m besitzt. Daher ist ϕ teilbar durch r , d.h. es existiert $\chi \in \Pi_{n-1}$ mit $\phi = r \cdot \chi$, und da ϕ orthogonal zu Π_{n-1} auf $[a, b]$ bzgl. der Gewichtsfunktion w ist, folgt

$$0 = \int_a^b w(x)\phi(x)\chi(x) dx = \int_a^b w(x)r(x)\chi^2(x) dx,$$

d.h. $\chi(x) \equiv 0$ im Widerspruch zu $\mathbf{d} \neq \mathbf{0}$. ■

Wir betrachten nun den Spezialfall $[a, b] = [-1, 1]$ und $w(x) \equiv 1$.

In den Beweisen der letzten beiden Sätze wurde nur benutzt, dass die Polynome p_j orthogonal sind, nicht aber die Normierung, dass der führende Koeffizient den Wert 1 besitzt. Wir normieren nun die Legendre Polynome \tilde{p}_n so, dass $\tilde{p}_n(1) = 1$ für alle $n \in \mathbb{N}_0$ gilt.

Geben wir den rechten Randpunkt $y_1 = 1$ des Intervalls als Knoten fest vor, so müssen die n freien Knoten x_1, \dots, x_n der Quadraturformel mit maximaler Ordnung die Nullstellen des Polynoms $q_n \in \Pi_n$ sein, das definiert ist durch

$$(x - 1)q_n(x) = \det \begin{pmatrix} \tilde{p}_n(x) & \tilde{p}_{n+1}(x) \\ \tilde{p}_n(1) & \tilde{p}_{n+1}(1) \end{pmatrix} = \det \begin{pmatrix} \tilde{p}_n(x) & \tilde{p}_{n+1}(x) \\ 1 & 1 \end{pmatrix} = \tilde{p}_n(x) - \tilde{p}_{n+1}(x).$$

Die Gewichte zu den Knoten y_1 und x_1, \dots, x_n kann man dann wie in Abschnitt 3.1 bestimmen.

Die entstehenden Quadraturformeln heißen **Radau Formeln**. Durch sie werden Polynome vom Höchstgrad $2n$ exakt integriert.

Geben wir beide Randpunkte $y_1 = -1$ und $y_2 = 1$ des Intervalls als Knoten vor, so müssen die freien Knoten x_1, \dots, x_n der Quadraturformel mit maximaler Ordnung die Nullstellen des Polynoms $q_n \in \Pi_n$ sein, das definiert ist durch

$$(x^2 - 1)q_n(x) = \pm \det \begin{pmatrix} \tilde{p}_n(x) & \tilde{p}_{n+1}(x) & \tilde{p}_{n+2}(x) \\ 1 & -1 & 1 \\ 1 & 1 & 1 \end{pmatrix} = \pm 2(\tilde{p}_{n+2}(x) - \tilde{p}_n(x)).$$

Man beachte, dass $\tilde{p}_n(x)$ für ungerades n eine ungerade Funktion ist und daher $\tilde{p}_n(-1) = (-1)^n$ gilt. Die entstehenden Quadraturformeln heißen **Lobatto Formeln**. Durch sie werden Polynome vom Höchstgrad $2n + 1$ exakt integriert.

Als Beispiel geben wir die Lobatto Formel der Ordnung 5 an:

$$Q(f) = \frac{1}{6} \left(f(-1) + 5f\left(-\frac{1}{\sqrt{5}}\right) + 5f\left(\frac{1}{\sqrt{5}}\right) + f(1) \right).$$

Weitere Knoten und Gewichte von Lobatto Formeln findet man in Abramowitz und Stegun [1], p. 920.

Bemerkung 3.35. Auch für unbeschränkte Integrationsintervalle kann man Gaußsche Quadraturformeln wie in Satz 3.26. mit Hilfe von orthogonalen Polynomen entwickeln. So erhält man für Integrale der Gestalt

$$\int_0^{\infty} e^{-x} f(x) dx$$

(mit den Nullstellen der **Laguerre Polynome** als Knoten) die **Gauß–Laguerre Quadraturformeln**, und für Integrale der Gestalt

$$\int_{-\infty}^{\infty} e^{-x^2} f(x) dx$$

(mit den Nullstellen der **Hermite Polynome** als Knoten) die **Gauß–Hermite Quadraturformeln**.

Knoten und Gewichte der Gauß–Laguerre Formeln (für $n \leq 15$) findet man in Abramowitz und Stegun [1], p. 923, und für die Gauß–Hermite Formeln (bis $n = 20$) auf Seite 924. \square

Beispiel 3.36. Für das Integral

$$\int_0^{\infty} \frac{x}{1 + e^x} dx = \int_0^{\infty} e^{-x} \frac{x}{1 + e^{-x}} dx = \frac{\pi^2}{12} = 0.8224670334241132$$

enthält Tabelle 3.5 die Näherungen mit den Gauß–Laguerre Quadraturformeln und die Fehler. Man sieht, dass man auch mit wenigen Knoten zu sehr guten Näherungen gelangt. \square

Tabelle 3.5: Quadraturformeln von Gauß–Laguerre

| n | Q_n | Fehler |
|-----|--------------------|------------|
| 1 | 0.7310585786300049 | $9.14e-2$ |
| 2 | 0.8052717896130982 | $1.72e-2$ |
| 3 | 0.8238172597250991 | $-1.35e-3$ |
| 4 | 0.8236994602380588 | $-1.23e-3$ |
| 5 | 0.8226695411616926 | $-2.03e-4$ |
| 6 | 0.8224050273750929 | $6.20e-5$ |

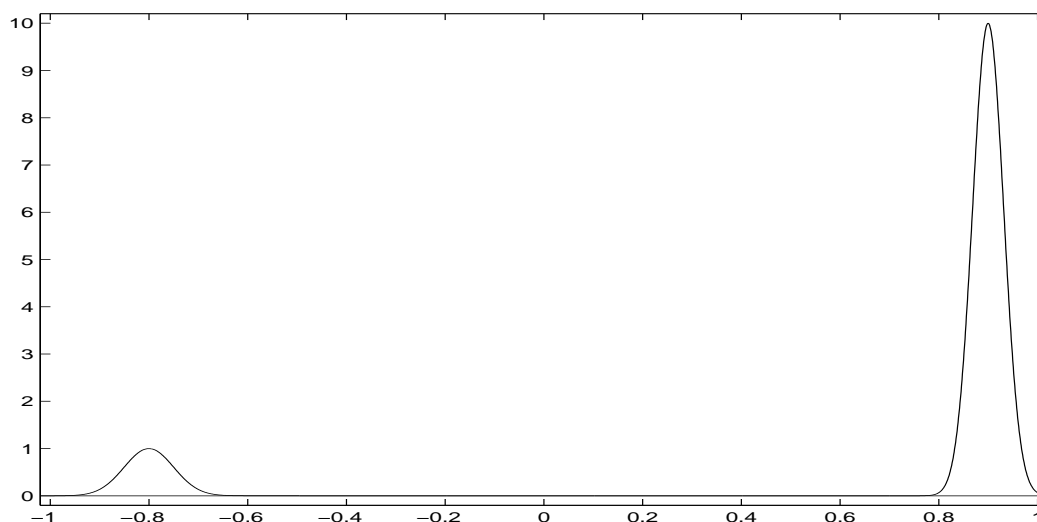


Abbildung 3.3: Zu Beispiel 3.37.

3.5 Adaptive Quadratur

Wir haben summierte Quadraturformeln nur mit konstanter Schrittweite $h > 0$ betrachtet. Es ist klar, dass man dabei für Funktionen mit unterschiedlichem Verhalten in verschiedenen Teilen des Integrationsintervalls entweder bei zu groß gewähltem h ein ungenaues Ergebnis erhält oder bei zu kleinem h Arbeit in den Bereichen verschwendet, in denen die Funktion “gutartig” ist.

Beispiel 3.37. (siehe dazu Abbildung 3.3)

$$f(x) = \exp(-200(x + 0.8)^2) + 10 \cdot \exp(-500(x - 0.9)^2), \quad -1 \leq x \leq 1. \quad \square$$

Wir entwickeln nun eine Vorgehensweise, mit der bei gegebenem Integranden, gegebenen Grenzen a und b und gegebener Genauigkeitsschranke $\varepsilon > 0$ eine Quadraturformel

$$I(f) = \sum_{j=0}^k w_j f(x_j)$$

erzeugt wird, für die gilt

$$\left| \int_a^b f(x) dx - I(f) \right| < \varepsilon,$$

wobei der Punkt über dem Ungleichungszeichen besagt, dass die Ungleichung nur asymptotisch für feine Zerlegungen $a \leq x_0 < x_1 < \dots < x_k \leq b$ von $[a, b]$ gilt.

Die Knoten x_j und Gewichte w_j werden adaptiv durch das Verfahren in Abhängigkeit von f und ε erzeugt.

Es sei

$$Q(f) = \sum_{i=1}^n w_i f(t_i)$$

eine Quadraturformel der Ordnung m für das Referenzintervall $[-1, 1]$.

Es sei das Integral bis zum Punkt $x_j \in [a, b)$ schon näherungsweise bestimmt, und es sei $x_{j+1} \in (x_j, b]$ gegeben. Dann berechnen wir die zwei Näherungen

$$Q_{[x_j, x_{j+1}]}(f) = \frac{x_{j+1} - x_j}{2} \sum_{i=1}^n w_i f\left(\frac{x_{j+1} + x_j}{2} + t_i \frac{x_{j+1} - x_j}{2}\right)$$

und

$$\begin{aligned} \tilde{Q}_{[x_j, x_{j+1}]}(f) &:= \frac{x_{j+\frac{1}{2}} - x_j}{2} \sum_{i=1}^n w_i f\left(\frac{x_{j+\frac{1}{2}} + x_j}{2} + t_i \frac{x_{j+\frac{1}{2}} - x_j}{2}\right) \\ &\quad + \frac{x_{j+1} - x_{j+\frac{1}{2}}}{2} \sum_{i=1}^n w_i f\left(\frac{x_{j+1} + x_{j+\frac{1}{2}}}{2} + t_i \frac{x_{j+1} - x_{j+\frac{1}{2}}}{2}\right) \end{aligned}$$

für $\int_{x_j}^{x_{j+1}} f(x) dx$, wobei $x_{j+\frac{1}{2}} := 0.5(x_j + x_{j+1})$ gesetzt ist, und hiermit

$$\hat{Q}_{[x_j, x_{j+1}]} := \frac{1}{2^m - 1} \left(2^m \cdot \tilde{Q}_{[x_j, x_{j+1}]}(f) - Q_{[x_j, x_{j+1}]}(f) \right).$$

Dann ist mit \tilde{Q} und Q auch \hat{Q} eine Quadraturformel von mindestens der Ordnung m , und man kann leicht mit Hilfe der Fehlerdarstellung aus Satz 3.10. zeigen, dass durch \hat{Q} sogar Polynome vom Grade m exakt integriert werden, \hat{Q} also wenigstens die Ordnung $m + 1$ hat.

Wir benutzen nun \hat{Q} , um den Fehler von \tilde{Q} (der genaueren der beiden Ausgangsformeln) zu schätzen.

Es gilt mit $h := x_{j+1} - x_j$

$$\tilde{E}_{[x_j, x_{j+1}]}(f) := \int_{x_j}^{x_{j+1}} f(x) dx - \tilde{Q}_{[x_j, x_{j+1}]}(f)$$

$$\begin{aligned}
&= \hat{Q}_{[x_j, x_{j+1}]}(f) - \tilde{Q}_{[x_j, x_{j+1}]}(f) + O(h^{m+2}) \\
&= \frac{1}{2^m - 1} \left(2^m \tilde{Q}_{[x_j, x_{j+1}]}(f) - Q_{[x_j, x_{j+1}]}(f) \right) - \tilde{Q}_{[x_j, x_{j+1}]}(f) + O(h^{m+2}) \\
&= \frac{1}{2^m - 1} \left(\tilde{Q}_{[x_j, x_{j+1}]}(f) - Q_{[x_j, x_{j+1}]}(f) \right) + O(h^{m+2}).
\end{aligned}$$

Da andererseits $\tilde{E}_{[x_j, x_{j+1}]}(f) = O(h^{m+1})$ gilt, können wir für kleine h den Summanden $O(h^{m+2})$ vernachlässigen und erhalten die Fehlerschätzung

$$\tilde{E}_{[x_j, x_{j+1}]}(f) \approx \frac{1}{2^m - 1} \left(\tilde{Q}_{[x_j, x_{j+1}]}(f) - Q_{[x_j, x_{j+1}]}(f) \right). \quad (3.27)$$

Wir benutzen (3.27), um das Intervall $[a, b]$ durch Bisektion zu zerlegen in $a = x_0 < x_1 < \dots < x_k = b$, so dass für $j = 1, \dots, k$ gilt

$$\left| \tilde{Q}_{[x_{j-1}, x_j]}(f) - Q_{[x_{j-1}, x_j]}(f) \right| \leq \frac{2^m - 1}{b - a} (x_j - x_{j-1}) \varepsilon. \quad (3.28)$$

Dann folgt für die summierte Quadraturformel

$$Q(f) := \sum_{j=1}^k \tilde{Q}_{[x_{j-1}, x_j]}(f)$$

aus (3.27)

$$\begin{aligned}
E(f) &= \int_a^b f(x) dx - Q(f) = \sum_{j=1}^k \tilde{E}_{[x_{j-1}, x_j]}(f) \\
&\leq \frac{1}{2^m - 1} \sum_{j=1}^k \left| \tilde{Q}_{[x_{j-1}, x_j]}(f) - Q_{[x_{j-1}, x_j]}(f) \right| \\
&\leq \frac{1}{b - a} \sum_{j=1}^k (x_j - x_{j-1}) \varepsilon = \varepsilon.
\end{aligned}$$

Das folgende PASCAL Programm ist eine Realisierung des adaptiven Verfahrens unter Benutzung der Gauß Formel der Ordnung $m = 6$.

Algorithmus 3.38. (Adaptive Gauß Integration; rekursiv)

FUNCTION Adaptive_Gauss (a, b, eps: REAL) : REAL;

```
{ Adaptive_Gauss berechnet das Integral der Funktion f in den
  Grenzen von a bis b mit der relativen Genauigkeit eps.
  f muss als FUNCTION bereitgestellt werden. }
```

```

VAR l, kn : REAL;

FUNCTION Gauss_3 (a, b: REAL) : REAL;
  VAR mp : REAL;
  BEGIN
    mp := (a+b) / 2;
    Gauss_3 := (b-a)/2 *
      (5 * f(mp-kn*(b-a)/2) + 5 * f(mp+kn*(b-a)/2) + 8 * f(mp)) / 9
  END; { OF GAUSS_3 }

FUNCTION Adaption (a, b, p: REAL) : REAL;
  VAR c, h, ac_int, cb_int, q : REAL;
  BEGIN
    c := (a+b) / 2;
    h := b - a;
    ac_int := Gauss_3 (a, c);
    cb_int := Gauss_3 (c, b);
    q := ac_int + cb_int;
    IF ABS(q-p) > h*1 THEN
      Adaption := Adaption (a, c, ac_int) + Adaption (c, b, cb_int)
    ELSE
      Adaption := q + (q-p) / 63 {+}
    END; { OF ADAPTION }

BEGIN
  l := 63 * eps / (b-a);
  kn := SQRT (0.6);
  Adaptive_Gauss := Adaption (a, b, Gauss_3 (a, b))
END; { OF ADAPTIVE_GAUSS }

```

Wir haben die Fehlerschätzung für die Quadraturformel \tilde{Q} durchgeführt. Da uns die Formel \hat{Q} höherer Ordnung zur Verfügung steht, haben wir im Programm \hat{Q} zur Berechnung einer Näherung für $\int_{x_j}^{x_{j+1}} f(x) dx$ verwendet (Anweisung $\{+\}$).

Tabelle 3.6: Adaptive Gauß Quadratur

| $x(i-1)$ | $x(i)$ | $x(i) - x(i-1)$ |
|----------|----------|-----------------|
| -1.00000 | -0.75000 | 0.25000 |
| -0.75000 | -0.50000 | 0.25000 |
| -0.50000 | 0.00000 | 0.50000 |
| 0.00000 | 0.50000 | 0.50000 |
| 0.50000 | 0.75000 | 0.25000 |
| 0.75000 | 0.87500 | 0.12500 |
| 0.87500 | 0.93750 | 0.06250 |
| 0.93750 | 1.00000 | 0.06250 |

Beispiel 3.39. Für das Beispiel 3.37.

$$f(x) = \exp(-200(x+0.8)^2) + 10 \cdot \exp(-500(x-0.9)^2), \quad -1 \leq x \leq 1,$$

erhält man mit Algorithmus 3.38. mit der Genauigkeitsschranke $\varepsilon = 1E-3$ die Knotenverteilung in Tabelle 3.6 und mit 93 Funktionsauswertungen den Näherungswert 0.917542. Der tatsächliche Fehler ist hierfür $1.7E-4$. \square

Eine andere Möglichkeit besteht darin, schrittweise vorzugehen. Ist also das Integral

$$\int_a^{x_j} f(x) dx$$

schon mit der gewünschten Genauigkeit bestimmt und wurden mit der Schrittweite h die Näherungen $Q_{[x_j, x_j+h]}(f)$ und $\tilde{Q}_{[x_j, x_j+h]}(f)$ berechnet, so kann man hiermit das Erfülltsein der lokalen Fehlerschranke (3.28) prüfen. Ist dies der Fall, so geht man zu dem neuen Intervall $[x_{j+1}, x_{j+1} + h_{\text{neu}}]$, $x_{j+1} := x_j + h$, über. Sonst wiederholt man den Schritt mit einer verkleinerten Schrittweite h_{neu} .

Die neue Schrittweite kann man so bestimmen: Es ist

$$E_{[x_j, x_j+h]} \approx \frac{1}{2^m - 1} |Q_{[x_j, x_j+h]}(f) - \tilde{Q}_{[x_j, x_j+h]}(f)| \approx Ch^{m+1},$$

d.h.

$$C \approx \frac{h^{-m-1}}{2^m - 1} |Q_{[x_j, x_j+h]}(f) - \tilde{Q}_{[x_j, x_j+h]}(f)|,$$

und daher kann man erwarten, dass mit

$$\varepsilon \frac{h_{\text{neu}}}{b-a} = Ch_{\text{neu}}^{m+1} = \frac{h^{-m-1}}{2^m - 1} |Q_{[x_j, x_j+h]}(f) - \tilde{Q}_{[x_j, x_j+h]}(f)| h_{\text{neu}}^{m+1},$$

d.h.

$$h_{\text{neu}} = h \left(\frac{(2^m - 1)h\varepsilon}{(b-a)|Q_{[x_j, x_j+h]}(f) - \tilde{Q}_{[x_j, x_j+h]}(f)|} \right)^{1/m}$$

die lokale Fehlerschranke (3.28) eingehalten wird. Hiermit erhält man das folgende Verfahren:

Tabelle 3.7: Adaptive Gauß Quadratur

| ε | Fehler | Funktionsauswertungen |
|---------------|------------|-----------------------|
| $1E-3$ | $8.78E-02$ | 54 |
| $1E-4$ | $2.99E-05$ | 153 |
| $1E-5$ | $2.63E-07$ | 270 |
| $1E-6$ | $8.22E-09$ | 351 |
| $1E-7$ | $2.10E-09$ | 531 |
| $1E-8$ | $8.02E-12$ | 747 |

Algorithmus 3.40. (Adaptive Gauß Quadratur)

```

function int=adap_gauss3(f,a,b,tol);
factor=63*tol/(b-a); h=0.1*(b-a); int=0;
while a < b
    q=gauss3(f,a,a+h);
    qs=gauss3(f,a,a+0.5*h)+gauss3(f,a+0.5*h,a+h);
    h_neu=0.9*h*(h*factor/abs(qs-q))^(1/6);
    if abs(q-qs) > h*factor;
        h=h_neu;
    else
        int=int+qs+(qs-q)/63;
        a=a+h;
        h=min(h_neu,b-a);
    end
end
end

```

Beispiel 3.41. Für das Beispiel 3.37. erhält man hiermit die Ergebnisse aus Tabelle 3.7. Wie erwartet wird der tatsächliche Fehler (jedenfalls für $\varepsilon \leq 1E-04$) deutlich kleiner als die vorgegebene Toleranz ε . \square

Verwendet man wie oben in einem adaptiven Verfahren dieselbe Gauß Formel für Q und \tilde{Q} , so kann man die an den Knoten von Q berechneten Funktionswerte bei der Auswertung von \tilde{Q} nicht wiederverwenden. Von Kronrod [67] wurde 1965 die folgende Vorgehensweise vorgeschlagen, die diesen Nachteil nicht hat:

Wir gehen aus von einer Gauß Formel

$$G_n(f) = \sum_{i=1}^n w_i f(x_i)$$

der Ordnung $2n$ mit den Knoten $x_1, \dots, x_n \in (-1, 1)$, und bestimmen $n + 1$ weitere Knoten $y_0, \dots, y_n \in (-1, 1)$ und Gewichte α_i, β_i , so dass die Quadraturformel

$$K_n(f) := \sum_{i=1}^n \alpha_i f(x_i) + \sum_{i=0}^n \beta_i f(y_i)$$

möglichst hohe Ordnung besitzt. K_n heißt die zu G_n gehörige **Kronrod Formel**.

Beispiel 3.42. Ausgehend von

$$G_2(f) = f\left(-\frac{1}{\sqrt{3}}\right) + f\left(\frac{1}{\sqrt{3}}\right)$$

machen wir für K_2 den Ansatz

$$K_2(f) = \alpha_1 f\left(-\frac{1}{\sqrt{3}}\right) + \alpha_2 f\left(\frac{1}{\sqrt{3}}\right) + \beta_0 f(y_0) + \beta_1 f(y_1) + \beta_2 f(y_2)$$

und bestimmen die 8 Unbekannten $\alpha_1, \alpha_2, \beta_0, \beta_1, \beta_2, y_0, y_1, y_2$ so, dass die Funktionen $x^j, j = 0, 1, 2, \dots, m$, für möglichst großes m durch K_2 exakt integriert werden.

Man kann zeigen, dass die Kronrod Formeln symmetrisch sind (hier: $\alpha_1 = \alpha_2, \beta_0 = \beta_2, y_0 = -y_2, y_1 = 0$). Unter Ausnutzung dieser Symmetrie folgt

$$K_2(x^{2j+1}) = 0 = \int_{-1}^1 x^{2j+1} dx \quad \text{für alle } j = 0, 1, 2, \dots$$

Für die geraden Potenzen ergibt sich das nichtlineare Gleichungssystem

$$\begin{aligned} x^0 &: 2\alpha_1 + 2\beta_0 + \beta_1 = 2, \\ x^2 &: \frac{2}{3}\alpha_1 + 2\beta_0 y_0^2 = \frac{2}{3}, \\ x^4 &: \frac{2}{9}\alpha_1 + 2\beta_0 y_0^4 = \frac{2}{5}, \\ x^6 &: \frac{2}{27}\alpha_1 + 2\beta_0 y_0^6 = \frac{2}{7}, \end{aligned}$$

mit der eindeutigen Lösung

$$y_0 = \sqrt{\frac{6}{7}}, \quad \alpha_1 = \frac{243}{495}, \quad \beta_0 = \frac{98}{495}, \quad \beta_1 = \frac{308}{495},$$

d.h.

$$K_2(f) = \frac{243}{495} G_2(f) + \frac{1}{495} \left(98 \left(f\left(-\sqrt{\frac{6}{7}}\right) + f\left(\sqrt{\frac{6}{7}}\right) \right) + 308 f(0) \right).$$

Nach Konstruktion hat diese Formel mindestens die Ordnung 8, und durch Berechnung von $E(x^8)$ sieht man, dass die Ordnung genau 8 ist. \square

Man kann zeigen, dass zu einer n -Punkt Gauß Formel G_n stets die $(2n + 1)$ -Punkt Kronrod Formel konstruiert werden kann, und dass ihre Ordnung $3n + 2$ ist, falls n gerade ist, und $3n + 3$, falls n ungerade ist.

Die Kronrod Formel K_n kann man nun auf folgende Weise nutzen, um den Fehler E_n der zugehörigen Gauß Formel zu schätzen. Es gilt für $[x_j, x_{j+1}] \subset [-1, 1]$

$$\int_{x_j}^{x_{j+1}} f(x) dx = K_n(f) + h^{3n+2} \cdot c_{3n+2} \cdot f^{(3n+2)}(\eta),$$

$h := x_{j+1} - x_j$, und daher folgt

$$\begin{aligned} E_n &:= \int_{x_j}^{x_{j+1}} f(x) dx - G_n(f) \\ &= K_n(f) - G_n(f) + h^{3n+2} \cdot c_{3n+2} \cdot f^{(3n+2)}(\eta). \end{aligned}$$

Da E_n proportional zu h^{2n} ist, können wir für kleine h den letzten Summanden vernachlässigen und erhalten

$$E_n \approx K_n(f) - G_n(f).$$

In dem folgenden Algorithmus schätzen wir hiermit den Fehler der Gauß Formel, verwenden aber als Näherung für das Integral $\int_{x_j}^{x_{j+1}} f(x) dx$ den mit der Kronrod Formel ermittelten Wert. Dies führt dazu, dass der Fehler wesentlich unterhalb der geforderten Toleranz liegt.

Algorithmus 3.43. (Adaptive Kronrod Quadratur; rekursiv)

```
FUNCTION Kronrod_Gauss (a, b, eps: REAL) : REAL;
```

```
{ Kronrod_Gauss berechnet das Integral der Funktion f in den
  Grenzen von a bis b mit der asymptotischen Genauigkeit eps.
  f muss als FUNCTION bereitgestellt werden. }
```

```
VAR k1, k2, Tol : REAL;
```

```
PROCEDURE Integral (a, b: REAL; VAR Kronrod, Gauss : REAL);
```

```

VAR mp, h, x : REAL;
BEGIN
  mp := (a+b) / 2;
  h := (b-a) / 2;
  x := f(mp-h*k1) + f(mp+h*k1);
  Gauss := h * x;
  Kronrod := h * (98 * (f(mp-h*k2) + f(mp+h*k2))
                 + 243*x + 308*f(mp)) / 495
END; { OF INTEGRAL }

FUNCTION Adaption (a, b: REAL) : REAL;
VAR mp, p, q, h : REAL;
BEGIN
  mp := (a+b) / 2;
  h := b - a;
  Integral (a, b, q, p);
  IF ABS(q-p) > Tol THEN
    Adaption := Adaption (a, mp) + Adaption (mp, b)
  ELSE
    Adaption := q
  END; { OF ADAPTION }

BEGIN
  Tol := eps / (b-a);
  k1 := 1 / SQRT (3);
  k2 := SQRT (6/7);
  Kronrod_Gauss := Adaption (a, b)
END; { OF KRONROD_GAUSS }

```

Beispiel 3.44. Mit $\varepsilon = 1E - 2$ erhält man für Beispiel 3.37. die Knotenverteilung in Tabelle 3.8 und mit 85 Funktionsauswertungen die Näherung 0.917405. Der tatsächliche Fehler ist $3.3E - 5$, also wesentlich kleiner als das vorgegebene ε . \square

Der folgende Algorithmus realisiert ähnlich wie Algorithmus 3.40. das schrittweise Vorgehen mit der Gauß-Kronrod Formel.

Tabelle 3.8: Kronrod Gauß Quadratur

| $x(i-1)$ | $x(i)$ | $x(i) - x(i-1)$ |
|----------|----------|-----------------|
| -1.00000 | -0.87500 | 0.12500 |
| -0.87500 | -0.75000 | 0.12500 |
| -0.75000 | -0.50000 | 0.25000 |
| -0.50000 | 0.00000 | 0.50000 |
| 0.00000 | 0.50000 | 0.50000 |
| 0.50000 | 0.75000 | 0.25000 |
| 0.75000 | 0.87500 | 0.12500 |
| 0.87500 | 0.93750 | 0.06250 |
| 0.93750 | 1.00000 | 0.06250 |

Tabelle 3.9: Adaptive Kronrod Quadratur

| ε | Fehler | Funktionsauswertungen |
|---------------|------------|-----------------------|
| $1E-1$ | $4.79E-05$ | 70 |
| $1E-2$ | $3.51E-07$ | 150 |
| $1E-3$ | $1.44E-08$ | 255 |
| $1E-4$ | $6.67E-11$ | 435 |

Algorithmus 3.45. (Adaptive Kronrod Quadratur)

```

function int=adap_kronrod(f,a,b,tol);
h=0.1*(b-a);
int=0;
eps=tol/(b-a);
while a < b
    [int1,int2]=kronrod(f,a,a+h);
    h_neu=0.9*h*(h*eps/abs(int1-int2))^(1/4);
    if abs(int1-int2) > h*eps;
        h=h_neu;
    else
        int=int+int2;
        a=a+h;
        h=min(h_neu,b-a);
    end
end

```

Beispiel 3.46. Für das Beispiel 3.37. erhält man hiermit die Ergebnisse aus Tabelle 3.9. Wie erwartet wird der tatsächliche Fehler wiederum deutlich kleiner als die vorgegebene Toleranz ε . □

Bemerkung 3.47. In dem Handbuch Piessens et al. [85] des Software Pakets QUADPACK finden sich die Knoten und Gewichte für Gauß Formeln und die zugehörigen Kronrod Formeln (bis hin zur 30-Punkt Gauß und 61-Punkt Kronrod Formel) und weiteres Material zu adaptiven Quadratur Methoden. Die FORTRAN 77 Subroutines von QUADPACK können als Public Domain Software bezogen werden von

<http://www.netlib.org/quadpack>

3.6 Numerische Differentiation

Wir betrachten eine Funktion $f : [a, b] \rightarrow \mathbb{R}$, von der nur die Funktionswerte $y_j := f(x_j)$ an diskreten Punkten $a \leq x_1 < x_2 < \dots < x_n \leq b$ bekannt sind. Aufgabe ist es, aus diesen diskreten Daten eine Näherung für den Wert einer Ableitung $f^{(m)}(x)$, $m \geq 1$, an einer Stelle x zu ermitteln.

Ähnlich wie bei der numerischen Integration interpolieren wir hierzu einige der gegebenen Daten (x_j, y_j) in der Nähe des Punktes x und wählen die m -te Ableitung der interpolierenden Funktion an der Stelle x als Näherung für $f^{(m)}(x)$. Gebräuchlich ist die Interpolation mit Polynomen und mit Splines. Sind in der Umgebung von x Daten mit verschiedenen Schrittweiten vorhanden und ist die Funktion f glatt genug, so kann wie beim Romberg Verfahren zur Quadratur auch Extrapolation verwendet werden.

Wir beschränken uns hier auf die Interpolation mit Polynomen und betrachten nur den Fall, dass die Stelle x , an der die Ableitung approximiert werden soll, ein Knoten ist.

Beispiel 3.48. Wir leiten Formeln für die Approximation der ersten Ableitung her.

Interpoliert man f linear mit den Daten (x_j, y_j) und (x_{j+1}, y_{j+1}) , so erhält man

$$p(x) = y_j + \frac{y_{j+1} - y_j}{x_{j+1} - x_j}(x - x_j),$$

und als Näherung für die Ableitung

$$f'(x_j) \approx p'(x_j) = \frac{y_{j+1} - y_j}{x_{j+1} - x_j}. \quad (3.29)$$

Dieser Ausdruck heißt der **vorwärtsgenommene Differenzenquotient**. Interpoliert man die Daten (x_{j-1}, y_{j-1}) und (x_j, y_j) linear, so erhält man genauso die Approximation

$$f'(x_j) \approx \frac{y_j - y_{j-1}}{x_j - x_{j-1}} \quad (3.30)$$

durch den **rückwärtsgenommenen Differenzenquotienten**.

Interpoliert man f quadratisch mit den Knoten (x_{j+k}, y_{j+k}) , $k = -1, 0, 1$, so erhält man

$$p(x) = y_j + [x_{j-1}, x_j](x - x_j) + [x_{j+1}, x_{j-1}, x_j](x - x_{j-1})(x - x_j)$$

mit der Ableitung

$$p'(x) = [x_{j-1}, x_j] + [x_{j+1}, x_{j-1}, x_j](2x - x_{j-1} - x_j).$$

Einsetzen von $x = x_j$ liefert nach kurzer Rechnung

$$f'(x_j) \approx \frac{x_{j+1} - x_j}{x_{j+1} - x_{j-1}}[x_{j-1}, x_j] + \frac{x_j - x_{j-1}}{x_{j+1} - x_{j-1}}[x_j, x_{j+1}]. \quad (3.31)$$

Ist speziell $x_{j+1} - x_j = x_j - x_{j-1} =: h$, so ist (3.31) der **zentrale Differenzenquotient**

$$f'(x_j) \approx \frac{1}{2}[x_{j-1}, x_j] + \frac{1}{2}[x_j, x_{j+1}] = \frac{y_{j+1} - y_{j-1}}{2h}. \quad (3.32)$$

Sind nur Funktionswerte von f auf einer Seite von x_j bekannt, so verwendet man einseitige Differenzenapproximationen. Z.B. erhält man mit $y_{j+k} = f(x_{j+k})$, $k = 0, 1, 2$, die Näherung

$$f'(x_j) \approx \frac{x_{j+2} + x_{j+1} - 2x_j}{x_{j+2} - x_j}[x_j, x_{j+1}] - \frac{x_{j+1} - x_j}{x_{j+2} - x_j}[x_{j+1}, x_{j+2}], \quad (3.33)$$

und im äquidistanten Fall

$$f'(x_j) \approx \frac{3}{2}[x_j, x_{j+1}] - \frac{1}{2}[x_{j+1}, x_{j+2}] = \frac{-y_{j+2} + 4y_{j+1} - 3y_j}{2h}. \quad (3.34)$$

Genauso erhält man mit 5 bzw. 7 äquidistanten Knoten die zentralen Differenzenapproximationen

$$f'(x_j) \approx \frac{1}{12h}(y_{j-2} - 8y_{j-1} + 8y_{j+1} - y_{j+2}) \quad (3.35)$$

und

$$f'(x_j) \approx \frac{1}{60h}(-y_{j-3} + 9y_{j-2} - 45y_{j-1} + 45y_{j+1} - 9y_{j+2} + y_{j+3}). \quad (3.36)$$

□

Den Fehler einer Differenzenformel kann man mit Hilfe des Taylorschen Satzes bestimmen. Für $f \in C^2$ gilt mit einem $\xi \in (x_j, x_j + h)$

$$f(x_j + h) = f(x_j) + f'(x_j)h + \frac{h^2}{2}f''(\xi),$$

d.h.

$$f'(x_j) = \frac{y_{j+1} - y_j}{h} - \frac{h}{2}f''(\xi),$$

und genauso mit einem $\eta \in (x_j - h, x_j)$

$$f'(x_j) = \frac{y_j - y_{j-1}}{h} - \frac{h}{2}f''(\eta).$$

Es gilt also für den Fehler sowohl des vorwärtsgenommenen als auch des rückwärtsgenommenen Differenzenquotienten die Asymptotik $O(h)$.

Definition 3.49. Eine Differenzenapproximation $D_r f(x; h)$ für die Ableitung $f^{(r)}(x)$ mit der Schrittweite h besitzt die **Fehlerordnung** p , falls gilt

$$D_r f(x; h) - f^{(r)}(x) = O(h^p).$$

Vorwärts- und rückwärtsgenommene Differenzenquotienten zur Approximation von $f'(x)$ besitzen also die Fehlerordnung 1.

Für $f \in C^4$ gilt

$$f(x_j \pm h) = f(x_j) \pm hf'(x_j) + \frac{h^2}{2}f''(x_j) \pm \frac{h^3}{6}f'''(x_j) + O(h^4),$$

und daher

$$\frac{y_{j+1} - y_{j-1}}{2h} - f'(x_j) = \frac{h^2}{6}f'''(x_j) + O(h^3).$$

Der zentrale Differenzenquotient besitzt also die Fehlerordnung 2. Genauso erhält man für die Approximationen in (3.35) und (3.36) die Fehlerordnungen 4 und 6.

Bei Rechnung in exakter Arithmetik beschreibt die Fehlerordnung das Verhalten des Fehlers für $h \rightarrow 0$. Die Differenzenformeln enthalten alle Differenzen von Funktionswerten von f , die bei kleinem h nahe beieinander liegen. Dies führt beim Rechnen mit endlicher Stellenzahl zu Auslöschungen.

Beispiel 3.50. Wir bestimmen für $f(x) := \cos x$ Näherungen für $f'(1)$ mit dem vorwärtsgenommenen Differenzenquotienten. Tabelle 3.10 enthält die Fehler der Approximationen für $h_j = 10^{-j}$, $j = 0, 1, \dots, 16$.

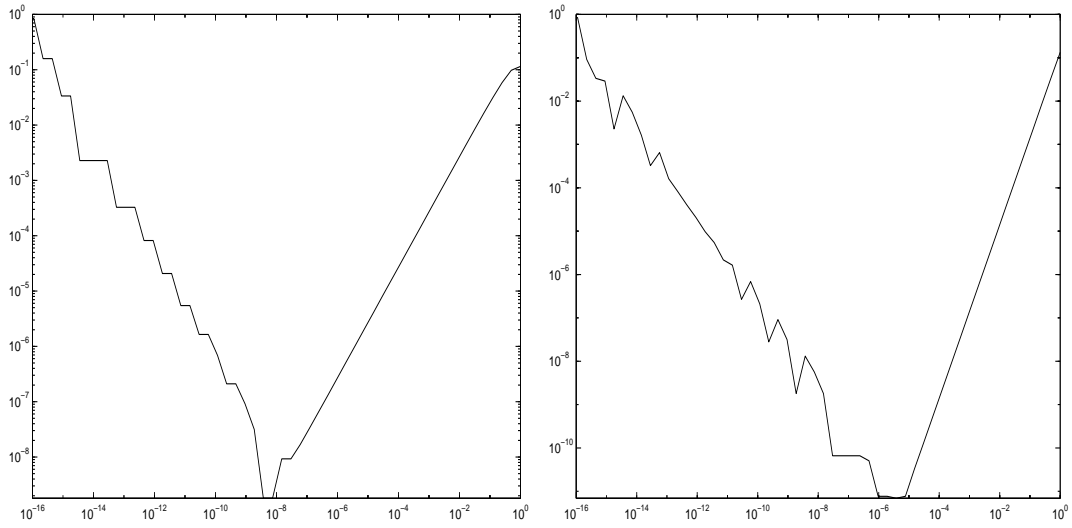


Abbildung 3.4 : Differenzenformel Ordnung 1 / Ordnung 2

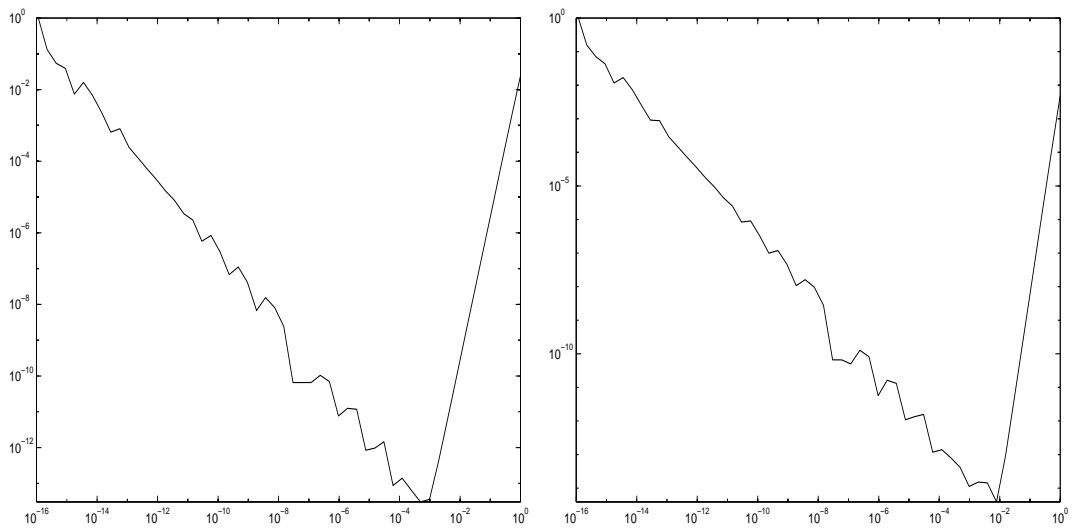


Abbildung 3.5 : Differenzenformel Ordnung 4 / Ordnung 6

Tabelle 3.10: Vorwärtsgenommene Differenzenquotienten

| j | Fehler | j | Fehler | j | Fehler |
|-----|-----------|-----|-----------|-----|-----------|
| 0 | $1.15e-1$ | 6 | $2.70e-7$ | 12 | $7.81e-5$ |
| 1 | $2.56e-2$ | 7 | $2.81e-8$ | 13 | $7.81e-5$ |
| 2 | $2.69e-3$ | 8 | $3.03e-9$ | 14 | $2.29e-3$ |
| 3 | $2.70e-4$ | 9 | $1.30e-7$ | 15 | $1.58e-1$ |
| 4 | $2.70e-5$ | 10 | $3.52e-7$ | 16 | $8.41e-1$ |
| 5 | $2.70e-6$ | 11 | $3.52e-7$ | | |

Der Fehler fällt also (wie durch die Fehlerordnung 1 vorausgesagt) zunächst bis $h = 10^{-8}$ linear, steigt aber danach durch Auslöschung in der Differenzenformel wieder an. Abbildung 3.4 enthält auf der linken Seite den Graphen des Fehlers in Abhängigkeit von h in doppeltlogarithmischer Darstellung.

Für andere Differenzenformeln beobachtet man dasselbe Verhalten. In Abbildung 3.4 rechts ist der Fehler für den zentralen Differenzenquotienten der Ordnung 2 dargestellt und in Abbildung 3.5 für die Formeln (3.35) und (3.36). \square

Das Verhalten im letzten Beispiel kann man auf folgende Weise erklären. Wir nehmen an, dass der errechnete Funktionswert $\tilde{y}_j \approx y_j$ die Größe

$$\tilde{y}_j = y_j(1 + \delta_j), \quad |\delta_j| \leq Ku \quad (3.37)$$

hat, wobei u die Maschinengenauigkeit bezeichnet und K eine “kleine” Konstante ist. Dann gilt für den errechneten vorwärtsgenommenen Differenzenquotienten

$$\tilde{D}_1(h) := \text{fl} \left(\frac{\tilde{y}_{j+1} - \tilde{y}_j}{h} \right) = \frac{\tilde{y}_{j+1} - \tilde{y}_j}{h} (1 + \varepsilon_1)(1 + \varepsilon_2), \quad |\varepsilon_1|, |\varepsilon_2| \leq u.$$

Unter Verwendung von (3.37) erhält man, wenn man Terme der Größenordnung u^2 vernachlässigt,

$$\tilde{D}_1(h) = \frac{y_{j+1} - y_j}{h} + \frac{y_{j+1}\delta_{j+1} - y_j\delta_j}{h} + \frac{y_{j+1} - y_j}{h}(\varepsilon_1 + \varepsilon_2),$$

und daher ist der Rundungsfehler bei der Auswertung der Differenzenformel $D_1(h) := (y_{j+1} - y_j)/h$ mit Konstanten C_1, C_2

$$|\tilde{D}_1(h) - D_1(h)| = \left| \frac{y_{j+1}\delta_{j+1} - y_j\delta_j}{h} + \frac{y_{j+1} - y_j}{h}(\varepsilon_1 + \varepsilon_2) \right| \leq \frac{C_1}{h}u + C_2u. \quad (3.38)$$

Da die vorwärtsgenommene Differenzenformel die Ordnung 1 hat, gibt es eine Konstante C_3 mit

$$|D_1(h) - f'(x_j)| \leq C_3h,$$

und daher folgt für den Gesamtfehler

$$|\tilde{D}_1(h) - f'(x_j)| \leq \frac{C_1}{h}u + C_2u + C_3h =: \Delta(h). \quad (3.39)$$

Der Graph dieser Funktion hat die Gestalt der Fehlerfunktion in Abbildung 3.4, links: Mit fallendem h fällt die Funktion $\Delta(h)$ bis zum Minimum, das durch

$$\Delta'(h) = -\frac{C_1 u}{h^2} + C_3 = 0$$

charakterisiert ist, d.h. bis

$$h_{\text{opt}} = \sqrt{\frac{C_1}{C_3}} \cdot \sqrt{u}, \quad (3.40)$$

und steigt danach wieder. In MATLAB ist $u \approx 10^{-16}$, das Minimum des Fehlers muss also in der Größenordnung von 10^{-8} liegen. Abbildung 3.4, links, zeigt, dass dies tatsächlich bei Beispiel 3.50. der Fall ist.

Besitzt die Differenzenformel D_1 die Fehlerordnung m , so bleibt (3.38) richtig, und man erhält entsprechend (3.39) den Gesamtfehler

$$|\tilde{D}_1(h) - f'(x_0)| \leq \frac{C_1}{h} u + C_2 u + C_3 h^m =: \Delta(h). \quad (3.41)$$

In diesem Fall erhält man als Größenordnung der optimalen Schrittweite

$$h_{\text{opt}} = C u^{1/(m+1)},$$

die ebenfalls durch die Beispiele in Abbildung 3.4, rechts, und Abbildung 3.5 bestätigt werden.

Kapitel 4

Lineare Systeme

Wir betrachten in diesem Abschnitt das Problem, ein lineares Gleichungssystem

$$\mathbf{Ax} = \mathbf{b} \tag{4.1}$$

mit einer gegebenen regulären Matrix $\mathbf{A} \in \mathbb{C}^{(n,n)}$ und einem gegebenen Vektor $\mathbf{b} \in \mathbb{C}^n$ zu lösen. Dies war schon einer der Hauptgegenstände in der Vorlesung “Lineare Algebra”. Wir werden in Abschnitt 4.1 vor allem die Ergebnisse aus dieser Vorlesung zusammentragen. In Abschnitt 4.2 werden wir auf einige weitergehende numerische Aspekte eingehen, insbesondere darauf, welche Einflüsse Speichertechniken auf die Konstruktion von Algorithmen haben, und die BLAS (Basic linear algebra subroutines) besprechen. In Abschnitt 4.3 behandeln wir Besonderheiten, die bei Bandmatrizen auftreten, in Abschnitt 4.4 fragen wir, wie sich Störungen der rechten Seite und der Matrixelemente auf die Lösung auswirken, und in Abschnitt 4.5 betrachten wir Matrizen, die eine spezielle Struktur aufweisen. In Abschnitt 4.6 geben wir einige Hinweise zur Software für lineare Gleichungssysteme. Wir gehen in dieser Vorlesung nicht auf das Problem ein, lineare Gleichungssysteme mit allgemeinen dünn besetzten Systemmatrizen (direkt oder iterativ) zu lösen. Dies ist der Gegenstand der Vorlesungen “Numerische Behandlung großer Systeme” und “Numerische Lineare Algebra”.

4.1 Zerlegung regulärer Matrizen

Es sei $\mathbf{A} \in \mathbb{C}^{(n,n)}$ eine gegebene reguläre Matrix. Dann existiert (vgl. LA Satz 4.42) eine Permutationsmatrix \mathbf{P} , eine untere Dreiecksmatrix \mathbf{L} , deren Diagonalelemente

zu 1 normiert sind, und eine obere Dreiecksmatrix \mathbf{R} mit

$$\mathbf{PA} = \mathbf{LR}. \quad (4.2)$$

Diese Zerlegung heißt die **LR Zerlegung** der Matrix \mathbf{A} . Es ist ferner bekannt (vgl. LA Satz 4.44), dass die Permutationsmatrix \mathbf{P} genau dann nicht benötigt wird, wenn alle Hauptuntermatrizen (engl.: principal submatrices)

$$\mathbf{A}(1:k, 1:k) := (a_{ij})_{i,j=1,\dots,k}, \quad k = 1, \dots, n,$$

von \mathbf{A} regulär sind. Wir haben hierbei die in MATLAB übliche Notation $\mathbf{A}(1:k, 1:k)$ für die Untermatrix von \mathbf{A} verwendet, die die erste bis k -te Zeile und erste bis k -te Spalte von \mathbf{A} enthält. Ist dies der Fall, so kann man die LR Zerlegung von \mathbf{A} mit dem **Gaußschen Eliminationsverfahren** bestimmen. Dabei speichern wir die von Null verschiedenen Elemente von \mathbf{R} in dem oberen Dreieck von \mathbf{A} und die Elemente unterhalb der Diagonale von \mathbf{L} in dem strikten unteren Dreieck von \mathbf{A} ab.

Algorithmus 4.1. (LR Zerlegung ohne Pivotsuche)

```

for i=1 : n-1
  for j=i+1 : n
    a(j,i)=a(j,i)/a(i,i);           \% Bestimme i-te Spalte von L
    for k=i+1 : n
      a(j,k)=a(j,k)-a(j,i)*a(i,k);  \% Datiere j-te Zeile auf
    end
  end
end
end

```

Ist die LR Zerlegung von \mathbf{A} bekannt, so kann man die Lösung des Gleichungssystems (4.1) schreiben als

$$\mathbf{Ax} = \mathbf{LRx} =: \mathbf{Ly} = \mathbf{b}, \quad \mathbf{Rx} = \mathbf{y},$$

und durch **Vorwärtseinsetzen**

Algorithmus 4.2. (Vorwärtseinsetzen)

```

for j=1 : n
  y(j)=b(j);
  for k=1 : j-1
    y(j)=y(j)-a(j,k)*y(k);
  end
end
end

```

die Lösung \mathbf{y} von $\mathbf{L}\mathbf{y} = \mathbf{b}$ bestimmen, und durch **Rückwärtseinsetzen**

Algorithmus 4.3. (Rückwärtseinsetzen)

```

for j=n : -1 : 1
  x(j)=y(j);
  for k=j+1 : n
    x(j)=x(j)-a(j,k)*x(k);
  end
  x(j)=x(j)/a(j,j);
end

```

die Lösung \mathbf{x} von $\mathbf{R}\mathbf{x} = \mathbf{y}$, d.h. von $\mathbf{A}\mathbf{x} = \mathbf{b}$.

Als Aufwand der LR Zerlegung erhält man

$$\sum_{i=1}^{n-1} \left(\sum_{j=i+1}^n (1 + \sum_{k=i+1}^n 2) \right) = \sum_{i=1}^{n-1} ((n-i) + 2(n-i)^2) = \frac{2}{3}n^3 + O(n^2).$$

flops (floating point operations). Das Vorwärts- und Rückwärtseinsetzen erfordert offenbar jeweils $O(n^2)$ flops. Existiert eine singuläre Hauptuntermatrix $\mathbf{A}(1:i, 1:i)$ der regulären Matrix \mathbf{A} , so bricht Algorithmus 4.1. ab, da das Pivotelement $a(i, i)$ bei der Elimination der i -ten Variable Null wird. In diesem Fall gibt es ein $a_{ij} \neq 0, j > i$, (das im Verlaufe des Algorithmus erzeugt worden ist), und man kann die aktuelle Zeile i mit der Zeile j vertauschen und den Algorithmus fortsetzen. Sammelt man diese Vertauschungen in der Permutationsmatrix \mathbf{P} , so erhält man am Ende des so modifizierten Algorithmus 4.1. eine LR Zerlegung der Matrix \mathbf{PA} :

$$\mathbf{PA} = \mathbf{LR}.$$

Aus Stabilitätsgründen empfiehlt es sich, auch dann eine Vertauschung vorzunehmen, wenn a_{ii} zwar von Null verschieden ist, in der Restmatrix $\mathbf{A}(i:n, i:n)$ aber Elemente vorhanden sind, deren Betrag wesentlich größer ist als der Betrag von a_{ii} .

Beispiel 4.4. Es sei

$$\mathbf{A} = \begin{pmatrix} 10^{-4} & 1 \\ 1 & 1 \end{pmatrix}$$

Wir bezeichnen mit $\text{fl}(a)$ die Gleitpunktdarstellung von a . Dann liefert Algorithmus 4.1. bei dreistelliger Rechnung

$$\mathbf{L} = \begin{pmatrix} 1 & 0 \\ \text{fl}(1/10^{-4}) & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 10^4 & 1 \end{pmatrix}$$

und

$$\mathbf{R} = \begin{pmatrix} 10^{-4} & 1 \\ 0 & \text{fl}(1 - 10^4) \end{pmatrix} = \begin{pmatrix} 10^{-4} & 1 \\ 0 & -10^4 \end{pmatrix},$$

und damit

$$\mathbf{LR} = \begin{pmatrix} 10^{-4} & 1 \\ 1 & 0 \end{pmatrix}. \quad \square$$

In vielen Fällen reicht es aus, vor dem Annullieren der Elemente der i -ten Spalte $j \in \{i, \dots, n\}$ zu bestimmen mit $|a_{ji}| \geq |a_{ki}|$ für alle $k = i, \dots, n$ und dann die i -te Zeile mit der j -ten Zeile zu vertauschen. Dieses Vorgehen heißt **Spaltenpivotsuche**. Man erhält folgende Modifikation von Algorithmus 4.1.:

Algorithmus 4.5. (LR Zerlegung mit Spaltenpivotsuche)

```

for i=1: n-1
  Waehle j >= i mit |a(j,i)| >= |a(k,i)| fuer alle k >= i
  und vertausche die i-te mit der j-ten Zeile
  for j=i+1 : n
    a(j,i)=a(j,i)/a(i,i);
    for k=i+1 : n
      a(j,k)=a(j,k)-a(j,i)*a(i,k);
    end
  end
end
end

```

Beispiel 4.6. Für Beispiel 4.4. erhält man mit Algorithmus 4.5.

$$\mathbf{L} = \begin{pmatrix} 1 & 0 \\ 10^{-4} & 1 \end{pmatrix}, \quad \mathbf{R} = \begin{pmatrix} 1 & 1 \\ 0 & \text{fl}(1 - 10^{-4}) \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

und

$$\mathbf{LR} = \begin{pmatrix} 1 & 1 \\ 10^{-4} & 1 + 10^{-4} \end{pmatrix}$$

ist eine gute Approximation von

$$\mathbf{PA} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \mathbf{A} = \begin{pmatrix} 1 & 1 \\ 10^{-4} & 1 \end{pmatrix}.$$

Nicht in allen Fällen führt die Spaltenpivotsuche zum Erfolg. Wendet man Algorithmus 4.5. bei dreistelliger Rechnung auf

$$\tilde{\mathbf{A}} = \begin{pmatrix} 1 & 10^4 \\ 1 & 1 \end{pmatrix}$$

an, so erhält man

$$\mathbf{L} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}, \quad \mathbf{R} = \begin{pmatrix} 1 & 10^4 \\ 0 & \text{fl}(1 - 10^4) \end{pmatrix} = \begin{pmatrix} 1 & 10^4 \\ 0 & -10^4 \end{pmatrix},$$

und hiermit

$$\mathbf{LR} = \begin{pmatrix} 1 & 10^4 \\ 1 & 0 \end{pmatrix}. \quad \square$$

Treten, wie in unserem Beispiel, in der Matrix Elemente sehr unterschiedlicher Größenordnung auf, so empfiehlt es sich, eine **vollständigen Pivotsuche** auszuführen. In diesem Fall werden im i -ten Eliminationsschritt ein Zeilenindex $j \geq i$ und ein Spaltenindex $k \geq i$ bestimmt mit $|a_{jk}| \geq |a_{\ell m}|$ für alle $\ell, m \geq i$, und es wird vor der Elimination die i -te mit der j -ten Zeile und die i -te Spalte mit der k -ten Spalte getauscht. Man erhält dann eine Zerlegung

$$\mathbf{PAQ} = \mathbf{LR},$$

wobei die Permutationsmatrix \mathbf{P} die Zeilenvertauschungen in \mathbf{A} vornimmt und die Permutationsmatrix \mathbf{Q} die Spaltenvertauschungen.

Ist die reguläre Matrix \mathbf{A} reell symmetrisch (oder Hermitesch; die Modifikationen hierfür sind offensichtlich) und existiert eine LR Zerlegung, so kann man \mathbf{R} als Produkt einer Diagonalmatrix \mathbf{D} und einer normierten oberen Dreiecksmatrix $\tilde{\mathbf{R}}$ schreiben. Hiermit gilt dann

$$\mathbf{A}^T = \tilde{\mathbf{R}}^T \mathbf{D} \mathbf{L}^T$$

mit einer normierten unteren Dreiecksmatrix $\tilde{\mathbf{R}}^T$ und einer oberen Dreiecksmatrix $\mathbf{D} \mathbf{L}^T$. Da die LR Zerlegung einer regulären Matrix eindeutig bestimmt ist, folgt $\tilde{\mathbf{R}}^T = \mathbf{L}$. Damit kann man \mathbf{A} auch schreiben als

$$\mathbf{A} = \mathbf{L} \mathbf{D} \mathbf{L}^T$$

mit einer Diagonalmatrix \mathbf{D} und \mathbf{L} wie oben. Diese Zerlegung heißt die **LDL^T Zerlegung** von \mathbf{A} . Ist die symmetrische Matrix \mathbf{A} zusätzlich positiv definit, so sind alle Hauptuntermatrizen von \mathbf{A} ebenfalls positiv definit, also regulär, und daher existiert in diesem Fall die **LDL^T Zerlegung**. Ferner sind die Diagonalelemente von $\mathbf{D} = \text{diag}\{d_1, \dots, d_n\}$ positiv, und man kann mit

$$\mathbf{C} = \mathbf{L} \text{diag}\{\sqrt{d_1}, \dots, \sqrt{d_n}\}$$

die LDL^T Zerlegung von \mathbf{A} schreiben als

$$\mathbf{A} = \mathbf{C}\mathbf{C}^T. \quad (4.3)$$

Diese Zerlegung heißt die **Cholesky Zerlegung** von \mathbf{A} .

Prinzipiell kann man die Cholesky Zerlegung mit Hilfe des Gaußschen Eliminationsverfahrens bestimmen. Man benötigt dann $\frac{2}{3}n^3 + O(n^2)$ Operationen und n^2 Speicherplätze. Durch direkten Vergleich der Elemente in (4.3) erhält man einen Algorithmus, der mit der Hälfte des Aufwandes auskommt. Da \mathbf{C} eine untere Dreiecksmatrix ist, gilt $c_{ij} = 0$ für $1 \leq i < j \leq n$, und daher ist

$$a_{ij} = \sum_{k=1}^n c_{ik}c_{jk} = \sum_{k=1}^i c_{ik}c_{jk}.$$

Speziell für $i = j = 1$ bedeutet dies

$$a_{11} = c_{11}^2, \quad \text{d.h. } c_{11} = \sqrt{a_{11}},$$

und für $i = 1$ und $j = 2, \dots, n$

$$a_{1j} = c_{11}c_{j1}, \quad \text{d.h. } c_{j1} = a_{j1}/c_{11}.$$

Damit ist die erste Spalte von \mathbf{C} bestimmt. Sind schon $c_{\mu\nu}$ für $\nu = 1, \dots, i-1$ und $\mu = \nu, \nu+1, \dots, n$ bestimmt, so erhält man aus

$$a_{ii} = c_{ii}^2 + \sum_{k=1}^{i-1} c_{ik}^2$$

das i -te Diagonalelement

$$c_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} c_{ik}^2}$$

von \mathbf{C} , und

$$a_{ij} = c_{ii}c_{ji} + \sum_{k=1}^{i-1} c_{ik}c_{jk}$$

liefert

$$c_{ji} = \frac{1}{c_{ii}} \left(a_{ij} - \sum_{k=1}^{i-1} c_{ik}c_{jk} \right), \quad j = i+1, \dots, n.$$

Damit erhält man das folgende Verfahren zur Bestimmung der Cholesky Zerlegung. Dabei überschreiben wir das untere Dreieck von \mathbf{A} durch die wesentlichen Elemente von \mathbf{C} .

Algorithmus 4.7. (Cholesky Zerlegung)

```

for i=1 : n
  for k=1 : i-1
    a(i,i)=a(i,i)-a(i,k)*a(i,k);
  end
  a(i,i)=sqrt(a(i,i));
  for j=i+1 : n
    for k=1 : i-1
      a(j,i)=a(j,i)-a(i,k)*a(j,k);
    end
    a(j,i)=a(j,i)/a(i,i);
  end
end

```

Der Aufwand des Cholesky Verfahrens ist

$$\sum_{i=1}^n \left(\sum_{k=1}^{i-1} 2 + \sum_{j=i+1}^n \left(1 + \sum_{k=1}^{i-1} 2 \right) \right) = \frac{1}{3}n^3 + O(n^2)$$

flops und n Quadratwurzeln. Besitzt die symmetrische Matrix \mathbf{A} eine LDL^T Zerlegung, so kann man diese mit einem ähnlichen Verfahren wie Algorithmus 4.7. bestimmen. Man beachte aber, dass eine Pivotsuche wie beim Gaußschen Eliminationsverfahren die Symmetrie der Matrix zerstört. Man muss also gleichlautende Zeilen- und Spaltenvertauschungen vornehmen. Auf diese Weise kann man nicht immer für eine reguläre Matrix ein von Null verschiedenes Pivotelement erzeugen, wie das Beispiel

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

zeigt. Auch wenn die LDL^T Zerlegung existiert, kann die Übertragung von Algorithmus 4.7. instabil sein.

4.2 Modifikationen des Gaußschen Verfahrens

In Algorithmus 4.1. haben wir die Elimination durchgeführt, indem wir im i -ten Schritt ein geeignetes Vielfaches der i -ten Zeile von den nachfolgenden Zeilen abgezogen haben. In Vektorschreibweise haben wir also

Algorithmus 4.8. (LR Zerlegung; zeilenorientiert)

```

for i=1: n-1
  for j=i+1 : n
    a(j,i)=a(j,i)/a(i,i);
    a(j,i+1:n)=a(j,i+1:n)-a(j,i)*a(i,i+1:n);
  end
end

```

Vertauscht man die beiden inneren Schleifen, so erhält man eine spaltenorientierte Version des Gaußschen Eliminationsverfahren

Algorithmus 4.9. (LR Zerlegung; spaltenorientiert)

```

for i=1: n-1
  a(i+1:n,i)=a(i+1:n,i)/a(i,i);
  for k=i+1 : n
    a(i+1:n,k)=a(i+1:n,k)-a(i,k)*a(i+1:n,i);
  end
end

```

Auch die i -Schleife kann man mit der j -Schleife und/oder der k -Schleife vertauschen. Man erhält insgesamt 6 Varianten der Gauß Elimination, die alle von der Zahl der Rechenoperationen her denselben Aufwand besitzen. Sie können sich aber auf verschiedenen Plattformen unter verschiedenen Programmiersprachen sehr unterschiedlich verhalten.

Der Grund hierfür ist, dass Speicher von Rechnern hierarchisch aufgebaut sind, beginnend mit sehr langsamen, sehr großen, sehr billigen Magnetband Speichern, über schnellere, kleinere, teurere Plattenspeicher, den noch schnelleren, noch kleineren, noch teureren Hauptspeicher, den schnellen, kleinen und teuren Cache und die sehr schnellen, sehr kleinen und sehr teuren Register der CPU. Arithmetische und logische Operationen können nur in den Registern ausgeführt werden. Daten, die sich nicht in den Registern befinden und mit denen Operationen ausgeführt werden sollen, können nur in den benachbarten Speicher bewegt werden, wobei der Datentransport zwischen den billigen Speicherarten (also z.B. zwischen der Platte und dem Hauptspeicher) sehr langsam geschieht, während der Transport zwischen den teuren Speichern an der Spitze der Hierarchie sehr schnell geschieht. Insbesondere ist die Geschwindigkeit, mit der arithmetische Operationen ausgeführt werden, sehr viel

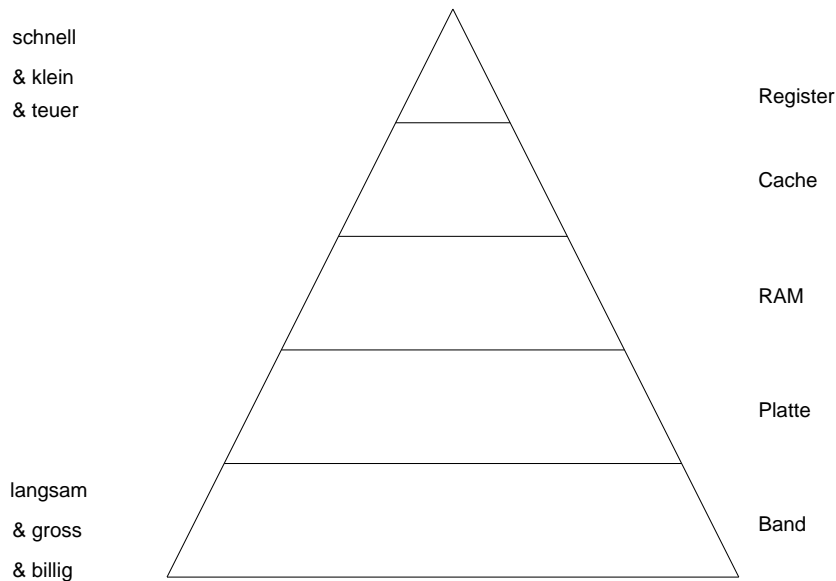


Abbildung 4.1: Hierarchischer Speicher

höher als die Geschwindigkeit, mit der Daten transportiert werden. Faktoren zwischen 10 und 10000 (je nach Speicherebene) sind nicht ungewöhnlich. Algorithmen müssen also so gestaltet werden, dass der Datentransport zwischen verschiedenen Speicherebenen möglichst klein ist.

Dass die Reihenfolge, in der die Schleifen im Gaußschen Eliminationsverfahren durchlaufen werden, auf die Laufzeit eines Programms einen Einfluss hat, sieht man so ein: Eine Matrix wird in einem langen (eindimensionalen) Array gespeichert. In C geschieht das zeilenweise:

$$\begin{array}{ccccccc}
 a(1,1) & \rightarrow & a(1,2) & \rightarrow & a(1,3) & \rightarrow & \dots & \rightarrow & a(1,n) \\
 \rightarrow & a(2,1) & \rightarrow & a(2,2) & \rightarrow & a(2,3) & \rightarrow & \dots & \rightarrow & a(2,n) \\
 \rightarrow & a(3,1) & \rightarrow & a(3,2) & \rightarrow & a(3,3) & \rightarrow & \dots & \rightarrow & a(3,n) \\
 & \vdots & & \vdots & & \vdots & & & & \vdots \\
 \rightarrow & a(n,1) & \rightarrow & a(n,2) & \rightarrow & a(n,3) & \rightarrow & \dots & \rightarrow & a(n,n)
 \end{array}$$

In der zeilenorientierten Version in Algorithmus 4.8. werden in der innersten Schleife Daten verwendet, die im Speicher nahe beieinander liegen. Es sind daher nur wenige Datentransporte zwischen den verschiedenen Speicherebenen erforderlich. Für die spaltenorientierte Version liegen die nacheinander benutzten Daten (wenigstens für große Dimensionen n) sehr weit auseinander, und daher ist ein “cache miss” (das benötigte Wort liegt nicht im Cache, und es müssen zwei Blöcke zwischen dem Cache und dem Hauptspeicher ausgetauscht werden) oder sogar ein “page fault” (das

benötigte Wort liegt nicht einmal im Hauptspeicher, und es müssen zwei Seiten zwischen dem Hauptspeicher und der Platte ausgetauscht werden) wahrscheinlicher. In FORTRAN ist die Situation umgekehrt. Matrizen werden spaltenweise gespeichert, und für Algorithmus 4.9. ist die Datenlokalität hoch, während sie in Algorithmus 4.8. gering ist. Um nicht für jeden neuen Rechner neue Software schreiben zu müssen, um die Modularität und Effizienz von Programmen zu erhöhen und um ihre Pflege zu erleichtern, wurden Standardoperationen der (numerischen) linearen Algebra, die basic linear algebra subprograms (BLAS), definiert, und es wurden Standardschnittstellen für ihren Aufruf festgelegt. Diese werden (jedenfalls für Hochleistungsrechner) von den Herstellern auf der Hardwareebene realisiert (nicht zuletzt, da die Hersteller wissen, dass die üblichen Benchmarktests Programme enthalten, die aus den BLAS Routinen aufgebaut sind!). Die Benutzung der BLAS in eigenen Programmen bietet eine Reihe von Vorteilen:

- Die Robustheit von Berechnungen der linearen Algebra wird durch die BLAS erhöht, denn in Ihnen werden Details der Algorithmen und der Implementierung berücksichtigt, die bei der Programmierung eines Anwendungsproblems leicht übersehen werden, wie z.B. die Berücksichtigung von Overflow Problemen.
- Die Portabilität von Programmen wird erhöht, ohne auf Effizienz zu verzichten, denn es werden optimierte Versionen der BLAS auf Rechnern verwendet, für die diese existieren. Für alle anderen Plattformen existieren kompatible Standard Fortran oder C Implementierungen. Diese können als Public Domain Software bezogen werden von

<http://www.netlib.org/blas/>

bzw.

<http://www.netlib.org/clapack/cblas/>

Im ATLAS Projekt (Automatically Tuned Linear Algebra Software) wurden und werden empirische Methoden entwickelt, um Bibliotheken hoher Performance zu erzeugen und zu pflegen, und so in der durch die Software diktierten Geschwindigkeit Schritt mit der Hardware Entwicklung zu halten. Es werden z.B. für den benutzten Rechner die Größen des Registers und Caches ermittelt und für die Level 2 und Level 3 BLAS die Blockgrößen angepasst. Die im

ATLAS Projekt entwickelte BLAS, für die es Fortran und C Interfaces gibt, kann herunter geladen werden von

<http://www.netlib.org/atlas/>

- Die Lesbarkeit von Programmen wird dadurch erhöht, dass mnemonische Namen für Standardoperationen verwendet werden und der Programmablauf nicht durch Codierungsdetails unterbrochen wird. Dies erleichtert auch die Dokumentation von Programmen.

Die erste Stufe der BLAS Definitionen (**Level 1 BLAS** oder **BLAS1**) wurde 1979 durchgeführt [70] und enthielt Vektor-Vektor Operationen wie das Skalarprodukt oder die Summe eines Vektors und des Vielfachen eines weiteren Vektors. Es wurde eine Namenskonvention eingeführt, die einen drei- bis fünfbuchstabigen, einprägsamen Namen verwendet, dem ein Buchstabe zur Kennzeichnung des Datentyps vorangestellt wird (S, D, C oder Z). Als Beispiele nennen wir nur die Function `_DOT`, für die durch “ALPHA=DDOT(N,X,1,Y,1)” das innere Produkt der doppelgenauen Vektoren \mathbf{x} und \mathbf{y} der Dimension n berechnet wird, oder die Subroutine `_AXPY` (“a x plus y”) mit dem Aufruf “CAXPY(N,ALPHA,X,1,Y,1)” durch die für die komplexen Vektoren \mathbf{x} und \mathbf{y} der Dimension n der komplexe Vektor $\alpha\mathbf{x} + \mathbf{y}$ berechnet wird und im Speicher von \mathbf{y} abgelegt wird. Statt der Einsen in den Aufrufen kann man Inkremente angeben, so dass auch innere Produkte der Art

$$\sum_{j=0}^{n-1} a_{1+mj} a_{2+mj}$$

berechnet werden können, also bei spaltenweiser Speicherung einer (m, n) -Matrix \mathbf{A} das innere Produkt der ersten und zweiten Zeile.

Beispiel 4.10. Als Beispiel betrachten wir die Bestimmung des inneren Produktes zweier Vektoren der Dimension $n = 10^7$. Wir verwenden (wie auch bei den folgenden Beispielen zur Performance der BLAS Routinen) eine HP 9000/785/C3000 Workstation mit einer CPU 2.0.PA8500 mit der Taktfrequenz 400 MHz und den Fortran90 Compiler f90-HP. Die folgende Tabelle enthält die Laufzeiten eines naiven Codes mit einer Laufanweisung, einer BLAS1 Implementierung in FORTRAN77, die aus der netlib heruntergeladen werden kann, einer BLAS Implementierung, die mit dem Fortran90 Compiler von HP geliefert wird, und einer optimierten BLAS Routine der `veclib` von HP.

| Implementierung | CPU Zeit | Relation |
|-----------------|----------|----------|
| naiv | 0.92 | 7.9 |
| BLAS (netlib) | 0.52 | 4.5 |
| BLAS (f90-HP) | 0.29 | 2.4 |
| BLAS (ATLAS) | 0.23 | 2.0 |
| veclib | 0.12 | 1.0 |

□

Die BLAS1 haben zu effizienten Implementierungen von Algorithmen auf skalaren Maschinen geführt, auf Vektorrechnern oder Parallelrechnern werden weitere Standardoperationen benötigt. Man kann z.B. das Produkt einer Matrix \mathbf{A} mit einem Vektor \mathbf{x} codieren als

Algorithmus 4.11. (Matrix-Vektor Produkt mit DAXPY)

```

Y=zeros(n,1);
for j=1 : n
    DAXPY(n,X(j),A(:,j),1,Y,1)
end

```

Dabei wird aber nicht berücksichtigt, dass der Ergebnisvektor \mathbf{y} im Register gehalten werden könnte. BLAS1 hat also den Nachteil, dass zu wenige (nützliche) flops ausgeführt werden im Verhältnis zu (nutzlosen) Datenbewegungen. Für die Ausführung eines `_AXPYs` sind z.B. $3n + 1$ Speicherzugriffe erforderlich (die Vektoren \mathbf{x} und \mathbf{y} und der Skalar α müssen gelesen werden, und der Ergebnisvektor \mathbf{y} muss geschrieben werden) und es werden $2n$ flops ausgeführt. Das Verhältnis ist also $2/3$. Dies wurde verbessert mit der Definition der **Level 2 BLAS** oder **BLAS2** in [28], die Matrix-Vektor Operationen enthalten, wie z.B. subroutine `_GEMV`, durch die $\mathbf{y} \leftarrow \alpha \mathbf{A}\mathbf{x} + \beta \mathbf{y}$ berechnet wird, das Produkt einer Dreiecksmatrix mit einem Vektor, Rang-1- oder Rang-2-Aufdatierungen von Matrizen wie $\mathbf{A} \leftarrow \alpha \mathbf{x}\mathbf{y}^T + \mathbf{A}$, oder Lösungen von Gleichungssystemen mit Dreiecksmatrizen. Für die Subroutine `_GEMV` sind im n -dimensionalen Fall $n^2 + 3n + 2$ Speicherzugriffe erforderlich, und es werden $2n^2$ flops ausgeführt. Das Verhältnis ist also 2. Algorithmus 4.9. ist schon fast eine BLAS2 Implementierung der LR Zerlegung. Man muss nur noch die Modifikationen der Spalten zu einer Rang-1-Modifikation des unteren $(n-i, n-i)$ -Blocks umschreiben. In Algorithmus 4.12. haben wir gleich (für den späteren Gebrauch) den sich ergebenden Algorithmus für eine (m, n) -Matrix \mathbf{A} mit $m \geq n$ beschrieben, wobei

A durch die untere (m, n) -Dreiecksmatrix L und die obere (n, n) -Dreiecksmatrix R überschrieben wird.

Algorithmus 4.12. (LR Zerlegung; BLAS2)

```

for i=1: n-1
  a(i+1:m,i)=a(i+1:m,i)/a(i,i);
  a(i+1:m,i+1:n)=a(i+1:m,i+1:n)-a(i+1:m,i)*a(i,i+1:n);
end

```

Beispiel 4.13. Als Beispiel zur Performance der BLAS2 betrachten wir die Bestimmung des Produktes einer $(10^4, 10^3)$ -Matrix mit einem Vektor. Hierfür erhält man die Laufzeiten

| Implementierung | CPU Zeit | Relation |
|-----------------|----------|----------|
| naiv | 4.32 | 61.7 |
| BLAS2 (netlib) | 1.39 | 19.9 |
| BLAS2 (f90-HP) | 0.62 | 8.9 |
| BLAS2 (ATLAS) | 0.17 | 2.4 |
| veclib | 0.07 | 1.0 |

Führt man das Matrix-Vektorprodukt mit Hilfe der BLAS1 Routine DDOT aus, so erhält man die folgenden Laufzeiten. Wir vergleichen hier mit der BLAS2 Routine der veclib und der BLAS2 Routine derselben Quelle.

| Implementierung | CPU Zeit | rel. zu BLAS2 (veclib) | rel. zu BLAS2 |
|-----------------|----------|------------------------|---------------|
| BLAS1 (netlib) | 2.57 | 36.7 | 1.8 |
| BLAS1 (f90-HP) | 2.49 | 35.6 | 4.0 |
| BLAS1 (ATLAS) | 2.31 | 33.0 | 13.6 |
| veclib | 2.06 | 29.4 | 29.4 |

Führt man das Matrix-Vektor Produkt schließlich mit der BLAS1 Routine DAXPY durch (vgl. Algorithmus 4.11.), so erhält man

| Implementierung | CPU Zeit | rel. zu BLAS2 (veclib) | rel. zu BLAS2 |
|-----------------|----------|------------------------|---------------|
| BLAS1 (netlib) | 0.55 | 7.9 | 0.4 |
| BLAS1 (f90-HP) | 0.23 | 3.3 | 0.4 |
| BLAS1 (ATLAS) | 0.28 | 4.0 | 1.7 |
| veclib | 0.09 | 1.3 | 1.3 |

Dass die BLAS1 Ergebnisse mit DAXPY wesentlich besser sind als mit DDOT ist klar, da im ersten Fall auf die spaltenweise Speicherung der Matrix in Fortran Rücksicht genommen wird und die Datenlokalität gewahrt wird. Dass die BLAS1 Realisierung unter Verwendung der netlib schneller ist als die entsprechenden BLAS2 Tests, liegt daran, dass in der Prozedur DAXPY Loop unrolling verwendet wird, während dies in DGEMV nicht verwendet wird. \square

In **Level 3 BLAS** [27] oder **BLAS3** wurden schließlich Matrix-Matrix Operationen wie $\mathbf{C} \leftarrow \alpha \mathbf{A}\mathbf{B} + \beta \mathbf{C}$ aufgenommen. Um die Wiederverwendung von Daten in den Registern oder im Cache in möglichst hohem Maße zu erreichen, werden die beteiligten Matrizen in Blöcke unterteilt und die Operationen blockweise ausgeführt. Auf diese Weise erreicht man, dass für die obige Operation bei $4n^2 + 2$ Speicherzugriffen $2n^3 + O(n^2)$ flops ausgeführt werden, das Verhältnis von nützlicher Arbeit zu Speicherzugriffen also auf $n/2$ steigt. Als Beispiel betrachten wir wieder die LR Zerlegung einer (n, n) -Matrix ohne Pivotsuche. Wir nehmen an, dass schon die ersten $i - 1$ Spalten von \mathbf{L} und die ersten $i - 1$ Zeilen von \mathbf{R} bestimmt sind. Wir haben also schon die Zerlegung

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} & \mathbf{A}_{13} \\ \mathbf{A}_{21} & \mathbf{A}_{22} & \mathbf{A}_{23} \\ \mathbf{A}_{31} & \mathbf{A}_{32} & \mathbf{A}_{33} \end{pmatrix} = \begin{pmatrix} \mathbf{L}_{11} & \mathbf{O} & \mathbf{O} \\ \mathbf{L}_{21} & \mathbf{I}_b & \mathbf{O} \\ \mathbf{L}_{31} & \mathbf{O} & \mathbf{I}_{n-b-i+1} \end{pmatrix} \begin{pmatrix} \mathbf{R}_{11} & \mathbf{R}_{12} & \mathbf{R}_{13} \\ \mathbf{O} & \tilde{\mathbf{A}}_{22} & \tilde{\mathbf{A}}_{23} \\ \mathbf{O} & \tilde{\mathbf{A}}_{32} & \tilde{\mathbf{A}}_{33} \end{pmatrix}$$

mit $\mathbf{A}_{11} \in \mathbb{R}^{(i-1, i-1)}$, $\mathbf{A}_{22} \in \mathbb{R}^{(b, b)}$ und $\mathbf{A}_{33} \in \mathbb{R}^{(n-i-b+1, n-i-b+1)}$ (die Dimensionen der anderen Blöcke ergeben sich hieraus). Wir beschreiben, wie wir die nächsten b Spalten von \mathbf{L} und b Zeilen von \mathbf{R} erhalten. Dazu wenden wir Algorithmus 4.12. auf die Untermatrix $\begin{pmatrix} \tilde{\mathbf{A}}_{22} \\ \tilde{\mathbf{A}}_{32} \end{pmatrix}$ an und erhalten

$$\begin{pmatrix} \tilde{\mathbf{A}}_{22} \\ \tilde{\mathbf{A}}_{32} \end{pmatrix} = \begin{pmatrix} \mathbf{L}_{22} \\ \mathbf{L}_{32} \end{pmatrix} \mathbf{R}_{22}. \quad (4.4)$$

Hiermit gilt für den unteren $(n - i + 1, n - i + 1)$ -Block

$$\begin{aligned} \begin{pmatrix} \tilde{\mathbf{A}}_{22} & \tilde{\mathbf{A}}_{23} \\ \tilde{\mathbf{A}}_{32} & \tilde{\mathbf{A}}_{33} \end{pmatrix} &= \begin{pmatrix} \mathbf{L}_{22} \mathbf{R}_{22} & \tilde{\mathbf{A}}_{23} \\ \mathbf{L}_{32} \mathbf{R}_{22} & \tilde{\mathbf{A}}_{33} \end{pmatrix} \\ &= \begin{pmatrix} \mathbf{L}_{22} & \mathbf{O} \\ \mathbf{L}_{32} & \mathbf{I}_{n-b-i+1} \end{pmatrix} \begin{pmatrix} \mathbf{R}_{22} & \mathbf{L}_{22}^{-1} \tilde{\mathbf{A}}_{23} \\ \mathbf{O} & \tilde{\mathbf{A}}_{33} - \mathbf{L}_{32} (\mathbf{L}_{22}^{-1} \tilde{\mathbf{A}}_{23}) \end{pmatrix} \\ &=: \begin{pmatrix} \mathbf{L}_{22} & \mathbf{O} \\ \mathbf{L}_{32} & \mathbf{I}_{n-b-i+1} \end{pmatrix} \begin{pmatrix} \mathbf{R}_{22} & \mathbf{R}_{23} \\ \mathbf{O} & \tilde{\mathbf{A}}_{33} - \mathbf{L}_{32} \mathbf{R}_{23} \end{pmatrix} \end{aligned}$$

$$=: \begin{pmatrix} \mathbf{L}_{22} & \mathbf{O} \\ \mathbf{L}_{32} & \mathbf{I}_{n-b-i+1} \end{pmatrix} \begin{pmatrix} \mathbf{R}_{22} & \mathbf{R}_{23} \\ \mathbf{O} & \tilde{\mathbf{A}}_{33}^{\text{neu}} \end{pmatrix}.$$

Damit erhält Algorithmus 4.12. unter Benutzung von BLAS3 die folgende Gestalt:

Algorithmus 4.14. (LR Zerlegung; BLAS3)

(1) Faktorisiere

$$\begin{pmatrix} \tilde{\mathbf{A}}_{22} \\ \tilde{\mathbf{A}}_{32} \end{pmatrix} = \begin{pmatrix} \mathbf{L}_{22} \\ \mathbf{L}_{32} \end{pmatrix} \mathbf{R}_{22}$$

unter Benutzung von Algorithmus 4.12..

(2) Berechne $\mathbf{R}_{23} = \mathbf{L}_{22}^{-1} \tilde{\mathbf{A}}_{23}$. Es muss also ein Gleichungssystem mit einer Dreiecksmatrix und mehreren rechten Seiten gelöst werden. Hierzu gibt es eine BLAS3 Subroutine.

(3) Bestimme

$$\tilde{\mathbf{A}}_{33}^{\text{neu}} = \tilde{\mathbf{A}}_{33} - \mathbf{L}_{32} \mathbf{R}_{23}.$$

Beispiel 4.15. Als Beispiel zur Performance der BLAS3 betrachten wir die Bestimmung des Produktes zweier $(10^3, 10^3)$ Matrizen. Hierfür erhält man die Laufzeiten

| Implementierung | CPU Zeit | Relation |
|-----------------|----------|----------|
| naiv | 462.42 | 247.34 |
| BLAS3 (netlib) | 320.00 | 171.1 |
| BLAS3 (f90-HP) | 7.25 | 3.9 |
| BLAS3 (ATLAS) | 4.26 | 2.3 |
| veclib | 1.87 | 1.0 |

Führt man das Produkt mit Hilfe der BLAS2 Routine DGEMV aus, so erhält man die folgenden Laufzeiten. Wir vergleichen hier wieder mit der BLAS3 Routine der veclib und der BLAS3 Routine derselben Quelle.

| Implementierung | CPU Zeit | rel. zu BLAS3 (veclib) | rel. zu BLAS3 |
|-----------------|----------|------------------------|---------------|
| BLAS2 (netlib) | 148.23 | 79.3 | 0.4 |
| BLAS2 (f90-HP) | 49.06 | 26.2 | 6.8 |
| BLAS2 (ATLAS) | 18.91 | 10.1 | 4.4 |
| veclib | 9.62 | 5.1 | 5.1 |

Fortran90 enthält schließlich noch die Routine MATMUL zur Multiplikation zweier Matrizen. Hiermit benötigt man eine Laufzeit von 4.91 Sekunden, d.h. das 2.6-fache der Zeit mit Hilfe der veclib Routine.

Die Verhältnisse der Laufzeiten sind (wie auch in den letzten beiden Beispielen) abhängig von den Dimensionen der beteiligten Matrizen bzw. Vektoren. Wir demonstrieren dies an dem Beispiel der Matrizenmultiplikation unter Benutzung der BLAS der netlib und veclib.

| Dimension | netlib | veclib | Verhältnis |
|-----------|--------|--------|------------|
| 50 | 0.03 | 0.001 | 30 |
| 70 | 0.08 | 0.001 | 80 |
| 100 | 0.26 | 0.003 | 87 |
| 300 | 7.79 | 0.050 | 156 |
| 500 | 40.03 | 0.216 | 185 |
| 700 | 119.30 | 0.619 | 193 |
| 1000 | 320.00 | 1.870 | 171 |

□

4.3 Bandmatrizen

In vielen Anwendungen besitzen die Koeffizientenmatrizen der linearen Gleichungssysteme Bandgestalt. Dabei heißt eine Matrix **A Bandmatrix**, wenn es Zahlen $p, q \in \mathbb{N}$ gibt mit $a_{ij} = 0$, falls $j > i + q$ oder $i > j + p$. q heißt die **obere Bandbreite** und p die **untere Bandbreite**. Wir sagen dann, dass **A** eine (p, q) -Bandmatrix ist.

Beispiel 4.16. Die Bestimmung eines interpolierenden kubischen Splines führt auf ein lineares Gleichungssystem mit einer tridiagonalen Matrix, d.h. einer $(1, 1)$ -Bandmatrix. □

Beispiel 4.17. Diskretisiert man die Randwertaufgabe

$$-y'' + q(x)y = f(x), \quad 0 < x < 1, \quad y(0) = 0, \quad y(1) = 0,$$

indem man auf einem äquidistanten Gitter $x_i = ih$, $h = 1/(n + 1)$, die zweite Ableitung y'' durch den zentralen Differenzenquotienten

$$y''(x) = \frac{1}{h^2}(y(x - h) - 2y(x) + y(x + h))$$

ersetzt, so erhält man für die Approximationen $Y_i \approx y(x_i)$, $i = 1, \dots, n$ ein lineares Gleichungssystem

$$\mathbf{AY} := \frac{1}{h^2} \begin{pmatrix} 2 + h^2q(x_1) & -1 & 0 & \dots & 0 \\ -1 & 2 + h^2q(x_2) & -1 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 2 + h^2q(x_n) \end{pmatrix} \mathbf{Y} = \begin{pmatrix} f(x_1) \\ f(x_2) \\ \vdots \\ \vdots \\ f(x_n) \end{pmatrix}$$

mit einer tridiagonalen Matrix A . □

Beispiel 4.18. Ähnlich wie im letzten Beispiel liefert die Differenzenapproximation

$$y^{(iv)}(x) \approx \frac{1}{h^4}(y((i - 2)h) - 4y((i - 1)h) + 6y(ih) - 4y((i + 1)h) + y((i + 2)h))$$

der vierten Ableitung (zusammen mit der Diskretisierung der zweiten Ableitung aus dem letzten Beispiel) die Diskretisierung der Randwertaufgabe

$$y^{(iv)} = f(x), \quad 0 < x < 1, \quad y(0) = y''(0) = y(1) = y''(1) = 0$$

durch ein Gleichungssystem mit einer pentdiagonalen Matrix

$$\frac{1}{h^4} \begin{pmatrix} 5 & -4 & 1 & 0 & \dots & 0 \\ -4 & 6 & -4 & 1 & \dots & 0 \\ 1 & -4 & 6 & -4 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 5 \end{pmatrix} \quad \square$$

Gleichungssysteme mit Bandmatrizen kann man mit wesentlich weniger Aufwand lösen als allgemeine Gleichungssysteme, da die Faktoren in einer LR Zerlegung einer Bandmatrix ebenfalls Bandgestalt besitzen.

Satz 4.19. Ist \mathbf{A} eine (p, q) -Bandmatrix und besitzt \mathbf{A} eine LR Zerlegung, so ist \mathbf{L} eine $(p, 0)$ -Bandmatrix und \mathbf{R} eine $(0, q)$ -Bandmatrix.

Beweis: Wir führen den Beweis durch Induktion. Man rechnet leicht nach, dass

$$\mathbf{A} =: \begin{pmatrix} \alpha & \mathbf{u}^T \\ \mathbf{v} & \mathbf{B} \end{pmatrix} = \begin{pmatrix} 1 & \mathbf{0}^T \\ \frac{1}{\alpha}\mathbf{v} & \mathbf{I}_{n-1} \end{pmatrix} \begin{pmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{B} - \frac{1}{\alpha}\mathbf{v}\mathbf{u}^T \end{pmatrix} \begin{pmatrix} \alpha & \mathbf{u}^T \\ \mathbf{0} & \mathbf{I}_{n-1} \end{pmatrix},$$

gilt. Da \mathbf{A} die untere Bandbreite p besitzt und die obere Bandbreite q , sind in \mathbf{u} nur die ersten q Komponenten von 0 verschieden und in \mathbf{v} nur die ersten p Komponenten. Daher ist auch \mathbf{B} eine (p, q) -Bandmatrix. Da $\mathbf{B} - \frac{1}{\alpha}\mathbf{v}\mathbf{u}^T$ diejenige Matrix ist, die man nach dem ersten Eliminationsschritt für \mathbf{A} erhält, besitzt diese Matrix eine LR Zerlegung $\tilde{\mathbf{L}}\tilde{\mathbf{R}}$. Definiert man hiermit

$$\mathbf{L} = \begin{pmatrix} 1 & \mathbf{0}^T \\ \frac{1}{\alpha}\mathbf{v} & \tilde{\mathbf{L}} \end{pmatrix} \quad \text{und} \quad \mathbf{R} = \begin{pmatrix} \alpha & \mathbf{u}^T \\ \mathbf{0} & \tilde{\mathbf{R}} \end{pmatrix},$$

so gilt $\mathbf{A} = \mathbf{LR}$, und \mathbf{L} und \mathbf{R} sind $(p, 0)$ - und $(0, q)$ -Bandmatrizen. ■

Die Übertragung der Gauß Elimination auf Bandmatrizen entsprechend Algorithmus 4.1. lautet

Algorithmus 4.20. (LR Zerlegung für Bandmatrizen)

```

for i=1 : n-1
  for j=i+1 : min(i+p,n)
    a(j,i)=a(j,i)/a(i,i);
    for k=i+1 : min(i+q,n)
      a(j,k)=a(j,k)-a(j,i)*a(i,k);
    end
  end
end
end

```

Man zählt sofort ab, dass dieser Algorithmus (für $p \ll n$ und $q \ll n$) nur $2npq$ flops benötigt. Da die offensichtlichen Übertragungen von Algorithmus 4.2. und Algorithmus 4.3. zum Vorwärts- und Rückwärtseinsetzen $2np$ und $2nq$ flops benötigen, kann man also ein lineares Gleichungssystem mit einer (p, q) -Bandmatrix mit $2(p + q + pq)n$ flops lösen. Ist eine vollständige Pivotsuche aus Stabilitätsgründen erforderlich, so wird die Bandstruktur der Faktoren offensichtlich zerstört. Wird nur eine Spaltenpivotsuche durchgeführt, so bleibt in der Zerlegung von \mathbf{PA} die Matrix \mathbf{L} eine $(p, 0)$ -Bandmatrix, die Bandbreite von \mathbf{R} kann aber erhöht werden zu $p + q$. Dies sieht man so ein: Im ersten Eliminationsschritt sind bei jeder Wahl des Pivotelements in der ersten Spalte von \mathbf{L} unterhalb der Diagonale höchstens p Elemente

von 0 verschieden. Die Bandbreite wird am stärksten vergrößert, wenn $a_{p+1,1}$ als Pivotelement gewählt wird. In diesem Fall sind nach der Vertauschung in der ersten Zeile höchstens $p + q$ Elemente rechts von der Diagonale besetzt, und es können durch die Elimination in der Matrix $a(2 : p, 2 : p + q)$ von 0 verschiedene Elemente erzeugt worden sein. Die untere Bandbreite p ist also nicht verändert worden und die obere auf höchstens $p + q$ vergrößert worden. Gleiche Überlegungen gelten auch für die folgenden Eliminationsschritte.

Wir haben Algorithmus 4.20. so notiert, als werde die Bandmatrix \mathbf{A} in einem Array der Dimension (n, n) gespeichert. Dies ist natürlich nicht sinnvoll, wenn die Bandbreiten p und q klein sind verglichen mit der Dimension n des Problems. Eine verbreitete Möglichkeit ist es, die Spalten von \mathbf{A} in einem $(p + q + 1, n)$ Array $\tilde{\mathbf{A}}$ zu speichern gemäß $a(i, j) = \tilde{a}(i - j + q + 1, j)$, d.h.

$$\tilde{\mathbf{A}} = \begin{pmatrix} 0 & 0 & 0 & \dots & 0 & a_{1,q+1} & a_{2,q+2} & \dots \\ 0 & 0 & 0 & \dots & a_{1,q} & a_{2,q+1} & a_{3,q+2} & \dots \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \\ 0 & a_{12} & a_{23} & \dots & a_{q-1,q} & a_{q,q+1} & a_{q+1,q+2} & \dots \\ a_{11} & a_{22} & a_{33} & \dots & a_{q,q} & a_{q+1,q+1} & a_{q+2,q+2} & \dots \\ a_{21} & a_{32} & a_{43} & \dots & a_{q+1,q} & a_{q+2,q+1} & a_{q+3,q+2} & \dots \\ \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \\ a_{p+1,1} & a_{p+2,2} & a_{p+3,3} & \dots & a_{p+q,q} & a_{p+q+1,q+1} & a_{p+q+2,q+2} & \dots \end{pmatrix}$$

Die Zeilen von $\tilde{\mathbf{A}}$ enthalten dann gerade die Diagonalen der Matrix \mathbf{A} . Soll auf diese Matrix die Gauß Elimination mit Spaltenpivotsuche angewandt werden, so wird man weitere p Zeilen an den Anfang des Arrays $\tilde{\mathbf{A}}$ stellen, um die von Null verschiedenen Elemente aufzunehmen, die im Verlauf des Algorithmus erzeugt werden.

4.4 Störungen linearer Systeme

Wir wollen nun den Begriff der **Kondition** einer Matrix einführen, deren Wert – wie oben angedeutet – verantwortlich ist für die numerische Behandelbarkeit eines linearen Gleichungssystems. Wir betrachten dazu neben dem linearen Gleichungssystem

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{4.5}$$

mit der regulären Matrix $\mathbf{A} \in \mathbb{C}^{(n,n)}$ ein gestörtes System

$$(\mathbf{A} + \Delta\mathbf{A})(\mathbf{x} + \Delta\mathbf{x}) = \mathbf{b} + \Delta\mathbf{b} \tag{4.6}$$

und fragen uns, wie die Lösung des ursprünglichen Systems auf diese Störungen reagiert.

Bemerkung 4.21. Kleine Störungen kann man bei der praktischen Lösung linearer Gleichungssysteme grundsätzlich nicht ausschließen. Einerseits können die Eingangsdaten des Systems aus Messungen herrühren und somit a priori fehlerbehaftet sein. Andererseits wird man bei der Benutzung eines elektronischen Rechners durch die endliche Genauigkeit der Zahlendarstellungen auf dem Rechner Eingabefehler machen müssen.

Man muss also grundsätzlich davon ausgehen, dass man mit gestörten Systemen anstelle der wirklich zu lösenden Systeme rechnen muss. Allerdings kann man meistens annehmen, dass die Störungen klein sind. \square

Bevor wir die Wirkung von Störungen untersuchen können, benötigen wir noch einige Aussagen über sogenannte Matrixnormen, durch die wir die Größe einer Matrix messen.

Definition 4.22. Es sei $\mathbf{A} \in \mathbb{C}^{(m,n)}$ eine Matrix, $\|\cdot\|_n$ eine Vektornorm auf \mathbb{C}^n und $\|\cdot\|_m$ eine Vektornorm auf \mathbb{C}^m . Dann heißt

$$\|\mathbf{A}\|_{m,n} := \max_{\mathbf{x} \neq \mathbf{0}} \frac{\|\mathbf{A}\mathbf{x}\|_m}{\|\mathbf{x}\|_n}$$

die **Matrixnorm** von \mathbf{A} , die den Normen $\|\cdot\|_n$ und $\|\cdot\|_m$ zugeordnet ist.

Bemerkung 4.23. Wegen $\frac{\|\mathbf{A}\mathbf{x}\|_m}{\|\mathbf{x}\|_n} = \left\| \mathbf{A} \left(\frac{\mathbf{x}}{\|\mathbf{x}\|_n} \right) \right\|_m$ genügt es, das Maximum über Vektoren der Länge 1 zu erstrecken:

$$\|\mathbf{A}\|_{m,n} = \max\{\|\mathbf{A}\mathbf{y}\|_m : \|\mathbf{y}\|_n = 1\}.$$

Die Existenz des Maximums (daß es also einen Vektor \mathbf{x} mit $\|\mathbf{x}\|_n = 1$ und $\|\mathbf{A}\mathbf{x}\|_m = \|\mathbf{A}\|_{m,n}$ gibt) folgt aus der Stetigkeit der Abbildung $\mathbf{x} \mapsto \|\mathbf{A}\mathbf{x}\|_m$. \square

Bemerkung 4.24. $\|\cdot\|_{m,n}$ ist eine Norm auf $\mathbb{C}^{(m,n)}$, denn

$$\begin{aligned} \|\mathbf{A}\|_{m,n} = 0 &\iff \|\mathbf{A}\mathbf{x}\|_m = 0 \quad \forall \mathbf{x} \in \mathbb{C}^n \iff \mathbf{A}\mathbf{x} = \mathbf{0} \quad \forall \mathbf{x} \in \mathbb{C}^n \iff \mathbf{A} = \mathbf{O}, \\ \|\lambda\mathbf{A}\|_{m,n} &= \max\{\|\lambda\mathbf{A}\mathbf{x}\|_m : \|\mathbf{x}\|_n = 1\} = \max\{|\lambda| \cdot \|\mathbf{A}\mathbf{x}\|_m : \|\mathbf{x}\|_n = 1\} \\ &= |\lambda| \cdot \|\mathbf{A}\|_{m,n}, \end{aligned}$$

$$\begin{aligned}
\|\mathbf{A} + \mathbf{B}\|_{m,n} &= \max\{\|\mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{x}\|_m : \|\mathbf{x}\|_n = 1\} \\
&\leq \max\{\|\mathbf{A}\mathbf{x}\|_m + \|\mathbf{B}\mathbf{x}\|_m : \|\mathbf{x}\|_n = 1\} \\
&\leq \max\{\|\mathbf{A}\mathbf{x}\|_m : \|\mathbf{x}\|_n = 1\} + \max\{\|\mathbf{B}\mathbf{x}\|_m : \|\mathbf{x}\|_n = 1\} \\
&= \|\mathbf{A}\|_{m,n} + \|\mathbf{B}\|_{m,n}.
\end{aligned}$$

Diese ist zusätzlich **submultiplikativ** im folgenden Sinne:

$$\|\mathbf{A}\mathbf{x}\|_m \leq \|\mathbf{A}\|_{m,n} \cdot \|\mathbf{x}\|_n \quad \text{für alle } \mathbf{x} \in \mathbb{C}^n, \mathbf{A} \in \mathbb{C}^{(m,n)}, \quad (4.7)$$

$$\|\mathbf{A}\mathbf{B}\|_{m,p} \leq \|\mathbf{A}\|_{m,n} \cdot \|\mathbf{B}\|_{n,p} \quad \text{für alle } \mathbf{A} \in \mathbb{C}^{(m,n)}, \mathbf{B} \in \mathbb{C}^{(n,p)}. \quad (4.8)$$

Die Ungleichung (4.7) folgt sofort aus der Definition 4.22.; denn es ist $\|\mathbf{A}\|_{m,n} \geq \frac{\|\mathbf{A}\mathbf{x}\|_m}{\|\mathbf{x}\|_n}$ für alle $\mathbf{x} \in \mathbb{C}^n$.

Die Ungleichung (4.8) erschließt man mit (4.7) wie folgt: Für alle $\mathbf{x} \in \mathbb{C}^p$ ist

$$\|\mathbf{A}\mathbf{B}\mathbf{x}\|_m = \|\mathbf{A}(\mathbf{B}\mathbf{x})\|_m \leq \|\mathbf{A}\|_{m,n} \cdot \|\mathbf{B}\mathbf{x}\|_n \leq \|\mathbf{A}\|_{m,n} \cdot \|\mathbf{B}\|_{n,p} \cdot \|\mathbf{x}\|_p,$$

Also ist $\|\mathbf{A}\mathbf{B}\|_{m,p} = \max\{\|\mathbf{A}\mathbf{B}\mathbf{x}\|_m : \|\mathbf{x}\|_p = 1\} \leq \|\mathbf{A}\|_{m,n} \cdot \|\mathbf{B}\|_{n,p}$. \square

Bemerkung 4.25. Die Matrixnorm $\|\mathbf{A}\|_{m,n}$ ist die kleinste nichtnegative reelle Zahl μ , mit der

$$\|\mathbf{A}\mathbf{x}\|_m \leq \mu \cdot \|\mathbf{x}\|_n \quad \forall \mathbf{x} \in \mathbb{C}^n$$

ist. $\|\mathbf{A}\|_{m,n}$ ist demnach die maximale Verlängerung, die ein \mathbf{x} durch Abbildung mit \mathbf{A} erfahren kann, wobei \mathbf{x} selbst in der $\|\cdot\|_n$ -Norm und $\mathbf{A}\mathbf{x}$ in der Norm $\|\cdot\|_m$ des Bildraumes gemessen wird. \square

Wir betrachten nun nur noch den Fall, daß im Urbildraum und im Bildraum dieselbe Norm verwendet wird, auch wenn beide Räume verschiedene Dimension haben, und verwenden für die Matrixnorm dasselbe Symbol wie für die Vektornorm. Für $\mathbf{A} \in \mathbb{R}^{(5,9)}$ bezeichnet also $\|\mathbf{A}\|_\infty$ die Matrixnorm von \mathbf{A} gemäß Definition 4.22., wenn sowohl im Urbildraum \mathbb{R}^9 als auch im Bildraum \mathbb{R}^5 die Maximumnorm verwendet wird.

Der folgende Satz 4.26. enthält die den wichtigsten Vektornormen zugeordneten Matrixnormen:

Satz 4.26. *Es sei $\mathbf{A} \in \mathbb{C}^{(m,n)}$ gegeben.*

(i) Die **Zeilensummennorm**

$$\|\mathbf{A}\|_\infty := \max_{i=1,\dots,m} \sum_{j=1}^n |a_{ij}|$$

ist der Maximumnorm $\|\mathbf{x}\|_\infty := \max_{i=1,\dots,n} |x_i|$ zugeordnet.

(ii) Die **Spektralnorm**

$$\|\mathbf{A}\|_2 := \max\{\sqrt{\lambda} : \lambda \text{ ist Eigenwert von } \mathbf{A}^H \mathbf{A}\}$$

ist der Euklidischen Norm zugeordnet.

(iii) Die **Spaltensummennorm**

$$\|\mathbf{A}\|_1 := \max_{j=1,\dots,n} \sum_{i=1}^m |a_{ij}|$$

ist der Summennorm $\|\mathbf{x}\|_1 := \sum_{i=1}^n |x_i|$ zugeordnet.

Beweis: (i): Für alle $\mathbf{x} \in \mathbb{C}^n$ gilt

$$\|\mathbf{Ax}\|_\infty = \max_{i=1,\dots,m} \left| \sum_{j=1}^n a_{ij} x_j \right| \leq \max_{i=1,\dots,m} \sum_{j=1}^n |a_{ij}| \cdot |x_j| \leq \|\mathbf{x}\|_\infty \cdot \max_{i=1,\dots,m} \sum_{j=1}^n |a_{ij}|,$$

und daher

$$\|\mathbf{A}\|_\infty \leq \max_{i=1,\dots,m} \sum_{j=1}^n |a_{ij}|. \quad (4.9)$$

Sei $k \in \{1, \dots, m\}$ mit $\sum_{j=1}^n |a_{kj}| \leq \sum_{j=1}^n |a_{ij}|$ für alle i . Wir definieren $\mathbf{x} \in \mathbb{C}^n$ durch $x_j := 1$, falls $a_{kj} = 0$, und $x_j := \overline{a_{kj}}/|a_{kj}|$, sonst. Dann gilt $\|\mathbf{x}\|_\infty = 1$ und

$$\|\mathbf{Ax}\|_\infty = \max_{i=1,\dots,m} \left| \sum_{j=1}^n a_{ij} x_j \right| \geq \left| \sum_{j=1}^n a_{kj} x_j \right| = \left| \sum_{j=1}^n |a_{kj}| \right| = \max_{i=1,\dots,m} \sum_{j=1}^n |a_{ij}|,$$

und daher

$$\|\mathbf{A}\|_\infty = \max\{\|\mathbf{Ay}\|_\infty : \|\mathbf{y}\|_\infty = 1\} \geq \|\mathbf{Ax}\|_\infty \geq \max_{i=1,\dots,m} \sum_{j=1}^n |a_{ij}|,$$

zusammen mit (4.9) also die Behauptung.

(ii): Es ist $\mathbf{A}^H \mathbf{A} \in \mathbb{C}^{(n,n)}$ eine Hermitesche Matrix, und daher ist nach dem Rayleighschen Prinzip

$$\max_{\mathbf{x} \neq \mathbf{0}} \frac{\|\mathbf{Ax}\|_2^2}{\|\mathbf{x}\|_2^2} = \max_{\mathbf{x} \neq \mathbf{0}} \frac{\mathbf{x}^H \mathbf{A}^H \mathbf{Ax}}{\mathbf{x}^H \mathbf{x}}$$

der maximale Eigenwert von $\mathbf{A}^H \mathbf{A}$.

(iii): Zeigt man ähnlich wie (i). ■

Beispiel 4.27.

$$\mathbf{A} = \begin{pmatrix} 1 & 0.1 & -0.1 \\ 0.1 & 2 & -0.4 \\ 0.2 & 0.4 & 3 \end{pmatrix}.$$

Es ist

$$\|\mathbf{A}\|_{\infty} = \max\{1.2, 2.5, 3.6\} = 3.6$$

$$\|\mathbf{A}\|_1 = \max\{1.3, 2.5, 3.5\} = 3.5$$

Die Spektralnorm von \mathbf{A} ist die Quadratwurzel aus dem maximalen Eigenwert von

$$\mathbf{A}^T \mathbf{A} = \begin{pmatrix} 1.05 & 0.38 & 0.46 \\ 0.38 & 4.17 & 0.39 \\ 0.46 & 0.39 & 9.17 \end{pmatrix}.$$

Nach einiger Rechnung ergibt dies

$$\|\mathbf{A}\|_2 = \sqrt{9.2294} = 3.04.$$

□

Die Spektralnorm einer Matrix ist nur schwer zu berechnen. Für viele Zwecke genügt jedoch die folgende Abschätzung:

Satz 4.28. Für alle $\mathbf{A} \in \mathbb{C}^{(m,n)}$ gilt

$$\|\mathbf{A}\|_2 \leq \|\mathbf{A}\|_S := \sqrt{\sum_{i=1}^m \sum_{j=1}^n |a_{ij}|^2}.$$

Bemerkung 4.29. $\|\mathbf{A}\|_S$ heißt die **Schur Norm** oder **Erhard Schmidt Norm** (oder – speziell in der anglo-amerikanischen Literatur – **Frobenius Norm**¹) der Matrix \mathbf{A} . $\|\cdot\|_S$ ist zwar eine Vektornorm auf $\mathbb{C}^{(m,n)}$ (= Euklidische Norm auf $\mathbb{C}^{m \cdot n}$), aber keine einer Vektornorm zugeordnete Matrixnorm, denn für eine solche gilt im Fall $n = m$ stets

$$\|\mathbf{E}\| = \max\{\|\mathbf{E}\mathbf{x}\| : \|\mathbf{x}\| = 1\} = 1.$$

Es ist aber $\|\mathbf{E}\|_S = \sqrt{n}$.

□

Beweis: (von Satz 4.28.)

Wegen der Cauchy-Schwarzschen Ungleichung gilt für alle $\mathbf{x} \in \mathbb{C}^n$

$$\|\mathbf{A}\mathbf{x}\|_2^2 = \sum_{i=1}^m \left| \sum_{j=1}^n a_{ij} x_j \right|^2 \leq \left(\sum_{i=1}^m \sum_{j=1}^n |a_{ij}|^2 \right) \sum_{j=1}^n |x_j|^2 = \|\mathbf{A}\|_S^2 \cdot \|\mathbf{x}\|_2^2,$$

und daher gilt $\|\mathbf{A}\|_2 \leq \|\mathbf{A}\|_S$. ■

¹Dort wird auch die Bezeichnung $\|\mathbf{A}\|_F$ unserer Bezeichnung $\|\mathbf{A}\|_S$ vorgezogen.

Beispiel 4.30. Für die Matrix \mathbf{A} aus Beispiel 4.27. erhält man

$$\|\mathbf{A}\|_2 \leq \|\mathbf{A}\|_S = \sqrt{14.39} \leq 3.80.$$

□

Wir betrachten nun das gestörte System (4.6) und nehmen an, dass die Störungen der Matrixelemente so klein sind, dass auch die Matrix $\mathbf{A} + \Delta\mathbf{A}$ regulär ist (dass dies für hinreichend kleine Störungen bei regulärer Matrix \mathbf{A} überhaupt stets möglich ist, wird aus dem Satz 4.31. unten folgen). Löst man mit dieser Annahme (4.6) nach $\Delta\mathbf{x}$ auf, so erhält man für den durch die Störungen $\Delta\mathbf{A}$ und $\Delta\mathbf{b}$ hervorgerufenen absoluten Fehler wegen $\mathbf{A}\mathbf{x} = \mathbf{b}$

$$\begin{aligned} \Delta\mathbf{x} &= (\mathbf{A} + \Delta\mathbf{A})^{-1}(\Delta\mathbf{b} - \Delta\mathbf{A}\mathbf{x}) \\ &= (\mathbf{I} + \mathbf{A}^{-1}\Delta\mathbf{A})^{-1}\mathbf{A}^{-1}(\Delta\mathbf{b} - \Delta\mathbf{A}\mathbf{x}). \end{aligned}$$

Mit einer beliebigen Vektornorm $\|\cdot\|$ auf dem \mathbb{C}^n und der gleich bezeichneten zugeordneten Matrixnorm kann man also abschätzen

$$\|\Delta\mathbf{x}\| \leq \|(\mathbf{I} + \mathbf{A}^{-1}\Delta\mathbf{A})^{-1}\| \cdot \|\mathbf{A}^{-1}\| (\|\Delta\mathbf{b}\| + \|\Delta\mathbf{A}\| \cdot \|\mathbf{x}\|).$$

Für $\mathbf{b} \neq \mathbf{0}$ und folglich $\mathbf{x} \neq \mathbf{0}$ erhält man daraus für den relativen Fehler $\|\Delta\mathbf{x}\|/\|\mathbf{x}\|$ die Abschätzung

$$\frac{\|\Delta\mathbf{x}\|}{\|\mathbf{x}\|} \leq \|(\mathbf{I} + \mathbf{A}^{-1}\Delta\mathbf{A})^{-1}\| \cdot \|\mathbf{A}^{-1}\| \left(\frac{\|\Delta\mathbf{b}\|}{\|\mathbf{x}\|} + \|\Delta\mathbf{A}\| \right). \quad (4.10)$$

Um in dieser Ungleichung $\|(\mathbf{I} + \mathbf{A}^{-1}\Delta\mathbf{A})^{-1}\|$ weiter abzuschätzen, benötigen wir das folgende Störungslemma, das gleichzeitig darüber Auskunft gibt, unter wie großen Störungen der Matrixelemente die Existenz der Inversen für die gestörte Matrix noch gesichert ist.

Satz 4.31. (Störungslemma) *Es sei $\mathbf{B} \in \mathbb{C}^{(n,n)}$ und es gelte für eine beliebige einer Vektornorm zugeordneten Matrixnorm die Ungleichung $\|\mathbf{B}\| < 1$. Dann ist die Matrix $\mathbf{I} - \mathbf{B}$ regulär, und es gilt*

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

Beweis: Für alle $\mathbf{x} \in \mathbb{C}^n$, $\mathbf{x} \neq \mathbf{0}$, gilt

$$\|(\mathbf{I} - \mathbf{B})\mathbf{x}\| \geq \|\mathbf{x}\| - \|\mathbf{B}\mathbf{x}\| \geq \|\mathbf{x}\| - \|\mathbf{B}\|\|\mathbf{x}\| = (1 - \|\mathbf{B}\|)\|\mathbf{x}\| > 0,$$

d.h. $(\mathbf{I} - \mathbf{B})\mathbf{x} = \mathbf{0}$ ist nur für $\mathbf{x} = \mathbf{0}$ lösbar, und daher ist $\mathbf{I} - \mathbf{B}$ regulär. Die Abschätzung der Norm der Inversen von $\mathbf{I} - \mathbf{B}$ erhält man so:

$$\begin{aligned} 1 &= \|(\mathbf{I} - \mathbf{B})^{-1}(\mathbf{I} - \mathbf{B})\| = \|(\mathbf{I} - \mathbf{B})^{-1} - (\mathbf{I} - \mathbf{B})^{-1}\mathbf{B}\| \\ &\geq \|(\mathbf{I} - \mathbf{B})^{-1}\| - \|(\mathbf{I} - \mathbf{B})^{-1}\mathbf{B}\| \\ &\geq \|(\mathbf{I} - \mathbf{B})^{-1}\| - \|(\mathbf{I} - \mathbf{B})^{-1}\| \cdot \|\mathbf{B}\| = (1 - \|\mathbf{B}\|) \cdot \|(\mathbf{I} - \mathbf{B})^{-1}\|. \end{aligned}$$

■

Mit dem Störungslemma (mit $\mathbf{B} = -\mathbf{A}^{-1}\Delta\mathbf{A}$) können wir nun den relativen Fehler des gestörten Problems aus (4.10) weiter abschätzen, wobei wir $\|\mathbf{A}^{-1}\Delta\mathbf{A}\| \leq \|\mathbf{A}^{-1}\| \cdot \|\Delta\mathbf{A}\|$ und $\|\mathbf{b}\| = \|\mathbf{A}\mathbf{x}\| \leq \|\mathbf{A}\| \cdot \|\mathbf{x}\|$ beachten.

$$\begin{aligned} \frac{\|\Delta\mathbf{x}\|}{\|\mathbf{x}\|} &\leq \frac{\|\mathbf{A}^{-1}\|}{1 - \|\mathbf{A}^{-1}\| \cdot \|\Delta\mathbf{A}\|} \left(\|\mathbf{A}\| \frac{\|\Delta\mathbf{b}\|}{\|\mathbf{b}\|} + \|\Delta\mathbf{A}\| \right) \\ &\leq \frac{\|\mathbf{A}^{-1}\| \cdot \|\mathbf{A}\|}{1 - \|\mathbf{A}^{-1}\| \cdot \|\mathbf{A}\| \frac{\|\Delta\mathbf{A}\|}{\|\mathbf{A}\|}} \left(\frac{\|\Delta\mathbf{A}\|}{\|\mathbf{A}\|} + \frac{\|\Delta\mathbf{b}\|}{\|\mathbf{b}\|} \right). \end{aligned} \quad (4.11)$$

Aus der Abschätzung (4.11) liest man ab, dass für kleine Störungen der Matrixelemente (so dass der Nenner nicht wesentlich von 1 abweicht) der relative Fehler der rechten Seite und der Matrixelemente um den Faktor $\|\mathbf{A}^{-1}\| \cdot \|\mathbf{A}\|$ verstärkt wird. Diesen Verstärkungsfaktor nennen wir die Kondition der Matrix \mathbf{A} .

Definition 4.32. Es sei $\mathbf{A} \in \mathbb{C}^{(n,n)}$ regulär und $\|\cdot\|_p$ eine einer (gleichbezeichneten) Vektornorm zugeordnete Matrixnorm auf $\mathbb{C}^{(n,n)}$. Dann heißt

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

die **Kondition** der Matrix \mathbf{A} (oder des linearen Gleichungssystems (4.5)) bezüglich der Norm $\|\cdot\|_p$.

Wir fassen unsere Überlegungen zusammen:

Satz 4.33. Es seien $\mathbf{A}, \Delta\mathbf{A} \in \mathbb{C}^{(n,n)}$ und $\mathbf{b}, \Delta\mathbf{b} \in \mathbb{C}^n$, $\mathbf{b} \neq \mathbf{0}$, gegeben, so dass \mathbf{A} regulär ist und $\|\mathbf{A}^{-1}\| \cdot \|\Delta\mathbf{A}\| < 1$ mit einer Vektornorm zugeordneten Matrixnorm $\|\cdot\|$ gilt. Dann existiert neben der Lösung des linearen Gleichungssystems (4.5) auch die Lösung $\mathbf{x} + \Delta\mathbf{x}$ des gestörten Systems (4.6), und es gilt mit der Kondition $\kappa(\mathbf{A}) := \|\mathbf{A}\| \cdot \|\mathbf{A}^{-1}\|$ die Abschätzung

$$\frac{\|\Delta\mathbf{x}\|}{\|\mathbf{x}\|} \leq \frac{\kappa(\mathbf{A})}{1 - \kappa(\mathbf{A}) \cdot \frac{\|\Delta\mathbf{A}\|}{\|\mathbf{A}\|}} \left(\frac{\|\Delta\mathbf{A}\|}{\|\mathbf{A}\|} + \frac{\|\Delta\mathbf{b}\|}{\|\mathbf{b}\|} \right).$$

Bemerkung 4.34. Für jede reguläre Matrix \mathbf{A} und jede Norm $\|\cdot\|$ gilt $\kappa(\mathbf{A}) \geq 1$, denn

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

□

Bemerkung 4.35. Werden die Rechnungen bei der Lösung eines linearen Gleichungssystems mit der Mantissenlänge ℓ ausgeführt, so haben die Daten von \mathbf{A} und \mathbf{b} bereits einen relativen Eingabefehler der Größe $5 \cdot 10^{-\ell}$. Gilt $\kappa(\mathbf{A}) = 10^\gamma$, so ist (abgesehen von Rundungsfehlern, die sich im numerischen Lösungsverfahren ergeben) für die numerische Lösung mit einem relativen Fehler der Größe $5 \cdot 10^{\gamma-\ell}$ zu rechnen. Grob gesprochen verliert man also beim Lösen eines linearen Gleichungssystems γ Stellen, wenn die Koeffizientenmatrix eine Kondition der Größenordnung 10^γ besitzt. Dieser Verlust von Stellen ist nicht dem jeweilig verwendeten Algorithmus zuzuschreiben. Er ist problemimmanent. □

Beispiel 4.36. Wir betrachten das lineare Gleichungssystem

$$\begin{pmatrix} 1 & 1 \\ 1 & 0.999 \end{pmatrix} \mathbf{x} = \begin{pmatrix} 2 \\ 1.999 \end{pmatrix},$$

das offensichtlich die Lösung $\mathbf{x} = (1, 1)^T$ besitzt. Für den Vektor $\mathbf{x} + \Delta\mathbf{x} := (5, -3.002)^T$ gilt

$$\mathbf{A}(\mathbf{x} + \Delta\mathbf{x}) = \begin{pmatrix} 1.998 \\ 2.001002 \end{pmatrix} =: \mathbf{b} + \Delta\mathbf{b}.$$

Es ist also

$$\frac{\|\Delta\mathbf{b}\|_\infty}{\|\mathbf{b}\|_\infty} = 1.001 \cdot 10^{-3} \quad \text{und} \quad \frac{\|\Delta\mathbf{x}\|_\infty}{\|\mathbf{x}\|_\infty} = 4.002,$$

und daher gilt für die Kondition

$$\kappa_\infty(\mathbf{A}) \geq \frac{4.002}{1.001} 10^3 = 3998.$$

Tatsächlich gilt $\mathbf{A}^{-1} = \begin{pmatrix} -999 & 1000 \\ 1000 & -1000 \end{pmatrix}$ und daher $\kappa_\infty(\mathbf{A}) = 4000$. Man sieht an diesem Beispiel, dass die Abschätzung in (4.11) scharf ist. □

Der nächste Satz gibt eine geometrische Charakterisierung der Konditionszahl. Er sagt, dass der relative Abstand einer regulären Matrix zur nächsten singulären Matrix in der Euklidischen Norm gleich dem Reziproken der Kondition ist.

Satz 4.37. *Es sei $\mathbf{A} \in \mathbb{C}^{(n,n)}$ regulär. Dann gilt*

$$\min \left\{ \frac{\|\Delta\mathbf{A}\|_2}{\|\mathbf{A}\|_2} : \mathbf{A} + \Delta\mathbf{A} \text{ singular} \right\} = \frac{1}{\kappa_2(\mathbf{A})}.$$

Beweis: Es genügt zu zeigen, dass

$$\min \{ \|\Delta \mathbf{A}\|_2 : \mathbf{A} + \Delta \mathbf{A} \text{ singular} \} = 1/\|\mathbf{A}^{-1}\|_2.$$

Dass das Minimum wenigstens $1/\|\mathbf{A}^{-1}\|_2$ ist, folgt aus dem Störungslemma, denn für $\|\Delta \mathbf{A}\|_2 < 1/\|\mathbf{A}^{-1}\|_2$ gilt $1 > \|\Delta \mathbf{A}\|_2 \|\mathbf{A}^{-1}\|_2 \geq \|\mathbf{A}^{-1} \Delta \mathbf{A}\|_2$, und daher besitzt $\mathbf{I} + \mathbf{A}^{-1} \Delta \mathbf{A} = \mathbf{A}^{-1}(\mathbf{A} + \Delta \mathbf{A})$, und damit auch $\mathbf{A} + \Delta \mathbf{A}$ eine Inverse.

Wir konstruieren nun eine Matrix $\Delta \mathbf{A}$, so dass $\mathbf{A} + \Delta \mathbf{A}$ singular ist und für die $\|\Delta \mathbf{A}\|_2 = 1/\|\mathbf{A}^{-1}\|_2$ gilt. Damit ist dann gezeigt, dass das Minimum größer oder gleich $1/\|\mathbf{A}^{-1}\|_2$ gilt. Wegen

$$\|\mathbf{A}^{-1}\|_2 = \max_{\mathbf{x} \neq \mathbf{0}} \frac{\|\mathbf{A}^{-1} \mathbf{x}\|_2}{\|\mathbf{x}\|_2}$$

existiert ein \mathbf{x} mit $\|\mathbf{x}\|_2 = 1$ und $\|\mathbf{A}^{-1}\|_2 = \|\mathbf{A}^{-1} \mathbf{x}\|_2 > 0$. Wir definieren hiermit

$$\mathbf{y} := \frac{\mathbf{A}^{-1} \mathbf{x}}{\|\mathbf{A}^{-1} \mathbf{x}\|_2} = \frac{\mathbf{A}^{-1} \mathbf{x}}{\|\mathbf{A}^{-1}\|_2} \quad \text{und} \quad \Delta \mathbf{A} := -\frac{\mathbf{x} \mathbf{y}^H}{\|\mathbf{A}^{-1}\|_2}.$$

Dann gilt $\|\mathbf{y}\|_2 = 1$ und

$$\|\Delta \mathbf{A}\|_2 = \max_{\mathbf{z} \neq \mathbf{0}} \frac{\|\mathbf{x} \mathbf{y}^H \mathbf{z}\|_2}{\|\mathbf{A}^{-1}\|_2 \|\mathbf{z}\|_2} = \max_{\mathbf{z} \neq \mathbf{0}} \frac{|\mathbf{y}^H \mathbf{z}|}{\|\mathbf{z}\|_2} \frac{\|\mathbf{x}\|_2}{\|\mathbf{A}^{-1}\|_2} = \frac{1}{\|\mathbf{A}^{-1}\|_2},$$

wobei das Maximum z.B. für $\mathbf{z} = \mathbf{y}$ angenommen wird. Wegen

$$(\mathbf{A} + \Delta \mathbf{A}) \mathbf{y} = \mathbf{A} \mathbf{y} - \frac{\mathbf{x} \mathbf{y}^H \mathbf{y}}{\|\mathbf{A}^{-1}\|_2} = \frac{\mathbf{x}}{\|\mathbf{A}^{-1}\|_2} - \frac{\mathbf{x}}{\|\mathbf{A}^{-1}\|_2} = \mathbf{0}$$

ist $\mathbf{A} + \Delta \mathbf{A}$ singular. ■

4.5 Lineare Systeme mit spezieller Struktur

In diesem Abschnitt behandeln wir Algorithmen zur Lösung von linearen Gleichungssystemen, wobei die Koeffizientenmatrix eine spezielle Struktur besitzt und die daher mit weniger als $O(n^3)$ Operationen gelöst werden können. Abweichend von den vorhergehenden Abschnitten beginnen wir bei der Numerierung der Zeilen und Spalten der Koeffizientenmatrix und der Komponenten der Vektoren in diesem Abschnitt nicht bei 1, sondern bei 0.

4.5.1 Vandermonde Systeme

Wir betrachten in diesem Abschnitt lineare Gleichungssysteme, deren Koeffizientenmatrix eine Vandermondesche Matrix ist. Es sei $\mathbf{x} = x(0:n) \in \mathbb{R}^{n+1}$ mit $x_j \neq x_k$ für $j, k \in \{0, 1, \dots, n\}$ und hiermit

$$\mathbf{V} = V(x_0, \dots, x_n) = \begin{pmatrix} 1 & x_0 & x_0^2 & \dots & x_0^n \\ 1 & x_1 & x_1^2 & \dots & x_1^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \dots & x_n^n \end{pmatrix}$$

Aus der Vorlesung Lineare Algebra ist bekannt, dass die Matrix \mathbf{V} regulär ist und dass bei gegebenem $\mathbf{f} = f(0:n) \in \mathbb{R}^{n+1}$ für die Lösung $\mathbf{a} = a(0:n)$ des Gleichungssystems $\mathbf{V}\mathbf{a} = \mathbf{f}$ das Polynom

$$p(x) = \sum_{j=0}^n a_j x^j \quad (4.12)$$

das eindeutige Polynom vom Grad n ist, das die Interpolationsbedingungen $p(x_j) = f_j$, $j = 0, 1, \dots, n$, erfüllt. Die a_j können wir auch mit Hilfe des Newtonschen Interpolationspolynoms bestimmen. Schreiben wir p in der Gestalt

$$p(x) = \sum_{k=0}^n c_k \prod_{i=0}^{k-1} (x - x_i),$$

so sind die c_k die dividierten Differenzen der x_j und können berechnet werden gemäß:

```

c(0:n)=f(0:n);
for k=0 : n-1
    for i=n : -1 : k+1
        c(i)=(c(i)-c(i-1))/(x(i)-x(i-k-1));
    end
end
end

```

Wir bestimmen nun die a_j aus den c_j . Dazu seien die Polynome

$$p_k(x) := a_k^{(k)} + a_{k+1}^{(k)}x + \dots + a_n^{(k)}x^{n-k}$$

rekursiv definiert durch

$$p_n(x) \equiv c_n, \quad p_k(x) := c_k + (x - x_k)p_{k+1}(x), \quad k = n - 1 : -1 : 0.$$

Durch Koeffizientenvergleich erhält man dann

$$a_n^{(n)} = c_n, \quad a_k^{(k)} = c_k - x_k a_{k+1}^{(k+1)}, \quad a_i^{(k)} = a_i^{(k+1)} - x_k a_{i+1}^{(k+1)}, \quad i = k+1 : n-1.$$

Damit ergibt sich die folgende Methode zur Berechnung der Lösung \mathbf{a} des Vandermonde Systems $\mathbf{V}\mathbf{a} = \mathbf{f}$.

Algorithmus 4.38. (Vandermonde Systeme)

```

for k=0 : n-1
  for i=n : -1 : k+1
    f(i)=(f(i)-f(i-1))/(x(i)-x(i-k-1));
  end
end
for k=n-1 : -1 : 0
  for i=k : n-1
    f(i)=f(i)-f(i+1)*x(k);
  end
end
end

```

Dabei überschreibt zunächst der Vektor \mathbf{c} den Vektor \mathbf{f} und danach der Lösungsvektor \mathbf{a} . Man zählt sofort ab, dass dieser Lösungsalgorithmus $2.5n^2$ flops erfordert. Ein effizientes Verfahren für das System $\mathbf{V}^T \mathbf{y} = \mathbf{b}$ erhält man, indem man Algorithmus 4.38. als Zerlegungsmethode interpretiert. Dazu definieren wir für $\alpha \in \mathbb{R}$ die Bidiagonalmatrix

$$\mathbf{L}_k(\alpha) := \left(\begin{array}{c|cccc} \mathbf{I}_k & & & & \mathbf{0}^T \\ \hline & 1 & 0 & \dots & 0 & 0 \\ & -\alpha & 1 & \dots & 0 & 0 \\ \mathbf{0} & \vdots & \ddots & \ddots & \ddots & \vdots \\ & 0 & 0 & \dots & 1 & 0 \\ & 0 & 0 & \dots & -\alpha & 1 \end{array} \right) \in \mathbb{R}^{(n+1, n+1)}$$

und die Diagonalmatrix

$$\mathbf{D}_k = \text{diag} \{1, \dots, 1, x_{k+1} - x_0, \dots, x_n - x_{n-k-1}\} \in \mathbb{R}^{(n+1, n+1)}.$$

Man rechnet leicht nach, dass man den Algorithmus zur Berechnung der dividierten Differenzen \mathbf{c} aus den Interpolationsdaten schreiben kann als

$$\mathbf{c} = \mathbf{D}_{n-1}^{-1} \mathbf{L}_{n-1}(1) \mathbf{D}_{n-2}^{-1} \dots \mathbf{D}_0^{-1} \mathbf{L}_0(1) \mathbf{f} =: \mathbf{R}^T \mathbf{f}.$$

Ähnlich kann man die Berechnung des Vektors \mathbf{a} aus den Koeffizienten c_j des Newtonschen Interpolationspolynoms schreiben als

$$\mathbf{a} = \mathbf{L}^T \mathbf{c}$$

mit der unteren Dreiecksmatrix \mathbf{L} , die definiert ist durch

$$\mathbf{L}^T := \mathbf{L}_0(x_0)^T \dots \mathbf{L}_{n-1}(x_{n-1})^T.$$

Zusammen gilt

$$\mathbf{a} = \mathbf{L}^T \mathbf{R}^T \mathbf{f}, \quad \text{d.h. } \mathbf{V}^{-1} = \mathbf{L}^T \mathbf{R}^T,$$

und daher erhält man die Lösung von $\mathbf{V}^T \mathbf{y} = \mathbf{b}$ durch

$$\mathbf{y} = \mathbf{V}^{-T} \mathbf{b} = \mathbf{R} \mathbf{L} \mathbf{b}.$$

Unter Benutzung der Definition von \mathbf{L} und \mathbf{R} erhält man damit den folgenden Algorithmus zur Lösung des transponierten Vandermondeschen Systems, der wiederum $2.5n^2$ flops benötigt:

Algorithmus 4.39. (Transponierte Vandermonde Systeme)

```

for k=0 : n-1
  for i=n : -1 : k+1
    b(i)=b(i)-x(k)*b(i-1);
  end
end
for k=n-1 : -1 : 0
  for i=k+1 : n
    b(i)=b(i)/(x(i)-x(i-k-1));
  end
  for i=k : n-1
    b(i)=b(i)-b(i+1);
  end
end

```

Die vorgestellten Algorithmen wurden von Björk und Pereyra [11] entwickelt. Es wurden von ihnen eine Reihe von Beispielen gerechnet. Die numerischen Ergebnisse sind (überraschend) gut, obwohl Vandermonde Matrizen häufig sehr schlechte Kondition besitzen.

Beispiel 4.40. Für $n = 16$ und $x_i = i/16$, $i = 0, \dots, 16$, hat die Vandermonde Matrix \mathbf{V} die Kondition $2.42E + 13$. Löst man also ein lineares Gleichungssystem mit der Koeffizientenmatrix \mathbf{V} oder \mathbf{V}^T in doppelter Genauigkeit, so kann man erwarten, dass der relative Fehler die Größenordnung $1E - 03$ hat.

Wählt man die rechte Seite \mathbf{b} jeweils so, dass $\mathbf{y} = (1, \dots, 1)^T$ die exakte Lösung ist, so erhält man in Übereinstimmung damit für $\tilde{\mathbf{y}} = \mathbf{V} \setminus \mathbf{b}$ den relativen Fehler $3.62E - 04$ und für $\tilde{\mathbf{y}} = \mathbf{V}^T \setminus \mathbf{b}$ den relativen Fehler $1.65E - 04$.

Mit Algorithmus 4.38. und Algorithmus 4.39. erhält man Näherungen mit den relativen Fehlern $1.58E - 05$ und $5.41E - 06$. \square

4.5.2 Schnelle Fourier Transformation

Eine sehr wichtige Operation in den Anwendungen, z.B. in der Systemtheorie, ist die Fourier Transformation. Wir betrachten hier die diskrete Version.

Definition 4.41. Es sei $\omega_n := \exp(-2\pi i/n)$ und

$$\mathbf{F}_n := (\omega_n^{jk})_{j,k=0,\dots,n-1} \in \mathbb{R}^n.$$

Dann heißt $\mathbf{y} := \mathbf{F}_n \mathbf{x}$ die **diskrete Fourier Transformation** von $\mathbf{x} \in \mathbb{R}^n$ und $\mathbf{x} := \mathbf{F}_n^{-1} \mathbf{y}$ die **inverse diskrete Fourier Transformation** von \mathbf{y} .

Satz 4.42.

$$\mathbf{F}_n^{-1} = \frac{1}{n} \mathbf{F}_n^H = \frac{1}{n} \bar{\mathbf{F}}_n.$$

Bis auf die Normierung ist \mathbf{F}_n also eine symmetrische und unitäre Matrix.

Beweis: Die Symmetrie von \mathbf{F}_n ist unmittelbar klar, und daher gilt $\mathbf{F}_n^H = \bar{\mathbf{F}}_n$. Es ist also nur $\mathbf{F}_n \bar{\mathbf{F}}_n = n\mathbf{I}$ zu zeigen. Wegen $\bar{\omega}_n = \omega_n^{-1}$ gilt

$$(\mathbf{F}_n \bar{\mathbf{F}}_n)_{jk} = \sum_{\ell=0}^{n-1} \omega_n^{j\ell} \bar{\omega}_n^{\ell k} = \sum_{\ell=0}^{n-1} \omega_n^{\ell(j-k)}.$$

Für $j = k$ ist das Ergebnis offensichtlich gleich n . Für $j \neq k$ erhält man für die geometrische Summe

$$(\mathbf{F}_n \bar{\mathbf{F}}_n)_{jk} = \frac{1 - \omega_n^{n(j-k)}}{1 - \omega_n^{j-k}} = 0$$

wegen $\omega_n^n = 1$. \blacksquare

Die diskrete Fourier Transformation (und wegen Satz 4.42. dann auch die inverse diskrete Fourier Transformation) kann man mit weniger als $2n^2$ Operationen ausführen. Um das Vorgehen übersichtlich zu gestalten, betrachten wir nur den Fall $n = 2^3$. Ordnet man die Spalten von \mathbf{F}_8 so um, dass zunächst die geraden Indizes und dann die ungeraden Indizes der Größe nach aufgeführt werden, so erhält \mathbf{F}_n wegen $\omega_8^8 = 1$ und $\omega_8^4 = -1$ die Gestalt (wobei wir zur Abkürzung $\omega := \omega_8$ setzen)

$$\tilde{\mathbf{F}}_n = \left(\begin{array}{cccc|cccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \omega^2 & \omega^4 & \omega^6 & \omega & \omega^3 & \omega^5 & \omega^7 \\ 1 & \omega^4 & 1 & \omega^4 & \omega^2 & \omega^6 & \omega^2 & \omega^6 \\ 1 & \omega^6 & \omega^4 & \omega^2 & \omega^3 & \omega & \omega^7 & \omega^5 \\ \hline 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & \omega^2 & \omega^4 & \omega^6 & -\omega & -\omega^3 & -\omega^5 & -\omega^7 \\ 1 & \omega^4 & 1 & \omega^4 & -\omega^2 & -\omega^6 & -\omega^2 & -\omega^6 \\ 1 & \omega^6 & \omega^4 & \omega^2 & -\omega^3 & -\omega & -\omega^7 & -\omega^5 \end{array} \right).$$

Mit der Diagonalmatrix

$$\mathbf{\Omega}_4 := \text{diag} \{1, \omega_8, \omega_8^2, \omega_8^3\}$$

kann man wegen $\omega_8^2 = \omega_4$ die umgeordnete Matrix schreiben als

$$\tilde{\mathbf{F}}_8 = \left(\begin{array}{c|c} \mathbf{F}_4 & \mathbf{\Omega}_4 \mathbf{F}_4 \\ \hline \mathbf{F}_4 & -\mathbf{\Omega}_4 \mathbf{F}_4 \end{array} \right).$$

Ordnet man die Komponenten des Vektors \mathbf{x} wie die Spalten von \mathbf{F}_8 um, so folgt

$$\mathbf{F}_8 \mathbf{x} = \left(\begin{array}{c|c} \mathbf{F}_4 & \mathbf{\Omega}_4 \mathbf{F}_4 \\ \hline \mathbf{F}_4 & -\mathbf{\Omega}_4 \mathbf{F}_4 \end{array} \right) \begin{pmatrix} x(0 : 2 : 6) \\ x(1 : 2 : 7) \end{pmatrix} = \left(\begin{array}{c|c} \mathbf{I} & \mathbf{\Omega}_4 \\ \hline \mathbf{I} & -\mathbf{\Omega}_4 \end{array} \right) \begin{pmatrix} \mathbf{F}_4 x(0 : 2 : 6) \\ \mathbf{F}_4 x(1 : 2 : 7) \end{pmatrix}$$

Man kann also die diskrete Fourier Transformation $\mathbf{y} = \mathbf{F}_8 \mathbf{x}$ leicht berechnen, wenn man die Fourier Transformationen $\mathbf{F}_4 x(0 : 2 : 6)$ und $\mathbf{F}_4 x(1 : 2 : 7)$ der halben Länge kennt. Ist allgemeiner $n = 2m$, so kann $\mathbf{y} = \mathbf{F}_n \mathbf{x}$ genauso berechnet werden als

$$\mathbf{y}_T = \mathbf{F}_m x(0 : 2 : n - 2); \quad \mathbf{y}_B = \mathbf{F}_m x(1 : 2 : n - 1),$$

$$\mathbf{d}_m = (1, \omega_n, \omega_n^2, \dots, \omega_n^{m-1})^T,$$

$$y(0 : m - 1) = \mathbf{y}_T + \mathbf{d}_m \cdot * \mathbf{y}_B, \quad y(m : n - 1) = \mathbf{y}_T - \mathbf{d}_m \cdot * \mathbf{y}_B,$$

wobei “ $\cdot *$ ” (wie in MATLAB) die komponentenweise Multiplikation zweier Vektoren bezeichnet. Ist $n = 2^p$ für ein $p \in \mathbb{N}$, so kann man diese Reduktion natürlich rekursiv verwenden. Das Ergebnis ist ein schneller Algorithmus zur diskreten Fourier Transformation (**fast Fourier transform, FFT**). Wegen $\mathbf{F}_1 x = x$ erhält man

Algorithmus 4.43. (FFT, rekursiv)

```

function y=FFT(x,n);
    if n==1
        y=x;
    else
        m=n/2; ω=exp(-2πi/n);
        yT=FFT(x(0:2:n-2),m); yB=FFT(x(1:2:n-1),m);
        d=[1;ω;...;ωm-1]; z=d.*yB;
        y=[yT+z;yT-z];
    end

```

Es gibt nicht rekursive Versionen der FFT (vgl. [115]). Eine sorgfältige Implementierung benötigt $5n \log_2 n$ flops.

4.5.3 Toeplitz Matrizen

Wir betrachten in diesem Abschnitt Toeplitz Matrizen; das sind Matrizen $\mathbf{T}_n \in \mathbb{R}^n$, die konstante Diagonalen besitzen. Sie gehören damit zur größeren Klasse der persymmetrischen Matrizen, die symmetrisch bzgl. der zweiten Diagonale sind, für die also $a_{ij} = a_{n+1-j, n+1-i}$ gilt.

Definition 4.44. Seien $t_{-n+1}, t_{-n+2}, \dots, t_0, t_1, \dots, t_{n-1} \in \mathbb{R}$ gegeben. Dann heißt

$$\mathbf{T}_n = (t_{j-i})_{i,j=1,\dots,n} = \begin{pmatrix} t_0 & t_1 & t_2 & \dots & t_{n-2} & t_{n-1} \\ t_{-1} & t_0 & t_1 & \dots & t_{n-3} & t_{n-2} \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ t_{-n+2} & t_{-n+3} & t_{-n+4} & \dots & t_0 & t_1 \\ t_{-n+1} & t_{-n+2} & t_{-n+3} & \dots & t_{-1} & t_0 \end{pmatrix}$$

Toeplitz Matrix der Dimension n . Es sei $\mathbf{J}_n := (\mathbf{e}^n, \dots, \mathbf{e}^2, \mathbf{e}^1)$ die Klappmatrix (in MATLAB: `flipud`). Eine Matrix $\mathbf{A} \in \mathbb{R}^{(n,n)}$ heißt **persymmetrisch**, wenn gilt

$$\mathbf{A} = \mathbf{J}_n \mathbf{A}^T \mathbf{J}_n.$$

Offensichtlich ist eine Toeplitz Matrix \mathbf{T}_n persymmetrisch. Ist \mathbf{T}_n regulär, so folgt aus $\mathbf{J}_n^{-1} = \mathbf{J}_n$

$$\mathbf{T}_n^{-1} = (\mathbf{J}_n \mathbf{T}_n^T \mathbf{J}_n)^{-1} = \mathbf{J}_n \mathbf{T}_n^{-T} \mathbf{J}_n,$$

und daher ist auch \mathbf{T}_n^{-1} persymmetrisch. Diese Eigenschaft ermöglicht es, lineare Gleichungssysteme mit Toeplitz Matrizen mit $O(n^2)$ Operationen zu lösen. Wir beschränken uns dabei auf den symmetrischen und positiv definiten Fall und nehmen ohne Beschränkung der Allgemeinheit an, dass $t_0 = 1$ gilt. Es ist klar, dass dann mit \mathbf{T}_n auch alle Hauptuntermatrizen

$$\mathbf{T}_k = \begin{pmatrix} 1 & t_1 & t_2 & \dots & t_{k-1} \\ t_1 & 1 & t_1 & \dots & t_{k-2} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ t_{k-1} & t_{k-2} & t_{k-3} & \dots & 1 \end{pmatrix}$$

positiv definit, also regulär, sind. Wir behandeln zunächst das **Yule–Walker System**

$$\mathbf{T}_n \mathbf{x} = -(t_1, t_2, \dots, t_{n-1}, t_n)^T,$$

das in der Signalverarbeitung eine besondere Rolle spielt. Wir leiten hierfür einen Lösungsalgorithmus her, der auch bei der Lösung des allgemeinen Systems $\mathbf{T}_n \mathbf{x} = \mathbf{b}$ von Nutzen ist. Wir nehmen an, dass die Lösung \mathbf{y} des Yule–Walker Systems

$$\mathbf{T}_k \mathbf{y} = -\mathbf{t} := -(t_1, \dots, t_k)^T$$

bereits bekannt ist, und lösen nun das System der Dimension $k + 1$:

$$\begin{pmatrix} \mathbf{T}_k & \mathbf{J}_k \mathbf{t} \\ \mathbf{t}^T \mathbf{J}_k & 1 \end{pmatrix} \begin{pmatrix} \mathbf{z} \\ \alpha \end{pmatrix} = - \begin{pmatrix} \mathbf{t} \\ t_{k+1} \end{pmatrix}.$$

Die erste Zeile liefert

$$\mathbf{z} = \mathbf{T}_k^{-1}(-\mathbf{t} - \alpha \mathbf{J}_k \mathbf{t}) = \mathbf{y} - \alpha \mathbf{T}_k^{-1} \mathbf{J}_k \mathbf{t}$$

und die zweite

$$\alpha = -t_{k+1} - \mathbf{t}^T \mathbf{J}_k \mathbf{z}.$$

Die Persymmetrie von \mathbf{T}_k^{-1} besagt $\mathbf{T}_k^{-1} \mathbf{J}_k = \mathbf{J}_k \mathbf{T}_k^{-1}$, und daher gilt

$$\mathbf{z} = \mathbf{y} - \alpha \mathbf{J}_k \mathbf{T}_k^{-1} \mathbf{t} = \mathbf{y} + \alpha \mathbf{J}_k \mathbf{y}.$$

Damit erhalten wir

$$\alpha = -t_{k+1} - \mathbf{t}^T \mathbf{J}_k (\mathbf{y} + \alpha \mathbf{J}_k \mathbf{y}).$$

Wegen der positiven Definitheit von \mathbf{T}_{k+1} und

$$\begin{pmatrix} \mathbf{I}_k & \mathbf{0} \\ \mathbf{y}^T \mathbf{J}_k & 1 \end{pmatrix} \begin{pmatrix} \mathbf{T}_k & \mathbf{J}_k \mathbf{t} \\ \mathbf{t}^T \mathbf{J}_k & 1 \end{pmatrix} \begin{pmatrix} \mathbf{I}_k & \mathbf{J}_k \mathbf{y} \\ \mathbf{0}^T & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{T}_k & \mathbf{0} \\ \mathbf{0}^T & 1 + \mathbf{t}^T \mathbf{y} \end{pmatrix}$$

ist $1 + \mathbf{t}^T \mathbf{J}_k^2 \mathbf{y} = 1 + \mathbf{t}^T \mathbf{y} > 0$, und wir erhalten

$$\alpha = -\frac{t_{k+1} + \mathbf{t}^T \mathbf{J}_k \mathbf{y}}{1 + \mathbf{t}^T \mathbf{y}}.$$

Diese Aufdatierung der Lösung

$$\mathbf{y}^{(k)} = \begin{pmatrix} \mathbf{z}^{(k-1)} \\ \alpha_{k-1} \end{pmatrix} \text{ von } \mathbf{T}_k \mathbf{y} = -\mathbf{t}^{(k)} := -(t_1, \dots, t_k)^T$$

ist die Grundlage der schnellen Lösung eines Yule–Walker Systems. Wir benötigen nur noch eine Rekursion für

$$\begin{aligned} \beta_k &:= 1 + (\mathbf{t}^{(k)})^T \mathbf{y}^{(k)} \\ &= 1 + ((\mathbf{t}^{(k-1)})^T, t_k) \begin{pmatrix} \mathbf{y}^{(k-1)} + \alpha_{k-1} \mathbf{J}_{k-1} \mathbf{y}^{(k-1)} \\ \alpha_{k-1} \end{pmatrix} \\ &= 1 + (\mathbf{t}^{(k-1)})^T \mathbf{y}^{(k-1)} + \alpha_{k-1} ((\mathbf{t}^{(k-1)})^T \mathbf{J}_{k-1} \mathbf{y}^{(k-1)} + t_k) \\ &= \beta_{k-1} (1 - \alpha_{k-1}^2). \end{aligned}$$

Zusammengenommen erhalten wir den **Algorithmus von Durbin** (vgl. [35]) zur Lösung des Yule–Walker Systems.

Algorithmus 4.45. (Durbin)

```

y(1)=-t(1); beta = 1; alpha=-t(1);
for k=1 : n-1
    beta = (1 - alpha^2)beta;
    alpha=-(t(k+1)+t(k:-1:1)'*y(1:k))/beta;
    z(1:k)=y(1:k)+alpha*y(k:-1:1);
    y(1:k+1)=(z(1:k)', alpha)'
end

```

Man zählt leicht ab, dass dieser Algorithmus $2n^2$ flops benötigt. Mit einer kleinen Erweiterung kann man auch das allgemeine Gleichungssystem $\mathbf{T}_n \mathbf{y} = \mathbf{b}$ lösen. Wir nehmen wieder an, dass die Lösung $\mathbf{y}^{(k)}$ des Yule–Walker Systems $\mathbf{T}_k \mathbf{y} = -\mathbf{t}^{(k)}$ und die Lösung $\mathbf{x}^{(k)}$ von

$$\mathbf{T}_k \mathbf{x} = \mathbf{b}^{(k)} := (b_1, \dots, b_k)^T$$

für ein $k \in \{1, \dots, n-1\}$ bekannt sind. Wir führen hierauf die Lösung von

$$\mathbf{T}_{k+1} \mathbf{x}^{(k+1)} = \begin{pmatrix} \mathbf{T}_k & \mathbf{J}_k \mathbf{t}^{(k)} \\ (\mathbf{t}^{(k)})^T \mathbf{J}_k & 1 \end{pmatrix} \begin{pmatrix} \mathbf{v}^{(k)} \\ \mu_k \end{pmatrix} = \mathbf{b}^{(k+1)}$$

zurück. Die erste Zeile liefert

$$\mathbf{v}^{(k)} = \mathbf{T}_k^{-1}(\mathbf{b}^{(k)} - \mu_k \mathbf{J}_k \mathbf{t}^{(k)}) = \mathbf{x}^{(k)} - \mu_k \mathbf{T}_k^{-1} \mathbf{J}_k \mathbf{t}^{(k)} = \mathbf{x}^{(k)} + \mu_k \mathbf{J}_k \mathbf{y}^{(k)},$$

und damit erhält man aus der zweiten Zeile

$$\begin{aligned} \mu_k &= b_{k+1} - (\mathbf{t}^{(k)})^T \mathbf{J}_k \mathbf{v}^{(k)} \\ &= b_{k+1} - (\mathbf{t}^{(k)})^T \mathbf{J}_k \mathbf{x}^{(k)} - \mu_k (\mathbf{t}^{(k)})^T \mathbf{y}^{(k)}, \end{aligned}$$

also

$$\mu_k = (b_{k+1} - (\mathbf{t}^{(k)})^T \mathbf{J}_k \mathbf{x}^{(k)}) / (1 + (\mathbf{t}^{(k)})^T \mathbf{y}^{(k)}).$$

Berechnet man die Lösungen $\mathbf{x}^{(k)}$ des allgemeinen Systems und $\mathbf{y}^{(k)}$ des Yule–Walker Systems gemeinsam, so erhält man den folgenden **Algorithmus von Levinson** (vgl. [72])

Algorithmus 4.46. (Levinson)

$y(1)=-t(1)$; $x(1)=b(1)$; $\beta = 1$; $\alpha=-t(1)$;

for $k=1$: $n-1$

$\beta = (1 - \alpha^2)\beta$;

$\mu=(b(k+1)-t(1:k)'*x(k:-1:1))/\beta$;

$v(1:k)=x(1:k)+\mu y(k:-1:1)$;

$x(1:k+1)=(v(1:k)', \mu)'$;

if $k < n-1$

$\alpha=-(t(k+1)+t(k:-1:1)'*y(1:k))/\beta$;

$z(1:k)=y(1:k)+\alpha y(k:-1:1)$;

$y(1:k+1)=(z(1:k)', \alpha)'$

end

end

Bemerkung 4.47. Mit dem Algorithmus von Levinson (manchmal auch: Algorithmus von Levinson-Durbin) erhält man die Lösung des Gleichungssystems mit einer Toeplitz Matrix mit $4n^2$ flops.

Es ist klar, dass man eine ähnliche Rekursion auch für den nichtsymmetrischen Fall herleiten kann. Es sind dann aber die Untermatrizen \mathbf{T}_k nicht mehr automatisch regulär, sondern dies muss für die Durchführbarkeit der Methode gefordert werden.

□

Bemerkung 4.48. Algorithmen wie der von Levinson und Durbin, mit denen man lineare Gleichungssysteme mit $O(n^2)$ Operationen lösen kann, heißen schnelle Algorithmen. Für positiv definite Toeplitz Matrizen wurde von Ammar und Gragg [3] ein superschneller Algorithmus konstruiert, der ein lineares System mit nur $O(n \log_2^2 n)$ Operationen löst. \square

Wir stellen nun noch einen Zusammenhang zwischen Toeplitz Matrizen und der schnellen Fourier Transformation her. Eine wichtige Klasse von Toeplitz Matrizen sind zirkulante Matrizen. Es sei

$$\mathbf{S}_n := (\mathbf{e}^2, \mathbf{e}^3, \dots, \mathbf{e}^n, \mathbf{e}^1)$$

und $\mathbf{v} := (v_0, v_1, \dots, v_{n-1})^T$. Dann heißt

$$\begin{aligned} \mathbf{C}(\mathbf{v}) &:= (\mathbf{v}, \mathbf{S}_n \mathbf{v}, \mathbf{S}_n^2 \mathbf{v}, \dots, \mathbf{S}_n^{n-1} \mathbf{v}) \\ &= \begin{pmatrix} v_0 & v_{n-1} & v_{n-2} & \dots & v_2 & v_1 \\ v_1 & v_0 & v_{n-1} & \dots & v_3 & v_2 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ v_{n-1} & v_{n-2} & v_{n-3} & \dots & v_1 & v_0 \end{pmatrix} \end{aligned}$$

eine **zirkulante Matrix**. Wir bezeichnen mit \mathbf{F}_n die Matrix der diskreten Fourier Transformation. Damit kann man zeigen, dass

$$\mathbf{C}(\mathbf{v}) = \mathbf{F}_n^{-1} \text{diag}(\mathbf{F}_n \mathbf{v}) \mathbf{F}_n$$

gilt. Man kann also das Matrix-Vektor-Produkt $\mathbf{y} = \mathbf{C}(\mathbf{v})\mathbf{x}$ mit Hilfe der FFT schnell ausführen gemäß

$$\tilde{\mathbf{x}} := \mathbf{F}_n \mathbf{x}, \quad \tilde{\mathbf{v}} := \mathbf{F}_n \mathbf{v}, \quad \mathbf{z} := \tilde{\mathbf{v}} \cdot * \tilde{\mathbf{x}}, \quad \mathbf{y} = \mathbf{F}_n^{-1} \mathbf{z}.$$

Das Produkt einer Toeplitz Matrix mit einem Vektor kann man auf diese Methode zurückführen. Man kann nämlich die Toeplitz Matrix

$$\mathbf{T}_n = \begin{pmatrix} t_0 & t_1 & t_2 & t_3 & \dots & t_{n-2} & t_{n-1} \\ s_1 & t_0 & t_1 & t_2 & \dots & t_{n-3} & t_{n-2} \\ s_2 & s_1 & t_0 & t_1 & \dots & t_{n-4} & t_{n-3} \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots & \\ s_{n-1} & s_{n-2} & s_{n-3} & s_{n-4} & \dots & s_1 & t_0 \end{pmatrix}$$

zu einer zirkulanten Matrix erweitern durch

$$C = \left(\begin{array}{cccccc|cccc} t_0 & t_1 & t_2 & t_3 & \dots & t_{n-2} & t_{n-1} & s_{n-1} & s_{n-2} & \dots & s_2 & s_1 \\ s_1 & t_0 & t_1 & t_2 & \dots & t_{n-3} & t_{n-2} & t_{n-1} & s_{n-1} & \dots & s_3 & s_2 \\ s_2 & s_1 & t_0 & t_1 & \dots & t_{n-4} & t_{n-3} & t_{n-2} & t_{n-1} & \dots & s_4 & s_3 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ s_{n-1} & s_{n-2} & s_{n-3} & s_{n-4} & \dots & s_1 & t_0 & t_1 & t_2 & \dots & t_{n-2} & t_{n-1} \\ \hline t_{n-1} & s_{n-1} & s_{n-2} & s_{n-3} & \dots & s_2 & s_1 & t_0 & t_1 & \dots & t_{n-3} & t_{n-2} \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ t_1 & t_2 & t_3 & t_4 & \dots & t_{n-1} & s_{n-1} & s_{n-2} & s_{n-3} & \dots & s_1 & t_0 \end{array} \right)$$

Erweitert man nun $\mathbf{x} \in \mathbb{R}^n$ durch $x(n+1 : 2n-1) = 0$ zu einem Vektor $\tilde{\mathbf{x}} \in \mathbb{R}^{2n-1}$ und gilt $\tilde{\mathbf{y}} = C\tilde{\mathbf{x}}$, so gilt offensichtlich $T\mathbf{x} = \mathbf{y}$ für $\mathbf{y} = \tilde{\mathbf{y}}(1 : n)$.

Man kann daher auch für Toeplitz Matrizen Matrix-Vektorprodukte mit 3 Anwendungen der FFT und einer komponentenweisen Multiplikationen von Vektoren bilden. Da die Anwendung einer FFT $O(n \log_2 n)$ Operationen benötigt, kann man auch das Produkt einer Toeplitz Matrix mit einem Vektor mit $O(n \log_2 n)$ Operationen berechnen.

4.6 Software für Lineare Gleichungssysteme

Sehr hochwertige Public Domain Software ist in den Bibliotheken **LAPACK** und **ScaLAPACK** erhältlich unter der Adresse

<http://www.netlib.org/lapack/>

bzw.

<http://www.netlib.org/scalapack/>

Die Fortran 77 Bibliothek LAPACK (und die Übertragungen in andere Sprachen: die C-Version CLAPACK, die C++ Version LAPACK++ und die Fortran 90 Version LAPACK90, die ebenfalls in der Netlib frei erhältlich sind,) ist für PCs, Workstations, Vektorrechner oder Parallelrechner mit gemeinsamen Speicher geeignet, ScaLAPACK für Parallelrechner mit verteiltem Speicher oder vernetzte Workstations. Beide Bibliotheken wurden unter Benutzung von BLAS3 geschrieben. Der Quellcode

ist frei zugänglich und sehr gut dokumentiert. Die kommerziellen Bibliotheken ISML oder NAG verwenden (zum Teil geringfügig modifizierte) LAPACK Routinen.

Eine kommentierte Übersicht über Public Domain Software der linearen Algebra findet sich auf der WWW Seite

<http://www.tu-harburg.de/mat/sommer/PUBLICATIONS/Softw.html>

des AB Mathematik, eine nicht kommentierte Liste unter

<http://www.netlib.org/utk/people/JackDongarra/la-sw.html>

Kapitel 5

Lineare Ausgleichsprobleme

Wir betrachten in diesem Abschnitt das **lineare Ausgleichsproblem**

$$\|\mathbf{Ax} - \mathbf{b}\|_2 = \min! \quad (5.1)$$

mit gegebenem $\mathbf{A} \in \mathbb{R}^{(m,n)}$, $m \geq n$, und $\mathbf{b} \in \mathbb{R}^m$. Dieses Problem wurde bereits in Kapitel 7 der Vorlesung Lineare Algebra behandelt. Wir werden in Abschnitt 5.1 und Abschnitt 5.2 vor allem die Ergebnisse dieser Vorlesung zusammentragen. In Abschnitt 5.3 werden wir die Singulärwertzerlegung einer Matrix betrachten, in Abschnitt 5.4 werden wir die Pseudoinverse einer Matrix einführen und in Abschnitt 5.5 auf das Konditionsproblem für Lineare Gleichungssysteme und Ausgleichsprobleme eingehen. Schließlich werden wir in Abschnitt 5.6 kurz auf das Problem der Regularisierung schlecht gestellter Probleme eingehen.

5.1 Normalgleichungen

Notwendig für eine Lösung des Ausgleichsproblems (5.1) ist, dass der Gradient des Funktionals

$$\phi(\mathbf{x}) = (\mathbf{Ax} - \mathbf{b})^T (\mathbf{Ax} - \mathbf{b})$$

verschwindet, d.h. $2\mathbf{A}^T(\mathbf{Ax} - \mathbf{b}) = \mathbf{0}$ oder

$$\mathbf{A}^T \mathbf{Ax} = \mathbf{A}^T \mathbf{b}. \quad (5.2)$$

Die Gleichungen (5.2) heißen die **Normalgleichungen** des Ausgleichsproblems (5.1), denn geometrisch besagen sie, dass der Defekt $\mathbf{r} := \mathbf{Ax} - \mathbf{b}$ für die Lösung \mathbf{x} senkrecht auf dem Bild $\{\mathbf{Ay} : \mathbf{y} \in \mathbb{R}^n\}$ von \mathbf{A} steht.

Dass jede Lösung von (5.2) auch das Ausgleichsproblem (5.1) löst, sieht man so: Es ist für alle $\mathbf{h} \in \mathbb{R}^n$

$$\begin{aligned}\|\mathbf{A}(\mathbf{x} + \mathbf{h}) - \mathbf{b}\|_2^2 &= (\mathbf{Ax} + \mathbf{Ah} - \mathbf{b})^T (\mathbf{Ax} + \mathbf{Ah} - \mathbf{b}) \\ &= \|\mathbf{Ax} - \mathbf{b}\|_2^2 + \|\mathbf{Ah}\|_2^2 + 2\mathbf{h}^T (\mathbf{A}^T \mathbf{Ax} - \mathbf{A}^T \mathbf{b}) \\ &= \|\mathbf{Ax} - \mathbf{b}\|_2^2 + \|\mathbf{Ah}\|_2^2 \geq \|\mathbf{Ax} - \mathbf{b}\|_2^2.\end{aligned}$$

Schließlich ist die Matrix $\mathbf{A}^T \mathbf{A}$ genau dann regulär, wenn die Matrix \mathbf{A} den Rang n besitzt. Damit ist das Ausgleichsproblem genau dann eindeutig lösbar, wenn $\text{Rang}(\mathbf{A}) = n$ gilt. Wir haben also

Satz 5.1. *Die Lösungen des Ausgleichsproblems*

$$\|\mathbf{Ax} - \mathbf{b}\|_2 = \min!$$

sind genau die Lösungen der Normalgleichungen

$$\mathbf{A}^T \mathbf{Ax} = \mathbf{A}^T \mathbf{b}.$$

(5.1) ist genau dann eindeutig lösbar, wenn \mathbf{A} linear unabhängige Spalten besitzt.

Besitzt \mathbf{A} unabhängige Spalten, so ist die Koeffizientenmatrix $\mathbf{A}^T \mathbf{A}$ positiv definit, und man kann daher die Normalgleichungen unter Benutzung der Cholesky Zerlegung lösen. Diese Methode erfordert zur Aufstellung der Normalgleichungen (unter Beachtung der Symmetrie von $\mathbf{A}^T \mathbf{A}$) $\frac{1}{2}n(n+1)+n$ innere Produkte von Vektoren der Länge m , also $n^2m + 3nm$ flops und zur Lösung der Normalgleichungen $\frac{1}{3}n^3 + O(n^2)$ flops.

Das Verfahren ist geeignet bei Problemen mit kleinem n . Bei größerem n können Stabilitätsprobleme auftreten und eines der folgenden Verfahren ist vorzuziehen.

5.2 Orthogonale Zerlegung von Matrizen

Ist $\mathbf{A} \in \mathbb{R}^{(m,n)}$, $m \geq n$, eine gegebene Matrix mit linear unabhängigen Spalten, so gibt es eine Matrix $\mathbf{Q} \in \mathbb{R}^{(m,n)}$ mit orthogonalen Spalten und eine obere Dreiecksmatrix $\mathbf{R} \in \mathbb{R}^{(n,n)}$, so dass gilt

$$\mathbf{A} = \mathbf{QR}. \quad (5.3)$$

(5.3) heißt eine **QR Zerlegung** der Matrix \mathbf{A} .

Wir haben bereits in der Vorlesung Lineare Algebra gesehen, dass man die QR Zerlegung einer Matrix mit Hilfe der Orthogonalisierung nach Gram–Schmidt oder durch Multiplikation mit geeigneten Householder Transformationen bestimmen kann.

Beim Gram–Schmidt Verfahren werden im j -ten Schritt, d.h. nachdem die ersten $j - 1$ Spalten $\mathbf{q}^1, \dots, \mathbf{q}^{j-1}$ von \mathbf{Q} bereits bestimmt wurden, von der j -ten Spalte \mathbf{a}^j von \mathbf{A} die Anteile in Richtung von \mathbf{q}^i , $i = 1, \dots, j - 1$, abgezogen und der so bestimmte, zu den \mathbf{q}^i orthogonale Vektor normiert. Dabei werden zugleich die Elemente $r_{ij} := (\mathbf{q}^i)^T \mathbf{a}^j$ bestimmt.

Algorithmus 5.2. (Klassische Gram–Schmidt Orthogonalisierung)

```

for i=1 : n
  q(:,i)=a(:,i);
  for j=1 : i-1
    r(j,i)=q(:,j)'\*a(:,i);
    q(:,i)=q(:,i)-r(j,i)*q(:,j);
  end
  r(i,i)=||q(:,i)||2;
  q(:,i)=q(:,i)/r(i,i);
end

```

Wir haben vorausgesetzt, dass die Spalten von \mathbf{A} linear unabhängig sind. Ist dies nicht der Fall, so wird Algorithmus 5.2. mit $r_{ii} = 0$ für ein i abbrechen. Diese Abfrage wird man natürlich noch in einen Algorithmus einbauen.

Sind die Spalten von \mathbf{A} nahezu linear abhängig, so ist das oben angegebene **klassische Gram–Schmidt Verfahren** numerisch instabil. Die berechneten Vektoren \mathbf{q}^j sind also nicht orthogonal. Etwas besser wird die Stabilität, wenn man die Berechnung der r_{ij} ersetzt durch $r_{ij} = (\mathbf{q}^j)^T \mathbf{q}^i$. Diese dem klassischen Gram–Schmidt Verfahren äquivalente Methode heißt **modifiziertes Gram–Schmidt Verfahren**.

Algorithmus 5.3. (Modifizierte Gram–Schmidt Orthogonalisierung)

```

for i=1 : n
  q(:,i)=a(:,i);
  for j=1 : i-1
    r(j,i)=q(:,j)'\*q(:,i);
    q(:,i)=q(:,i)-r(j,i)*q(:,j);
  end
end

```

```

r(i, i) = ||q(:, i)||2;
q(:, i) = q(:, i) / r(i, i);
end

```

Beispiel 5.4. Das folgende Beispiel von Björck zeigt die Überlegenheit des modifizierten Gram–Schmidt Verfahrens. Sei

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

wobei ε so klein ist, dass $\text{fl}(1 + \varepsilon^2) = 1$ gilt. Dann erhält man mit dem klassischen und dem modifizierten Gram–Schmidt Verfahren (bis auf Normierung der Spalten) die Matrizen

$$Q_{kGS} = \begin{pmatrix} 1 & 0 & 0 \\ \varepsilon & -\varepsilon & -\varepsilon \\ 0 & \varepsilon & 0 \\ 0 & 0 & \varepsilon \end{pmatrix} \quad \text{und} \quad Q_{mGS} = \begin{pmatrix} 1 & 0 & 0 \\ \varepsilon & -\varepsilon & -\varepsilon/2 \\ 0 & \varepsilon & -\varepsilon/2 \\ 0 & 0 & \varepsilon \end{pmatrix}.$$

Für die minimalen Winkel φ_{kGS} und φ_{mGS} zwischen zwei Spalten gilt

$$\cos \varphi_{kGS} = \frac{1}{2} \quad \text{und} \quad \cos \varphi_{mGS} = -\frac{\varepsilon}{\sqrt{2}}. \quad \square$$

Stabiler als das Gram–Schmidt Verfahren sind Methoden zur Bestimmung der QR Zerlegung einer Matrix, die auf der Multiplikation von \mathbf{A} mit geeigneten orthogonalen Matrizen beruhen. Wir betrachten zunächst die Orthogonalisierung mit Hilfe von Householder Matrizen.

Definition 5.5. Es sei $\mathbf{w} \in \mathbb{R}^n$ mit $\|\mathbf{w}\|_2 = 1$. Dann heißt die Matrix

$$\mathbf{H} := \mathbf{I} - 2\mathbf{w}\mathbf{w}^T$$

Householder Matrix.

Die Householder Matrix \mathbf{H} beschreibt eine Spiegelung an der Hyperebene \mathbf{w}^\perp .

Offensichtlich ist \mathbf{H} eine symmetrische und orthogonale Matrix, d.h. $\mathbf{H}^T = \mathbf{H}$ und $\mathbf{H}^T \mathbf{H} = \mathbf{H}^2 = \mathbf{I}$.

Die explizite Matrixgestalt von \mathbf{H} wird außerordentlich selten benötigt. $\mathbf{H} = \mathbf{I} - \beta \mathbf{u} \mathbf{u}^T$ ist vollständig charakterisiert durch einen Normalenvektor \mathbf{u} der Hyperebene, an der gespiegelt wird, und durch den Skalierungsfaktor $\beta = 2/\|\mathbf{u}\|_2^2$. Sind diese beiden bekannt, so kann man eine Multiplikation eines Vektors \mathbf{x} mit \mathbf{H} ausführen gemäß

$$\mathbf{H}\mathbf{x} = \mathbf{x} - \beta(\mathbf{u}^T \mathbf{x})\mathbf{u}.$$

Es sind also ein inneres Produkt und ein „axpy“, d.h. $4n$ flops, erforderlich. Wegen $\mathbf{H}^{-1} = \mathbf{H}^T = \mathbf{H}$ gilt dasselbe für das Lösen eines Gleichungssystems mit der Koeffizientenmatrix \mathbf{H} .

Um die Orthogonalisierung einer Matrix mit Householder Transformationen auszuführen, benötigen wir zu gegebenem Vektor $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{x} \neq \mathbf{0}$, eine Householder Matrix \mathbf{H} mit

$$\mathbf{H}\mathbf{x} = \pm \|\mathbf{x}\|_2 \mathbf{e}^1, \quad \mathbf{e}^1 = (1, 0, \dots, 0)^T.$$

In der Vorlesung Lineare Algebra wurde gezeigt (und dies rechnet man auch leicht nach), dass für den Fall, dass \mathbf{x} kein Vielfaches von \mathbf{e}^1 ist, die Householder Matrizen

$$\mathbf{H}_{\pm} = \mathbf{I} - \beta \mathbf{w}_{\pm} \mathbf{w}_{\pm}^T,$$

die durch die Normalenvektoren

$$\mathbf{w}_{\pm} = \mathbf{x} \pm \|\mathbf{x}\|_2 \mathbf{e}^1,$$

bestimmt sind, das Gewünschte leisten. Ist \mathbf{x} ein Vielfaches des ersten Einheitsvektors \mathbf{e}^1 , so ist einer der beiden Vektoren der Nullvektor, und zwar gilt $\tilde{\mathbf{w}}_- = \mathbf{0}$ im Fall $x_1 > 0$ und $\tilde{\mathbf{w}}_+ = \mathbf{0}$ im Fall $x_1 < 0$. Ist $\mathbf{x} \approx c\mathbf{e}^1$, so erhält man für $\tilde{\mathbf{w}}_-$ im Fall $c > 0$ und für $\tilde{\mathbf{w}}_+$ im Fall $c < 0$ Auslöschung. Um diese zu vermeiden, wählen wir daher stets

$$\mathbf{H} = \mathbf{I} - \frac{2}{\|\tilde{\mathbf{w}}\|_2^2} \tilde{\mathbf{w}} \tilde{\mathbf{w}}^T, \quad \tilde{\mathbf{w}} = \mathbf{x} + \text{sign}(x_1) \|\mathbf{x}\|_2 \mathbf{e}^1.$$

Damit erhalten wir den folgenden Algorithmus zur Berechnung der Matrix $\mathbf{H} = \mathbf{I} - \beta \mathbf{u} \mathbf{u}^T$, die \mathbf{x} in $\text{span}(\mathbf{e}^1)$ abbildet:

Algorithmus 5.6. (Householder Vektor)

```
function [u, beta]=householder(x)
    mu=x(2:n)'*x(2:n);
    u=[1; x(2:n)];
    if mu == 0
```

```

    beta=0;
else
    u(1)=x(1)+sign(x(1))*sqrt(x(1)^2 + mu);
    beta=2/(u(1)^2+mu);
end

```

Wir verwenden nun Householder Matrizen, um die Matrix \mathbf{A} orthogonal auf obere Dreiecksgestalt zu transformieren. Sei dazu \mathbf{H}_1 wie oben beschrieben gewählt, so dass

$$\mathbf{H}_1 \mathbf{a}^1 = \pm \|\mathbf{a}^1\|_2 \mathbf{e}^1 =: r_{11} \mathbf{e}^1.$$

Dann gilt mit $\mathbf{P}_1 = \mathbf{H}_1$ (und mit einer Matrix $\mathbf{A}_1 \in \mathbb{R}^{(m-1, n-1)}$)

$$\mathbf{P}_1 \mathbf{A} = \begin{pmatrix} r_{11} & r(1, 2:n) \\ \mathbf{0} & \mathbf{A}_1 \end{pmatrix}.$$

Ist nach j Schritten, $1 \leq j < n$, die Gestalt

$$\left(\begin{array}{cccc|cccc} r_{11} & r_{12} & \dots & r_{1j} & r_{1,j+1} & r_{1,j+2} & \dots & r_{1n} \\ 0 & r_{22} & \dots & r_{2j} & r_{2,j+1} & r_{2,j+2} & \dots & r_{2n} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & r_{jj} & r_{j,j+1} & r_{j,j+2} & \dots & r_{j,n} \\ \hline 0 & 0 & \dots & 0 & r_{j+1,j+1} & r_{j+1,j+2} & \dots & r_{j+1,n} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & 0 & r_{m,j+1} & r_{m,j+2} & \dots & r_{m,n} \end{array} \right)$$

erreicht, so wählen wir eine Householder Matrix $\mathbf{H}_{j+1} \in \mathbb{R}^{(m-j, n-j)}$, so dass $\mathbf{H}_{j+1} r(j+1 : m, j+1)$ eine Vielfaches des ersten Einheitsvektors ist, und setzen hiermit

$$\mathbf{P}_{j+1} = \begin{pmatrix} \mathbf{I}_j & \mathbf{O}_{j, n-j} \\ \mathbf{O}_{m-j, j} & \mathbf{H}_{j+1} \end{pmatrix}.$$

Dann gilt

$$\mathbf{P}_{j+1} \mathbf{P}_j \dots \mathbf{P}_1 \mathbf{A} = \left(\begin{array}{cccc|cccc} r_{11} & r_{12} & \dots & r_{1j} & r_{1,j+1} & r_{1,j+2} & \dots & r_{1n} \\ 0 & r_{22} & \dots & r_{2j} & r_{2,j+1} & r_{2,j+2} & \dots & r_{2n} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & r_{jj} & r_{j,j+1} & r_{j,j+2} & \dots & r_{j,n} \\ 0 & 0 & \dots & 0 & r_{j+1,j+1} & r_{j+1,j+2} & \dots & r_{j+1,n} \\ \hline 0 & 0 & \dots & 0 & 0 & r_{j+2,j+2} & \dots & r_{j+2,n} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & 0 & 0 & r_{m,j+2} & \dots & r_{m,n} \end{array} \right).$$

Nach n Schritten ist damit die obere Dreiecksgestalt erreicht (wobei wir vorausgesetzt haben, dass die Matrix \mathbf{A} linear unabhängige Spalten besitzt, da sonst das Verfahren vorher abgebrochen wäre).

Setzt man

$$\mathbf{Q}^T := \mathbf{P}_n \mathbf{P}_{n-1} \dots \mathbf{P}_1,$$

so ist \mathbf{Q} eine orthogonale Matrix, und es gilt

$$\mathbf{A} = \mathbf{Q} \begin{pmatrix} \mathbf{R} \\ \mathbf{O}_{m-n,n} \end{pmatrix} = \mathbf{P}_1 \mathbf{P}_2 \dots \mathbf{P}_n \begin{pmatrix} \mathbf{R} \\ \mathbf{O}_{m-n,n} \end{pmatrix}.$$

Die Householder Matrizen \mathbf{H}_j (und damit die Faktoren \mathbf{P}_j in \mathbf{Q}) sind jeweils durch Vektoren \mathbf{u}^j und Skalare β_j vollständig bestimmt. Die \mathbf{u}^j können (bis auf die erste Komponente) im Verlauf eines Algorithmus unterhalb der Diagonalen von \mathbf{A} gespeichert werden. Tatsächlich müssen diese Elemente wegen der Gestalt der \mathbf{u}^j nicht einmal geändert werden. Die ersten Komponenten von \mathbf{u}^j und die β_j müssen in einem zusätzlichen Array gespeichert werden. Die Elemente von \mathbf{R} können das obere Dreieck von \mathbf{A} einschließlich der Diagonale überschreiben.

Algorithmus 5.7. (QR Zerlegung mit Householder Matrizen)

```

for i=1 : min(m-1,n)
    [u,beta]=householder(a(i:m,i));
    a(i:m,i+1:n)=a(i:m,i+1:n)-beta*u*(u'*a(i:m,i+1:n));
    a(i,i)=a(i,i)-beta*u(1)*u'*a(i:m,i);
    uu(i)=u(1);
    be(i)=beta;
end

```

Man zählt leicht ab, dass dieser Algorithmus im wesentlichen $2n^2m - \frac{2}{3}n^3$ flops benötigt für die QR Zerlegung von \mathbf{A} .

Eine weitere Möglichkeit zur Bestimmung der QR Zerlegung bieten die Givens Rotationen. Es ist bekannt, dass eine Rotation im \mathbb{R}^2 um den Winkel θ gegeben ist durch

$$\mathbf{x} \mapsto \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix} \mathbf{x}.$$

Entsprechend kann man eine Multiplikation eines Vektors $\mathbf{x} \in \mathbb{R}^n$ mit der Matrix

$$\mathbf{R}(i, j, \theta) = \begin{pmatrix} \mathbf{I}_{i-1} & \mathbf{0} & \mathbf{O} & \mathbf{0} & \mathbf{O} \\ \mathbf{0}^T & \cos \theta & \mathbf{0}^T & -\sin \theta & \mathbf{0}^T \\ \mathbf{O} & \mathbf{0} & \mathbf{I}_{j-i-1} & \mathbf{0} & \mathbf{O} \\ \mathbf{0}^T & \sin \theta & \mathbf{0}^T & \cos \theta & \mathbf{0}^T \\ \mathbf{O} & \mathbf{0} & \mathbf{O} & \mathbf{0} & \mathbf{I}_{n-j} \end{pmatrix}$$

als Rotation in der von \mathbf{e}^i und \mathbf{e}^j aufgespannten Ebene auffassen. $\mathbf{R}(i, j, \theta)$ heißt eine **Givens Rotation** oder seltener auch **Jacobi Rotation**. Diese Abbildungen wurden 1958 von Givens [47] benutzt, um eine Matrix orthogonal auf obere Dreiecksgestalt zu transformieren. Jacobi [58] verwandte diese Abbildung schon im vorletzten Jahrhundert zur Lösung des vollständigen symmetrischen Eigenwertproblems (vgl. Abschnitt 7.6).

Es sei nun für $\mathbf{x} \in \mathbb{R}^n$ die j -te Komponente x_j von Null verschieden. Wir wählen θ so, dass

$$\cos \theta = \frac{x_i}{\sqrt{x_i^2 + x_j^2}} \quad \text{und} \quad \sin \theta = \frac{-x_j}{\sqrt{x_i^2 + x_j^2}}.$$

Dann gilt

$$\begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix} \begin{pmatrix} x_i \\ x_j \end{pmatrix} = \begin{pmatrix} \sqrt{x_i^2 + x_j^2} \\ 0 \end{pmatrix},$$

und daher wird in $\mathbf{R}(i, j, \theta)\mathbf{x}$ die j -te Komponente annulliert. Damit ist klar, wie man mit einer Folge von Multiplikationen mit Givens Rotationen eine QR Zerlegung einer Matrix $\mathbf{A} \in \mathbb{R}^{(m,n)}$ bestimmen kann.

Man annulliert nacheinander die Elemente der ersten Spalte unterhalb der Diagonale mit geeigneten Rotationen $\mathbf{R}(1, 2, \theta_{12})$, $\mathbf{R}(1, 3, \theta_{13})$, \dots , $\mathbf{R}(1, m, \theta_{1m})$, und dann mit Rotationen $\mathbf{R}(i, j, \theta_{ij})$, $i = 2, \dots, n$, $j = i + 1, \dots, m$ die Elemente der weiteren Spalten von links nach rechts und von oben nach unten.

Genauso kann man mit Rotationen $\mathbf{R}(m-1, m, \theta_{m-1,m})$, \dots , $\mathbf{R}(1, 2, \theta_{12})$ die Elemente der ersten Spalte unterhalb der Diagonale von unten nach oben annullieren, und dann die weiteren Spalten von links nach rechts auf gleiche Weise behandeln.

Analog kann man auch die sog. **Givens Reflexionen** verwenden, die ausgehend von der Spiegelung

$$\mathbf{x} \mapsto \begin{pmatrix} \cos \theta & \sin \theta \\ \sin \theta & -\cos \theta \end{pmatrix} \mathbf{x}$$

im \mathbb{R}^2 wie oben definiert werden.

Ein Vorteil der Verwendung von Givens Rotationen oder Reflexionen gegenüber den Householder Transformationen zur QR Zerlegung einer Matrix \mathbf{A} besteht darin, dass man auf übersichtliche Weise gewisse Nullen in \mathbf{A} berücksichtigen und damit die Arbeit vermindern kann.

Ein Nachteil ist, dass die direkte Implementierung des beschriebenen Verfahrens doppelt so teuer ist wie die QR Zerlegung mit Householder Matrizen. Man beachte aber, dass Gentleman [44] und Hammarling [53] eine Methode, die **schnelle Givens Transformation**, angegeben haben, mit der der Aufwand genauso hoch ist, wie mit Householder Matrizen. Diese wurde in der BLAS1 verwendet.

Um eine Givens Transformation auf eine Matrix anzuwenden, muss eine Multiplikation \mathbf{QA} ausgeführt werden mit

$$\mathbf{Q} = \begin{pmatrix} c & -s \\ s & c \end{pmatrix}, \quad \mathbf{A} = \begin{pmatrix} a_1 & a_2 & \dots & a_n \\ b_1 & b_2 & \dots & b_n \end{pmatrix}$$

Die Zahl der Multiplikationen bei der Berechnung von \mathbf{QA} kann dadurch halbiert werden, dass die Matrix \mathbf{A} in skaliertes Form $\mathbf{A} = \mathbf{D}\tilde{\mathbf{A}}$ geschrieben wird mit einer Diagonalmatrix $\mathbf{D} = \text{diag}\{d_1, d_2\}$ und die beiden Matrizen $\tilde{\mathbf{A}}$ und \mathbf{D} getrennt aufdatiert werden:

$$\mathbf{QA} = \mathbf{QD}\tilde{\mathbf{A}} = \tilde{\mathbf{D}}\tilde{\mathbf{Q}}\tilde{\mathbf{A}}.$$

Dabei wird $\tilde{\mathbf{D}}$ so gewählt, dass zwei Elemente der Matrix $\tilde{\mathbf{Q}}$ den Wert 1 haben. Hierdurch werden $2n$ Multiplikationen eingespart.

In der tatsächlichen Umsetzung dieser Idee wird \mathbf{D}^2 an Stelle von \mathbf{D} gespeichert, da hierdurch Quadratwurzeln vermieden werden können. Sei zunächst $|c| \geq |s|$. Dann setzen wir

$$\mathbf{QD} = \begin{pmatrix} d_1c & -d_2s \\ d_1s & d_2c \end{pmatrix} = c\mathbf{D} \begin{pmatrix} 1 & -\frac{d_2s}{d_1c} \\ \frac{d_1s}{d_2c} & 1 \end{pmatrix} =: \tilde{\mathbf{D}}\tilde{\mathbf{Q}}.$$

Um das Element b_1 zu annullieren wählen wir

$$\frac{s}{c} = -\frac{b_1}{a_1} = -\frac{d_2\tilde{b}_1}{d_1\tilde{a}_1},$$

und hiermit ist

$$\tilde{\mathbf{Q}} = \begin{pmatrix} 1 & \tilde{q}_{12} \\ \tilde{q}_{21} & 1 \end{pmatrix}, \quad \tilde{q}_{21} = -\frac{\tilde{b}_1}{\tilde{a}_1}, \quad \tilde{q}_{12} = -\left(\frac{d_2}{d_1}\right)^2 \tilde{q}_{21}.$$

Man benötigt also nur die Quadrate der Skalierungsfaktoren d_1 und d_2 , und diese können wegen $c^2 = (1 + s^2/c^2)^{-1}$ aufdatiert werden gemäß

$$\tilde{d}_1^2 = \frac{d_1^2}{t}, \quad \tilde{d}_2^2 = \frac{d_2^2}{t}, \quad t = 1 + \frac{s^2}{c^2} = 1 + \tilde{q}_{12}\tilde{q}_{21}.$$

Für den Fall $|c| < |s|$ schreiben wir

$$\mathbf{QD} = \begin{pmatrix} d_1c & -d_2s \\ d_1s & d_2c \end{pmatrix} = s \begin{pmatrix} d_2 & 0 \\ 0 & d_1 \end{pmatrix} \begin{pmatrix} \frac{d_1c}{d_2s} & -1 \\ 1 & \frac{d_2c}{d_1s} \end{pmatrix} = \tilde{\mathbf{D}}\tilde{\mathbf{Q}}.$$

Dann gilt

$$\tilde{\mathbf{Q}} = \begin{pmatrix} \tilde{q}_{11} & -1 \\ 1 & \tilde{q}_{22} \end{pmatrix}, \quad \tilde{q}_{22} = -\frac{\tilde{a}_1}{\tilde{b}_1}, \quad \tilde{q}_{11} = \left(\frac{d_1}{d_2}\right)^2 \tilde{q}_{22}$$

und

$$\tilde{d}_2^2 = \frac{d_2^2}{t}, \quad \tilde{d}_1^2 = \frac{d_1^2}{t}, \quad t = 1 + \frac{c^2}{s^2} = 1 + \tilde{q}_{11}\tilde{q}_{22}.$$

Verwendet man diese Art der Givens Transformationen, bei denen die Quadrate der Skalierungsfaktoren jeweils mit einem Faktor zwischen 1 und 0.5 multipliziert werden, so besteht nach einer großen Zahl von Transformationen die Gefahr des Exponentenunterlaufs. Es muss daher die Größe der Skalierungsfaktoren kontrolliert werden und hin und wieder eine Reskalierung vorgenommen werden. Dies reduziert die Effizienz. Abhängig vom benutzten Rechner wird die Effizienz gegenüber der klassischen Givens Transformation um Faktoren zwischen 1.2 und 1.6 gesteigert.

Ist eine QR Zerlegung

$$\mathbf{A} = \mathbf{Q} \begin{pmatrix} \mathbf{R} \\ \mathbf{O}_{m-n,n} \end{pmatrix}$$

der Matrix $\mathbf{A} \in \mathbb{R}^{(m,n)}$ bekannt, so ist es leicht, das lineare Ausgleichsproblem

$$\|\mathbf{Ax} - \mathbf{b}\|_2 = \min!$$

zu lösen. Da die Multiplikation eines Vektors mit einer orthogonalen Matrix seine Euklidische Länge nicht verändert, gilt

$$\|\mathbf{Ax} - \mathbf{b}\|_2 = \|\mathbf{Q}^T(\mathbf{Ax} - \mathbf{b})\|_2 = \left\| \begin{pmatrix} \mathbf{R} \\ \mathbf{O}_{m-n,n} \end{pmatrix} \mathbf{x} - \mathbf{Q}^T\mathbf{b} \right\|_2$$

Mit

$$\mathbf{Q}^T\mathbf{b} =: \begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{pmatrix}, \quad \mathbf{b}_1 \in \mathbb{R}^n, \quad \mathbf{b}_2 \in \mathbb{R}^{m-n}$$

folgt

$$\|\mathbf{Ax} - \mathbf{b}\|_2^2 = \|\mathbf{Rx} - \mathbf{b}_1\|_2^2 + \|\mathbf{b}_2\|_2^2.$$

Den zweiten Summanden kann man durch die Wahl von \mathbf{x} nicht beeinflussen. Daher ist die Lösung des Ausgleichsproblems

$$\mathbf{x} = \mathbf{R}^{-1}\mathbf{b}_1,$$

und der Defekt ist $\|\mathbf{b}_2\|$.

5.3 Singulärwertzerlegung

Eine für viele Anwendungen wichtige Zerlegung einer Matrix ist die Singulärwertzerlegung.

Definition 5.8. Sei $\mathbf{A} \in \mathbb{R}^{(m,n)}$. Gilt für orthogonale Matrizen $\mathbf{U} \in \mathbb{R}^{(m,m)}$ und $\mathbf{V} \in \mathbb{R}^{(n,n)}$ und eine Diagonalmatrix $\mathbf{\Sigma} = (\sigma_i \delta_{ij})_{i,j} \in \mathbb{R}^{(m,n)}$

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T, \quad (5.4)$$

so heißt (5.4) eine **Singulärwertzerlegung (SVD¹)** von \mathbf{A} .

Beispiel 5.9. Ist $\mathbf{A} \in \mathbb{R}^{(n,n)}$ symmetrisch mit den Eigenwerten μ_1, \dots, μ_n und ist $\mathbf{v}^1, \dots, \mathbf{v}^n$ ein zugehöriges System von orthonormalen Eigenvektoren, so gilt mit der Diagonalmatrix $\mathbf{\Sigma} := \text{diag}\{\mu_1, \dots, \mu_n\}$ und mit $\mathbf{V} := (\mathbf{v}^1, \dots, \mathbf{v}^n)$

$$\mathbf{A} = \mathbf{V}\mathbf{\Sigma}\mathbf{V}^T,$$

und dies ist eine Singulärwertzerlegung von \mathbf{A} . □

Setzt man $\mathbf{U} = (\mathbf{u}^1, \dots, \mathbf{u}^m)$ und $\mathbf{V} = (\mathbf{v}^1, \dots, \mathbf{v}^n)$, so ist (5.4) äquivalent zu

$$\mathbf{A}\mathbf{v}^i = \begin{cases} \sigma_i \mathbf{u}^i, & i = 1, \dots, \min(m, n), \\ \mathbf{0} & i = m + 1, \dots, n \text{ (falls } n > m). \end{cases} \quad (5.5)$$

Ohne Einschränkung können die σ_i nichtnegativ gewählt werden. Ist etwa $\sigma_j < 0$, so ersetze man die j -te Spalte \mathbf{v}^j von \mathbf{V} durch $-\mathbf{v}^j$. Ferner können wir durch Umordnung von Zeilen von \mathbf{V}^T bzw. Spalten von \mathbf{U} erreichen, dass $\sigma_1 \geq \sigma_2 \geq \dots$ gilt. Wir werden nun nur noch Singulärwertzerlegungen betrachten, in denen die Diagonalelemente $\sigma_1, \dots, \sigma_{\min(m,n)}$ nichtnegativ und der Größe nach geordnet sind.

¹**SVD** ist die Abkürzung der englischen Bezeichnung *Singular Value Decomposition*. Da diese Abkürzung meistens auch in der deutschen Literatur verwendet wird, schließen wir uns dieser Sprachregelung an.

Definition 5.10. *Es seien in einer Singulärwertzerlegung (5.4) von $\mathbf{A} \in \mathbb{R}^{(m,n)}$ die Diagonalelemente $\sigma_1, \dots, \sigma_{\min(m,n)}$ von Σ nicht negativ. Dann heißen die σ_i die **Singulärwerte** oder die **singulären Werte** von \mathbf{A} .*

Beispiel 5.11. (Fortsetzung von Beispiel 5.9.)

Es sei $\mathbf{A} \in \mathbb{R}^{(n,n)}$ symmetrisch und μ_i und \mathbf{v}^i wie in Beispiel 5.9., wobei die Reihenfolge so gewählt ist, dass $|\mu_1| \geq |\mu_2| \geq \dots \geq |\mu_r| > \mu_{r+1} = \dots = \mu_n = 0$ gilt.

Es sei $\mathbf{u}^i := \mathbf{v}^i$, falls $\mu_i \geq 0$, und $\mathbf{u}^i := -\mathbf{v}^i$, falls $\mu_i < 0$, und hiermit $\mathbf{U} := (\mathbf{u}^1, \dots, \mathbf{u}^n)$. Dann gilt

$$\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T, \quad \Sigma = (|\mu_i|\delta_{ij}),$$

und $|\mu_1|, \dots, |\mu_r|$ sind die singulären Werte von \mathbf{A} . □

Aus (5.4) folgt

$$\mathbf{A}^T = \mathbf{V}\Sigma^T\mathbf{U}^T; \tag{5.6}$$

besitzt also \mathbf{A} eine Singulärwertzerlegung, so auch \mathbf{A}^T , und die von Null verschiedenen singulären Werte stimmen überein. (5.6) kann man (entsprechend (5.5)) schreiben als

$$\mathbf{A}^T\mathbf{u}^i = \begin{cases} \sigma_i\mathbf{v}^i, & i=1, \dots, \min(m, n), \\ \mathbf{0} & i=n+1, \dots, m \text{ (falls } m > n). \end{cases}$$

Aus (5.4) und (5.6) folgt

$$\begin{aligned} \mathbf{A}^T\mathbf{A} &= \mathbf{V}\Sigma^T\Sigma\mathbf{V}^T, \\ \mathbf{A}\mathbf{A}^T &= \mathbf{U}\Sigma\Sigma^T\mathbf{U}^T, \end{aligned} \tag{5.7}$$

d.h. die Quadrate der singulären Werte von \mathbf{A} (und \mathbf{A}^T) sind Eigenwerte von $\mathbf{A}^T\mathbf{A}$ und $\mathbf{A}\mathbf{A}^T$, und $\{\mathbf{v}^1, \dots, \mathbf{v}^n\}$ bzw. $\{\mathbf{u}^1, \dots, \mathbf{u}^m\}$ ist ein Orthonormalsystem von Eigenvektoren von $\mathbf{A}^T\mathbf{A}$ bzw. $\mathbf{A}\mathbf{A}^T$.

Dies zeigt, dass die singulären Werte σ_i eindeutig bestimmt sind, nicht aber die Matrizen \mathbf{U} und \mathbf{V} , da mehrfache Eigenwerte von $\mathbf{A}\mathbf{A}^T$ und $\mathbf{A}^T\mathbf{A}$ auftreten können.

Satz 5.12. *Jedes $\mathbf{A} \in \mathbb{R}^{(m,n)}$ besitzt eine Singulärwertzerlegung.*

Beweis: Da mit \mathbf{A} auch \mathbf{A}^T eine Singulärwertzerlegung besitzt, können wir $m \geq n$ voraussetzen.

Wir zeigen die Behauptung durch vollständige Induktion über m und n . Wir nehmen also an, dass jede $(m-1, n-1)$ -Matrix eine Singulärwertzerlegung besitzt, und zeigen, dass dies dann auch für jede (m, n) -Matrix gilt. Wir nehmen an, dass $\mathbf{A} \neq \mathbf{O}$, denn sonst können wir \mathbf{U} und \mathbf{V} als beliebige orthogonale Matrizen wählen und $\mathbf{\Sigma} = \mathbf{O}$.

Für $\mathbf{A} \in \mathbb{R}^{(m,1)}$, $m \in \mathbb{N}$, können wir $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ schreiben, wobei $\mathbf{U} \in \mathbb{R}^{(m,m)}$ eine orthogonale Matrix ist mit der ersten Spalte $\mathbf{A}/\|\mathbf{A}\|_2$, $\mathbf{\Sigma} = \|\mathbf{A}\|_2 \mathbf{e}^1$ und $\mathbf{V} = (1) \in \mathbb{R}^{(1,1)}$.

Um den Induktionsschritt auszuführen, wählen wir $\mathbf{v} \in \mathbb{R}^n$ mit $\|\mathbf{v}\|_2 = 1$ und $\|\mathbf{A}\|_2 = \|\mathbf{A}\mathbf{v}\|_2 =: \sigma > 0$. Ein solches \mathbf{v} existiert, denn es gilt nach Definition $\|\mathbf{A}\|_2 = \max_{\|\mathbf{v}\|_2=1} \|\mathbf{A}\mathbf{v}\|_2$. Es sei $\mathbf{u} := \mathbf{A}\mathbf{v}/\|\mathbf{A}\mathbf{v}\|_2$.

Wir ergänzen die Vektoren \mathbf{v} und \mathbf{u} zu orthogonalen Matrizen $\mathbf{V} = (\mathbf{v}, \mathbf{V}_1) \in \mathbb{R}^{(n,n)}$, $\mathbf{U} = (\mathbf{u}, \mathbf{U}_1) \in \mathbb{R}^{(m,m)}$. Dann gilt

$$\tilde{\mathbf{A}} := \mathbf{U}^T \mathbf{A} \mathbf{V} = \begin{pmatrix} \mathbf{u}^T \\ \mathbf{U}_1^T \end{pmatrix} (\mathbf{A}\mathbf{v}, \mathbf{A}\mathbf{V}_1) = \begin{pmatrix} \mathbf{u}^T \\ \mathbf{U}_1^T \end{pmatrix} (\sigma \mathbf{u}, \mathbf{A}\mathbf{V}_1) = \begin{pmatrix} \sigma & \mathbf{w}^T \\ \mathbf{0} & \mathbf{A}_1 \end{pmatrix}$$

mit $\mathbf{w} := \mathbf{V}_1^T \mathbf{A}^T \mathbf{u}$ und $\mathbf{A}_1 = \mathbf{U}_1^T \mathbf{A} \mathbf{V}_1 \in \mathbb{R}^{(m-1, n-1)}$. Aus

$$\|\tilde{\mathbf{A}} \begin{pmatrix} \sigma \\ \mathbf{w} \end{pmatrix}\|_2^2 = \left\| \begin{pmatrix} \sigma^2 + \mathbf{w}^T \mathbf{w} \\ \mathbf{A}_1 \mathbf{w} \end{pmatrix} \right\|_2^2 \geq (\sigma^2 + \mathbf{w}^T \mathbf{w})^2$$

folgt $\|\tilde{\mathbf{A}}\|_2^2 \geq \sigma^2 + \mathbf{w}^T \mathbf{w}$. Andererseits ist

$$\sigma^2 = \|\mathbf{A}\|_2^2 = \rho(\mathbf{A}^T \mathbf{A}) = \rho(\mathbf{V} \tilde{\mathbf{A}}^T \mathbf{U}^T \mathbf{U} \tilde{\mathbf{A}} \mathbf{V}^T) = \rho(\tilde{\mathbf{A}}^T \tilde{\mathbf{A}}) = \|\tilde{\mathbf{A}}\|_2^2,$$

und daher folgt $\mathbf{w} = \mathbf{0}$, d.h.

$$\tilde{\mathbf{A}} = \begin{pmatrix} \sigma & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_1 \end{pmatrix}.$$

Der Rest des Beweises folgt nun durch einen trivialen Induktionsschluss. ■

Der folgende Satz 5.13. sagt, welche Bedeutungen die in der Singulärwertzerlegung von \mathbf{A} auftretenden Größen für die Matrix \mathbf{A} haben:

Satz 5.13. *Ist die Singulärwertzerlegung von \mathbf{A} durch (5.4) gegeben und gilt $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_r > \sigma_{r+1} = \dots = \sigma_{\min(m,n)} = 0$, so ist*

(i) r der Rang von \mathbf{A} ,

(ii) $\text{Kern}(\mathbf{A}) := \{\mathbf{x} \in \mathbb{R}^n : \mathbf{A}\mathbf{x} = \mathbf{0}\} = \text{span}\{\mathbf{v}^{r+1}, \dots, \mathbf{v}^n\},$

(iii) $\text{Bild}(\mathbf{A}) := \{\mathbf{A}\mathbf{x} : \mathbf{x} \in \mathbb{R}^n\} = \text{span}\{\mathbf{u}^1, \dots, \mathbf{u}^r\},$

(iv)

$$\mathbf{A} = \sum_{i=1}^r \sigma_i \mathbf{u}^i (\mathbf{v}^i)^T = \mathbf{U}_r \mathbf{\Sigma}_r \mathbf{V}_r^T$$

mit $\mathbf{U}_r = (\mathbf{u}^1, \dots, \mathbf{u}^r), \quad \mathbf{V}_r = (\mathbf{v}^1, \dots, \mathbf{v}^r), \quad \mathbf{\Sigma}_r = \text{diag}(\sigma_1, \dots, \sigma_r),$

(v)

$$\|\mathbf{A}\|_S^2 := \sum_{i=1}^m \sum_{j=1}^n a_{ij}^2 = \sum_{i=1}^r \sigma_i^2,$$

(vi)

$$\|\mathbf{A}\|_2 := \sigma_1.$$

Beweis: (i): Da die Multiplikation mit den regulären Matrizen \mathbf{U}^T und \mathbf{V} den Rang nicht verändert, gilt $\text{Rang } \mathbf{A} = \text{Rang } \mathbf{\Sigma} = r.$

(ii): Es ist $\mathbf{V}^T \mathbf{v}^i = \mathbf{e}^i$. Somit ist

$$\mathbf{A}\mathbf{v}^i = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T \mathbf{v}^i = \mathbf{U}\mathbf{\Sigma}\mathbf{e}^i = \mathbf{0} \text{ für } i = r+1, \dots, n.$$

Also gilt

$$\mathbf{v}^{r+1}, \dots, \mathbf{v}^n \in \text{Kern}(\mathbf{A}).$$

Da $\dim \text{Kern}(\mathbf{A}) = n - r$, bilden diese Vektoren eine Basis von $\text{Kern}(\mathbf{A})$.

(iii): Wegen $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ ist

$$\text{Bild}(\mathbf{A}) = \mathbf{U} \cdot \text{Bild}(\mathbf{\Sigma}) = \mathbf{U} \cdot \text{span}(\mathbf{e}^1, \dots, \mathbf{e}^r) = \text{span}(\mathbf{u}^1, \dots, \mathbf{u}^r).$$

(iv): Durch Block-Matrix-Multiplikation erhält man

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \begin{pmatrix} \mathbf{u}^1 & \dots & \mathbf{u}^m \end{pmatrix} \mathbf{\Sigma} \begin{pmatrix} (\mathbf{v}^1)^T \\ \vdots \\ (\mathbf{v}^n)^T \end{pmatrix} = \sum_{i=1}^r \sigma_i \mathbf{u}^i (\mathbf{v}^i)^T.$$

(v): Es sei $\mathbf{A} = (\mathbf{a}^1, \dots, \mathbf{a}^n)$. Da die orthogonale Matrix \mathbf{U}^T die Euklidische Länge nicht verändert, gilt

$$\|\mathbf{A}\|_S^2 = \sum_{i=1}^n \|\mathbf{a}^i\|_2^2 = \sum_{i=1}^n \|\mathbf{U}^T \mathbf{a}^i\|_2^2 = \|\mathbf{U}^T \mathbf{A}\|_S^2.$$

Durch entsprechende Argumentation mit den Zeilen von $\mathbf{U}^T \mathbf{A}$ erhält man

$$\|\mathbf{A}\|_S^2 = \|\mathbf{U}^T \mathbf{A}\mathbf{V}\|_S^2 = \|\mathbf{\Sigma}\|_S^2 = \sum_{i=1}^r \sigma_i^2.$$

(vi): Wegen des Beweises von Satz 5.12. ist $\|\mathbf{A}\|_2$ ein singulärer Wert von \mathbf{A} , d.h. $\|\mathbf{A}\|_2 \leq \sigma_1$, und

$$\|\mathbf{A}\|_2 = \max\{\|\mathbf{Ax}\|_2 : \|\mathbf{x}\|_2 = 1\} \geq \|\mathbf{Av}^1\|_2 = \sigma_1.$$

■

Bemerkung 5.14. Besitzt die reguläre Matrix \mathbf{A} die Singulärwertzerlegung $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$, so gilt $\|\mathbf{A}\|_2 = \sigma_1$, und wegen $\mathbf{A}^{-1} = \mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^T$ ist $\|\mathbf{A}^{-1}\|_2 = \frac{1}{\sigma_n}$. Daher ist die Kondition von \mathbf{A} bzgl. der Euklidischen Norm

$$\kappa_2(\mathbf{A}) := \frac{\sigma_1}{\sigma_n}.$$

□

Bemerkung 5.15. Ist $\mathbf{A} \in \mathbb{R}^{(n,n)}$ mit den Eigenwerten μ_1, \dots, μ_n , so folgt aus $\mathbf{Ax}^i = \mu_i \mathbf{x}^i$ die Gleichung

$$|\mu_i|^2 = \frac{(\mathbf{Ax}^i)^H \mathbf{Ax}^i}{(\mathbf{x}^i)^H \mathbf{x}^i} = \frac{(\mathbf{x}^i)^H \mathbf{A}^T \mathbf{Ax}^i}{(\mathbf{x}^i)^H \mathbf{x}^i}.$$

Das Rayleighsche Prinzip besagt

$$\lambda_{\min} \leq \frac{\mathbf{x}^H \mathbf{A}^T \mathbf{Ax}}{\mathbf{x}^H \mathbf{x}} \leq \lambda_{\max} \quad \text{für alle } \mathbf{x} \in \mathbb{C}^n, \mathbf{x} \neq \mathbf{0},$$

wobei λ_{\min} und λ_{\max} den minimalen und maximalen Eigenwert von $\mathbf{A}^T \mathbf{A}$ bezeichnen. Aus (5.7) folgt also $\sigma_1 \geq |\mu_i| \geq \sigma_n$ für alle i .

Ist \mathbf{A} symmetrisch, so gilt nach Beispiel 5.11. $\sigma_1 = |\mu_1|$ und $\sigma_r = |\mu_r|$. Für nicht symmetrische Matrizen ist dies i.A. nicht der Fall. □

Beispiel 5.16. Es sei $\mathbf{A} = \begin{pmatrix} 1 & 4 \\ 1 & 1 \end{pmatrix}$. Dann gilt $\mu_1 = 3$ und $\mu_2 = -1$.

Die singulären Werte von \mathbf{A} sind die Quadratwurzeln der Eigenwerte von $\mathbf{B} := \mathbf{AA}^T = \begin{pmatrix} 17 & 5 \\ 5 & 2 \end{pmatrix}$, d.h. $\sigma_1 = 4.3018 > \mu_1$ und $\sigma_2 = 0.6972 < |\mu_2|$. □

Bemerkung 5.17. Die Aussage (iv) in Satz 5.13. kann man wegen Beispiel 5.9. als Verallgemeinerung der Spektralzerlegung einer reellen symmetrischen Matrix betrachten. □

Bemerkung 5.18. Im Prinzip kann man mit Hilfe von (5.7) die Singulärwertzerlegung von \mathbf{A} berechnen (wenn man Eigenwertaufgaben numerisch lösen kann). Dazu hat man $\mathbf{A}^T \mathbf{A}$ und $\mathbf{A} \mathbf{A}^T$ zu berechnen. Dies ist zum einen sehr aufwendig, zum anderen kann — wie wir später sehen werden — die Kondition drastisch verschlechtert und damit die numerische Behandlung erheblich erschwert werden.

In der Praxis verwendet man einen Algorithmus von Golub und Reinsch (1971), der auf dem sogenannten QR Algorithmus zur Bestimmung der Eigenwerte von $\mathbf{A}^T \mathbf{A}$ beruht, aber die Berechnung von $\mathbf{A}^T \mathbf{A}$ bzw. $\mathbf{A} \mathbf{A}^T$ vermeidet. Wir kommen hierauf zurück. \square

Die Singulärwertzerlegung kann zur Datenkompression verwendet werden. Hintergrund liefert

Satz 5.19. *Es sei $\mathbf{A} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T$ die Singulärwertzerlegung, $\mathbf{U} = (\mathbf{u}^1, \dots, \mathbf{u}^m)$ und $\mathbf{V} = (\mathbf{v}^1, \dots, \mathbf{v}^n)$. Dann ist für $k < n$*

$$\mathbf{A}_k := \sum_{j=1}^k \sigma_j \mathbf{u}^j (\mathbf{v}^j)^T$$

eine beste Approximierende mit Rang k von \mathbf{A} im Sinne der Spektralnorm, und es gilt

$$\|\mathbf{A} - \mathbf{A}_k\|_2 = \sigma_{k+1}.$$

Beweis: Es gilt

$$\begin{aligned} \|\mathbf{A} - \mathbf{A}_k\|_2 &= \left\| \sum_{j=k+1}^n \sigma_j \mathbf{u}^j (\mathbf{v}^j)^T \right\|_2 \\ &= \|\mathbf{U} \text{diag}\{0, \dots, 0, \sigma_{k+1}, \dots, \sigma_n\} \mathbf{V}^T\|_2 = \sigma_{k+1}. \end{aligned}$$

Wir haben nur noch zu zeigen, dass es keine Matrix mit Rang k gibt, deren Abstand zu \mathbf{A} kleiner als σ_{k+1} ist.

Sei \mathbf{B} eine Matrix vom Rang k . Dann hat der Nullraum von \mathbf{B} die Dimension $n - k$. Der Raum $\text{span}\{\mathbf{v}^1, \dots, \mathbf{v}^{k+1}\}$ hat die Dimension $k + 1$. Nach der Dimensionsformel liegen im Durchschnitt dieser Räume nichttriviale Vektoren. Es liege \mathbf{w} im Durchschnitt, und es gelte $\|\mathbf{w}\|_2 = 1$. Dann gilt

$$\begin{aligned} \|\mathbf{A} - \mathbf{B}\|_2^2 &\geq \|(\mathbf{A} - \mathbf{B})\mathbf{w}\|_2^2 = \|\mathbf{A}\mathbf{w}\|_2^2 \\ &= \|\mathbf{U} \mathbf{\Sigma} \mathbf{V}^T \mathbf{w}\|_2^2 = \|\mathbf{\Sigma} (\mathbf{V}^T \mathbf{w})\|_2^2 \\ &\geq \sigma_{k+1}^2 \|\mathbf{V}^T \mathbf{w}\|_2^2 = \sigma_{k+1}^2. \quad \blacksquare \end{aligned}$$

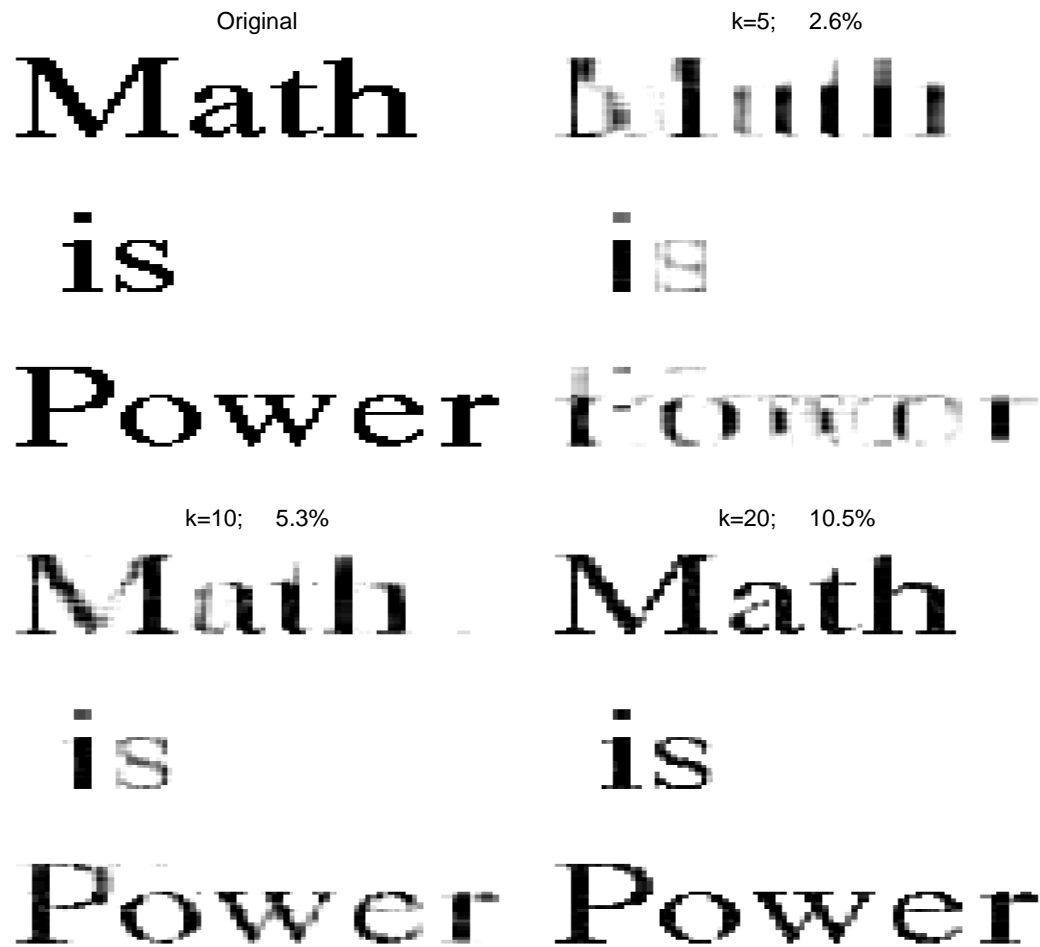


Abbildung 5.1: Datenkompression

Es sei nun $\mathbf{A} \in \mathbb{R}^{(m,n)}$ eine Matrix in die für a_{ij} ein Grauwert eingetragen ist (in MATLAB $0 \leq a_{ij} \leq 100$). Ist dann $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ die Singulärwertzerlegung von \mathbf{A} , so erhält man durch

$$\mathbf{A}_k = \sum_{j=1}^k \sigma_j \mathbf{u}^j (\mathbf{v}^j)^T, \quad k = 1, \dots, \min(n, m)$$

Approximationen für \mathbf{A} . Für die Speicherung oder die Übertragung dieser Daten benötigt man nur noch den Anteil $k * (n + m + 1) / (n * m)$ der Originaldaten.

Abbildung 5.1 enthält ein Original, das mit einer Matrix $\mathbf{A} \in \mathbb{R}^{(431,326)}$ dargestellt wurde, und die Approximationen mit $k = 5$, $k = 10$ und $k = 20$, also mit 2.6%, 5.3% und 10.5% der Originaldaten.

Wir weisen darauf hin, dass es für die Datenkompression in der Bildverarbeitung spezielle Algorithmen gibt, die wesentlich billiger sind als die Singulärwertzerlegung.

5.4 Pseudoinverse

Wir betrachten erneut das lineare Ausgleichsproblem

Es seien $\mathbf{A} \in \mathbb{R}^{(m,n)}$ und $\mathbf{b} \in \mathbb{R}^m$ gegeben mit $m \geq n$. Man bestimme $\mathbf{x} \in \mathbb{R}^n$ mit

$$\|\mathbf{Ax} - \mathbf{b}\|_2 = \min! \quad (5.8)$$

und untersuchen dieses nun mit Hilfe der Singulärwertzerlegung.

Wir bezeichnen in diesem ganzen Abschnitt mit $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_r > \sigma_{r+1} = \dots = \sigma_n = 0$ die singulären Werte von \mathbf{A} , mit $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ eine Singulärwertzerlegung von \mathbf{A} und mit \mathbf{u}^j bzw. \mathbf{v}^k die Spalten von \mathbf{U} bzw. \mathbf{V} .

Satz 5.20. *Es sei $\mathbf{c} := \mathbf{U}^T\mathbf{b} \in \mathbb{R}^m$. Dann ist die Lösungsmenge des linearen Ausgleichsproblems (5.8) gegeben durch*

$$L = \bar{\mathbf{x}} + \text{Kern}(\mathbf{A}), \quad (5.9)$$

wobei $\bar{\mathbf{x}}$ die folgende spezielle Lösung von (5.8) bezeichnet:

$$\bar{\mathbf{x}} := \sum_{i=1}^r \frac{c_i}{\sigma_i} \mathbf{v}^i. \quad (5.10)$$

Beweis: Da die Multiplikation mit einer orthogonalen Matrix die Euklidische Länge nicht ändert, gilt mit $\mathbf{z} := \mathbf{V}^T\mathbf{x}$

$$\begin{aligned} \|\mathbf{Ax} - \mathbf{b}\|_2^2 &= \|\mathbf{U}^T(\mathbf{Ax} - \mathbf{b})\|_2^2 = \|\mathbf{\Sigma}\mathbf{V}^T\mathbf{x} - \mathbf{U}^T\mathbf{b}\|_2^2 \\ &= \|\mathbf{\Sigma}\mathbf{z} - \mathbf{c}\|_2^2 = \|(\sigma_1 z_1 - c_1, \dots, \sigma_r z_r - c_r, -c_{r+1}, \dots, -c_m)^T\|_2^2. \end{aligned}$$

Wie in Abschnitt 5.2 liest man hieraus die Lösung von (5.8) sofort ab: $z_i := \frac{c_i}{\sigma_i}$, $i = 1, \dots, r$, und $z_i \in \mathbb{R}$ beliebig für $i = r+1, \dots, n$, d.h.

$$\mathbf{x} = \sum_{i=1}^r \frac{c_i}{\sigma_i} \mathbf{v}^i + \sum_{i=r+1}^n z_i \mathbf{v}^i, \quad z_i \in \mathbb{R}, \quad i = r+1, \dots, n. \quad (5.11)$$

Da die letzten $n - r$ Spalten von \mathbf{V} nach Satz 5.13. den Kern von \mathbf{A} aufspannen, kann man die Lösungsmenge L von (5.8) schreiben als (5.9), (5.10). ■

In Satz 5.1. wurde gezeigt (und das liest man auch aus Satz 5.20. ab), dass die Lösung des Ausgleichsproblems (5.8) genau dann eindeutig ist, wenn $r = \text{Rang } \mathbf{A} = n$ gilt. Wir erzwingen nun auch im Fall $r < n$ die Eindeutigkeit, indem wir zusätzlich fordern, dass die Euklidische Norm der Lösung möglichst klein werden soll.

Definition 5.21. Es sei L die Lösungsmenge des Ausgleichsproblems (5.8). $\tilde{\mathbf{x}} \in L$ heißt **Pseudonormallösung** von (5.8), falls

$$\|\tilde{\mathbf{x}}\|_2 \leq \|\mathbf{x}\|_2 \quad \text{für alle } \mathbf{x} \in L.$$

Aus der Darstellung (5.11) der allgemeinen Lösung von (5.8) liest man ab, dass $\bar{\mathbf{x}}$ aus (5.10) Pseudonormallösung von (5.8) ist, denn

$$\left\| \bar{\mathbf{x}} + \sum_{i=r+1}^n z_i \mathbf{v}^i \right\|_2^2 = \|\bar{\mathbf{x}}\|_2^2 + \sum_{i=r+1}^n |z_i|^2 \cdot \|\mathbf{v}^i\|_2^2 \geq \|\bar{\mathbf{x}}\|_2^2.$$

Ferner ist die Pseudonormallösung eindeutig bestimmt, und $\bar{\mathbf{x}}$ ist offensichtlich die einzige Lösung von (5.8) mit $\mathbf{x} \in \text{Kern}(\mathbf{A})^\perp \cap L$. Daher gilt

Satz 5.22. Es gibt genau eine Pseudonormallösung $\bar{\mathbf{x}}$ von (5.8). Diese ist charakterisiert durch $\bar{\mathbf{x}} \in \text{Kern}(\mathbf{A})^\perp \cap L$.

Für jedes $\mathbf{A} \in \mathbb{R}^{(m,n)}$ ist durch

$$\mathbb{R}^m \ni \mathbf{b} \mapsto \bar{\mathbf{x}} \in \mathbb{R}^n : \|\mathbf{A}\bar{\mathbf{x}} - \mathbf{b}\|_2 \leq \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2 \quad \forall \mathbf{x} \in \mathbb{R}^n, \|\bar{\mathbf{x}}\|_2 \text{ minimal}$$

eine Abbildung erklärt. Diese ist offensichtlich linear (vgl. die Darstellung von $\bar{\mathbf{x}}$ in (5.10)), kann also durch eine Matrix $\mathbf{A}^\dagger \in \mathbb{R}^{(n,m)}$ dargestellt werden.

Definition 5.23. Es sei $\mathbf{A} \in \mathbb{R}^{(m,n)}$. Dann heißt die Matrix $\mathbf{A}^\dagger \in \mathbb{R}^{(n,m)}$, für die durch $\bar{\mathbf{x}} := \mathbf{A}^\dagger \mathbf{b}$ für alle $\mathbf{b} \in \mathbb{R}^m$ die Pseudonormallösung des Ausgleichsproblems (5.8) gegeben ist, die **Pseudoinverse** (oder **Moore-Penrose Inverse**) von \mathbf{A} .

Bemerkung 5.24. Die Pseudoinverse wurde von Penrose auf ganz andere Weise eingeführt und wird in der Literatur bisweilen mit Hilfe der Moore-Penrose Bedingungen definiert. Wir werden die Äquivalenz der Definitionen in Satz 5.30. zeigen. \square

Bemerkung 5.25. Ist $\text{Rang } \mathbf{A} = n$, so ist das Ausgleichsproblem (5.8) eindeutig lösbar, und aus den Normalgleichungen folgt, dass die Lösung $\bar{\mathbf{x}} = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{b}$ ist. In diesem Fall ist also $\mathbf{A}^\dagger = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T$.

Ist noch spezieller $n = m$ und \mathbf{A} regulär, so ist $\mathbf{A}^\dagger = \mathbf{A}^{-1}$.

Die Pseudo-Inverse wird also zur "normalen Inversen", wenn diese existiert, und ist somit eine konsistente Erweiterung dieses Begriffes. \square

Aus unserer Konstruktion ergibt sich sofort im allgemeinen Fall

Satz 5.26. Sei $\mathbf{A} \in \mathbb{R}^{(m,n)}$ mit der Singulärwertzerlegung

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T, \quad \mathbf{\Sigma} = (\sigma_i \delta_{ij})_{i,j}.$$

Dann gilt

$$(i) \quad \mathbf{\Sigma}^\dagger = (\tau_i \delta_{ij})_{j,i}, \quad \tau_i = \begin{cases} \sigma_i^{-1}, & \text{falls } \sigma_i \neq 0 \\ 0, & \text{falls } \sigma_i = 0 \end{cases},$$

$$(ii) \quad \mathbf{A}^\dagger = \mathbf{V}\mathbf{\Sigma}^\dagger\mathbf{U}^T.$$

Bemerkung 5.27. Die explizite Matrix-Darstellung der Pseudoinverse braucht man genauso häufig wie die der inversen Matrix, nämlich fast nie. \square

Aus der Darstellung der Pseudoinversen in Satz 5.26.(ii) folgt unmittelbar

Korollar 5.28. Für jede Matrix $\mathbf{A} \in \mathbb{R}^{(m,n)}$ gilt

$$\mathbf{A}^{\dagger\dagger} = \mathbf{A}$$

und

$$(\mathbf{A}^\dagger)^T = (\mathbf{A}^T)^\dagger.$$

\mathbf{A}^\dagger besitzt also die üblichen Eigenschaften der Inversen \mathbf{A}^{-1} für reguläres \mathbf{A} . Es gilt jedoch i.A.

$$(\mathbf{A}\mathbf{B})^\dagger \neq \mathbf{B}^\dagger\mathbf{A}^\dagger.$$

Beispiel 5.29. Es gilt

$$\mathbf{A} = \begin{pmatrix} 1 & -1 \\ 0 & 0 \end{pmatrix} = \mathbf{I} \begin{pmatrix} \sqrt{2} & 0 \\ 0 & 0 \end{pmatrix} \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix},$$

und daher besitzt \mathbf{A} die Pseudoinverse

$$\mathbf{A}^\dagger = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & 0 \\ 0 & 0 \end{pmatrix} \mathbf{I} = \frac{1}{2} \begin{pmatrix} 1 & 0 \\ -1 & 0 \end{pmatrix}.$$

Es ist $\mathbf{A}^2 = \mathbf{A}$ und $(\mathbf{A}^\dagger)^2 = \frac{1}{2}\mathbf{A}^\dagger$, d.h. $(\mathbf{A}^2)^\dagger \neq (\mathbf{A}^\dagger)^2$. \square

Wir zeigen nun die Äquivalenz unserer Definition der Pseudoinversen mit den Moore-Penrose-Bedingungen.

Satz 5.30. Sei $\mathbf{A} \in \mathbb{R}^{(m,n)}$.

(i) Es gibt genau eine Matrix $\mathbf{B} \in \mathbb{R}^{(n,m)}$ mit

$$\mathbf{AB} = (\mathbf{AB})^T, \quad \mathbf{BA} = (\mathbf{BA})^T, \quad \mathbf{ABA} = \mathbf{A}, \quad \mathbf{BAB} = \mathbf{B}, \quad (5.12)$$

(ii) \mathbf{A}^\dagger erfüllt (5.12).

Bemerkung 5.31. Die vier Forderungen in (5.12) heißen die **Moore-Penrose Bedingungen**. \square

Beweis: (ii): Wir zeigen zunächst (ii). Damit ist zugleich die Existenz einer Matrix gesichert, die den Moore-Penrose-Bedingungen genügt.

Es ist

$$(\mathbf{A}^\dagger \mathbf{A})^T = (\mathbf{V} \Sigma^\dagger \underbrace{\mathbf{U}^T \mathbf{U}}_{\mathbf{I}} \Sigma \mathbf{V}^T)^T = \mathbf{V} (\Sigma^\dagger \Sigma)^T \mathbf{V}^T = \mathbf{V} \Sigma^\dagger \Sigma \mathbf{V}^T = \mathbf{A}^\dagger \mathbf{A},$$

denn $\Sigma^\dagger \Sigma = \begin{pmatrix} \mathbf{E}_r & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{pmatrix} = (\Sigma^\dagger \Sigma)^T$, und

$$\mathbf{AA}^\dagger \mathbf{A} = \mathbf{U} \Sigma \mathbf{V}^T \mathbf{V} \Sigma^\dagger \mathbf{U}^T \mathbf{U} \Sigma \mathbf{V}^T = \mathbf{U} \Sigma \Sigma^\dagger \Sigma \mathbf{V}^T = \mathbf{U} \Sigma \mathbf{V}^T = \mathbf{A};$$

genauso zeigt man $(\mathbf{AA}^\dagger)^T = \mathbf{AA}^\dagger$ und $\mathbf{A}^\dagger \mathbf{AA}^\dagger = \mathbf{A}^\dagger$.

(i): Wir haben nur noch die Eindeutigkeit zu zeigen.

Erfüllt \mathbf{B} die Bedingung (5.12), so gilt für $\mathbf{P} := \mathbf{BA}$ und $\tilde{\mathbf{P}} := \mathbf{AB}$

$$\mathbf{P}^T = \mathbf{P}, \quad \mathbf{P}^2 = (\mathbf{BAB})\mathbf{A} = \mathbf{BA} = \mathbf{P}, \quad \text{und genauso } \tilde{\mathbf{P}}^T = \tilde{\mathbf{P}}, \quad \tilde{\mathbf{P}}^2 = \tilde{\mathbf{P}},$$

so dass \mathbf{P} und $\tilde{\mathbf{P}}$ orthogonale Projektionen beschreiben.

Für $\mathbf{x} \in \text{Kern}(\mathbf{P})$ gilt $\mathbf{Ax} = \mathbf{ABAx} = \mathbf{APx} = \mathbf{0}$, für $\mathbf{x} \in \text{Kern}(\mathbf{A})$ gilt $\mathbf{Px} = \mathbf{BAx} = \mathbf{0}$.

Daher folgt $\text{Kern}(\mathbf{A}) = \text{Kern}(\mathbf{P})$, d.h. \mathbf{P} ist die (von \mathbf{B} unabhängige) orthogonale Projektion von \mathbb{R}^n auf $\text{Kern}(\mathbf{A})^\perp$.

Analog gilt für $\mathbf{y} \in M := \{\mathbf{y} : \tilde{\mathbf{P}}\mathbf{y} = \mathbf{y}\}$ die Gleichung $\mathbf{AB}\mathbf{y} = \mathbf{y}$, d.h. $\mathbf{y} \in \text{Bild}(\mathbf{A})$. Andererseits folgt für $\mathbf{y} := \mathbf{Ax} \in \text{Bild}(\mathbf{A})$, dass $\tilde{\mathbf{P}}\mathbf{y} = \mathbf{ABAx} = \mathbf{Ax} = \mathbf{y}$, d.h. $\mathbf{y} \in M$. Es ist also $M = \text{Bild}(\mathbf{A})$, und auch $\tilde{\mathbf{P}}$ ist unabhängig von \mathbf{B} .

Sind \mathbf{B}_1 und \mathbf{B}_2 zwei Matrizen mit der Eigenschaft (5.12), so gilt $\mathbf{AB}_1 = \mathbf{AB}_2$ und $\mathbf{B}_1\mathbf{A} = \mathbf{B}_2\mathbf{A}$, und es folgt

$$\mathbf{B}_1 = \mathbf{B}_1\mathbf{AB}_1 = \mathbf{B}_2\mathbf{AB}_1 = \mathbf{B}_2\mathbf{AB}_2 = \mathbf{B}_2.$$

■

Beispiel 5.32. Es sei

$$\mathbf{A} = \begin{pmatrix} 1 & -1 \\ 0 & 0 \end{pmatrix} \quad \text{und} \quad \mathbf{C} = \frac{1}{2} \begin{pmatrix} 1 & 0 \\ -1 & 0 \end{pmatrix}.$$

Dann sind

$$\mathbf{AC} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \quad \text{und} \quad \mathbf{CA} = \frac{1}{2} \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

symmetrisch, und es gilt $\mathbf{CAC} = \mathbf{C}$ und $\mathbf{ACA} = \mathbf{A}$. Dies zeigt erneut, dass \mathbf{C} die Pseudoinverse von \mathbf{A} ist. \square

Bemerkung 5.33. Der Beweis von Satz 5.30. zeigt, dass $\mathbf{AA}^\dagger : \mathbb{R}^m \rightarrow \text{Bild}(\mathbf{A})$ und $\mathbf{A}^\dagger\mathbf{A} : \mathbb{R}^n \rightarrow \text{Kern}(\mathbf{A})^\perp = \text{Bild}(\mathbf{A}^T)$ Projektionsmatrizen sind. \square

Beispiel 5.34. Es sei $\mathbf{A} = \begin{pmatrix} 1 & -1 \\ 0 & 0 \end{pmatrix}$ wie in Beispiel 5.29. und Beispiel 5.32. Dann beschreibt

$$\mathbf{AA}^\dagger = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \quad \text{die orthogonale Projektion auf} \quad \text{Bild}(\mathbf{A}) = \text{span} \left\{ \begin{pmatrix} 1 \\ 0 \end{pmatrix} \right\}$$

und

$$\mathbf{A}^\dagger\mathbf{A} = \frac{1}{2} \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \quad \text{die orthogonale Projektion auf} \quad \text{Bild}(\mathbf{A}^T) = \text{span} \left\{ \begin{pmatrix} 1 \\ -1 \end{pmatrix} \right\}.$$

\square

5.5 Störung von Ausgleichsproblemen

Wir übertragen nun den Begriff der Kondition einer Matrix auf singuläre und allgemeiner auf nicht quadratische Matrizen. Es ist klar, dass für den quadratischen, nicht regulären Fall dieser verallgemeinerte Konditionsbegriff nicht mehr als Verstärkungsfaktor für die Störung linearer Systeme gedeutet werden kann. Er spielt aber eine ähnliche Rolle für lineare Ausgleichsprobleme.

Wir beschränken uns auf die Euklidische Norm.

Zur Motivation betrachten wir das lineare Ausgleichsproblem

$$\|\mathbf{Ax} - \mathbf{b}\|_2 = \min! \tag{5.13}$$

mit $\mathbf{A} \in \mathbb{R}^{(m,n)}$, $\text{Rang}(\mathbf{A}) = r$, und eine Störung hiervon

$$\|\mathbf{A}(\mathbf{x} + \Delta\mathbf{x}) - (\mathbf{b} + \Delta\mathbf{b})\|_2 = \min!, \tag{5.14}$$

wobei wir zunächst nur Störungen von \mathbf{b} , nicht aber der Koeffizientenmatrix \mathbf{A} zulassen.

Es sei $\mathbf{x} = \mathbf{A}^\dagger \mathbf{b}$ bzw. $\mathbf{x} + \Delta \mathbf{x} = \mathbf{A}^\dagger (\mathbf{b} + \Delta \mathbf{b})$ die Pseudonormallösung von (5.13) bzw. (5.14). Dann gilt $\Delta \mathbf{x} = \mathbf{A}^\dagger \Delta \mathbf{b}$, und aus $\|\mathbf{A}^\dagger\|_2 = \frac{1}{\sigma_r}$ folgt

$$\|\Delta \mathbf{x}\|_2 \leq \|\mathbf{A}^\dagger\|_2 \cdot \|\Delta \mathbf{b}\|_2 = \frac{1}{\sigma_r} \|\Delta \mathbf{b}\|_2.$$

Ferner gilt (vgl. (5.11)):

$$\|\mathbf{x}\|_2^2 = \sum_{i=1}^r \frac{c_i^2}{\sigma_i^2} \geq \frac{1}{\sigma_1^2} \sum_{i=1}^r c_i^2 = \frac{1}{\sigma_1^2} \left\| \sum_{i=1}^r (\mathbf{u}^i)^T \mathbf{b} \mathbf{u}^i \right\|_2^2.$$

Da offenbar $\sum_{i=1}^r (\mathbf{u}^i)^T \mathbf{b} \mathbf{u}^i$ die Projektion von \mathbf{b} auf den Bildbereich $\text{Bild}(\mathbf{A})$ von \mathbf{A} ist, folgt also für den relativen Fehler

$$\frac{\|\Delta \mathbf{x}\|_2}{\|\mathbf{x}\|_2} \leq \frac{\sigma_1}{\sigma_r} \cdot \frac{\|\Delta \mathbf{b}\|_2}{\|\mathbf{P}_{\text{Bild}(\mathbf{A})} \mathbf{b}\|_2}. \quad (5.15)$$

Diese Ungleichung beschreibt wieder, wie sich der relative Fehler der rechten Seite eines Ausgleichsproblems auf die Lösung auswirken kann. Wir definieren daher

Definition 5.35. $\mathbf{A} \in \mathbb{R}^{(m,n)}$ besitze die Singulärwertzerlegung $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$. Dann heißt $\kappa_2(\mathbf{A}) := \frac{\sigma_1}{\sigma_r}$ die **Kondition** der Matrix \mathbf{A} .

Bemerkung 5.36. Ist $\mathbf{A} \in \mathbb{R}^{(n,n)}$ regulär, so stimmt diese Definition der Kondition mit der vorher gegebenen (für die Euklidische Norm) überein. \square

Bemerkung 5.37. Wegen $\kappa_2(\mathbf{A}^T \mathbf{A}) = \kappa_2(\mathbf{A})^2$ und $\kappa_2(\mathbf{A}) > 1$ sind die Normalgleichungen eines linearen Ausgleichsproblems schlechter konditioniert als die Koeffizientenmatrix des Ausgleichsproblems. \square

Lässt man auch Störungen der Koeffizientenmatrix \mathbf{A} zu, so erhält man (vgl. Demmel [23] p. 117)

Satz 5.38. Die Matrix $\mathbf{A} \in \mathbb{R}^{(m,n)}$, $m \geq n$, besitze den vollen Rang n . Es sei \mathbf{x} die Lösung des Ausgleichsproblems (5.13) und $\tilde{\mathbf{x}}$ die Lösung des gestörten Problems

$$\|(\mathbf{A} + \Delta \mathbf{A})\mathbf{x} - (\mathbf{b} + \Delta \mathbf{b})\|_2 = \min!, \quad (5.16)$$

wobei

$$\varepsilon := \max \left(\frac{\|\Delta \mathbf{A}\|_2}{\|\mathbf{A}\|_2}, \frac{\|\Delta \mathbf{b}\|_2}{\|\mathbf{b}\|_2} \right) < \frac{1}{\kappa_2(\mathbf{A})} = \frac{\sigma_n(\mathbf{A})}{\sigma_1(\mathbf{A})}. \quad (5.17)$$

Dann gilt

$$\frac{\|\mathbf{x} - \tilde{\mathbf{x}}\|_2}{\|\mathbf{x}\|_2} \leq \varepsilon \left(\frac{2\kappa_2(\mathbf{A})}{\cos \theta} + \tan \theta \cdot \kappa_2^2(\mathbf{A}) \right) + O(\varepsilon^2), \quad (5.18)$$

wobei θ den Winkel zwischen \mathbf{b} und seiner Projektion auf den Raum $\text{Bild}(\mathbf{A})$ bezeichnet.

5.6 Regularisierung schlecht konditionierter Probleme

Einige Probleme der Anwendungen (z.B. in der Tomographie oder bei der Ausbreitung von Rissen in den Materialwissenschaften) führen auf lineare Gleichungssysteme oder Ausgleichsprobleme mit schlecht konditionierten Koeffizientenmatrizen. In diesen Fällen lassen die Störungsüberlegungen in Abschnitt 4.4 und Abschnitt 5.5 erwarten, dass die bisher betrachteten Verfahren zu schlechten oder gar unbrauchbaren Ergebnissen führen.

Beispiel 5.39. Das Problem, die orthogonale Projektion einer gegebenen Funktion $f : [0, 1] \rightarrow \mathbb{R}$ auf den Raum Π_{n-1} der Polynome vom Höchstgrad $n - 1$ bzgl. des inneren Produkts

$$\langle f, g \rangle := \int_0^1 f(x)g(x) dx$$

zu berechnen, führt bei der Wahl der Basis $\{1, x, \dots, x^{n-1}\}$ auf das lineare Gleichungssystem

$$\mathbf{A}\mathbf{y} = \mathbf{b} \quad (5.19)$$

mit der Matrix

$$\mathbf{A} = (a_{ij})_{i,j=1,\dots,n}, \quad a_{ij} := \frac{1}{i+j-1}, \quad (5.20)$$

der sogenannten **Hilbert Matrix**, und $\mathbf{b} \in \mathbb{R}^n$, $b_i := \langle f, x^{i-1} \rangle$.

Wir wählen für die Dimensionen $n = 10$, $n = 20$ und $n = 40$ die rechte Seite von (5.19) so, dass $\mathbf{y} = (1, \dots, 1)^T$ die eindeutige Lösung ist, und behandeln (5.19) mit den bekannten numerischen Verfahren. Unter MATLAB erhält man mit der LR-Zerlegung mit Spaltenpivotsuche (in MATLAB $\mathbf{A} \setminus \mathbf{b}$), mit dem Cholesky Verfahren, der QR Zerlegung der Matrix \mathbf{A} und der Singulärwertzerlegung von \mathbf{A} die folgenden Fehler in der Euklidischen Norm:

| | $n = 10$ | $n = 20$ | $n = 40$ |
|--------------|----------|------------------------|----------|
| LR Zerlegung | 5.24 E-4 | 8.25 E+1 | 3.78 E+2 |
| Cholesky | 7.15 E-4 | numer. nicht pos. def. | |
| QR Zerlegung | 1.41 E-3 | 1.67 E+2 | 1.46 E+3 |
| SVD | 8.24 E-4 | 3.26 E+2 | 8.35 E+2 |

Die Ergebnisse sind also (wenigstens für die Dimensionen $n = 20$ oder $n = 40$) unbrauchbar.

Ein ähnliches Verhalten zeigt sich bei dem Ausgleichsproblem. Wir betrachten für $n = 10$, $n = 20$ und $n = 40$ und $m = n + 10$ die Ausgleichsprobleme

$$\|\mathbf{A}\mathbf{x} - \mathbf{b}\|_2 = \min!$$

mit der Hilbert Matrix $\mathbf{A} \in \mathbb{R}^{(m,n)}$, wobei \mathbf{b} so gewählt ist, dass $\mathbf{x} = (1, \dots, 1)^T$ die Lösung ist mit dem Residuum $\mathbf{A}\mathbf{x} - \mathbf{b} = \mathbf{0}$. Die folgende Tabelle enthält wieder die Fehler in der Euklidischen Norm für die Lösung der Normalgleichungen mit der LR-Zerlegung (das Cholesky Verfahren führte schon bei $n = 10$ zu der Meldung, dass die Koeffizientenmatrix nicht positiv definit ist), mit der QR Zerlegung und der Singulärwertzerlegung. Die Lösungen sind ebenfalls unbrauchbar.

| | $n = 10$ | $n = 20$ | $n = 40$ |
|-------------------|----------|----------|----------|
| Normalgleichungen | 2.91 E+2 | 2.40 E+2 | 8.21 E+2 |
| QR Zerlegung | 1.93 E-5 | 5.04 E+0 | 1.08 E+1 |
| SVD | 4.67 E-5 | 6.41 E+1 | 3.72 E+2 |

□

Bei schlecht konditionierten Ausgleichsproblemen oder Gleichungssystemen ($n = m$) ist das folgende Vorgehen zu empfehlen:

Bestimme die Singulärwertzerlegung $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ von \mathbf{A} ; setze

$$\mathbf{\Sigma}_\tau^\dagger = \text{diag}(\eta_i \delta_{ji}), \quad \eta_i := \begin{cases} \sigma_i^{-1} & \text{falls } \sigma_i \geq \tau, \\ 0 & \text{sonst,} \end{cases}$$

wobei $\tau > 0$ eine gegebene Zahl ist,

$$\mathbf{A}_\tau^\dagger := \mathbf{V}\mathbf{\Sigma}_\tau^\dagger\mathbf{U}^T, \quad \mathbf{x} := \mathbf{A}_\tau^\dagger\mathbf{b}.$$

\mathbf{A}_τ^\dagger heißt **effektive Pseudoinverse** von \mathbf{A} . Von den Bedingungen aus Satz 5.30. bleiben nur

$$\mathbf{A}_\tau^\dagger \mathbf{A} \mathbf{A}_\tau^\dagger = \mathbf{A}_\tau^\dagger, \quad \mathbf{A}_\tau^\dagger \mathbf{A} = (\mathbf{A}_\tau^\dagger \mathbf{A})^T, \quad \mathbf{A} \mathbf{A}_\tau^\dagger = (\mathbf{A} \mathbf{A}_\tau^\dagger)^T$$

erhalten. An Stelle von $\mathbf{A} \mathbf{A}^\dagger \mathbf{A} = \mathbf{A}$ gilt nur

$$\|\mathbf{A} \mathbf{A}_\tau^\dagger \mathbf{A} - \mathbf{A}\|_2 \leq \tau.$$

Das obige Verfahren nennt man eine **Regularisierung**. Zu kleine singuläre Werte werden unschädlich gemacht, um die Kondition zu verbessern. Dafür nimmt man einen Verfahrensfehler in Kauf. Man löst an Stelle des Gleichungssystems $\mathbf{A}\mathbf{x} = \mathbf{b}$ das System $\mathbf{A}\mathbf{x} = \mathbf{P}\mathbf{b}$, wobei \mathbf{P} die orthogonale Projektion auf den Teilraum $\text{span}\{\mathbf{u}^i : \sigma_i \geq \tau\}$ bezeichnet.

Die bekannteste Regularisierung wurde unabhängig von Philips [84] und Tichonov [107] eingeführt und wird als **Tichonov Regularisierung** bezeichnet. Sie entspricht einer Dämpfung des Einflusses kleiner singulärer Werte bei der Lösung. Man löst an Stelle des Systems $\mathbf{A}\mathbf{x} = \mathbf{b}$ das Gleichungssystem

$$(\mathbf{A}^T \mathbf{A} + \alpha \mathbf{I}_n) \mathbf{x} = \mathbf{A}^T \mathbf{b} \quad (5.21)$$

mit einem Regularisierungsparameter $\alpha > 0$.

Offenbar ist (5.21) äquivalent zu

$$\|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2 + \alpha \|\mathbf{x}\|^2 = \min! \quad (5.22)$$

(dies ist die übliche Formulierung der Tichonov Regularisierung) oder zu

$$\|\tilde{\mathbf{A}}\mathbf{x} - \tilde{\mathbf{b}}\|^2 = \min!, \quad \tilde{\mathbf{A}} = \begin{pmatrix} \mathbf{A} \\ \sqrt{\alpha} \mathbf{I}_n \end{pmatrix}, \quad \tilde{\mathbf{b}} = \begin{pmatrix} \mathbf{b} \\ \mathbf{0} \end{pmatrix}. \quad (5.23)$$

Diese letzte Formulierung wurde zusammen mit der QR Zerlegung der Matrix $\tilde{\mathbf{A}}$ von Golub [48] erstmals verwendet, um die Regularisierung stabil auszuführen.

Wegen $\tilde{\mathbf{A}}^T \tilde{\mathbf{A}} = \mathbf{A}^T \mathbf{A} + \alpha \mathbf{I}_n$ besitzt $\tilde{\mathbf{A}}$ die singulären Werte $\sqrt{\sigma_i^2 + \alpha}$, wenn die σ_i die singulären Werte von \mathbf{A} sind, und die Kondition von (5.23) wird verkleinert zu $\sqrt{\frac{\sigma_1^2 + \alpha}{\sigma_n^2 + \alpha}}$.

Ist $\boldsymbol{\beta} := \mathbf{U}^T \mathbf{b}$, so ist (5.23) wegen (5.21) äquivalent zu

$$\mathbf{V}(\boldsymbol{\Sigma}^T \mathbf{U}^T \mathbf{U} \boldsymbol{\Sigma} + \alpha \mathbf{I}_n) \mathbf{V}^T \mathbf{x} = \mathbf{V} \boldsymbol{\Sigma}^T \mathbf{U}^T \mathbf{b} = \mathbf{V} \boldsymbol{\Sigma}^T \boldsymbol{\beta},$$

d.h.

$$\mathbf{x} = \mathbf{V}(\boldsymbol{\Sigma}^T \boldsymbol{\Sigma} + \alpha \mathbf{I}_n)^{-1} \boldsymbol{\Sigma}^T \boldsymbol{\beta} = \sum_{i=1}^n \frac{\beta_i \sigma_i}{\sigma_i^2 + \alpha} \mathbf{v}^i.$$

Man benötigt also nur die Singulärwertzerlegung von \mathbf{A} , um das regularisierte Problem für verschiedene Regularisierungsparameter α zu lösen.

Wir wenden auf das Gleichungssystem (5.19) die Regularisierungen durch Abschneiden der kleinen singulären Werte an und die verschiedenen Formen der Tichonov Regularisierungen. Dabei wird der Regularisierungsparameter α jeweils so gewählt, dass der Fehler minimal wird. Dies ist bei praktischen Problemen natürlich nicht möglich (die Lösung des Problems wird ja erst noch gesucht). Strategien zur Wahl des Parameters findet man in Engl [38] oder Louis [73]. Als grobe Richtlinie kann man sagen, dass man eine numerische Lösung, die bedingt durch die schlechte Kondition eines Problems verfälscht ist, häufig daran erkennt, dass sie stark oszilliert. Man variiert dann interaktiv den Parameter solange, bis man die Lösung glaubt (d.h. bis die gefundene Lösung die physikalischen Eigenschaften des modellierten Problems richtig wiedergibt).

Für das lineare Gleichungssystem (5.19), (5.20) erhält man die Fehler der folgenden Tabelle. Dabei wurden die Normalgleichungen der Tichonov Regularisierung (5.21) mit der Cholesky Zerlegung gelöst (die LR-Zerlegung mit dem MATLAB Befehl \ lieferte ähnliche Ergebnisse) und das regularisierte Ausgleichsproblem (5.23) wurde mit der QR Zerlegung von $\tilde{\mathbf{A}}$ und der Singulärwertzerlegung von \mathbf{A} gelöst.

| | $n = 10$ | $n = 20$ | $n = 40$ |
|--------------------|----------|----------|----------|
| Tichonov Cholesky | 1.41 E-3 | 2.03 E-3 | 3.51 E-3 |
| Tichonov QR | 3.50 E-6 | 5.99 E-6 | 7.54 E-6 |
| Tichonov SVD | 3.43 E-6 | 6.33 E-6 | 9.66 E-6 |
| Abgeschnittene SVD | 2.77 E-6 | 3.92 E-6 | 7.35 E-6 |

Für das Ausgleichsproblem erhält man

| | $n = 10$ | $n = 20$ | $n = 40$ |
|--------------------|----------|----------|----------|
| Tichonov Cholesky | 3.85 E-4 | 1.19 E-3 | 2.27 E-3 |
| Tichonov QR | 2.24 E-7 | 1.79 E-6 | 6.24 E-6 |
| Tichonov SVD | 8.51 E-7 | 1.61 E-6 | 3.45 E-6 |
| Abgeschnittene SVD | 7.21 E-7 | 1.94 E-6 | 7.70 E-6 |

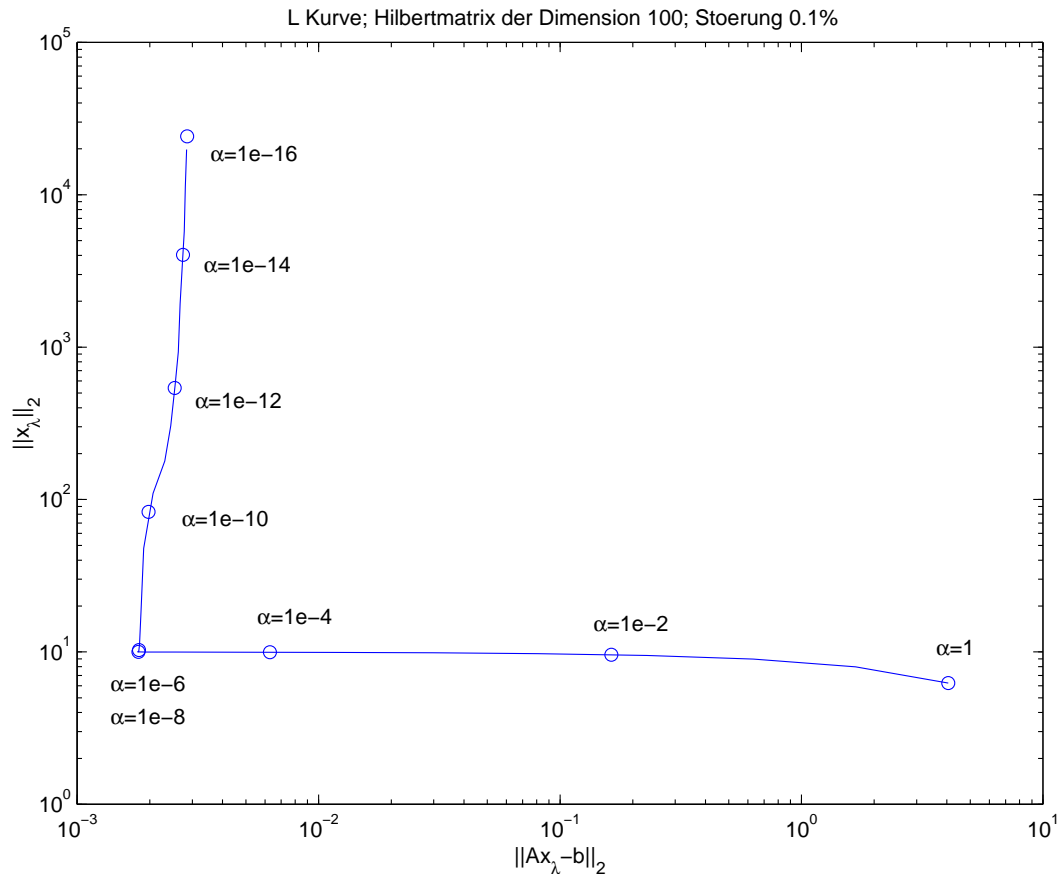


Abbildung 5.2: L-Kurve

Ein häufig verwendetes Verfahren zur Schätzung des optimalen Regularisierungsparameters ist die **L-Kurven Methode** [54], [55]. In ihr betrachtet man die Kurve

$$\alpha \mapsto (\|Ax_\alpha - b\|_2, \|x_\alpha\|_2).$$

Diese sog. **L-Kurve** hat häufig die Gestalt des Buchstaben *L* wie in Abbildung 5.2, wobei der Parameter α , der zu dem “Knick” gehört, optimal ist.

Für Systeme, für die Koeffizienten $b^T u^j$ der (ungestörten) rechten Seite bzgl. der singulären Vektoren u^j von A schneller abfallen als die singulären Werte σ_j von A , kann man die L-Kurven Methode begründen (vgl. Hansen [54]). Im allgemeinen Fall kann diese Methode aber versagen.

Abbildung 5.2 enthält die L-Kurve für das Gleichungssystem mit der Hilbertmatrix der Dimension 100 und der Lösung $\bar{x} = (1, \dots, 1)^T$, wobei zusätzlich die rechte Seite mit einem zufälligen Fehler von höchstens 0.1% verfälscht worden ist (Für dieses Beispiel ist die im letzten Absatz genannte Bedingung übrigens nicht erfüllt). Die folgende Tabelle enthält $\|Ax_\alpha - b\|_2$, $\|x_\alpha\|_2$ und den Fehler $\|x_\alpha - \bar{x}\|_2$ für

verschiedene Parameter α . Tatsächlich ist der Fehler minimal für $\alpha = 1e - 06$, und bei diesem Parameter liegt auch der Knick der L-Kurve.

| α | $\ \mathbf{Ax}_\alpha - \mathbf{b}\ _2$ | $\ \mathbf{x}_\alpha\ _2$ | $\ \mathbf{x}_\alpha - \bar{\mathbf{x}}\ _2$ |
|----------|---|---------------------------|--|
| 1e+00 | 4.0508e+00 | 6.2357e+00 | 5.3199e+00 |
| 1e-01 | 8.7802e-01 | 8.6987e+00 | 2.9868e+00 |
| 1e-02 | 1.6321e-01 | 9.5915e+00 | 1.6413e+00 |
| 1e-03 | 3.0354e-02 | 9.8697e+00 | 9.0701e-01 |
| 1e-04 | 6.2919e-03 | 9.9553e+00 | 5.2049e-01 |
| 1e-05 | 2.2522e-03 | 9.9819e+00 | 2.5714e-01 |
| 1e-06 | 1.7941e-03 | 9.9944e+00 | 1.0069e-01 |
| 1e-07 | 1.7862e-03 | 1.0017e+01 | 4.9295e-01 |
| 1e-08 | 1.8052e-03 | 1.0287e+01 | 2.3157e+00 |
| 1e-09 | 1.8355e-03 | 1.7458e+01 | 1.4260e+01 |
| 1e-10 | 1.9748e-03 | 8.3028e+01 | 8.2396e+01 |
| 1e-11 | 2.3529e-03 | 2.0179e+02 | 2.0153e+02 |
| 1e-12 | 2.5359e-03 | 5.3920e+02 | 5.3911e+02 |
| 1e-13 | 2.6525e-03 | 1.4070e+03 | 1.4070e+03 |
| 1e-14 | 2.7449e-03 | 4.0380e+03 | 4.0380e+03 |
| 1e-15 | 2.8050e-03 | 1.0709e+04 | 1.0709e+04 |
| 1e-16 | 2.8535e-03 | 2.4176e+04 | 2.4176e+04 |

Kapitel 6

Nichtsymmetrische Eigenwertaufgaben

Wir betrachten in diesem Kapitel die numerische Behandlung von vollbesetzten Eigenwertaufgaben. In Abschnitt 6.1 stellen wir einige Ergebnisse aus der Vorlesung Lineare Algebra zusammen und Abschnitt 6.2 enthält Störungsresultate für Eigenwerte und Eigenvektoren für nichtsymmetrische Eigenwertaufgaben. In Abschnitt 6.3 betrachten wir die Potenzmethode zur Bestimmung des betragsmaximalen Eigenwerts einer Matrix und Modifikationen, und in Abschnitt 6.4 behandeln wir den QR Algorithmus, das wichtigste Verfahren zur Bestimmung aller Eigenwerte (und Eigenvektoren) einer nichtsymmetrischen Matrix. In Abschnitt 6.5 betrachten wir schließlich den QZ Algorithmus für allgemeine Eigenwertaufgaben.

6.1 Vorbetrachtungen

Da Eigenwerte und Eigenvektoren reeller Matrizen komplex sein können, betrachten wir gleich für $\mathbf{A} \in \mathbb{C}^{(n,n)}$ das **spezielle Eigenwertproblem**

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}. \quad (6.1)$$

Besitzt (6.1) eine nichttriviale Lösung $\mathbf{x} \in \mathbb{C}^n \setminus \{\mathbf{0}\}$, so heißt λ ein **Eigenwert** von \mathbf{A} und \mathbf{x} ein zugehöriger **Eigenvektor**. Die Menge aller Eigenwerte einer Matrix \mathbf{A} heißt das **Spektrum** von \mathbf{A} und wird mit $\sigma(\mathbf{A})$ bezeichnet.

Genauer heißt $\mathbf{x} \neq \mathbf{0}$ mit (6.1) ein **Rechtseigenvektor** von \mathbf{A} zu λ . Ist λ ein Eigenwert von \mathbf{A} , so besitzt auch die Gleichung

$$\mathbf{y}^H(\mathbf{A} - \lambda\mathbf{I}) = \mathbf{0} \quad (6.2)$$

nichttriviale Lösungen. Jedes $\mathbf{y} \in \mathbb{C}^n$, das die Gleichung (6.2) erfüllt, heißt ein **Linkseigenvektor** von \mathbf{A} . Die Notation ist hier nicht einheitlich. In einigen Büchern werden die nichttrivialen Lösungen von $\mathbf{y}^T(\mathbf{A} - \lambda\mathbf{I}) = \mathbf{0}$ die Linkseigenvektoren von \mathbf{A} genannt. Wenn wir nur von Eigenvektoren sprechen, so sind stets Rechtseigenvektoren gemeint.

Die Eigenwerte von \mathbf{A} sind die Nullstellen des **charakteristischen Polynoms**

$$\chi(\lambda) := \det(\mathbf{A} - \lambda\mathbf{I}).$$

Ist $\tilde{\lambda}$ eine k -fache Nullstelle von χ (ist also das Polynom $\chi(\lambda)$ durch $(\lambda - \tilde{\lambda})^k$ aber nicht durch $(\lambda - \tilde{\lambda})^{k+1}$ teilbar), so heißt k die **algebraische Vielfachheit** von $\tilde{\lambda}$. Da χ den Grad n besitzt, besitzt also \mathbf{A} genau n Eigenwerte, wenn man jeden Eigenwert entsprechend seiner algebraischen Vielfachheit zählt.

Neben der algebraischen Vielfachheit betrachtet man die **geometrische Vielfachheit** eines Eigenwerts $\tilde{\lambda}$. Dies ist die Dimension des Lösungsraums des linearen Gleichungssystems

$$(\mathbf{A} - \tilde{\lambda}\mathbf{I})\mathbf{x} = \mathbf{0}.$$

Nach LA Satz 8.18 ist für jeden Eigenwert die geometrische Vielfachheit kleiner oder gleich der algebraischen Vielfachheit.

Gilt $\mathbf{B} = \mathbf{X}^{-1}\mathbf{A}\mathbf{X}$ mit einer regulären Matrix $\mathbf{X} \in \mathbb{C}^{(n,n)}$, so heißen die Matrizen \mathbf{A} und \mathbf{B} **ähnlich**, den Übergang von \mathbf{A} zu \mathbf{B} nennt man eine **Ähnlichkeitstransformation**. Ähnliche Matrizen besitzen (vgl. LA Satz 8.15) dieselben Eigenwerte einschließlich ihrer algebraischen und geometrischen Vielfachheit.

Naheliegender ist es nun, die Matrix \mathbf{A} , deren Eigenwerte bestimmt werden sollen, durch geeignete Ähnlichkeitstransformationen auf eine Gestalt zu bringen, aus der man die Eigenwerte (oder jedenfalls Näherungen hierfür) von \mathbf{A} leicht ablesen kann oder für die das Eigenwertproblem leichter gelöst werden kann.

Das folgende Lemma ermöglicht es, die Dimension des Eigenwertproblems zu reduzieren.

Lemma 6.1. *Besitzt $\mathbf{A} \in \mathbb{C}^{(n,n)}$ eine Blockzerlegung*

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{O} & \mathbf{A}_{22} \end{pmatrix}$$

mit $\mathbf{A}_{11} \in \mathbb{C}^{(m,m)}$, $\mathbf{A}_{22} \in \mathbb{C}^{(n-m,n-m)}$, so gilt

$$\sigma(\mathbf{A}) = \sigma(\mathbf{A}_{11}) \cup \sigma(\mathbf{A}_{22}). \quad (6.3)$$

Beweis: Es gelte

$$\mathbf{A}\mathbf{x} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{O} & \mathbf{A}_{22} \end{pmatrix} \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} = \lambda \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix}$$

mit $\mathbf{x}_1 \in \mathbb{C}^m$ und $\mathbf{x}_2 \in \mathbb{C}^{n-m}$. Gilt $\mathbf{x}_2 \neq \mathbf{0}$, so ist $\mathbf{A}_{22}\mathbf{x}_2 = \lambda\mathbf{x}_2$ und $\lambda \in \sigma(\mathbf{A}_{22})$. Ist $\mathbf{x}_2 = \mathbf{0}$, so gilt $\mathbf{A}_{11}\mathbf{x}_1 = \lambda\mathbf{x}_1$ und $\lambda \in \sigma(\mathbf{A}_{11})$. Es ist also

$$\sigma(\mathbf{A}) \subset (\sigma(\mathbf{A}_{11}) \cup \sigma(\mathbf{A}_{22})).$$

Zählt man alle Eigenwerte entsprechend ihrer algebraischen Vielfachheit, so besitzt \mathbf{A} n Eigenwerte, \mathbf{A}_{11} m Eigenwerte und \mathbf{A}_{22} $n - m$ Eigenwerte. Damit kann die Inklusion nicht echt sein, und es ist (6.3) bewiesen. ■

Ist \mathbf{J} die **Jordansche Normalform** der Matrix \mathbf{A} und $\mathbf{X}^{-1}\mathbf{A}\mathbf{X} = \mathbf{J}$, so kann man (vgl. LA Abschnitt 8.5) alle Eigenwerte und Eigenvektoren (und auch Hauptvektoren) von \mathbf{A} aus \mathbf{J} und der Transformationsmatrix \mathbf{X} sofort ablesen. Trotzdem wird die Jordansche Normalform in der numerischen Mathematik nicht verwendet. Den Grund zeigt

Beispiel 6.2. Die Störung

$$\mathbf{J}_\alpha := \begin{pmatrix} 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \alpha & 0 & 0 & \dots & 0 \end{pmatrix} \in \mathbb{R}^{(n,n)}$$

der Jordanschen Normalform \mathbf{J}_0 mit dem n -fachen Eigenwert 0 besitzt das charakteristische Polynom

$$\chi_\alpha(\lambda) = (-1)^n(\lambda^n - \alpha).$$

\mathbf{J}_α besitzt also n verschiedene Nullstellen, und ist daher auf Diagonalgestalt transformierbar. Dies zeigt, dass die Struktur der Jordanschen Normalform nicht stabil unter kleinen Störungen ist. □

Wir betrachten ferner nicht beliebige Ähnlichkeitstransformationen, denn die nicht-singuläre Transformationsmatrix \mathbf{X} kann schlecht konditioniert sein. In diesem Fall ist die Transformation $\mathbf{X}^{-1}\mathbf{A}\mathbf{X}$ anfällig für Rundungsfehler. Um dies auszuschließen, beschränken wir uns auf unitäre Transformationsmatrizen \mathbf{U} , für die also gilt $\mathbf{U}^H\mathbf{U} = \mathbf{I}$. Für diese ist

$$\|\mathbf{U}\mathbf{x}\|_2^2 = (\mathbf{U}\mathbf{x})^H(\mathbf{U}\mathbf{x}) = \mathbf{x}^H\mathbf{U}^H\mathbf{U}\mathbf{x} = \mathbf{x}^H\mathbf{x} = \|\mathbf{x}\|_2^2$$

für alle $\mathbf{x} \in \mathbb{C}^n$, daher gilt $\|\mathbf{U}\|_2 = 1$, und da mit \mathbf{U} auch $\mathbf{U}^{-1} = \mathbf{U}^H$ unitär ist, erhält man $\kappa_2(\mathbf{U}) = 1$.

Wir wissen (vgl. LA Satz 8.31), dass man genau die **normalen** Matrizen mit unitären Matrizen auf Diagonalgestalt transformieren kann. Allgemeine Matrizen kann man auf obere Dreiecksgestalt transformieren, aus der man dann wieder die Eigenwerte unmittelbar ablesen kann.

Satz 6.3. (Schursche Normalform)

Es sei $\mathbf{A} \in \mathbb{C}^{(n,n)}$. Dann existiert eine unitäre Matrix \mathbf{U} , so dass

$$\mathbf{U}^H \mathbf{A} \mathbf{U} = \mathbf{T} \quad (6.4)$$

obere Dreiecksgestalt besitzt. \mathbf{T} heißt **Schursche Normalform** der Matrix \mathbf{A} .

Beweis: Wir führen den Beweis durch Induktion bzgl. der Dimension n . Für $n = 1$ ist die Aussage trivial. Sie sei für Matrizen der Dimension $\leq n - 1$ bewiesen, und es sei $\mathbf{A} \in \mathbb{C}^{(n,n)}$. Ist \mathbf{x} ein Eigenvektor von \mathbf{A} zum Eigenwert λ und $\|\mathbf{x}\|_2 = 1$, so ergänzen wir \mathbf{x} zu einer unitären Matrix $\mathbf{W} = (\mathbf{x}, \mathbf{W}_1)$. Hierfür gilt

$$\mathbf{W}^H \mathbf{A} \mathbf{W} = \begin{pmatrix} \mathbf{x}^H \\ \mathbf{W}_1^H \end{pmatrix} \mathbf{A} (\mathbf{x}, \mathbf{W}_1) = \begin{pmatrix} \lambda & \mathbf{x}^H \mathbf{A} \mathbf{W}_1 \\ \mathbf{0} & \mathbf{W}_1^H \mathbf{A} \mathbf{W}_1 \end{pmatrix}.$$

Die Matrix $\mathbf{A}_1 := \mathbf{W}_1^H \mathbf{A} \mathbf{W}_1 \in \mathbb{C}^{(n-1, n-1)}$ besitzt nach Induktionsvoraussetzung eine Schur Zerlegung $\mathbf{R}_1 = \mathbf{V}^H \mathbf{A}_1 \mathbf{V}$ mit einer unitären Matrix \mathbf{V} und einer oberen Dreiecksmatrix \mathbf{R}_1 . Wir definieren hiermit $\mathbf{U} := \mathbf{W} \begin{pmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{V} \end{pmatrix}$. Dann gilt

$$\begin{aligned} \mathbf{U}^H \mathbf{A} \mathbf{U} &= \begin{pmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{V}^H \end{pmatrix} \mathbf{W}^H \mathbf{A} \mathbf{W} \begin{pmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{V} \end{pmatrix} \\ &= \begin{pmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{V}^H \end{pmatrix} \begin{pmatrix} \lambda & \mathbf{x}^H \mathbf{A} \mathbf{W}_1 \\ \mathbf{0} & \mathbf{W}_1^H \mathbf{A} \mathbf{W}_1 \end{pmatrix} \begin{pmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{V} \end{pmatrix} \\ &= \begin{pmatrix} \lambda & \mathbf{x}^H \mathbf{A} \mathbf{W}_1 \mathbf{V} \\ \mathbf{0} & \mathbf{R}_1 \end{pmatrix}. \quad \blacksquare \end{aligned}$$

Bemerkung 6.4. Besitzt \mathbf{A} die Schur Zerlegung

$$\mathbf{U}^H \mathbf{A} \mathbf{U} = \text{diag} \{ \lambda_1, \dots, \lambda_n \} + \mathbf{N}$$

mit einer strikten oberen Dreiecksmatrix \mathbf{N} , so gilt unabhängig von der Wahl von \mathbf{U} für die Schur Norm

$$\|\mathbf{N}\|_S^2 = \|\mathbf{U}^H \mathbf{A} \mathbf{U}\|_S^2 - \sum_{j=1}^n |\lambda_j|^2 = \|\mathbf{A}\|_S^2 - \sum_{j=1}^n |\lambda_j|^2.$$

Es gilt $\|\mathbf{N}\|_S^2 = 0$ genau dann, wenn \mathbf{A} eine normale Matrix ist. $\|\mathbf{N}\|_S =: \Delta(\mathbf{A})$ heißt die **Abweichung von \mathbf{A} von der Normalität**. \square

Bei der bisherigen Behandlung der Ähnlichkeitstransformationen haben wir einen wichtigen Aspekt nicht berücksichtigt. Ist die Ausgangsmatrix \mathbf{A} reell, so wird man nur orthogonale Matrizen \mathbf{U} (d.h. reelle unitäre Matrizen) zur Transformation benutzen, da sonst $\mathbf{U}^H \mathbf{A} \mathbf{U}$ komplexe Elemente enthält. Besitzt \mathbf{A} komplexe Eigenwerte, so kann \mathbf{A} hiermit offenbar nicht auf obere Dreiecksgestalt transformiert werden (die Diagonale enthält ja die Eigenwerte). Da mit $\lambda \in \mathbb{C} \setminus \mathbb{R}$ auch $\bar{\lambda}$ Eigenwert von \mathbf{A} ist (das charakteristische Polynom hat reelle Koeffizienten!), kann man \mathbf{A} auf **Quasidreiecksgestalt** transformieren. Dabei heißt eine Matrix \mathbf{R} Quasidreiecksmatrix, falls

$$\mathbf{R} = \begin{pmatrix} \mathbf{R}_{11} & \mathbf{R}_{12} & \mathbf{R}_{13} & \dots & \mathbf{R}_{1k} \\ \mathbf{O} & \mathbf{R}_{22} & \mathbf{R}_{23} & \dots & \mathbf{R}_{2k} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \mathbf{O} & \mathbf{O} & \mathbf{O} & \dots & \mathbf{R}_{kk} \end{pmatrix}$$

obere Blockdreiecksgestalt besitzt und die Diagonalblöcke \mathbf{R}_{jj} alle die Dimension 1 oder 2 besitzen.

Satz 6.5. (Reelle Schursche Normalform)

Sei $\mathbf{A} \in \mathbb{R}^{(n,n)}$. Dann existiert eine orthogonale Matrix \mathbf{U} , so dass $\mathbf{U}^T \mathbf{A} \mathbf{U}$ quasidreieckig ist. \mathbf{U} kann so gewählt werden, dass jeder 2×2 -Diagonalblock \mathbf{R}_{jj} nur komplexe Eigenwerte besitzt.

Beweis: Der Beweis wird ähnlich wie der von Satz 6.3. durch Induktion geführt, wobei man konjugiert komplexe Eigenwerte folgendermaßen behandeln kann: Ist $\lambda \notin \mathbb{R}$ Eigenwert von \mathbf{A} mit Eigenvektor \mathbf{x} , so ist $\bar{\lambda}$ Eigenwert von \mathbf{A} mit Eigenvektor $\bar{\mathbf{x}}$. Wegen $\lambda \neq \bar{\lambda}$ sind \mathbf{x} und $\bar{\mathbf{x}}$ linear unabhängig, und daher sind auch $\mathbf{x}_1 := \operatorname{Re} \mathbf{x}$ und $\mathbf{x}_2 := \operatorname{Im} \mathbf{x}$ linear unabhängig.

Mit $\lambda := \mu + i\nu$ gilt

$$\begin{aligned} \mathbf{A} \mathbf{x}_1 &= \mathbf{A} \left(\frac{\mathbf{x} + \bar{\mathbf{x}}}{2} \right) = \frac{1}{2} (\lambda \mathbf{x} + \bar{\lambda} \bar{\mathbf{x}}) \\ &= \frac{1}{2} ((\mu + \nu i)(\mathbf{x}_1 + i\mathbf{x}_2) + (\mu - \nu i)(\mathbf{x}_1 - i\mathbf{x}_2)) = \mu \mathbf{x}_1 - \nu \mathbf{x}_2 \end{aligned}$$

und genauso erhält man

$$\mathbf{A}\mathbf{x}_2 = \mathbf{A} \left(-i \frac{\mathbf{x} - \bar{\mathbf{x}}}{2} \right) = \mu \mathbf{x}_2 + \nu \mathbf{x}_1.$$

Zusammengenommen gilt

$$\mathbf{A}(\mathbf{x}_1, \mathbf{x}_2) = (\mathbf{x}_1, \mathbf{x}_2) \begin{pmatrix} \mu & \nu \\ -\nu & \mu \end{pmatrix}.$$

Wählt man $\alpha, \beta, \gamma \in \mathbb{R}$ geeignet, so sind $\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2$ gemäß

$$(\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2) := (\mathbf{x}_1, \mathbf{x}_2) \begin{pmatrix} \alpha & \beta \\ 0 & \gamma \end{pmatrix}$$

orthonormale Vektoren und

$$\mathbf{A}(\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2) = (\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2) \begin{pmatrix} \alpha & \beta \\ 0 & \gamma \end{pmatrix}^{-1} \begin{pmatrix} \mu & \nu \\ -\nu & \mu \end{pmatrix} \begin{pmatrix} \alpha & \beta \\ 0 & \gamma \end{pmatrix} =: (\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2) \cdot \mathbf{T}_1,$$

wobei $\mathbf{T}_1 \in \mathbb{R}^{(2,2)}$ genau die Eigenwerte λ und $\bar{\lambda}$ besitzt.

Ergänzt man nun $\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2$, durch $\mathbf{u}_3, \dots, \mathbf{u}_n$ zu einer orthogonalen Matrix $\mathbf{U} := (\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \mathbf{u}_3, \dots, \mathbf{u}_n)$, so gilt

$$\mathbf{U}^T \mathbf{A} \mathbf{U} = \begin{pmatrix} \mathbf{T}_1 & \mathbf{B} \\ \mathbf{0} & \mathbf{C} \end{pmatrix}.$$

Hiermit kann man wie vorher die Deflation durchführen. ■

6.2 Störungssätze für Eigenwerte und Eigenvektoren

Viele Verfahren zur Bestimmung von Eigenwerten bestehen darin, Matrizen \mathbf{X}_k zu bestimmen, so dass $\mathbf{X}_k^{-1} \mathbf{A} \mathbf{X}_k$ mehr und mehr einer Diagonalmatrix gleicht. Wir fragen daher, wie gut die Eigenwerte einer Matrix durch ihre Diagonalelemente approximiert werden.

Satz 6.6. (Gerschgorin) Sei

$$\mathbf{A} := (a_{ij}) \in \mathbb{C}^{(n,n)}, \quad z_i := \sum_{\substack{j=1 \\ j \neq i}}^n |a_{ij}|, \quad s_j := \sum_{\substack{i=1 \\ i \neq j}}^n |a_{ij}|.$$

Dann gilt

(i) Alle Eigenwerte von \mathbf{A} liegen in der Vereinigung der Kreise

$$Z_i := \{z \in \mathbb{C} : |z - a_{ii}| \leq z_i\}$$

(ii) Alle Eigenwerte von \mathbf{A} liegen in der Vereinigung der Kreise

$$S_j := \{z \in \mathbb{C} : |z - a_{jj}| \leq s_j\}$$

(iii) Jede Zusammenhangskomponente von $\bigcup_{i=1}^n Z_i$ bzw. $\bigcup_{j=1}^n S_j$ enthält genau so viele Eigenwerte von \mathbf{A} wie Kreise an der Komponente beteiligt sind, wobei Kreise und Eigenwerte entsprechend ihrer (algebraischen) Vielfachheit gezählt sind.

Beweis: (i): Sei $\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$, $\mathbf{x} \neq \mathbf{0}$. Dann gilt

$$(\lambda - a_{ii})x_i = \sum_{j \neq i} a_{ij}x_j \quad \text{für alle } i.$$

Sei $i \in \{1, \dots, n\}$ ein Index mit $|x_i| = \|\mathbf{x}\|_\infty$. Dann folgt

$$|\lambda - a_{ii}| \leq \sum_{j \neq i} |a_{ij}| \cdot \underbrace{\left| \frac{x_j}{x_i} \right|}_{\leq 1} \leq z_i,$$

und daher gilt $\lambda \in Z_i$.

(ii) folgt aus (i), da \mathbf{A} und \mathbf{A}^T dieselben Eigenwerte besitzen.

(iii) Wir betrachten für $t \in [0, 1]$

$$\mathbf{A}(t) := \mathbf{D} + t(\mathbf{A} - \mathbf{D}) \quad \text{mit } \mathbf{D} := \text{diag}\{a_{11}, \dots, a_{nn}\}.$$

Dann sind für $\mathbf{A}(t)$ die Radien der Kreise $Z_i(t)$ bzw. $S_j(t)$ gleich tz_i bzw. ts_j und die Mittelpunkte a_{ii} bzw. a_{jj} .

Offenbar ist die Behauptung für $t = 0$ richtig. Wächst nun t , so bleibt wegen der stetigen Abhängigkeit der Eigenwerte von den Elementen der Matrix $\mathbf{A}(t)$, also von t , die Anzahl der Kreise und Eigenwerte in einer Komponente konstant, solange sie nicht mit einer anderen Komponente zusammenstößt. Geschieht dies, so addieren sich die Kreise und Eigenwerte. Die Behauptung ist also für alle $t \geq 0$ und damit für $\mathbf{A} = \mathbf{A}(1)$ richtig. ■

Bemerkung 6.7. Die Aussage (iii) lässt sich nicht verschärfen zu: In jedem Kreis Z_i bzw. S_j liegt mindestens ein Eigenwert von \mathbf{A} , denn sei $\mathbf{A} = \begin{pmatrix} 0 & 1 \\ 2 & 0 \end{pmatrix}$. Dann sind die Eigenwerte $\lambda_{1/2} = \pm\sqrt{2}$, und es liegt kein Eigenwert in $Z_1 = S_2 = \{z : |z| \leq 1\}$.
□

Für viele Eigenwert Routinen kann man zeigen, dass die berechneten Eigenwerte $\lambda_1, \dots, \lambda_n$ die exakten Eigenwerte einer gestörten Matrix $\mathbf{A} + \mathbf{E}$ sind, wobei \mathbf{E} eine kleine Norm besitzt. Wir fragen daher, wie Störungen die Eigenwerte einer Matrix beeinflussen. Beispielhaft hierfür ist:

Satz 6.8. (Bauer, Fike) Ist μ Eigenwert von $\mathbf{A} + \mathbf{E} \in \mathbb{C}^{(n,n)}$ und

$$\mathbf{X}^{-1}\mathbf{A}\mathbf{X} = \mathbf{\Lambda} = \text{diag}\{\lambda_1, \dots, \lambda_n\},$$

so gilt

$$\min_{\lambda \in \sigma(\mathbf{A})} |\lambda - \mu| \leq \kappa_p(\mathbf{X}) \|\mathbf{E}\|_p.$$

Dabei bezeichnet $\|\cdot\|_p$ die von der Höldernorm

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

induzierte Matrixnorm und κ_p die Kondition bzgl. $\|\cdot\|_p$.

Beweis: Für $\mu \in \sigma(\mathbf{A})$ ist die Aussage trivial. Sei also $\mu \notin \sigma(\mathbf{A})$.

Da $\mathbf{X}^{-1}(\mathbf{A} + \mathbf{E} - \mu\mathbf{I})\mathbf{X}$ singularär ist, ist auch

$$\mathbf{I} + (\mathbf{\Lambda} - \mu\mathbf{I})^{-1}(\mathbf{X}^{-1}\mathbf{E}\mathbf{X}) = (\mathbf{\Lambda} - \mu\mathbf{I})^{-1}\mathbf{X}^{-1}(\mathbf{A} + \mathbf{E} - \mu\mathbf{I})\mathbf{X}$$

singularär.

Also existiert $\mathbf{y} \in \mathbb{C}^n$, $\|\mathbf{y}\|_p = 1$ mit

$$\mathbf{y} + (\mathbf{\Lambda} - \mu\mathbf{I})^{-1}(\mathbf{X}^{-1}\mathbf{E}\mathbf{X})\mathbf{y} = \mathbf{0}$$

Hieraus folgt

$$\begin{aligned} 1 &= \|\mathbf{y}\|_p = \|(\mathbf{\Lambda} - \mu\mathbf{I})^{-1}(\mathbf{X}^{-1}\mathbf{E}\mathbf{X})\mathbf{y}\|_p \\ &\leq \max_{\|\mathbf{z}\|_p=1} \|(\mathbf{\Lambda} - \mu\mathbf{I})^{-1}(\mathbf{X}^{-1}\mathbf{E}\mathbf{X})\mathbf{z}\|_p \\ &= \|(\mathbf{\Lambda} - \mu\mathbf{I})^{-1}\mathbf{X}^{-1}\mathbf{E}\mathbf{X}\|_p \leq \frac{1}{\min_{\lambda \in \sigma(\mathbf{A})} |\lambda - \mu|} \|\mathbf{X}^{-1}\|_p \|\mathbf{E}\|_p \|\mathbf{X}\|_p, \end{aligned}$$

wobei ausgenutzt wurde, dass die p -Norm einer Diagonalmatrix gleich dem Betrag des betragsmaximalen Elements der Matrix ist. ■

Für normale Matrizen erhält man insbesondere

Korollar 6.9. *Es sei \mathbf{A} eine normale Matrix und μ ein Eigenwert von $\mathbf{A} + \mathbf{E} \in \mathbb{C}^{(n,n)}$. Dann gilt*

$$\min_{\lambda \in \sigma(\mathbf{A})} |\lambda - \mu| \leq \|\mathbf{E}\|_2$$

Beweis: Da \mathbf{A} normal, kann man in Satz 6.8. \mathbf{X} unitär wählen, und die Behauptung folgt dann aus $\kappa_2(\mathbf{X}) = 1$. ■

Korollar 6.9. kann man gleichzeitig als Folgerung des folgenden Satzes auffassen, in dem die Zerlegung aus dem Satz von Schur ausgenutzt wird.

Satz 6.10. *Sei $\mathbf{A} \in \mathbb{C}^{(n,n)}$ und \mathbf{U} unitär, so dass*

$$\mathbf{U}^H \mathbf{A} \mathbf{U} = \mathbf{\Lambda} + \mathbf{N} = \text{diag}\{\lambda_1, \dots, \lambda_n\} + \mathbf{N}$$

mit einer strikten oberen Dreiecksmatrix \mathbf{N} . Ist $\mu \in \sigma(\mathbf{A} + \mathbf{E})$ und ist $p \in \mathbb{N}$ die kleinste natürliche Zahl mit $|\mathbf{N}|^p = \mathbf{O}$, so gilt

$$\min_{\lambda \in \sigma(\mathbf{A})} |\lambda - \mu| \leq \max(\theta, \theta^{1/p}).$$

wobei

$$\theta = \|\mathbf{E}\|_2 \sum_{k=0}^{p-1} \|\mathbf{N}\|_2^k$$

Beweis: Es sei

$$\delta := \min_{\lambda \in \sigma(\mathbf{A})} |\lambda - \mu|.$$

Für $\delta = 0$ ist die Behauptung trivial. Für $\delta > 0$ ist $\mathbf{I} - (\mu\mathbf{I} - \mathbf{A})^{-1}\mathbf{E}$ singular, und es gilt wie im letzten Beweis

$$1 \leq \|(\mu\mathbf{I} - \mathbf{A})^{-1}\mathbf{E}\|_2 \leq \|(\mu\mathbf{I} - \mathbf{A})^{-1}\|_2 \|\mathbf{E}\|_2 = \|[(\mu\mathbf{I} - \mathbf{\Lambda}) - \mathbf{N}]^{-1}\|_2 \|\mathbf{E}\|_2.$$

Da $(\mu\mathbf{I} - \mathbf{\Lambda})$ eine Diagonalmatrix ist, gilt nach dem folgenden Lemma 6.11. $[(\mu\mathbf{I} - \mathbf{\Lambda})^{-1}\mathbf{N}]^p = \mathbf{O}$, und daher

$$[(\mu\mathbf{I} - \mathbf{\Lambda}) - \mathbf{N}]^{-1} = \sum_{k=0}^{p-1} [(\mu\mathbf{I} - \mathbf{\Lambda})^{-1}\mathbf{N}]^k (\mu\mathbf{I} - \mathbf{\Lambda})^{-1}.$$

Geht man hier zu Normen über und setzt oben ein, so folgt

$$1 \leq \frac{1}{\delta} \|\mathbf{E}\|_2 \sum_{k=0}^{p-1} \delta^{-k} \|\mathbf{N}\|_2^k \leq \frac{1}{\delta} \|\mathbf{E}\|_2 \max(1, \delta^{1-p}) \sum_{k=0}^{p-1} \|\mathbf{N}\|_2^k = \theta \max(\delta^{-1}, \delta^{-p}),$$

und hieraus folgt die Behauptung. ■

Lemma 6.11. *Es sei \mathbf{N} eine strikte obere Dreiecksmatrix und p die kleinste natürliche Zahl mit $|\mathbf{N}|^p = \mathbf{O}$. Dann gilt für jede Diagonalmatrix \mathbf{D} auch $(\mathbf{DN})^p = \mathbf{O}$*

Beweis: Es sei das (i, j) -Element von $|\mathbf{N}|^2$ gleich 0. Ist dann die k -te Komponente der i -ten Zeile von \mathbf{N} von 0 verschieden, so muss das k -te Element der j -ten Spalte von \mathbf{N} gleich 0 sein. Dies zeigt, dass das (i, j) -Element von $(\mathbf{DN})^2$ gleich 0 sein muss.

Mit dieser Überlegung und einem trivialen Induktionsschluss erhält man die Behauptung. ■

Satz 6.8. und Satz 6.10. geben an, wie empfindlich Eigenwerte auf Störungen in \mathbf{A} reagieren können. Sind $\kappa_p(\mathbf{X})$ oder $\|\mathbf{N}\|_2^{p-1}$ groß, so können kleine Änderungen in den Elementen von \mathbf{A} große Änderungen in den Eigenwerten bewirken.

Beispiel 6.12. Es seien

$$\mathbf{A} = \begin{pmatrix} \mathbf{0} & \mathbf{I}_9 \\ 0 & \mathbf{0}^T \end{pmatrix} = \mathbf{J}_0^{(10)}, \quad \text{und} \quad \mathbf{E} = \begin{pmatrix} \mathbf{0} & \mathbf{O} \\ 10^{-10} & \mathbf{0}^T \end{pmatrix}.$$

Dann hat \mathbf{A} den 10-fachen Eigenwert 0 und alle Eigenwerte von $\mathbf{A} + \mathbf{E}$ haben Betrag 0.1 (Das charakteristische Polynom ist $\chi(\lambda) = \lambda^{10} - 10^{-10}$). Eine Störung in \mathbf{A} der Größe 10^{-10} bewirkt also eine Änderung der Eigenwerte von der Größe 10^{-1} .

Mit den Bezeichnungen aus Satz 6.10. gilt $\mathbf{N} = \mathbf{A}$, $\|\mathbf{N}\|_2 = 1$, $p = 10$ und $\|\mathbf{E}\|_2 = 10^{-10}$. Daher ist $\theta = 10^{-9}$ und

$$|\mu| \leq \max\{\theta, \theta^{1/10}\} = 10^{-0.9} \leq 0.126. \quad \square$$

Die im letzten Beispiel beobachtete empfindliche Abhängigkeit der Eigenwerte kann nicht bei normalen Matrizen auftreten (vgl. Korollar 6.9.). Andererseits impliziert Nichtnormalität nicht notwendig empfindliche Abhängigkeit. Ein nichtnormale Matrix kann gut- und schlechtkonditionierte Eigenwerte haben. Wir verbessern daher unsere Störungsüberlegungen, so dass sie auf einzelne Eigenwerte anwendbar sind, und nicht das ganze Spektrum gleichzeitig erfassen.

Satz 6.13. *Seien $\mathbf{A}, \mathbf{F} \in \mathbb{C}^{(n,n)}$. Sei $\tilde{\lambda}$ ein algebraisch einfacher Eigenwert von \mathbf{A} und $\tilde{\mathbf{x}}$ ein zugehöriger Eigenvektor. Es sei $\|\mathbf{F}\|_2 = 1$ und $\boldsymbol{\ell} \in \mathbb{C}^n$ mit $\boldsymbol{\ell}^H \tilde{\mathbf{x}} = 1$.*

Dann existieren $\eta > 0$ und beliebig oft stetig differenzierbare Abbildungen $\mathbf{x} : (-\eta, \eta) \rightarrow \mathbb{C}^n$, $\lambda : (-\eta, \eta) \rightarrow \mathbb{C}$ mit $\lambda(0) = \tilde{\lambda}$, $\mathbf{x}(0) = \tilde{\mathbf{x}}$ und

$$(\mathbf{A} + \varepsilon \mathbf{F})\mathbf{x}(\varepsilon) = \lambda(\varepsilon)\mathbf{x}(\varepsilon), \quad \boldsymbol{\ell}^H \mathbf{x}(\varepsilon) = 1.$$

Beweis: Wir betrachten die Abbildung $\Phi : \mathbb{C}^{n+2} \rightarrow \mathbb{C}^{n+1}$,

$$\Phi(\mathbf{x}, \lambda, \varepsilon) := \begin{pmatrix} (\mathbf{A} + \varepsilon \mathbf{F} - \lambda \mathbf{I})\mathbf{x} \\ \ell^H \mathbf{x} - 1 \end{pmatrix}.$$

Dann gilt

$$\Phi(\tilde{\mathbf{x}}, \tilde{\lambda}, 0) = \begin{pmatrix} (\mathbf{A} - \tilde{\lambda} \mathbf{I})\tilde{\mathbf{x}} \\ \ell^H \tilde{\mathbf{x}} - 1 \end{pmatrix} = \mathbf{0},$$

und die Behauptung folgt aus dem Satz über implizite Funktionen, falls

$$\frac{\partial}{\partial(\mathbf{x}, \lambda)} \Phi(\tilde{\mathbf{x}}, \tilde{\lambda}, 0) = \begin{pmatrix} \mathbf{A} - \tilde{\lambda} \mathbf{I} & , & -\tilde{\mathbf{x}} \\ \ell^H & , & 0 \end{pmatrix}$$

nichtsingulär ist.

Angenommen es gibt $\begin{pmatrix} \mathbf{z} \\ \mu \end{pmatrix} \in \mathbb{C}^{n+1}$, $\begin{pmatrix} \mathbf{z} \\ \mu \end{pmatrix} \neq \mathbf{0}$, mit

$$\frac{\partial}{\partial(\mathbf{x}, \lambda)} \Phi(\tilde{\mathbf{x}}, \tilde{\lambda}, 0) \begin{pmatrix} \mathbf{z} \\ \mu \end{pmatrix} = \mathbf{0},$$

d.h.

$$(\mathbf{A} - \tilde{\lambda} \mathbf{I})\mathbf{z} - \mu \tilde{\mathbf{x}} = \mathbf{0}, \quad \ell^H \mathbf{z} = 0.$$

Aus $\mu = 0$ folgt $\mathbf{z} \neq \mathbf{0}$ und $(\mathbf{A} - \tilde{\lambda} \mathbf{I})\mathbf{z} = \mathbf{0}$. Da $\tilde{\lambda}$ ein einfacher Eigenwert ist, folgt $\mathbf{z} = \alpha \tilde{\mathbf{x}}$ für ein $\alpha \in \mathbb{C}$, und man erhält den Widerspruch

$$0 = \ell^H \mathbf{z} = \alpha \ell^H \tilde{\mathbf{x}} = \alpha, \quad \text{d.h. } \mathbf{z} = \mathbf{0}.$$

Aus $\mu \neq 0$ folgt

$$(\mathbf{A} - \tilde{\lambda} \mathbf{I})\mathbf{z} = \mu \tilde{\mathbf{x}} \neq \mathbf{0}, \quad (\mathbf{A} - \tilde{\lambda} \mathbf{I})^2 \mathbf{z} = \mu (\mathbf{A} - \tilde{\lambda} \mathbf{I})\tilde{\mathbf{x}} = \mathbf{0},$$

d.h. \mathbf{z} ist ein Hauptvektor 2. Stufe zu $\tilde{\lambda}$, und $\tilde{\lambda}$ hat wenigstens die algebraische Vielfachheit 2 im Widerspruch zur Voraussetzung. ■

Bevor wir weiter untersuchen, wie empfindlich die Eigenwerte von Störungen abhängen, benötigen wir zunächst

Lemma 6.14. *Sei λ ein algebraisch einfacher Eigenwert der Matrix $\mathbf{A} \in \mathbb{C}^{(n,n)}$ mit dem Rechtseigenvektor \mathbf{x} und dem Linkseigenvektor \mathbf{y} . Dann gilt $\mathbf{y}^H \mathbf{x} \neq 0$.*

Beweis: Es sei ohne Beschränkung der Allgemeinheit $\|\mathbf{x}\|_2 = 1$. Wir ergänzen \mathbf{x} zu einer unitären Matrix $\mathbf{U} := (\mathbf{x}, \mathbf{U}_1)$. Dann gilt (vgl. den Beweis von Satz 6.3.)

$$\mathbf{U}^H \mathbf{A} \mathbf{U} = \begin{pmatrix} \lambda & \mathbf{z}^H \\ \mathbf{0} & \mathbf{A}_1 \end{pmatrix}.$$

Da λ algebraisch einfacher Eigenwert von \mathbf{A} ist, ist λ nicht Eigenwert der Matrix \mathbf{A}_1 , und daher besitzt die Gleichung $\mathbf{w}^H(\lambda\mathbf{I} - \mathbf{A}_1) = \mathbf{z}^H$ eine eindeutige Lösung \mathbf{w} . Für den hiermit gebildeten Vektor $\begin{pmatrix} 1 \\ \mathbf{w} \end{pmatrix}$ gilt

$$(1, \mathbf{w}^H)(\mathbf{U}^H \mathbf{A} \mathbf{U} - \lambda \mathbf{I}) = (\lambda, \mathbf{z}^H + \mathbf{w}^H \mathbf{A}_1) - \lambda(1, \mathbf{w}^H) = \mathbf{0}^T.$$

Den Ausdruck auf der linken Seite kann man schreiben als

$$(1, \mathbf{w}^H) \mathbf{U}^H (\mathbf{A} - \lambda \mathbf{I}) \mathbf{U} = \mathbf{0}^T,$$

und dies zeigt, dass

$$\mathbf{y} := \mathbf{U} \begin{pmatrix} 1 \\ \mathbf{w} \end{pmatrix}$$

ein Linkseigenvektor von \mathbf{A} zum Eigenwert λ ist. Für diesen gilt wegen $\mathbf{x} = (\mathbf{x}, \mathbf{U}_1) \mathbf{e}^1 = \mathbf{U} \mathbf{e}^1$

$$\mathbf{y}^H \mathbf{x} = (1, \mathbf{w}^H) \mathbf{U}^H \mathbf{x} = (1, \mathbf{w}^H) \mathbf{U}^H \mathbf{U} \mathbf{e}^1 = 1 \neq 0.$$

Da jeder andere Linkseigenvektor von \mathbf{A} zu λ ein Vielfaches von \mathbf{y} ist, ist die Behauptung gezeigt. \blacksquare

Wir betrachten die Situation aus Satz 6.13. Es sei $\tilde{\lambda}$ ein algebraisch einfacher Eigenwert von \mathbf{A} und $\tilde{\mathbf{x}}$ bzw. $\tilde{\mathbf{y}}$ ein zugehöriger Rechtseigenvektor bzw. Linkseigenvektor. $\mathbf{x}(\varepsilon)$ und $\lambda(\varepsilon)$ seien wie in Satz 6.13. definiert.

Differenziert man die Gleichung

$$(\mathbf{A} + \varepsilon \mathbf{F}) \mathbf{x}(\varepsilon) = \lambda(\varepsilon) \mathbf{x}(\varepsilon)$$

nach ε , so folgt für $\varepsilon = 0$

$$\mathbf{A} \dot{\mathbf{x}}(0) + \mathbf{F} \tilde{\mathbf{x}} = \dot{\lambda}(0) \tilde{\mathbf{x}} + \tilde{\lambda} \dot{\mathbf{x}}(0),$$

wobei der Punkt die Ableitung nach ε bezeichnet.

Multipliziert man mit $\tilde{\mathbf{y}}^H$, so erhält man

$$\tilde{\mathbf{y}}^H \mathbf{A} \dot{\mathbf{x}}(0) + \tilde{\mathbf{y}}^H \mathbf{F} \tilde{\mathbf{x}} = \tilde{\lambda} \tilde{\mathbf{y}}^H \dot{\mathbf{x}}(0) + \tilde{\mathbf{y}}^H \mathbf{F} \tilde{\mathbf{x}} = \dot{\lambda}(0) \tilde{\mathbf{y}}^H \tilde{\mathbf{x}} + \tilde{\lambda} \tilde{\mathbf{y}}^H \dot{\mathbf{x}}(0).$$

Geht man zu Beträgen über, so folgt wegen $\|\mathbf{F}\|_2 = 1$ aus der Cauchy Schwarzschen Ungleichung

$$|\dot{\lambda}(0)| = \frac{|\tilde{\mathbf{y}}^H \mathbf{F} \tilde{\mathbf{x}}|}{|\tilde{\mathbf{y}}^H \tilde{\mathbf{x}}|} \leq \frac{\|\tilde{\mathbf{y}}\|_2 \|\tilde{\mathbf{x}}\|_2}{|\tilde{\mathbf{y}}^H \tilde{\mathbf{x}}|},$$

wobei die obere Schranke für den Fall $\mathbf{F} := \tilde{\mathbf{y}}\tilde{\mathbf{x}}^H / (\|\tilde{\mathbf{y}}\|_2\|\tilde{\mathbf{x}}\|_2)$ angenommen wird.

Wegen

$$|\lambda(\varepsilon) - \tilde{\lambda}| = |\dot{\lambda}(0)|\varepsilon + O(\varepsilon^2)$$

nennt man $\|\tilde{\mathbf{y}}\|_2\|\tilde{\mathbf{x}}\|_2/|\tilde{\mathbf{y}}^H\tilde{\mathbf{x}}|$ die **Kondition** des einfachen Eigenwerts $\tilde{\lambda}$.

$s(\tilde{\lambda}) := |\tilde{\mathbf{y}}^H\tilde{\mathbf{x}}|/(\|\tilde{\mathbf{y}}\|_2\|\tilde{\mathbf{x}}\|_2)$ ist der Kosinus des Winkels zwischen Links- und Rechts-Eigenvektor zu $\tilde{\lambda}$.

Bemerkung 6.15. In einem gewissen Sinne misst $s(\tilde{\lambda})$ den Abstand, wie weit $\tilde{\lambda}$ von einem mehrfachen Eigenwert entfernt ist: Von Wilkinson [122] (vgl. auch Demmel [23], p. 152) wurde gezeigt, dass $\mathbf{E} \in \mathbb{C}^{(n,n)}$ derart existiert, dass $\tilde{\lambda}$ mehrfacher Eigenwert von $\mathbf{A} + \mathbf{E}$ ist und

$$\|\mathbf{E}\|_2 \leq \frac{s(\tilde{\lambda})}{\sqrt{1 - s(\tilde{\lambda})^2}}. \quad \square$$

Bemerkung 6.16. Ist der Eigenwert nicht einfach, so ist die Sensitivitätsfrage komplizierter. Ist z.B.

$$\mathbf{A} := \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix}, \quad \mathbf{F} := \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix},$$

so ist

$$\lambda(\mathbf{A} + \varepsilon\mathbf{F}) = \{1 \pm \sqrt{\varepsilon a}\}.$$

Für $a \neq 0$ hängen also die Eigenwerte nicht differenzierbar von ε ab. Dies ist typisch für Eigenwerte zu nichtlinearen Elementarteilern. Eine genauere Diskussion findet man in Wilkinson [121], pp. 77 ff. \square

Abschließend geben wir ein einfaches Resultat über die Empfindlichkeit der Eigenvektoren bei kleinen Störungen.

Es sei $\mathbf{A} \in \mathbb{C}^{(n,n)}$ mit verschiedenen Eigenwerten $\lambda_1, \dots, \lambda_n$ und $\mathbf{F} \in \mathbb{C}^{(n,n)}$ mit $\|\mathbf{F}\|_2 = 1$. Wie in Satz 6.13. gilt dann für genügend kleine $|\varepsilon|$

$$\begin{aligned} (\mathbf{A} + \varepsilon\mathbf{F})\mathbf{x}^k(\varepsilon) &= \lambda_k(\varepsilon)\mathbf{x}^k(\varepsilon), \\ \mathbf{y}^k(\varepsilon)^H(\mathbf{A} + \varepsilon\mathbf{F}) &= \lambda_k(\varepsilon)\mathbf{y}^k(\varepsilon)^H \end{aligned}$$

für $k = 1, \dots, n$, wobei $\lambda_k(\varepsilon)$, $\mathbf{x}^k(\varepsilon)$ und $\mathbf{y}^k(\varepsilon)$ beliebig oft differenzierbar sind.

Differenziert man die erste Gleichung nach ε , so erhält man an der Stelle $\varepsilon = 0$

$$\mathbf{A}\dot{\mathbf{x}}_k(0) + \mathbf{F}\mathbf{x}^k = \dot{\lambda}_k(0)\mathbf{x}^k + \lambda_k\dot{\mathbf{x}}_k(0)$$

mit $\lambda_k = \lambda_k(0)$, $\mathbf{x}^k = \mathbf{x}^k(0)$.

Da die \mathbf{x}^k eine Basis des \mathbb{C}^n bilden, gilt $\dot{\mathbf{x}}_k(0) = \sum_{i=1}^n a_i \mathbf{x}^i$, und daher

$$\mathbf{A}\dot{\mathbf{x}}^k(0) + \mathbf{F}\mathbf{x}^k = \sum_{i=1}^n a_i \lambda_i \mathbf{x}^i + \mathbf{F}\mathbf{x}^k = \dot{\lambda}_k(0) \mathbf{x}^k + \lambda_k \sum_{i=1}^n a_i \mathbf{x}^i.$$

Hieraus folgt

$$\sum_{\substack{i=1 \\ i \neq k}}^n a_i (\lambda_i - \lambda_k) \mathbf{x}^i + \mathbf{F}\mathbf{x}^k = \dot{\lambda}_k(0) \mathbf{x}^k. \quad (6.5)$$

Es sind Links- und Rechtseigenvektoren einer Matrix zu verschiedenen Eigenwerten orthogonal, denn aus $\mathbf{y}^H \mathbf{A} = \mu \mathbf{y}^H$, $\mathbf{A}\mathbf{x} = \lambda \mathbf{x}$ und $\mu \neq \lambda$ folgt

$$\mathbf{y}^H \mathbf{A}\mathbf{x} = \mu \mathbf{y}^H \mathbf{x} = \lambda \mathbf{y}^H \mathbf{x},$$

und daher $(\lambda - \mu) \mathbf{y}^H \mathbf{x} = 0$, d.h. $\mathbf{y}^H \mathbf{x} = 0$. Da alle λ_i verschieden sind, gilt $(\mathbf{y}^i)^H \mathbf{x}^k = 0$ für $i \neq k$, und daher folgt aus Gleichung (6.5)

$$a_i (\lambda_i - \lambda_k) (\mathbf{y}^i)^H \mathbf{x}^i + (\mathbf{y}^i)^H \mathbf{F}\mathbf{x}^k = 0 \quad \text{für } i \neq k.$$

Damit ist

$$a_i = \frac{(\mathbf{y}^i)^H \mathbf{F}\mathbf{x}^k}{(\lambda_k - \lambda_i) (\mathbf{y}^i)^H \mathbf{x}^i},$$

und man erhält schließlich

$$\mathbf{x}^k(\varepsilon) = \mathbf{x}^k + \varepsilon \sum_{\substack{i=1 \\ i \neq k}}^n \left\{ \frac{(\mathbf{y}^i)^H \mathbf{F}\mathbf{x}^k}{(\lambda_k - \lambda_i) (\mathbf{y}^i)^H \mathbf{x}^i} \right\} \mathbf{x}^i + O(\varepsilon^2).$$

Die Empfindlichkeit eines Eigenvektors gegenüber Störungen hängt damit auch von der Trennung des entsprechenden Eigenwerts vom übrigen Spektrum ab.

6.3 Potenzmethode

Wir betrachten in diesem Abschnitt ein Verfahren zur Bestimmung des betragsgrößten Eigenwerts und zugehörigen Eigenvektors, eine Modifikation hiervon, um auch andere Eigenwerte zu berechnen und eine simultane Variante. Wir besprechen die Methoden vor allem zum besseren Verständnis des dann im nächsten Abschnitt folgenden Arbeitspferdes zur Berechnung von Eigenwerten und Eigenvektoren, des QR Algorithmus.

6.3.1 Potenzmethode; von Mises Iteration

Wir betrachten für $\mathbf{A} \in \mathbb{C}^{(n,n)}$ die spezielle Eigenwertaufgabe

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}. \quad (6.6)$$

Es sei \mathbf{A} diagonalisierbar, $\mathbf{A}\mathbf{x}^i = \lambda_i\mathbf{x}^i$, $i = 1, \dots, n$, und es besitze \mathbf{A} einen dominanten Eigenwert λ_1 , d.h.

$$|\lambda_1| > |\lambda_2| \geq \dots \geq |\lambda_n|. \quad (6.7)$$

Es sei $\mathbf{u}^0 \in \mathbb{C}^n$, $\mathbf{u}^0 \neq \mathbf{0}$, gegeben. Multipliziert man

$$\mathbf{u}^0 =: \sum_{i=1}^n \alpha_i \mathbf{x}^i$$

m mal mit der Matrix \mathbf{A} , so folgt

$$\mathbf{A}^m \mathbf{u}^0 = \sum_{i=1}^n \alpha_i \lambda_i^m \mathbf{x}^i = \lambda_1^m \left\{ \alpha_1 \mathbf{x}^1 + \sum_{i=2}^n \alpha_i \left(\frac{\lambda_i}{\lambda_1} \right)^m \mathbf{x}^i \right\}. \quad (6.8)$$

Wegen (6.7) konvergiert daher die Folge $\{\lambda_1^{-m} \mathbf{A}^m \mathbf{u}^0\}$ gegen ein Vielfaches von \mathbf{x}^1 , falls $\alpha_1 \neq 0$ gilt, d.h. $\{\mathbf{A}^m \mathbf{u}^0\}$ konvergiert gegen die Eigenrichtung $\text{span}\{\mathbf{x}^1\}$ zum dominanten Eigenwert λ_1 , wenn der Startvektor \mathbf{u}^0 eine Komponente in Richtung von \mathbf{x}^1 besitzt.

Gilt $|\lambda_1| \neq 1$, so erhält man für große m Exponentenüberlauf oder -unterlauf. Man wird also die Folge $\{\mathbf{A}^m \mathbf{u}^0\}$ noch geeignet normieren und erhält die **Potenzmethode** oder das **von Mises Verfahren**.

Sei $\|\cdot\|$ irgendeine Norm auf \mathbb{C}^n und $\mathbf{u}^0 \in \mathbb{C}^n \setminus \{\mathbf{0}\}$.

Algorithmus 6.17. (Potenzmethode)

```

for m=0,1,... until convergence do
  v_{m+1}=Au_m;
  k_{m+1}=\|v_{m+1}\|;
  u_{m+1}=v_{m+1}/k_{m+1};
end

```

Häufig wird eine andere Skalierung der erzeugten Folge gewählt, die billiger ist als die Normierung. Sei dazu $\boldsymbol{\ell} \in \mathbb{C}^n$ ein gegebener Vektor. Hiermit iterieren wir gemäß

Algorithmus 6.18. (Potenzmethode)

```

for m=0,1,... until convergence do
  v_{m+1}=Au_m;
  k_{m+1}=1'*v_{m+1};
  u_{m+1}=v_{m+1}/k_{m+1};
end

```

Satz 6.19. *Es sei λ_1 dominanter Eigenwert von \mathbf{A} und*

$$\mathbf{u}^0 := \sum_{i=1}^n \alpha_i \mathbf{x}^i$$

mit $\alpha_1 \neq 0$ und $\ell \in \mathbb{C}^n$.

Es seien \mathbf{u}^m und k_m nach Algorithmus 6.18. berechnet. Gilt dann $\ell^H \mathbf{u}^m \neq 0$ für alle $m \in \mathbb{N}$ und $\ell^H \mathbf{x}^1 \neq 0$, so ist

$$\lim_{m \rightarrow \infty} k_m = \lambda_1, \quad \lim_{m \rightarrow \infty} \mathbf{u}^m = \frac{1}{\ell^H \mathbf{x}^1} \mathbf{x}^1.$$

Beweis: Offensichtlich gilt

$$\mathbf{u}^m = \mathbf{A}^m \mathbf{u}^0 / \ell^H \mathbf{A}^m \mathbf{u}^0.$$

Daher ist

$$k_m = \ell^H \mathbf{v}^m = \ell^H \mathbf{A} \mathbf{u}^{m-1} = \ell^H \mathbf{A} \frac{\mathbf{A}^{m-1} \mathbf{u}^0}{\ell^H \mathbf{A}^{m-1} \mathbf{u}^0} = \frac{\ell^H \mathbf{A}^m \mathbf{u}^0}{\ell^H \mathbf{A}^{m-1} \mathbf{u}^0},$$

und aus

$$\lim_{m \rightarrow \infty} \lambda_1^{-m} \mathbf{A}^m \mathbf{u}^0 = \alpha_1 \mathbf{x}^1$$

folgt

$$\lim_{m \rightarrow \infty} \lambda_1^{-1} k_m = \frac{\lim_{m \rightarrow \infty} \ell^H (\lambda_1^{-m} \mathbf{A}^m \mathbf{u}^0)}{\lim_{m \rightarrow \infty} \ell^H (\lambda_1^{-(m-1)} \mathbf{A}^{m-1} \mathbf{u}^0)} = 1,$$

d.h.

$$\lim_{m \rightarrow \infty} k_m = \lambda_1$$

und

$$\lim_{m \rightarrow \infty} \mathbf{u}^m = \lim_{m \rightarrow \infty} \frac{\mathbf{A}^m \mathbf{u}^0}{\ell^H \mathbf{A}^m \mathbf{u}^0} = \lim_{m \rightarrow \infty} \frac{\alpha_1 \mathbf{x}^1 + \sum_{i=2}^n \alpha_i (\lambda_i / \lambda_1)^m \mathbf{x}^i}{\alpha_1 \ell^H \mathbf{x}^1 + \sum_{i=2}^n \alpha_i (\lambda_i / \lambda_1)^m \ell^H \mathbf{x}^i} = \frac{\mathbf{x}^1}{\ell^H \mathbf{x}^1}. \quad \blacksquare$$

- Bemerkung 6.20.**
1. Eine häufige Wahl von ℓ ist $\ell = e^k$ für ein $k \in \{1, \dots, n\}$, d.h. man normiert im Eigenvektor die k -te Komponente zu 1, wobei man nicht vorhersagen kann, welches k geeignet ist.
 2. Ist $\mathbf{A} \in \mathbb{R}^{(n,n)}$ nichtnegativ und irreduzibel, so ist der zum Spektralradius gehörende Eigenvektor positiv, und man kann $\ell = (1, 1, \dots, 1)^T$ wählen, wenn \mathbf{u}^0 positive Komponenten hat.
 3. Wir haben $\alpha_1 \neq 0$ vorausgesetzt. Diese Voraussetzung ist nicht einschneidend, denn durch Rundungsfehler wird (fast immer) ein \mathbf{u}^m erzeugt, das eine Komponente in Richtung von \mathbf{x}^1 besitzt.
 4. Ist λ_1 mehrfacher Eigenwert von \mathbf{A} und \mathbf{A} diagonalähnlich, so bleiben alle Überlegungen richtig, wenn nur \mathbf{u}^0 eine Komponente in Richtung des Eigenraumes von \mathbf{A} zu λ_1 besitzt oder eine solche durch Rundungsfehler erzeugt wird.
 5. Ist $\mathbf{A} \in \mathbb{R}^{(n,n)}$ mit $|\lambda_1| = |\lambda_2| > |\lambda_3| \geq \dots \geq |\lambda_n|$ und $\lambda_1 \neq \lambda_2$ (also $\lambda_1 = \bar{\lambda}_2$ oder $\lambda_1 = -\lambda_2$), so erhält man keine Konvergenz, sondern es treten in der Folge $\{\mathbf{u}^m\}$ Oszillationen auf. Auch in diesem Fall kann man aus (drei aufeinanderfolgenden Elementen) der Folge $\{\mathbf{u}^m\}$ Näherungen für die zu λ_1 und λ_2 gehörenden Eigenvektoren ermitteln. Eine ausführliche Diskussion findet man in Zurmühl [123], pp. 283 ff.
 6. Schließlich haben wir vorausgesetzt, dass \mathbf{A} diagonalisierbar ist. Ist dies nicht erfüllt, so muss man \mathbf{u}^0 als Linearkombination von Hauptvektoren darstellen und erhält ebenfalls Konvergenz des Verfahrens (vgl. Collatz [20], pp. 325 ff.).
 7. Für die Konvergenzgeschwindigkeit gilt mit

$$q := \max_{i=2, \dots, n} \left| \frac{\lambda_i}{\lambda_1} \right|$$

$$\begin{aligned}
|k_{m+1} - \lambda_1| &= \left| \frac{\ell^H \mathbf{A}^{m+1} \mathbf{u}^0}{\ell^H \mathbf{A}^m \mathbf{u}^0} - \lambda_1 \right| = \left| \frac{1}{\ell^H \mathbf{A}^m \mathbf{u}^0} \ell^H (\mathbf{A}^{m+1} \mathbf{u}^0 - \lambda_1 \mathbf{A}^m \mathbf{u}^0) \right| \\
&= \left| \frac{\lambda_1^{m+1}}{\ell^H \mathbf{A}^m \mathbf{u}^0} \ell^H \left(\sum_{i=2}^n \alpha_i \left(\left(\frac{\lambda_i}{\lambda_1} \right)^{m+1} - \left(\frac{\lambda_i}{\lambda_1} \right)^m \right) \mathbf{x}^i \right) \right| \\
&\leq \left| \frac{\lambda_1^{m+1}}{\ell^H \mathbf{A}^m \mathbf{u}^0} \right| \|\ell\|_2 \sum_{i=2}^n |\alpha_i| \left| \frac{\lambda_i}{\lambda_1} \right|^m \left| \frac{\lambda_i}{\lambda_1} - 1 \right| \|\mathbf{x}^i\|_2 \leq C q^m \quad (6.9)
\end{aligned}$$

d.h. wir haben lineare Konvergenz mit dem Konvergenzfaktor q .

Ist also $|\lambda_1|$ von den Beträgen der übrigen Eigenwerte gut getrennt, so erhält man rasche Konvergenz, i.A. ist das Verfahren aber sehr langsam.

8. Bisweilen kann die Konvergenzgeschwindigkeit durch eine Spektralverschiebung verbessert werden, d.h. betrachte $\mathbf{A} + \alpha\mathbf{I}$, $\alpha \in \mathbb{C}$. Dann gehen die Eigenwerte über in $\lambda_i + \alpha$, und die Konvergenzgeschwindigkeit wird bestimmt durch

$$q := \max_{i=2,\dots,n} \left| \frac{\lambda_i + \alpha}{\lambda_1 + \alpha} \right|.$$

Die Verbesserung hierdurch ist meistens unwesentlich.

6.3.2 Inverse Iteration

Eine erhebliche Beschleunigung der von Mises Iteration erhält man durch die inverse Iteration, die 1944 von Wielandt [120] eingeführt wurde bei der Stabilitätsanalyse von Strukturen, die kleine Störungen bekannter Systeme sind. In diesem Fall sind gute Approximationen für die relevanten Eigenwerte bekannt, und man erhält (wie wir noch sehen werden) rasche Konvergenz. Heute wird die inverse Iteration vor allem verwendet zur Bestimmung von Eigenvektoren, wenn wenige Eigenwerte und Eigenvektoren gesucht sind.

Sei $\lambda \neq \lambda_i$ für alle i . Dann besitzt die Matrix $(\lambda\mathbf{I} - \mathbf{A})^{-1}$ die Eigenwerte $1/(\lambda - \lambda_i)$ und die Eigenvektoren stimmen mit denen von \mathbf{A} überein.

Führt man die von Mises Iteration für $(\lambda\mathbf{I} - \mathbf{A})^{-1}$ aus und gilt $|\lambda - \lambda_i| < |\lambda - \lambda_j|$, $j = 1, \dots, n$, $j \neq i$, mit einem einfachen Eigenwert λ_i von \mathbf{A} , so ist $\frac{1}{\lambda - \lambda_i}$ dominanter Eigenwert von $(\lambda\mathbf{I} - \mathbf{A})^{-1}$ und die Konvergenzgeschwindigkeit wird bestimmt durch

$$q := \max_{j \neq i} \left| \frac{\lambda - \lambda_i}{\lambda - \lambda_j} \right|.$$

Ist also λ eine gute Näherung für λ_i , so erhält man sehr schnelle Konvergenz.

Algorithmus 6.21. (Inverse Iteration; feste Shifts)

```

for m=0,1,... until convergence do
  L\{o}se  $(\lambda\mathbf{I}-\mathbf{A})\mathbf{v}_{\{m+1\}}=\mathbf{u}_m$ ;
   $\mathbf{k}_{\{m+1\}}=\mathbf{1}'*\mathbf{v}_{\{m+1\}}$ ;
   $\mathbf{u}_{\{m+1\}}=\mathbf{v}_{\{m+1\}}/\mathbf{k}_{\{m+1\}}$ ;
end

```

Wie in Satz 6.19. erhält man im Fall $|\lambda - \lambda_i| < |\lambda - \lambda_j|$ für alle $j \neq i$

$$\mathbf{u}^m \rightarrow \frac{\mathbf{x}^i}{\ell^H \mathbf{x}^i}, \quad k_m \rightarrow \frac{1}{\lambda - \lambda_i}$$

unter den entsprechenden Voraussetzungen $\alpha_i \neq 0$, $\ell^H \mathbf{v}^m \neq 0$, $\ell^H \mathbf{x}^i \neq 0$.

Die Konvergenz ist natürlich linear (vgl. (6.9)). Dies wird verbessert, wenn man λ gleichzeitig iteriert:

Algorithmus 6.22. (Inverse Iteration; variable Shifts)

```

for m=0,1,... until convergence do
  L\{"o"}se  $(\lambda_m I - A)v_{m+1} = u_m$ ;
   $k_{m+1} = 1 / v_{m+1}$ ;
   $u_{m+1} = v_{m+1} / k_{m+1}$ ;
   $\lambda_{m+1} = \lambda_m - 1 / k_{m+1}$ ;
end

```

Es gilt ja $k_{m+1} \rightarrow \frac{1}{\lambda - \lambda_i}$. Hieraus erhält man eine Schätzung $\lambda_{(m+1)}$ für λ_i durch

$$k_{m+1} \approx 1 / (\lambda_{(m)} - \lambda_{(m+1)}),$$

und damit die Aufdatierung des Shift Parameters in Algorithmus 6.22.

Satz 6.23. Sei $\tilde{\lambda}$ ein algebraisch einfacher Eigenwert von \mathbf{A} mit zugehörigem Eigenvektor $\tilde{\mathbf{u}}$, und es gelte $\ell^H \tilde{\mathbf{u}} = \ell^H \mathbf{u}^0 = 1$. Dann konvergiert das Verfahren in Algorithmus 6.22. lokal quadratisch gegen $\tilde{\lambda}$ bzw. $\tilde{\mathbf{u}}$.

Beweis: Wir zeigen, dass das Verfahren genau die Iterationsvorschrift des Newton-Verfahrens für das Gleichungssystem

$$F(\mathbf{u}, \lambda) := \begin{pmatrix} \lambda \mathbf{u} - \mathbf{A} \mathbf{u} \\ \ell^H \mathbf{u} - 1 \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ 0 \end{pmatrix}$$

ist und dass $F'(\tilde{\mathbf{u}}, \tilde{\lambda})$ nichtsingulär ist. Dann folgt die quadratische Konvergenz sowohl für den Eigenvektor als auch den Eigenwert aus einem Satz in Mathematik III. Wir werden auf das Newton Verfahren in Kapitel 8 eingehen.

Es gilt

$$F'(\mathbf{u}, \lambda) = \begin{pmatrix} \lambda \mathbf{I} - \mathbf{A} & \mathbf{u} \\ \ell^H & 0 \end{pmatrix}.$$

Das Newton-Verfahren ist also gegeben durch

$$\begin{pmatrix} \lambda_{(m)}\mathbf{I} - \mathbf{A} & \mathbf{u}^m \\ \boldsymbol{\ell}^H & 0 \end{pmatrix} \begin{pmatrix} \mathbf{u}^{m+1} - \mathbf{u}^m \\ \lambda_{(m+1)} - \lambda_{(m)} \end{pmatrix} = - \begin{pmatrix} \lambda_{(m)}\mathbf{u}^m - \mathbf{A}\mathbf{u}^m \\ \boldsymbol{\ell}^H\mathbf{u}^m - 1 \end{pmatrix}$$

d.h.

$$\begin{aligned} (\lambda_{(m)}\mathbf{I} - \mathbf{A})\mathbf{u}^{m+1} + (\lambda_{(m+1)} - \lambda_{(m)})\mathbf{u}^m &= \mathbf{0} \\ \boldsymbol{\ell}^H\mathbf{u}^{m+1} &= 1, \end{aligned}$$

und dies ist äquivalent Algorithmus 6.22.

Es sei

$$F'(\tilde{\mathbf{u}}, \tilde{\lambda}) \begin{pmatrix} \mathbf{v} \\ \mu \end{pmatrix} = \mathbf{0}$$

d.h.

$$(\tilde{\lambda}\mathbf{I} - \mathbf{A})\mathbf{v} + \mu\tilde{\mathbf{u}} = \mathbf{0}, \quad \boldsymbol{\ell}^H\mathbf{v} = 0.$$

Im Falle $\mu = 0$ gilt $(\tilde{\lambda}\mathbf{I} - \mathbf{A})\mathbf{v} = \mathbf{0}$, d.h., da $\tilde{\lambda}$ ein einfacher Eigenwert von \mathbf{A} ist, $\mathbf{v} = \alpha\tilde{\mathbf{u}}$ für ein $\alpha \in \mathbb{C}$, und es folgt $0 = \boldsymbol{\ell}^H\mathbf{v} = \alpha\boldsymbol{\ell}^H\tilde{\mathbf{u}} = \alpha$.

Im Falle $\mu \neq 0$ gilt

$$(\tilde{\lambda}\mathbf{I} - \mathbf{A})\mathbf{v} = -\mu\tilde{\mathbf{u}} \neq \mathbf{0},$$

und daher

$$(\tilde{\lambda}\mathbf{I} - \mathbf{A})^2\mathbf{v} = -\mu(\tilde{\lambda}\mathbf{I} - \mathbf{A})\tilde{\mathbf{u}} = \mathbf{0}.$$

\mathbf{v} ist also ein Hauptvektor 2. Stufe, und daher besitzt $\tilde{\lambda}$ mindestens die algebraische Vielfachheit 2. ■

Bemerkung 6.24. Man entnimmt dem Beweis sofort, dass das Verfahren in Algorithmus 6.21. auch dann lokal konvergiert, wenn \mathbf{A} nicht diagonalisierbar ist. □

Wir werden noch sehen, dass die inverse Iteration sogar kubisch konvergiert, wenn die Matrix \mathbf{A} symmetrisch ist und wenn die Näherung für den Eigenwert aufdatiert wird mit dem Rayleighschen Quotienten

$$\lambda_{(m)} = \frac{(\mathbf{u}^m)^H \mathbf{A} \mathbf{u}^m}{\|\mathbf{u}^m\|_2^2}.$$

Diese Aufdatierung ist auch bei nicht symmetrischen Matrizen sinnvoll wegen des folgenden Lemmas. Man erhält dann wie in Satz 6.23. quadratische Konvergenz (s. Stewart [97], pp. 346 ff.).

Lemma 6.25. Sei $\mathbf{A} \in \mathbb{C}^{(n,n)}$ beliebig, $\mathbf{x} \in \mathbb{C}^n$, und für $\mu \in \mathbb{C}$

$$\mathbf{r}(\mu) := \mathbf{A}\mathbf{x} - \mu\mathbf{x}$$

der zugehörige Defekt. Dann gilt

$$\|\mathbf{r}(\mu)\|_2 = \min! \iff \mu = \frac{\mathbf{x}^H \mathbf{A}\mathbf{x}}{\|\mathbf{x}\|_2^2}.$$

Beweis: $\|\mathbf{r}(\mu)\|_2 = \min!$ ist ein lineares Ausgleichsproblem zur Bestimmung von μ mit der Koeffizientenmatrix \mathbf{x} und der rechten Seite $\mathbf{A}\mathbf{x}$. Die Normalgleichung hierzu lautet

$$\mathbf{x}^H \mathbf{x} \mu = \mathbf{x}^H \mathbf{A}\mathbf{x}. \quad \blacksquare$$

Auf den ersten Blick scheint es eine Schwierigkeit bei der inversen Iteration zu sein, dass die Koeffizientenmatrix $\lambda_{(m)}\mathbf{I} - \mathbf{A}$ des zu lösenden linearen Gleichungssystems bei Konvergenz von $\lambda_{(m)}$ gegen einen Eigenwert von \mathbf{A} mehr und mehr singular wird, die Kondition also wächst und damit der Fehler der Lösung des Gleichungssystems wächst. Man kann jedoch zeigen (vgl. Chatelin [17], pp. 217 ff.), dass (bei gut konditionierten Problemen) der Fehler, der bei der Lösung von $(\lambda_{(m)}\mathbf{I} - \mathbf{A})\mathbf{v}^{m+1} = \mathbf{u}^m$ gemacht wird, vornehmlich die Richtung des Eigenvektors zu λ , also die gewünschte Richtung, hat.

6.3.3 Deflation

Hat man mit einem numerischen Verfahren einen Eigenwert λ_1 und einen zugehörigen Eigenvektor \mathbf{x}^1 bestimmt und möchte man weitere Eigenwerte und Eigenvektoren berechnen, so muss man den bereits gefundenen Eigenwert “unschädlich” machen, um zu verhindern, dass das Verfahren gegen den bereits bekannten Eigenwert bzw. Eigenvektor erneut konvergiert. Dies geschieht mit Hilfe der Deflation.

Wir beschreiben drei Typen:

Deflation nach Hotelling: Sei $\mathbf{A} \in \mathbb{C}^{(n,n)}$ Hermitesch, und seien \mathbf{x}^1 und λ_1 mit $\mathbf{A}\mathbf{x}^1 = \lambda_1\mathbf{x}^1$, $\|\mathbf{x}^1\|_2 = 1$, bekannt.

Es sei

$$\mathbf{B} := \mathbf{A} - \lambda_1\mathbf{x}^1(\mathbf{x}^1)^H.$$

Ergänzt man \mathbf{x}^1 zu einem Orthonormalsystem $\mathbf{x}^1, \dots, \mathbf{x}^n$ von Eigenvektoren von \mathbf{A} , so gilt

$$\mathbf{B}\mathbf{x}^i = \lambda_i \mathbf{x}^i, \quad i = 2, \dots, n, \quad \mathbf{B}\mathbf{x}^1 = \mathbf{0}.$$

Es bleiben also alle Eigenwerte und Eigenvektoren von \mathbf{A} erhalten, nur λ_1 wird durch 0 ersetzt; genauer: Ist λ_1 ein k -facher Eigenwert von \mathbf{A} , so besitzt \mathbf{B} den $(k-1)$ -fachen Eigenwert λ_1 .

Deflation nach Wielandt: Diese Methode ist auf beliebige Matrizen anwendbar. Sei also $\mathbf{A} \in \mathbb{C}^{(n,n)}$ beliebig.

Ist \mathbf{x} ein Rechtseigenvektor von \mathbf{A} zum Eigenwert λ und \mathbf{y} ein Linkseigenvektor zum Eigenwert $\mu \neq \lambda$, so sind (vgl. Abschnitt 6.2) \mathbf{x} und \mathbf{y} orthogonal.

Es seien nun ein Eigenwert λ_1 von \mathbf{A} und ein zugehöriger (Rechts-) Eigenvektor \mathbf{x}^1 bekannt, und es sei $\mathbf{u} \in \mathbb{C}^n$ mit $\mathbf{u}^H \mathbf{x}^1 \neq 0$. Dann gilt für die Matrix $\mathbf{B} := \mathbf{A} - \mathbf{x}^1 \mathbf{u}^H$

$$\mathbf{B}\mathbf{x}^1 = (\lambda_1 - \mathbf{u}^H \mathbf{x}^1) \mathbf{x}^1.$$

Ist $\lambda \neq \lambda_1$ ein Eigenwert von \mathbf{A} mit Linkseigenvektor \mathbf{y} , so gilt

$$\mathbf{y}^H \mathbf{B} = \mathbf{y}^H \mathbf{A} - \mathbf{y}^H \mathbf{x}^1 \mathbf{u}^H = \lambda \mathbf{y}^H.$$

Die von λ_1 verschiedenen Eigenwerte von \mathbf{A} bleiben erhalten (es ändern sich nur die Rechtseigenvektoren), während man durch Wahl von \mathbf{u} den Eigenwert λ_1 an jede gewünschte Stelle transportieren kann.

Deflation durch Ähnlichkeitstransformation: Sei $\mathbf{A} \in \mathbb{C}^{(n,n)}$ beliebig und \mathbf{x}^1 und λ_1 wie oben gegeben. Wir bestimmen eine nichtsinguläre Matrix \mathbf{H} mit $\mathbf{H}\mathbf{x}^1 = k\mathbf{e}^1$ (z.B. als Householder Transformation). Dann gilt

$$\mathbf{H}\mathbf{A}\mathbf{H}^{-1}\mathbf{H}\mathbf{x}^1 = \lambda_1 \mathbf{H}\mathbf{x}^1, \quad \text{d.h.} \quad \mathbf{H}\mathbf{A}\mathbf{H}^{-1}\mathbf{e}^1 = \lambda_1 \mathbf{e}^1,$$

und $\mathbf{H}\mathbf{A}\mathbf{H}^{-1}$ besitzt die Gestalt

$$\mathbf{H}\mathbf{A}\mathbf{H}^{-1} = \begin{pmatrix} \lambda_1 & \mathbf{b}^H \\ \mathbf{0} & \mathbf{B} \end{pmatrix},$$

wobei \mathbf{B} dieselben Eigenwerte wie \mathbf{A} besitzt und die Vielfachheit von λ_1 um 1 reduziert worden ist.

6.3.4 Unterraum Iteration

Führt man die Potenzmethode simultan mit $p > 1$ Vektoren $\mathbf{y}^1, \dots, \mathbf{y}^p$ aus, so werden (wenn \mathbf{A} einen dominanten Eigenwert besitzt) die Folgen $\mathbf{A}^m \mathbf{y}^1, \dots, \mathbf{A}^m \mathbf{y}^p$ alle gegen die Eigenrichtung zu dem dominanten Eigenwert konvergieren und mehr und mehr parallel werden. Wir verhindern dies, indem wir die Iterierten nicht nur normieren, sondern auch die Orthogonalität in jedem Schritt erzwingen. Das Ergebnis ist die **Unterraum Iteration** oder **orthogonale Iteration**. Es sei $\mathbf{Y}_0 \in \mathbb{C}^{(n,p)}$ eine Matrix mit orthonormalen Spalten.

Algorithmus 6.26. (Unterraum Iteration)

```

for m=0,1,... until convergence do
  Z_{m+1}=AY_m;
  Bestimme eine obere Dreiecksmatrix R_{m+1} und
  eine Matrix Y_{m+1} mit orthonormalen Spalten mit
  Z_{m+1}=Y_{m+1}R_{m+1};
end

```

Wir skizzieren das Konvergenzverhalten der Unterraum Iteration und geben so einen Eindruck von diesem Verfahren (ohne die Schlüsse sauber auszuführen) für den Fall, dass die Matrix \mathbf{A} diagonalisierbar ist und dass für ihre Eigenwerte gilt:

$$|\lambda_1| \geq |\lambda_2| \geq \dots \geq |\lambda_p| > |\lambda_{p+1}| \geq \dots \geq |\lambda_n|.$$

Wir werden sehen, dass die Unterraumiteration der Grundform des QR Algorithmus äquivalent ist, und hierfür die Konvergenz nachweisen.

Es ist

$$\text{span} \{\mathbf{Y}_{i+1}\} = \text{span} \{\mathbf{Z}_{i+1}\} = \text{span} \{\mathbf{A}\mathbf{Y}_i\}.$$

Mit $\mathbf{A} = \mathbf{X}\mathbf{\Lambda}\mathbf{X}^{-1}$ gilt also

$$\text{span} \{\mathbf{Y}_i\} = \text{span} \{\mathbf{A}^i \mathbf{Y}_0\} = \text{span} \{\mathbf{X}\mathbf{\Lambda}^i \mathbf{X}^{-1} \mathbf{Y}_0\}$$

und

$$\mathbf{X}\mathbf{\Lambda}^i \mathbf{X}^{-1} \mathbf{Y}_0 = \lambda_p^i \mathbf{X} \text{diag} \left\{ \left(\frac{\lambda_1}{\lambda_p} \right)^i, \dots, \left(\frac{\lambda_{p-1}}{\lambda_p} \right)^i, 1, \left(\frac{\lambda_{p+1}}{\lambda_p} \right)^i, \dots, \left(\frac{\lambda_n}{\lambda_p} \right)^i \right\} \mathbf{X}^{-1} \mathbf{Y}_0.$$

Wir zerlegen die Matrizen $\mathbf{X}^{-1} \mathbf{Y}_0$ und \mathbf{X} gemäß

$$\mathbf{X}^{-1} \mathbf{Y}_0 = \begin{pmatrix} \mathbf{U} \\ \tilde{\mathbf{U}} \end{pmatrix}, \quad \mathbf{U} \in \mathbb{R}^{(p,p)}, \quad \tilde{\mathbf{U}} \in \mathbb{R}^{(n-p,p)}$$

und

$$\mathbf{X} = (\mathbf{X}_p, \tilde{\mathbf{X}}_p), \quad \mathbf{X}_p \in \mathbb{R}^{(n,p)}, \quad \tilde{\mathbf{X}}_p \in \mathbb{R}^{(n,n-p)},$$

und setzen

$$\mathbf{\Lambda}_1 = \text{diag} \left(\frac{\lambda_1}{\lambda_p}, \dots, \frac{\lambda_{p-1}}{\lambda_p}, 1 \right), \quad \mathbf{\Lambda}_2 = \text{diag} \left(\frac{\lambda_{p+1}}{\lambda_p}, \dots, \frac{\lambda_n}{\lambda_p} \right).$$

Dann gilt

$$\mathbf{X} \mathbf{\Lambda}^i \mathbf{X}^{-1} \mathbf{Y}_0 = \lambda_p^i (\mathbf{X}_p, \tilde{\mathbf{X}}_p) \begin{pmatrix} \mathbf{\Lambda}_1^i & \mathbf{O} \\ \mathbf{O} & \mathbf{\Lambda}_2^i \end{pmatrix} \begin{pmatrix} \mathbf{U} \\ \tilde{\mathbf{U}} \end{pmatrix} = \lambda_p^i (\mathbf{X}_p \mathbf{\Lambda}_1^i \mathbf{U} + \tilde{\mathbf{X}}_p \mathbf{\Lambda}_2^i \tilde{\mathbf{U}}).$$

Wegen

$$\left| \frac{\lambda_j}{\lambda_p} \right| \geq 1 \text{ für } j \leq p \quad \text{und} \quad \left| \frac{\lambda_j}{\lambda_p} \right| < 1 \text{ für } j > p$$

konvergiert $\mathbf{\Lambda}_2^i$ und damit auch der zweite Summand $\tilde{\mathbf{X}}_p \mathbf{\Lambda}_2^i \tilde{\mathbf{U}}$ gegen die Nullmatrix, nicht aber $\mathbf{\Lambda}_1^i$. Der erste Summand wird also die Oberhand gewinnen, jedenfalls wenn die Matrix \mathbf{U} den Rang p besitzt (Wie wir noch sehen werden, ist dies eine Verallgemeinerung der Annahme, dass der Startvektor bei der Potenzmethode eine Komponente in Richtung des Eigenvektors zum betragsmaximalen Eigenwert von \mathbf{A} hat). Auch wenn man keine Konvergenz der Matrizen

$$\mathbf{X}_p \mathbf{\Lambda}_1^i \mathbf{U} + \tilde{\mathbf{X}}_p \mathbf{\Lambda}_2^i \tilde{\mathbf{U}}$$

erwarten kann, da die führenden Diagonalelemente der Matrix $\mathbf{\Lambda}_1^i$ dem Betrage nach über alle Grenzen wachsen, kann man sich vorstellen, dass wegen

$$\text{span} \{ \mathbf{Y}_i \} = \text{span} \{ \mathbf{X} \mathbf{\Lambda}^i \mathbf{X}^{-1} \mathbf{Y}_0 \} = \text{span} \{ \mathbf{X}_p \mathbf{\Lambda}_1^i \mathbf{U} + \tilde{\mathbf{X}}_p \mathbf{\Lambda}_2^i \tilde{\mathbf{U}} \}$$

und $\text{span} \{ \mathbf{X}_p \mathbf{\Lambda}_1^i \mathbf{U} \} = \text{span} \{ \mathbf{X}_p \}$ für genügend große i die Spalten der Matrix \mathbf{Y}_i einer Basis des invarianten Unterraums von \mathbf{A} nahe kommen, der von den Eigenvektoren zu den p betragsmaximalen Eigenwerten aufgespannt wird.

Führt man die Unterraum Iteration nur mit den ersten $q < p$ Spalten von \mathbf{Y}_0 aus, so erhält man offensichtlich im i -ten Schritt genau die ersten q Spalten von \mathbf{Y}_i . Gilt also sogar

$$|\lambda_1| > |\lambda_2| > \dots > |\lambda_p| > |\lambda_{p+1}| \geq \dots \geq |\lambda_n|$$

und sind alle führenden Hauptuntermatrizen von \mathbf{U} regulär, so können wir für jedes q den ‘‘Konvergenzbeweis’’ für die ersten q Spalten von \mathbf{Y}_0 ausführen, und wir erhalten für jedes $q \leq p$ ‘‘Konvergenz’’ des von den ersten q Spalten von \mathbf{Y}_i aufgespannten Raumes gegen den invarianten Unterraum von \mathbf{A} , der durch die zu

$\lambda_1, \dots, \lambda_q$ gehörenden Eigenvektoren aufgespannt wird (Was es dabei heißt, dass ein Unterraum des \mathbb{C}^n gegen einen anderen Unterraum konvergiert, wollen wir hier nicht problematisieren).

Für den Fall $p = n$ und $\mathbf{Y}_0 = \mathbf{I}$ kann man schließlich erwarten, dass die orthogonale Iteration für genügend großes i eine unitäre Matrix \mathbf{Y}_i mit der folgenden Eigenschaft liefert: Für alle $q = 1, \dots, n$ bilden die ersten q Spalten eine approximative Basis des Raumes, der durch die Eigenvektoren zu den q betragsgrößten Eigenwerten aufgespannt wird. Dass dies tatsächlich der Fall ist, wenn alle Eigenwerte von \mathbf{A} dem Betrage nach verschieden sind und alle führenden Hauptuntermatrizen der Matrix \mathbf{X}^{-1} regulär sind, zeigen wir im nächsten Abschnitt.

6.4 Der QR Algorithmus

Der QR Algorithmus dient dazu, alle Eigenwerte einer Matrix $\mathbf{A} \in \mathbb{C}^{(n,n)}$ zu bestimmen. Er wurde von Francis [40], [41] und Kublanovskaya [68] unabhängig voneinander im Jahr 1961 eingeführt. Er wird heute als das beste Verfahren zur Lösung der vollständigen Eigenwertaufgabe für nichtsymmetrische Probleme angesehen. Seine Grundform haben wir schon im letzten Abschnitt als orthogonale Iteration für den Fall $p = n$ betrachtet. In diesem Abschnitt werden wir Modifikationen angeben, durch die der QR Algorithmus wesentlich beschleunigt wird.

Es sei $\mathbf{A}_0 := \mathbf{A}$. Die Grundform des QR Algorithmus ist gegeben durch

Algorithmus 6.27. (QR Algorithmus; Grundform)

```

for i=0,1,... until convergence do
  Zerlege  $\mathbf{A}_i = \mathbf{Q}_i \mathbf{R}_i$ ;                (QR Zerlegung)
   $\mathbf{A}_{i+1} = \mathbf{R}_i \mathbf{Q}_i$ ;
end

```

Die durch den QR Algorithmus erzeugten Matrizen \mathbf{A}_i sind einander ähnlich und damit ähnlich zu \mathbf{A} , denn es gilt

$$\mathbf{A}_{i+1} = \mathbf{R}_i \mathbf{Q}_i = \mathbf{Q}_i^H (\mathbf{Q}_i \mathbf{R}_i) \mathbf{Q}_i = \mathbf{Q}_i^H \mathbf{A}_i \mathbf{Q}_i.$$

Um zu zeigen, dass der QR Algorithmus mit der orthogonalen Iteration übereinstimmt, zeigen wir zunächst:

Lemma 6.28. *Es sei $\mathbf{A} \in \mathbb{C}^{(n,n)}$ regulär. Dann gibt es genau eine unitäre Matrix \mathbf{Q} und eine obere Dreiecksmatrix \mathbf{R} mit positiven Diagonalelementen, so dass $\mathbf{A} = \mathbf{Q}\mathbf{R}$ gilt.*

Beweis: Die Matrix $\mathbf{A}^H \mathbf{A}$ ist positiv definit. Daher besitzt sie eine eindeutige Cholesky Zerlegung $\mathbf{A}^H \mathbf{A} = \mathbf{R}^H \mathbf{R}$ mit einer oberen Dreiecksmatrix \mathbf{R} mit positiven Diagonalelementen. Es sei hiermit $\mathbf{Q} := \mathbf{A}\mathbf{R}^{-1}$. Dann gilt

$$\mathbf{Q}^H \mathbf{Q} = (\mathbf{R}^H)^{-1} \mathbf{A}^H \mathbf{A} \mathbf{R}^{-1} = (\mathbf{R}^H)^{-1} \mathbf{R}^H \mathbf{R} \mathbf{R}^{-1} = \mathbf{I},$$

d.h. \mathbf{Q} ist unitär.

Sei $\mathbf{A} = \tilde{\mathbf{Q}}\tilde{\mathbf{R}}$ eine weitere Zerlegung mit den angegebenen Eigenschaften. Dann gilt

$$\mathbf{A}^H \mathbf{A} = \tilde{\mathbf{R}}^H \tilde{\mathbf{Q}}^H \tilde{\mathbf{Q}} \tilde{\mathbf{R}} = \tilde{\mathbf{R}}^H \tilde{\mathbf{R}},$$

und wegen der Eindeutigkeit der Cholesky Zerlegung folgt $\mathbf{R} = \tilde{\mathbf{R}}$, und dann $\tilde{\mathbf{Q}} = \mathbf{A}\tilde{\mathbf{R}}^{-1} = \mathbf{A}\mathbf{R}^{-1} = \mathbf{Q}$. ■

Satz 6.29. *Die QR Zerlegungen in Algorithmus 6.26. und Algorithmus 6.27. seien so bestimmt, dass die Diagonalelemente der Dreiecksmatrizen jeweils positiv sind.*

Es sei \mathbf{A}_i mit Algorithmus 6.27. bestimmt. Dann gilt

$$\mathbf{A}_i = \mathbf{Y}_i^H \mathbf{A} \mathbf{Y}_i, \quad (6.10)$$

wenn die \mathbf{Y}_i mit der orthogonalen Iteration mit der Startmatrix $\mathbf{Y}_0 = \mathbf{I}$ berechnet werden.

Beweis: Für $i = 0$ ist die Aussage trivial. Es gelte (6.10) für ein $i \geq 0$. Nach Algorithmus 6.26. gilt $\mathbf{A}\mathbf{Y}_i = \mathbf{Y}_{i+1}\mathbf{R}_{i+1}$, wobei \mathbf{Y}_{i+1} unitär ist und \mathbf{R}_{i+1} eine obere Dreiecksmatrix mit positiver Diagonale. Dann ist

$$\mathbf{Y}_i^H \mathbf{A} \mathbf{Y}_i = \mathbf{Y}_i^H (\mathbf{Y}_{i+1} \mathbf{R}_{i+1}) \quad (6.11)$$

das Produkt der unitären Matrix $\mathbf{Q} = \mathbf{Y}_i^H \mathbf{Y}_{i+1}$ und der oberen Dreiecksmatrix

$$\mathbf{R} = \mathbf{R}_{i+1} = \mathbf{Y}_{i+1}^H \mathbf{A} \mathbf{Y}_i.$$

(6.11) ist also die QR Zerlegung von \mathbf{A}_i , und es folgt

$$\mathbf{A}_{i+1} = \mathbf{R}\mathbf{Q} = (\mathbf{Y}_{i+1}^H \mathbf{A} \mathbf{Y}_i)(\mathbf{Y}_i^H \mathbf{Y}_{i+1}) = \mathbf{Y}_{i+1}^H \mathbf{A} \mathbf{Y}_{i+1}. \quad \blacksquare$$

Bevor wir die Konvergenz der Grundform des QR Algorithmus beweisen, benötigen wir zunächst zwei einfache Beziehungen. Es sei

$$U_i := Q_1 Q_2 \cdots Q_i \quad \text{und} \quad S_i := R_i R_{i-1} \cdots R_1.$$

Dann folgt aus $A_{i+1} = Q_i^H A_i Q_i$, $i \geq 0$, durch Induktion

$$A_{i+1} = U_i^H A U_i, \quad i = 1, 2, \dots \quad (6.12)$$

Ferner gilt

$$A^i = U_i S_i, \quad (6.13)$$

denn für $i = 1$ ist dies gerade $A = Q_1 R_1$, und ist (6.13) für ein i bewiesen, so folgt

$$A^{i+1} = A A^i = A U_i S_i = U_i A_{i+1} S_i = U_i Q_{i+1} R_{i+1} S_i = U_{i+1} S_{i+1}.$$

Satz 6.30. *Es seien die Eigenwerte von $A \in \mathbb{C}^{(n,n)}$ dem Betrage nach voneinander verschieden und angeordnet gemäß*

$$|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n| > 0. \quad (6.14)$$

Es sei $A = X \Lambda X^{-1}$, und es besitze die Matrix $Y := X^{-1}$ eine LR Zerlegung.

Dann konvergiert der QR Algorithmus im Wesentlichen, d.h. für $A_i := (a_{jk}^{(i)})_{j,k}$ gilt

$$\begin{aligned} \lim_{i \rightarrow \infty} a_{jk}^{(i)} &= 0 \quad \text{für } j > k \\ \lim_{i \rightarrow \infty} a_{kk}^{(i)} &= \lambda_k \quad \text{für } k = 1, \dots, n. \end{aligned}$$

Beweis: Sei $X = QR$ die QR Zerlegung von X mit $r_{ii} > 0$ und $Y = LU$ die LR Zerlegung von Y mit $\ell_{ii} = 1$.

Wegen $A = X \Lambda X^{-1} = Q R \Lambda R^{-1} Q^H$ gilt

$$Q^H A Q = R \Lambda R^{-1}, \quad (6.15)$$

d.h. $Q^H A Q$ ist obere Dreiecksmatrix mit den Diagonalelementen λ_i in der durch (6.14) gegebenen Reihenfolge.

Es ist

$$A^m = X \Lambda^m X^{-1} = Q R \Lambda^m L U = Q R \Lambda^m L \Lambda^{-m} \Lambda^m U.$$

Wegen

$$(\Lambda^m \mathbf{L} \Lambda^{-m})_{ij} = \ell_{ij} \left(\frac{\lambda_i}{\lambda_j} \right)^m = \begin{cases} 0 & \text{für } i < j \\ 1 & \text{für } i = j \\ \rightarrow 0 & \text{für } i > j \end{cases}$$

ist

$$\Lambda^m \mathbf{L} \Lambda^{-m} = \mathbf{I} + \mathbf{E}_m \quad \text{mit} \quad \lim_{m \rightarrow \infty} \mathbf{E}_m = \mathbf{O}.$$

Daher gilt

$$\begin{aligned} \mathbf{A}^m &= \mathbf{Q} \mathbf{R} (\mathbf{I} + \mathbf{E}_m) \Lambda^m \mathbf{U} = \mathbf{Q} (\mathbf{I} + \mathbf{R} \mathbf{E}_m \mathbf{R}^{-1}) \mathbf{R} \Lambda^m \mathbf{U} \\ &=: \mathbf{Q} (\mathbf{I} + \mathbf{F}_m) \mathbf{R} \Lambda^m \mathbf{U} \quad \text{mit} \quad \lim_{m \rightarrow \infty} \mathbf{F}_m = \mathbf{O}. \end{aligned}$$

Da die Cholesky Zerlegung stetig von den Matrixelementen abhängt, gilt dies auch (vgl. Lemma 6.28.) für \mathbf{Q} und \mathbf{R} in der QR Zerlegung mit positiver Diagonale in \mathbf{R} . Da $\mathbf{I} = \mathbf{I} \cdot \mathbf{I}$ die QR Zerlegung von \mathbf{I} ist, folgt also für die QR Zerlegung

$$\mathbf{I} + \mathbf{F}_m = \tilde{\mathbf{Q}}_m \tilde{\mathbf{R}}_m, \quad \tilde{\mathbf{R}}_m \quad \text{mit positiver Diagonale,}$$

$\tilde{\mathbf{Q}}_m \rightarrow \mathbf{I}$ und $\tilde{\mathbf{R}}_m \rightarrow \mathbf{I}$ wegen $\mathbf{F}_m \rightarrow \mathbf{O}$.

Wegen (6.13) ist

$$\mathbf{A}^m = (\mathbf{Q} \tilde{\mathbf{Q}}_m) (\tilde{\mathbf{R}}_m \mathbf{R} \Lambda^m \mathbf{U}) = \mathbf{U}_m \mathbf{S}_m,$$

und da die QR Zerlegung bis auf Multiplikation mit einer unitären Diagonalmatrix eindeutig ist, folgt hieraus: Es existieren unitäre Diagonalmatrizen \mathbf{D}_m mit

$$\mathbf{U}_m \mathbf{D}_m = \mathbf{Q} \tilde{\mathbf{Q}}_m \rightarrow \mathbf{Q},$$

und daher folgt aus (6.12)

$$\mathbf{D}_m^H \mathbf{A}_{m+1} \mathbf{D}_m = \mathbf{D}_m^H \mathbf{U}_m^H \mathbf{A} \mathbf{U}_m \mathbf{D}_m \rightarrow \mathbf{Q}^H \mathbf{A} \mathbf{Q} = \mathbf{R} \Lambda \mathbf{R}^{-1} \quad (6.16)$$

d.h.

$$\lim_{m \rightarrow \infty} a_{ij}^{(m+1)} \frac{d_j^{(m)}}{d_i^{(m)}} = \begin{cases} \lambda_i & \text{falls } i = j \\ 0 & \text{falls } i > j \end{cases}$$

wegen $|d_i^{(m)}| = 1$ für alle i also

$$\lim_{m \rightarrow \infty} a_{ij}^{(m)} = \begin{cases} \lambda_i & \text{falls } i = j \\ 0 & \text{falls } i > j \end{cases} \quad \blacksquare$$

Bemerkung 6.31. Am Ende sind die Eigenwerte von \mathbf{A} in \mathbf{A}_i der Größe der Beträge nach geordnet. \square

Beispiel 6.32. Die Matrix

$$\mathbf{A} = \begin{pmatrix} 1 & -1 & -1 \\ 4 & 6 & 3 \\ -4 & -4 & -1 \end{pmatrix}$$

besitzt die Eigenwerte $\lambda_1 = 3$, $\lambda_2 = 2$ und $\lambda_3 = 1$, und für die Matrix der Eigenvektoren gilt (bis auf die Normierung)

$$\mathbf{X} = \begin{pmatrix} 0 & -1 & -1 \\ 1 & 1 & 2 \\ -1 & 0 & -2 \end{pmatrix} \quad \text{und} \quad \mathbf{X}^{-1} = \begin{pmatrix} 2 & 2 & 1 \\ 0 & 1 & 1 \\ -1 & -1 & -1 \end{pmatrix}.$$

Die Voraussetzungen von Satz 6.30. sind also erfüllt. Man erhält

$$\mathbf{A}_{10} = \begin{pmatrix} 3.000034 & -0.577363 & 8.981447 \\ 9.78e-6 & 1.999995 & 1.4142593 \\ -6.91e-6 & 3.99e-6 & 0.999972 \end{pmatrix} \quad \square$$

Bemerkung 6.33. Wegen (6.14) ist \mathbf{A} diagonalisierbar, d.h. \mathbf{X} und damit \mathbf{Y} existieren. Die einzige Forderung in Satz 6.30. an die Matrix \mathbf{X} ist also, dass $\mathbf{Y} := \mathbf{X}^{-1}$ eine LR Zerlegung besitzt. Verzichtet man hierauf, so erhält man

$$\lim_{i \rightarrow \infty} a_{kk}^{(i)} = \lambda_{\pi(k)}$$

für eine Permutation π von $\{1, \dots, n\}$ (vgl. Wilkinson [121], p. 519 ff.). \square

Beispiel 6.34. Für die Matrix

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

mit den Eigenwerten $\lambda_1 = 3$, $\lambda_2 = 2$, $\lambda_3 = 1$ und den Eigenvektoren

$$\mathbf{X} = \begin{pmatrix} -1 & -1 & -1 \\ 2 & 1 & 1 \\ -2 & -1 & 0 \end{pmatrix} \quad \text{und} \quad \mathbf{X}^{-1} = \begin{pmatrix} 1 & 1 & 0 \\ -2 & -2 & -1 \\ 0 & 1 & 1 \end{pmatrix}.$$

ist die Voraussetzung von Satz 6.30. nicht erfüllt. Die Matrix $\mathbf{X}^{-1}(1 : 2, 1 : 2)$ ist singulär. Die orthogonale Iteration liefert nach 30 Schritten

$$\mathbf{A}_{30} = \begin{pmatrix} 3.0000058 & -2.0000014 & 2.9999975 \\ 1.16e-6 & 0.9999989 & -0.9999978 \\ -1.16e-6 & 6.69e-5 & 1.9999953 \end{pmatrix}.$$

Die Diagonalelemente scheinen also gegen die Eigenwerte zu konvergieren, wobei sie allerdings nicht nach der Betragsgöße geordnet sind. Diese Konvergenz würde auch bei exakter Rechnung eintreten (vgl. Bemerkung 6.33.). Iteriert man weiter, so erhält man (durch Rundungsfehler)

$$\mathbf{A}_{70} = \begin{pmatrix} 3 + 5.2e - 13 & -3.5355303 & 0.7071247 \\ 1.48e - 13 & 1.9999949 & -1.0000051 \\ -7.52e - 19 & -5.07e - 6 & 1.0000051 \end{pmatrix},$$

also doch noch Konvergenz gegen eine Dreiecksmatrix mit der Betragsgröße nach geordneten Eigenwerten. \square

Die Voraussetzung, dass \mathbf{Y} eine LR Zerlegung besitzt, ist natürlich, denn es gilt

Lemma 6.35. *Sei $\mathbf{X} \in \mathbb{C}^{(n,n)}$ regulär. $\mathbf{Y} = \mathbf{X}^{-1}$ besitzt genau dann eine LR Zerlegung, wenn*

$$\text{span}\{\mathbf{e}^1, \dots, \mathbf{e}^k\} \cap \text{span}\{\mathbf{x}^{k+1}, \dots, \mathbf{x}^n\} = \{\mathbf{0}\}, \quad k = 1, \dots, n-1 \quad (6.17)$$

Beweis: Zunächst gilt

$$\det \begin{pmatrix} y_{11} & \cdots & y_{1k} \\ \cdots & \cdots & \cdots \\ y_{k1} & \cdots & y_{kk} \end{pmatrix} = 0 \iff \det \begin{pmatrix} x_{k+1,k+1} & \cdots & x_{k+1,n} \\ \cdots & \cdots & \cdots \\ x_{n,k+1} & \cdots & x_{nn} \end{pmatrix} = 0, \quad (6.18)$$

denn sei

$$\mathbf{X} = \begin{pmatrix} \mathbf{X}_{11} & \mathbf{X}_{12} \\ \mathbf{X}_{21} & \mathbf{X}_{22} \end{pmatrix}, \mathbf{Y} = \begin{pmatrix} \mathbf{Y}_{11} & \mathbf{Y}_{12} \\ \mathbf{Y}_{21} & \mathbf{Y}_{22} \end{pmatrix} \quad \text{mit } \mathbf{X}_{11}, \mathbf{Y}_{11} \in \mathbb{C}^{(k,k)}.$$

Dann gilt

$$\mathbf{X}\mathbf{Y} = \begin{pmatrix} \mathbf{X}_{11}\mathbf{Y}_{11} + \mathbf{X}_{12}\mathbf{Y}_{21} & \mathbf{X}_{11}\mathbf{Y}_{12} + \mathbf{X}_{12}\mathbf{Y}_{22} \\ \mathbf{X}_{21}\mathbf{Y}_{11} + \mathbf{X}_{22}\mathbf{Y}_{21} & \mathbf{X}_{21}\mathbf{Y}_{12} + \mathbf{X}_{22}\mathbf{Y}_{22} \end{pmatrix} = \begin{pmatrix} \mathbf{I}_k & \mathbf{O} \\ \mathbf{O} & \mathbf{I}_{n-k} \end{pmatrix}.$$

Aus der Existenz von \mathbf{X}_{22}^{-1} folgt mit dem Block an der Stelle (2,1) zunächst $\mathbf{Y}_{21} = -\mathbf{X}_{22}^{-1}\mathbf{X}_{21}\mathbf{Y}_{11}$, und dann aus dem Block an der Stelle (1,1)

$$(\mathbf{X}_{11} - \mathbf{X}_{12}\mathbf{X}_{22}^{-1}\mathbf{X}_{21})\mathbf{Y}_{11} = \mathbf{I}_k.$$

Die Rangformel liefert also, dass auch \mathbf{Y}_{11}^{-1} existiert.

Umgekehrt erhält man genauso aus den Blöcken an den Stellen (2,1) und (2,2), dass aus der Existenz von \mathbf{Y}_{11}^{-1} auch die Existenz von \mathbf{X}_{22}^{-1} folgt.

(6.17) ist nun genau dann verletzt, wenn ein $\mathbf{a} = (\alpha_1, \dots, \alpha_n)^T \neq \mathbf{0}$ existiert mit

$$\sum_{i=1}^k \alpha_i \mathbf{e}^i = \sum_{i=k+1}^n \alpha_i \mathbf{x}^i,$$

und dies ist genau dann der Fall, wenn das Gleichungssystem

$$\begin{pmatrix} x_{k+1,k+1} & \cdots & x_{k+1,n} \\ \cdots & \cdots & \cdots \\ x_{n,k+1} & \cdots & x_{n,k+1} \end{pmatrix} \begin{pmatrix} \alpha_{k+1} \\ \vdots \\ \alpha_n \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}$$

nichttrivial lösbar ist. Nach (6.18) ist hierzu äquivalent

$$\det \begin{pmatrix} y_{11} & \cdots & y_{1k} \\ \cdots & \cdots & \cdots \\ y_{k1} & \cdots & y_{kk} \end{pmatrix} = 0,$$

und daher bricht der Gaußsche Algorithmus ohne Spaltenpivotsuche spätestens mit $y_{kk}^{(k)} = 0$ ab. ■

Da man den QR Algorithmus als Unterraum Iteration deuten kann mit den Startwerten $\mathbf{e}^1, \mathbf{e}^2, \dots, \mathbf{e}^n$, kann man ohne die Voraussetzung, dass \mathbf{Y} eine LR Zerlegung besitzt, nach Lemma 6.35. nicht erwarten, dass die ersten k Spalten von $\lim_{m \rightarrow \infty} \mathbf{U}_m$ eine approximative Basis des invarianten Unterraums $\text{span}\{\mathbf{x}^1, \dots, \mathbf{x}^k\}$ bilden.

Ist z.B. $y_{11} = 0$, so gilt $\mathbf{e}^1 \in \text{span}\{\mathbf{x}^2, \dots, \mathbf{x}^n\}$, und die Potenzmethode mit dem Startwert \mathbf{e}^1 wird nicht gegen die Richtung von \mathbf{x}^1 konvergieren. Ist die Untermatrix $(y_{ij})_{i,j=1,\dots,k}$ von \mathbf{Y} für ein $k \in \{2, \dots, n\}$ singular, so existiert ein $\mathbf{z} \in \text{span}\{\mathbf{e}^1, \dots, \mathbf{e}^k\} \cap \text{span}\{\mathbf{x}^{k+1}, \dots, \mathbf{x}^n\}$. Daher gilt $\mathbf{A}^m \mathbf{z} \in \text{span}\{\mathbf{x}^{k+1}, \dots, \mathbf{x}^n\}$, und $\text{span}\{\mathbf{A}^m \mathbf{e}^1, \dots, \mathbf{A}^m \mathbf{e}^k\}$ konvergiert nicht gegen den von $\mathbf{x}^1, \dots, \mathbf{x}^k$ aufgespannten invarianten Unterraum von \mathbf{A} .

Bemerkung 6.36. Man kann zeigen, dass

$$\lim_{m \rightarrow \infty} \mathbf{Q}_m = \text{diag} \left(\frac{\lambda_i}{|\lambda_i|} \right)$$

gilt, d.h. dass \mathbf{Q}_m i.A. nicht gegen \mathbf{I} konvergiert. Damit konvergiert \mathbf{U}_m überhaupt nicht und daher auch \mathbf{D}_m nicht gegen \mathbf{I} .

Aus (6.16) folgt, dass die Elemente von \mathbf{A}_m oberhalb der Diagonale oszillieren und nur dem Betrage nach konvergieren. Dies erklärt die Formulierung “der QR-Algorithmus konvergiert im Wesentlichen”. □

Bemerkung 6.37. Ist $\mathbf{A} \in \mathbb{R}^{(n,n)}$, so gilt $\mathbf{Q}_k \in \mathbb{R}^{(n,n)}$, $\mathbf{R}_k \in \mathbb{R}^{(n,n)}$ für alle k , und der ganze Algorithmus verläuft in \mathbb{R} . Sind alle Eigenwerte betragsmäßig verschieden, so sind sie alle reell, und der QR-Algorithmus konvergiert. Besitzt \mathbf{A} jedoch komplexe Eigenwerte, so kann die Grundform nicht konvergieren.

Ist $\mathbf{A} \in \mathbb{C}^{(n,n)}$ Hermitesch, so sind alle \mathbf{A}_m Hermitesch (vgl. (6.12)), und \mathbf{A}_m konvergiert unter den Voraussetzungen von Satz 6.30. gegen eine Diagonalmatrix. \square

Ein Nachteil der Grundform des QR Algorithmus ist, dass sie sehr aufwändig ist. Ist \mathbf{A} voll besetzt, so benötigt man in jedem Schritt $O(n^3)$ Operationen zur Bestimmung der QR Zerlegung. Dies wird im nächsten Unterabschnitt verbessert.

6.4.1 Beschleunigung des QR Algorithmus, explizite Shifts

Wir beschleunigen nun den QR Algorithmus auf zwei Weisen. Erstens erhöhen wir mit Hilfe von Shifts die Konvergenzgeschwindigkeit, und zweitens zeigen wir, dass die Hessenberg Gestalt einer Matrix im Laufe eines QR Schritts erhalten bleibt. Da die QR Zerlegung einer Hessenberg Matrix nur $O(n^2)$ Operationen benötigt, werden wir Matrizen vor Anwendung des QR Algorithmus zunächst unitär auf Hessenberg Gestalt transformieren.

Sei $\mathbf{A}_1 := \mathbf{A}$. Wir betrachten dann die geschiftete Version des QR Algorithmus:

Algorithmus 6.38. (QR Algorithmus mit Shifts)

```

for i=0,1,... until convergence do
  Wähle geeigneten Shift kappa_i;
  Zerlege  $\mathbf{A}_i - \kappa_i \mathbf{I} = \mathbf{Q}_i \mathbf{R}_i$ ;
   $\mathbf{A}_{i+1} = \mathbf{R}_i \mathbf{Q}_i + \kappa_i \mathbf{I}$ ;
end

```

Für die hierdurch erzeugten Matrizen gilt

$$\mathbf{R}_i = \mathbf{Q}_i^H (\mathbf{A}_i - \kappa_i \mathbf{I}),$$

und

$$\mathbf{A}_{i+1} = \mathbf{Q}_i^H (\mathbf{A}_i - \kappa_i \mathbf{I}) \mathbf{Q}_i + \kappa_i \mathbf{I} = \mathbf{Q}_i^H \mathbf{A}_i \mathbf{Q}_i. \quad (6.19)$$

Die \mathbf{A}_i sind also alle der Matrix \mathbf{A} ähnlich und haben dieselben Eigenwerte. Um die Parameter κ_i geeignet zu wählen, überlegen wir uns einen Zusammenhang von Algorithmus 6.38. und der inversen Iteration.

Satz 6.39. *Es seien*

$$\mathbf{U}_i := \mathbf{Q}_1 \mathbf{Q}_2 \dots \mathbf{Q}_i, \quad \mathbf{S}_i := \mathbf{R}_i \mathbf{R}_{i-1} \dots \mathbf{R}_1.$$

Dann gilt

$$\mathbf{U}_i \mathbf{S}_i = (\mathbf{A} - \kappa_i \mathbf{I})(\mathbf{A} - \kappa_{i-1} \mathbf{I}) \dots (\mathbf{A} - \kappa_1 \mathbf{I}). \quad (6.20)$$

Beweis: Zunächst folgt aus (6.19) durch Induktion

$$\mathbf{A}_{i+1} = \mathbf{U}_i^H \mathbf{A} \mathbf{U}_i. \quad (6.21)$$

Für $i = 1$ besagt (6.20)

$$\mathbf{U}_1 \mathbf{S}_1 = \mathbf{Q}_1 \mathbf{R}_1 = \mathbf{A} - \kappa_1 \mathbf{I},$$

und dies ist gerade die Zerlegung im ersten Schritt von Algorithmus 6.38.

Sei (6.20) bereits für $i - 1$ bewiesen. Dann folgt aus der Definition von \mathbf{A}_{i+1}

$$\mathbf{R}_i = (\mathbf{A}_{i+1} - \kappa_i \mathbf{I}) \mathbf{Q}_i^H = \mathbf{U}_i^H (\mathbf{A} - \kappa_i \mathbf{I}) \mathbf{U}_i \mathbf{Q}_i^H = \mathbf{U}_i^H (\mathbf{A} - \kappa_i \mathbf{I}) \mathbf{U}_{i-1},$$

und hieraus erhält man durch Rechtsmultiplikation mit \mathbf{S}_{i-1} und Linksmultiplikation mit \mathbf{U}_i

$$\mathbf{U}_i \mathbf{S}_i = (\mathbf{A} - \kappa_i \mathbf{I}) \mathbf{U}_{i-1} \mathbf{S}_{i-1} = (\mathbf{A} - \kappa_i \mathbf{I})(\mathbf{A} - \kappa_{i-1} \mathbf{I}) \dots (\mathbf{A} - \kappa_1 \mathbf{I})$$

nach Induktionsannahme. ■

Aus der Darstellung (6.20) folgt sofort

$$(\mathbf{A}^H - \bar{\kappa}_i \mathbf{I})^{-1} \dots (\mathbf{A}^H - \bar{\kappa}_1 \mathbf{I})^{-1} \mathbf{e}^n = \mathbf{U}_i (\mathbf{S}_i^H)^{-1} \mathbf{e}^n.$$

Da mit \mathbf{S}_i^H auch $(\mathbf{S}_i^H)^{-1}$ eine untere Dreiecksmatrix ist, gilt

$$\mathbf{U}_i (\mathbf{S}_i^H)^{-1} \mathbf{e}^n = \sigma_i \mathbf{U}_i \mathbf{e}^n$$

für ein $\sigma_i \in \mathbb{C}$, d.h. die letzte Spalte von \mathbf{U}_i ist eine Näherung für einen Eigenvektor von \mathbf{A}^H , die man ausgehend von \mathbf{e}^n mit der inversen Iteration mit den Shifts $\bar{\kappa}_1, \dots, \bar{\kappa}_i$ erhält.

Man kann also schnelle Konvergenz erwarten, wenn man $\bar{\kappa}_i$ als Rayleighquotienten von \mathbf{A}^H an der Stelle \mathbf{u}_n^{i-1} , der letzten Spalte von \mathbf{U}_{i-1} , wählt. Es ist

$$\bar{\kappa}_i = (\mathbf{u}_n^{i-1})^H \mathbf{A}^H \mathbf{u}_n^{i-1} = \overline{(\mathbf{u}_n^{i-1})^H \mathbf{A} \mathbf{u}_n^{i-1}} \stackrel{(6.21)}{=} \overline{(\mathbf{e}^n)^T \mathbf{A}_i \mathbf{e}^n} =: \overline{a_{nn}^{(i)}},$$

d.h.

$$\kappa_i = a_{nn}^{(i)}.$$

Dieser Shift heißt Rayleigh Quotienten Shift.

Eine weitere Verbesserung des Verfahrens (Verkleinerung des Aufwandes) erhält man, wenn man \mathbf{A} zunächst unitär auf obere Hessenberg Gestalt transformiert. Dabei sagt man, dass eine Matrix \mathbf{A} **Hessenberg Gestalt** hat, wenn

$$a_{ij} = 0 \quad \text{für alle } i > j + 1.$$

Diese Gestalt bleibt nämlich, wie wir noch sehen werden, während des gesamten QR Algorithmus erhalten. Wir bestimmen daher eine unitäre Matrix \mathbf{U} , so dass $\mathbf{U}^H \mathbf{A} \mathbf{U}$ obere Hessenberg Gestalt hat.

\mathbf{U} können wir als Produkt von $n - 2$ Spiegelungen der Gestalt $\mathbf{I} - 2\mathbf{w}\mathbf{w}^H$ aufbauen: Sei

$$\mathbf{A} = \begin{pmatrix} a_{11} & \mathbf{c}^T \\ \mathbf{b} & \mathbf{B} \end{pmatrix} \quad \text{mit } \mathbf{b} \neq \mathbf{0}.$$

Wir bestimmen dann $\mathbf{w} \in \mathbb{C}^{n-1}$ mit $\|\mathbf{w}\|_2 = 1$ und

$$\mathbf{Q}_1 \mathbf{b} := (\mathbf{I}_{n-1} - 2\mathbf{w}\mathbf{w}^H) \mathbf{b} = k \mathbf{e}^1,$$

und hiermit $\mathbf{P}_1 = \begin{pmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{Q}_1 \end{pmatrix}$.

Dann gilt

$$\mathbf{A}_1 := \mathbf{P}_1 \mathbf{A} \mathbf{P}_1 = \left(\begin{array}{c|c} a_{11} & \mathbf{c}^T \mathbf{Q}_1 \\ \hline k & \\ 0 & \\ \vdots & \mathbf{Q}_1 \mathbf{B} \mathbf{Q}_1 \\ 0 & \end{array} \right).$$

Damit hat die erste Spalte schon die Gestalt, die wir in einer Hessenberg Matrix benötigen. Die Spalten $2, \dots, n - 2$ können wir dann genauso behandeln (vgl. die Vorgehensweise bei der QR Zerlegung einer Matrix).

Eine andere Methode, \mathbf{A} unitär auf Hessenberg Gestalt zu transformieren, baut \mathbf{U} als Produkt von $\frac{1}{2}(n-1)(n-2)$ ebenen Drehungen (oder Reflexionen) auf. Wir bestimmen dazu \mathbf{U}_{23} , so dass das Element $(3, 1)$ in $\mathbf{U}_{23} \mathbf{A}$ verschwindet. Multiplikation von rechts mit \mathbf{U}_{23}^H ändert dann nur die zweite und dritte Spalte, zerstört also nicht die bereits erzeugte 0 in der Position $(3, 1)$.

Zu $\mathbf{A}_1 := \mathbf{U}_{23}\mathbf{A}\mathbf{U}_{23}^H$ bestimmen wir dann \mathbf{U}_{24} , so dass $\mathbf{U}_{24}\mathbf{A}_1$ an der Stelle $(4, 1)$ eine Null hat. Diese (und auch die an der Stelle $(3, 1)$) bleibt erhalten bei der Multiplikation von rechts mit \mathbf{U}_{24}^H .

Man erhält eine Hessenberg Matrix, wenn man Matrizen \mathbf{U}_{ij} in der Reihenfolge

$$(i, j) = (2, 3), (2, 4), \dots, (2, n), (3, 4), (3, 4), \dots, (n-1, n)$$

wählt, so dass das Element an der Stelle

$$(3, 1), (4, 1), \dots, (n, 1), (4, 2), (5, 2), \dots, (n, n-1)$$

annulliert wird.

Sei $\mathbf{A} =: \mathbf{A}_1$ obere Hessenberg Matrix und $\kappa_1 \in \mathbb{C}$ ein Shift-Parameter, z.B. $\kappa_1 := a_{nn}^{(1)}$. Wir multiplizieren dann für $i = 1, \dots, n-1$ die Matrix

$$\mathbf{U}_{i-1,i} \dots \mathbf{U}_{12}(\mathbf{A} - \kappa_1 \mathbf{I})$$

mit einer ebenen Drehung $\mathbf{U}_{i,i+1}$, so dass das Element an der Stelle $(i+1, i)$ annulliert wird. Dann besitzt

$$\mathbf{R}_1 := \mathbf{U}_{n-1,n} \dots \mathbf{U}_{12}(\mathbf{A} - \kappa_1 \mathbf{I})$$

obere Dreiecksgestalt, und $\mathbf{Q}_1 := \mathbf{U}_{12}^H \dots \mathbf{U}_{n-1,n}^H$ ist der unitäre Anteil in der QR Zerlegung von $\mathbf{A} - \kappa_1 \mathbf{I}$ in Algorithmus 6.38. Die neue Iterierte \mathbf{A}_2 erhält man dann gemäß

$$\mathbf{A}_2 = \mathbf{R}_1 \mathbf{U}_{12}^H \dots \mathbf{U}_{n-1,n}^H + \kappa_1 \mathbf{I}.$$

Da die Multiplikation von rechts mit $\mathbf{U}_{i,i+1}^H$ nur die i -te und $(i+1)$ -te Spalte verändert, ist klar, dass \mathbf{A}_2 wieder obere Hessenberg Matrix ist. Die obere Hessenberg Gestalt der Matrizen bleibt also im Verlaufe des QR Algorithmus erhalten. Da ein QR Schritt für eine Hessenberg Matrix nur $O(n^2)$ Operationen erfordert, lohnt sich diese vorbereitende Transformation von \mathbf{A} .

Ist \mathbf{A} Hermitesch, so sind alle \mathbf{A}_i Hermitesch (vgl. (6.21)). Transformiert man also \mathbf{A} zunächst auf Tridiagonalgestalt, so bleiben alle \mathbf{A}_m wie eben tridiagonal, und man benötigt sogar in jedem QR Schritt nur $O(n)$ Operationen.

Die Matrizen $\mathbf{U}_{i,i+1}$ müssen nicht während des ganzen QR Schritts abgespeichert werden, denn die Multiplikation von links mit $\mathbf{U}_{i,i+1}$ ändert nur die Zeilen i und $i+1$. In $\mathbf{U}_{23}\mathbf{U}_{12}(\mathbf{A}_1 - \kappa_1 \mathbf{I})$ und \mathbf{R}_1 stimmen also bereits die beiden ersten Zeilen und Spalten (beachte die Hessenberg Gestalt) überein. Da die Multiplikation von

rechts mit U_{12}^H nur die Spalten 1 und 2 verändert, kann man die Transformation eines QR Schritts in folgender Reihenfolge durchführen ($\tilde{\mathbf{A}} := \mathbf{A}_1 - \kappa_1 \mathbf{I}$):

$$\begin{aligned} \tilde{\mathbf{A}} &\rightarrow U_{12} \tilde{\mathbf{A}} \rightarrow U_{23} U_{12} \tilde{\mathbf{A}} \rightarrow U_{23} U_{12} \tilde{\mathbf{A}} U_{12}^H \rightarrow U_{34} U_{23} U_{12} \tilde{\mathbf{A}} U_{12}^H \\ &\rightarrow U_{34} U_{23} U_{12} \tilde{\mathbf{A}} U_{12}^H U_{23}^H \rightarrow \dots \rightarrow U_{n-1,n} \dots U_{12} \tilde{\mathbf{A}} U_{12}^H \dots U_{n-1,n}^H \\ &=: \mathbf{A}_2 - \kappa_1 \mathbf{I}. \end{aligned}$$

Dies wird in dem folgenden Algorithmus, der einen QR Schritt mit Shift für eine Hessenberg Matrix beschreibt, bereits berücksichtigt. Wir verwenden darin die Prozedur $(\gamma, \sigma, \nu) = \text{rot}(\alpha, \beta)$, die bei gegebenen $\alpha, \beta \in \mathbb{C}$ die Parameter γ, σ, ν berechnet, so dass

$$\begin{pmatrix} \gamma & \sigma \\ -\sigma & \gamma \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} \nu \\ 0 \end{pmatrix} \text{ gilt.}$$

1. $\kappa := a_{nn}$
 $a_{11} := a_{11} - \kappa$
2. for $k = 1, \dots, n$ do
 - (i) If $k = n$, then goto (iv)
 - (ii) $(\gamma_k, \sigma_k, \nu_k) := \text{rot}(a_{kk}, a_{k+1,k})$
 $a_{kk} := \nu_k$
 $a_{k+1,k} := 0$
 $a_{k+1,k+1} := a_{k+1,k+1} - \kappa$
 for $j=k+1, \dots, n$ do
 $\begin{pmatrix} a_{kj} \\ a_{k+1,j} \end{pmatrix} := \begin{pmatrix} \gamma_k & \sigma_k \\ -\sigma_k & \gamma_k \end{pmatrix} \begin{pmatrix} a_{kj} \\ a_{k+1,j} \end{pmatrix}$
 end
 - (iii) If $k=1$, then step k
 - (iv) for $i=1, 2, \dots, k$ do
 $(a_{i,k-1}, a_{ik}) := (a_{i,k-1}, a_{ik}) \begin{pmatrix} \bar{\gamma}_{k-1} & -\bar{\sigma}_{k-1} \\ \bar{\sigma}_{k-1} & \bar{\gamma}_{k-1} \end{pmatrix},$
 $a_{k-1,k-1} := a_{k-1,k-1} + \kappa$
 end
3. $a_{nn} := a_{nn} + \kappa.$

Wendet man diesen Algorithmus wiederholt an, so wird $a_{n,n-1}^{(i)}$ rasch (quadratisch; vgl. Lemma 6.25.) gegen 0 gehen. Ist $|a_{n,n-1}^{(i)}|$ klein genug, so setzt man es gleich 0

und erhält eine Matrix der Gestalt

$$\mathbf{A}_i = \left(\begin{array}{ccccc|c} + & + & \dots & + & + & * \\ + & + & \dots & + & + & * \\ 0 & + & \dots & + & + & * \\ \dots & \dots & \dots & \dots & \dots & \vdots \\ 0 & 0 & \dots & + & + & * \\ \hline 0 & 0 & \dots & 0 & \approx 0 & * \end{array} \right) = \mathbf{U}_i^H \mathbf{A} \mathbf{U}_i.$$

$a_{nn}^{(i)}$ ist also eine Näherung für einen Eigenwert von \mathbf{A} (nicht notwendig wie in Satz 6.30. für den betragskleinsten Eigenwert, da die Shifts diese Eigenschaft zerstören), und die Eigenwerte der führenden $(n - 1, n - 1)$ Hauptuntermatrix (oben +) von \mathbf{A}_i sind Näherungen für die übrigen Eigenwerte von \mathbf{A} . Man kann also mit einer verkleinerten Matrix den Algorithmus fortsetzen.

Darüber hinaus gehen auch die übrigen Subdiagonalelemente von \mathbf{A}_i gegen 0 (vgl. Satz 6.30.). Ist eines dieser Elemente klein genug, etwa $a_{k+1,k}^{(i)} \approx 0$, so kann man offenbar zwei kleinere Hessenberg Matrizen

$$\begin{pmatrix} a_{11}^{(i)} & a_{12}^{(i)} & \dots & a_{1,k-1}^{(i)} & a_{1k}^{(i)} \\ a_{21}^{(i)} & a_{22}^{(i)} & \dots & a_{2,k-1}^{(i)} & a_{2k}^{(i)} \\ 0 & a_{32}^{(i)} & \dots & a_{3,k-1}^{(i)} & a_{3k}^{(i)} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & a_{k,k-1}^{(i)} & a_{kk}^{(i)} \end{pmatrix}, \begin{pmatrix} a_{k+1,k+1}^{(i)} & a_{k+1,k+2}^{(i)} & \dots & a_{k+1,n-1}^{(i)} & a_{k+1,n}^{(i)} \\ a_{k+2,k+1}^{(i)} & a_{k+2,k+2}^{(i)} & \dots & a_{k+2,n-1}^{(i)} & a_{k+2,n}^{(i)} \\ 0 & a_{k+3,k+2}^{(i)} & \dots & a_{k+3,n-1}^{(i)} & a_{k+3,n}^{(i)} \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & a_{n,n-1}^{(i)} & a_{nn}^{(i)} \end{pmatrix}$$

mit dem QR Algorithmus behandeln.

Als Kriterium für klein genug wählt man mit $\varepsilon = 10^{-t}$, wobei t die Zahl der signifikanten Stellen bezeichnet,

$$|a_{k+1,k}^{(i)}| \leq \varepsilon \min\{|a_{kk}^{(i)}|, |a_{k+1,k+1}^{(i)}|\}$$

oder (weniger einschneidend)

$$|a_{k+1,k}^{(i)}| \leq \varepsilon (|a_{kk}^{(i)}| + |a_{k+1,k+1}^{(i)}|).$$

Beispiel 6.40. Wir demonstrieren den Konvergenzverlauf für die Hessenberg Matrix

$$\mathbf{A} = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 1 & 0 \\ 0 & 3 & 2 & 1 \\ 0 & 0 & 2 & 4 \end{pmatrix} \in \mathbb{R}^{(4,4)}.$$

Die folgende Tabelle enthält die Subdiagonalelemente bei dem oben beschriebenen Abbruch mit $t = 16$:

| i | a_{21} | a_{32} | a_{43} |
|-----|---------------|---------------|---------------|
| 1 | 2 | 3 | 2 |
| 2 | $1.83e0$ | $-2.23e0$ | $1.61e0$ |
| 3 | $1.65e0$ | $1.43e0$ | $3.55e - 1$ |
| 4 | $6.78e - 1$ | $-3.65e0$ | $3.82e - 2$ |
| 5 | $7.63e - 1$ | $1.17e0$ | $-1.63e - 3$ |
| 6 | $8.32e - 1$ | $-5.32e - 1$ | $3.93e - 6$ |
| 7 | $-1.18e0$ | $1.52e - 1$ | $-3.03e - 11$ |
| 8 | $1.64e0$ | $-4.97e - 2$ | $1.62e - 21$ |
| 9 | $-3.22e0$ | $2.89e - 5$ | |
| 10 | $1.81e0$ | $5.33e - 10$ | |
| 11 | $-5.32e - 1$ | $-2.48e - 19$ | |
| 12 | $-4.46e - 3$ | | |
| 13 | $5.89e - 8$ | | |
| 14 | $-1.06e - 17$ | | |

Nach einer Anlaufphase wird die Zahl der gültigen Stellen des letzten berücksichtigten Diagonalelements von Schritt zu Schritt ungefähr verdoppelt. Dies zeigt die quadratische Konvergenz des Verfahrens.

Für die Grundform des QR Algorithmus wird diese Genauigkeit erst nach ca. 300 Schritten erreicht. \square

6.4.2 Implizite Shifts

Der Algorithmus des letzten Abschnitts zusammen mit der beschriebenen Deflationstechnik liefert ein gutes Verfahren zur Bestimmung aller Eigenwerte von \mathbf{A} . Es gibt jedoch eine Schwierigkeit. Ist \mathbf{A} reell, so ist $\kappa_1 = a_{nn}$ reell und alle Matrizen \mathbf{A}_i sowie alle Shiftparameter κ_i bleiben reell.

Da die inverse Iteration nur dann rasch konvergiert, wenn die Shifts gegen einen Eigenwert von \mathbf{A} konvergieren, kann der Algorithmus nicht effizient sein, wenn \mathbf{A} komplexe Eigenwerte besitzt.

Eine Möglichkeit, diese Schwierigkeit zu umgehen, ist, komplexe Shifts einzuführen und in komplexer Arithmetik zu rechnen. Eine andere (bessere) besteht darin, mit reellen Shifts zu arbeiten und eine Quasidreiecksgestalt (vgl. Satz 6.5.) herzustellen, und dann die komplexen Eigenwerte aus den $(2, 2)$ -Diagonalblöcken zu berechnen. Zu

diesem Zweck wird eine Variante des QR Algorithmus hergeleitet, bei der die Shifts nicht explizit durchgeführt werden. Diese Variante findet eine weitere Anwendung bei der Berechnung der Singulärwertzerlegung von Matrizen.

Definition 6.41. Sei \mathbf{B} eine obere Hessenberg Matrix. \mathbf{B} heißt **nichtreduziert**, falls $b_{i+1,i} \neq 0$ für $i = 1, \dots, n-1$ gilt.

Für den QR Algorithmus genügt es, sich mit nichtreduzierten Hessenberg Matrizen zu beschäftigen, da sonst eine Deflation durchgeführt werden kann.

Satz 6.42. Es seien $\mathbf{A}, \mathbf{B}, \mathbf{Q} \in \mathbb{C}^{(n,n)}$, \mathbf{Q} unitär und \mathbf{B} eine nichtreduzierte Hessenberg Matrix mit positiven Subdiagonalelementen $b_{k+1,k}$. Ist $\mathbf{B} = \mathbf{Q}^H \mathbf{A} \mathbf{Q}$, so sind \mathbf{B} und \mathbf{Q} eindeutig durch die erste Spalte von \mathbf{Q} festgelegt.

Beweis: Wir geben einen Algorithmus zur Berechnung von \mathbf{B} und \mathbf{Q} an.

Sei $\mathbf{Q} := (\mathbf{q}^1, \mathbf{q}^2, \dots, \mathbf{q}^n)$, und es seien bereits $\mathbf{q}^1, \dots, \mathbf{q}^k$ und die ersten $k-1$ Spalten von \mathbf{B} berechnet. Für $k=1$ ist \mathbf{q}^1 festgelegt, der Algorithmus kann also gestartet werden.

Wegen $\mathbf{Q} \mathbf{B} = \mathbf{A} \mathbf{Q}$ und, da \mathbf{B} obere Hessenberg Matrix ist, gilt

$$b_{k+1,k} \mathbf{q}^{k+1} + b_{k,k} \mathbf{q}^k + \dots + b_{1k} \mathbf{q}^1 = \mathbf{A} \mathbf{q}^k. \quad (6.22)$$

Multiplikation dieser Gleichung mit $(\mathbf{q}^i)^H$ liefert

$$b_{ik} = (\mathbf{q}^i)^H \mathbf{A} \mathbf{q}^k, \quad i = 1, \dots, k.$$

Hierdurch ist die k -te Spalte von \mathbf{B} außer $b_{k+1,k}$ festgelegt.

Wegen $b_{k+1,k} \neq 0$ gilt

$$\mathbf{q}^{k+1} = \frac{1}{b_{k+1,k}} \cdot \left(\mathbf{A} \mathbf{q}^k - \sum_{i=1}^k b_{ik} \mathbf{q}^i \right),$$

und aus $(\mathbf{q}^{k+1})^H \mathbf{q}^{k+1} = 1$ erhält man wegen der Positivität von $b_{k+1,k}$ auf eindeutige Weise $b_{k+1,k}$. ■

Bemerkung 6.43. Die Voraussetzung $b_{k+1,k} > 0$ wurde nur getroffen, um die Eindeutigkeit zu sichern. Anderenfalls sind \mathbf{q}^{k+1} und $b_{k+1,k}$ nur bis auf einen gemeinsamen Faktor vom Betrag 1 bestimmt.

Wir wollen nun Satz 6.42. auf den QR Algorithmus anwenden.

Satz 6.44. *Es sei \mathbf{A} eine nichtreduzierte obere Hessenberg Matrix, κ ein Shiftparameter,*

$$\mathbf{A} - \kappa \mathbf{I} = \mathbf{QR}, \quad \mathbf{B} = \mathbf{RQ} + \kappa \mathbf{I}$$

und \mathbf{B} nichtreduziert. Dann gilt (vgl. (6.19))

$$\mathbf{B} = \mathbf{Q}^H \mathbf{A} \mathbf{Q},$$

und man kann \mathbf{B} auch mit dem folgenden Algorithmus berechnen:

- (1) Bestimme eine unitäre Matrix $\mathbf{P} \in \mathbb{C}^{(n,n)}$, so dass die ersten Spalten von \mathbf{P}^H und \mathbf{Q} übereinstimmen.
- (2) Reduziere \mathbf{PAP}^H auf die obere Hessenberg Matrix $\tilde{\mathbf{B}}$.

Beweis: Sind $\mathbf{U}_1, \mathbf{U}_2, \dots, \mathbf{U}_{n-2}$ die Householder Transformationen zur Transformation von \mathbf{PAP}^H auf Hessenberg Gestalt (wie im letzten Unterabschnitt) und ist $\tilde{\mathbf{Q}}^H = \mathbf{U}_{n-2} \mathbf{U}_{n-3} \dots \mathbf{U}_1 \mathbf{P}$, so gilt $\tilde{\mathbf{B}} = \tilde{\mathbf{Q}}^H \mathbf{A} \tilde{\mathbf{Q}}$.

Wegen der speziellen Gestalt der Matrizen

$$\mathbf{U}_i = \begin{pmatrix} \mathbf{I}_i & \mathbf{O} \\ \mathbf{O} & \tilde{\mathbf{U}}_i \end{pmatrix}$$

verändert die Linksmultiplikation mit \mathbf{U}_i nicht die erste Zeile, d.h. $\tilde{\mathbf{Q}}^H$ und \mathbf{P} haben dieselbe erste Zeile, und \mathbf{P} hat dieselbe erste Zeile wie \mathbf{Q}^H . Nach Satz 6.42. gilt also $\mathbf{Q} = \tilde{\mathbf{Q}}$ und $\mathbf{B} = \tilde{\mathbf{B}}$. ■

Der obige Algorithmus konzentriert die Wirkung des Shifts κ in der Matrix \mathbf{P} . Nachdem \mathbf{PAP}^H berechnet worden ist, hat man nur noch mit einem Standardalgorithmus \mathbf{PAP}^H auf Hessenberg Gestalt zu transformieren.

Um \mathbf{P} zu bestimmen, haben wir die erste Spalte von \mathbf{Q} zu berechnen. Es ist $\mathbf{A} - \kappa \mathbf{I} = \mathbf{QR}$, wobei \mathbf{R} eine obere Dreiecksmatrix ist. Ist $\mathbf{Q} = (\mathbf{q}^1, \dots, \mathbf{q}^n)$, so gilt

$$r_{11} \mathbf{q}^1 = \mathbf{QR} \mathbf{e}^1 = (\mathbf{A} - \kappa \mathbf{I}) \mathbf{e}^1,$$

d.h. die erste Spalte von \mathbf{Q} , ist ein Vielfaches der ersten Spalte

$$\mathbf{a} = (a_{11} - \kappa, a_{21}, 0, \dots, 0)^T$$

von $\mathbf{A} - \kappa \mathbf{I}$.

Wählt man \mathbf{P} mit $\mathbf{P}\mathbf{a} = \pm \|\mathbf{a}\|_2 \mathbf{e}^1$, so gilt $\mathbf{P}^H \mathbf{e}^1 = \pm \frac{\mathbf{a}}{\|\mathbf{a}\|_2}$, und die ersten Spalten von \mathbf{P}^H und \mathbf{Q} stimmen überein. Man kann also \mathbf{P} als ebene Drehung wählen, die die zweite Komponente a_{21} annulliert.

Bildet man hiermit $\mathbf{P}\mathbf{A}\mathbf{P}^H$, so erhält man eine gestörte obere Hessenberg Matrix, in der zusätzlich das Element an der Stelle $(3, 1)$ besetzt ist. Durch Multiplikation von links mit einer Drehung \mathbf{U}_{23} kann man dieses annullieren, und die Multiplikation mit \mathbf{U}_{23}^H von rechts erzeugt ein von 0 verschiedenes Element an der Stelle $(4, 2)$.

Allgemein hat man im k -ten Schritt ein von 0 verschiedenes Element an der Stelle $(k+1, k-1)$, das durch eine Rotation $\mathbf{U}_{k,k+1}$ an die Stelle $(k+2, k)$ "gejagt" wird ("chasing the bulge").

Als Shift wählt man wie vorher den Rayleigh Quotienten Shift $\kappa = \alpha_n$ oder auch den **Wilkinson Shift**, d.h. κ als den Eigenwert von

$$\begin{pmatrix} a_{n-1,n-1}^{(i)} & a_{n-1,n}^{(i)} \\ a_{n,n-1}^{(i)} & a_{nn}^{(i)} \end{pmatrix},$$

der $a_{nn}^{(i)}$ am nächsten liegt. Ferner wird wie vorher mit Hilfe der Deflationen die Größe der behandelten Eigenwertprobleme verkleinert.

Wir beschreiben nun, wie man die impliziten Shifts verwenden kann, um komplexe Eigenwerte zu bestimmen. Es werden zwei QR Schritte mit den Shifts κ_0 und κ_1 zu einem Schritt zusammengefasst. Ist \mathbf{A} reelle obere Hessenberg Matrix und sind κ_0 und κ_1 konjugiert komplex, so kann man diesen Doppelschritt in reeller Arithmetik ausführen.

Der erste Schritt mit κ_0 werde ausgeführt und führe auf die Matrix

$$\mathbf{A}_1 = \mathbf{Q}_0^H \mathbf{A} \mathbf{Q}_0,$$

und hierauf werde der zweite Schritt mit Shift κ_1 angewendet und liefere

$$\mathbf{A}_2 = \mathbf{Q}_1^H \mathbf{A}_1 \mathbf{Q}_1 = \mathbf{Q}_1^H \mathbf{Q}_0^H \mathbf{A} \mathbf{Q}_0 \mathbf{Q}_1.$$

Dies kann man wie vorher mit dem folgenden Algorithmus ausführen:

- (1) Bestimme eine unitäre Matrix \mathbf{U} , die dieselbe erste Zeile wie $\mathbf{Q}_1^H \mathbf{Q}_0^H$ besitzt.
- (2) Transformiere $\mathbf{U}\mathbf{A}\mathbf{U}^H$ auf obere Hessenberg Gestalt \mathbf{A}_2 .

Wir bestimmen zunächst \mathbf{U} . Es seien $\mathbf{R}_0, \mathbf{R}_1$ die oberen Dreiecksanteile der QR Zerlegung von $\mathbf{A} - \kappa_0 \mathbf{I}$ bzw. $\mathbf{A}_1 - \kappa_1 \mathbf{I}$. Dann gilt nach Satz 6.39.

$$\mathbf{Q}_0 \mathbf{Q}_1 \mathbf{R}_1 \mathbf{R}_0 = (\mathbf{A} - \kappa_1 \mathbf{I})(\mathbf{A} - \kappa_0 \mathbf{I}).$$

Da $\mathbf{R}_1 \mathbf{R}_0$ eine obere Dreiecksmatrix ist, ist die erste Spalte von $\mathbf{Q}_0 \mathbf{Q}_1$ Vielfaches der ersten Spalte \mathbf{a} von $(\mathbf{A} - \kappa_1 \mathbf{I})(\mathbf{A} - \kappa_0 \mathbf{I})$. Wir können also \mathbf{U} als Householder-Matrix wählen, die \mathbf{a} in ein Vielfaches von \mathbf{e}^1 transformiert.

Da \mathbf{A} obere Hessenberg Matrix ist, sind nur die ersten drei Komponenten von \mathbf{a} von Null verschieden. Man rechnet leicht nach, dass gilt

$$\begin{aligned} a_1 &= a_{11}^2 - (\kappa_0 + \kappa_1)a_{11} + \kappa_0 \kappa_1 + a_{12}a_{21} \\ a_2 &= a_{21}(a_{11} + a_{22} - (\kappa_0 + \kappa_1)) \\ a_3 &= a_{21}a_{32}. \end{aligned}$$

Wir wählen nun κ_0 und κ_1 als die Eigenwerte von

$$\begin{pmatrix} a_{n-1,n-1} & a_{n-1,n} \\ a_{n,n-1} & a_{nn} \end{pmatrix}.$$

Dann gilt

$$\kappa_0 + \kappa_1 = a_{n-1,n-1} + a_{nn}, \quad \kappa_0 \kappa_1 = a_{nn}a_{n-1,n-1} - a_{n,n-1}a_{n-1,n}.$$

Hiermit erhält man für die a_i , $i = 1, 2, 3$

$$\begin{aligned} a_1 &= a_{21} \left(\frac{(a_{nn} - a_{11})(a_{n-1,n-1} - a_{11}) - a_{n-1,n}a_{n,n-1}}{a_{21}} + a_{12} \right) \\ a_2 &= a_{21}(a_{22} + a_{11} - a_{nn} - a_{n-1,n-1}) \\ a_3 &= a_{21}a_{32}. \end{aligned}$$

Da man nur an der Richtung von \mathbf{a} interessiert ist, kann man den gemeinsamen Faktor a_{21} fortlassen und \mathbf{U} bestimmen.

Die Matrix $\mathbf{U} \mathbf{A} \mathbf{U}^H$ besitzt die Gestalt

$$\begin{pmatrix} * & * & * & * & * & * & * & \dots \\ * & * & * & * & * & * & * & \dots \\ * & * & * & * & * & * & * & \dots \\ * & * & * & * & * & * & * & \dots \\ 0 & 0 & 0 & * & * & * & * & \dots \\ 0 & 0 & 0 & 0 & * & * & * & \dots \\ 0 & 0 & 0 & 0 & 0 & * & * & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \end{pmatrix},$$

ist also wieder eine gestörte obere Hessenberg Matrix, die "Beule" besteht aber nun aus 3 Elementen.

Ähnlich wie vorher kann man die Beule Schritt für Schritt nach rechts unten jagen, wobei man nun aber Matrizen

$$\mathbf{U}_i = \begin{pmatrix} \mathbf{I}_i & \mathbf{O} & \mathbf{O} \\ \mathbf{O} & \mathbf{H}_i & \mathbf{O} \\ \mathbf{O} & \mathbf{O} & \mathbf{I}_{n-i-3} \end{pmatrix}$$

mit einer Householder Matrix $\mathbf{H}_i \in \mathbb{R}^{(3,3)}$ verwendet. Im letzten Schritt muss man eine ebene Drehung $\mathbf{U}_{n-1,n}$ benutzen.

Mit diesem Verfahren (zusammen mit Deflation) kann man \mathbf{A} in eine Blocktridiagonalmatrix unitär transformieren, und hieraus leicht die Eigenwerte bestimmen.

Ist $\mathbf{A} \in \mathbb{R}^{(n,n)}$, so können die Shifts κ_0, κ_1 zwar komplex werden, es bleiben aber a_1, a_2, a_3 reell, und der Algorithmus kann in reeller Arithmetik durchgeführt werden.

Es wurde beobachtet, dass der Algorithmus i.A. quadratisch konvergiert. Diese Konvergenz kann jedoch nicht gezeigt werden, denn z.B. bleibt die Matrix

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

unverändert. In der Praxis führt man einen Schritt mit zufällig gewählten Shifts κ_0, κ_1 aus. Dies zerstört die spezielle Gestalt. Danach führt man den Algorithmus mit der oben besprochenen Shift-Strategie durch.

6.5 Der QZ Algorithmus

Wir betrachten die **allgemeine Eigenwertaufgabe**

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{B}\mathbf{x} \tag{6.23}$$

und fragen wieder nach Parametern $\lambda \in \mathbb{C}$, für die (6.23) nichttrivial lösbar ist. Die von dem Parameter λ abhängige Familie $\mathbf{A} - \lambda\mathbf{B}$ von Matrizen heißt im Englischen **matrix pencil**. Im Deutschen hat sie keinen Namen. Wir werden daher in diesem Text auch von dem Matrix Pencil oder kurz Pencil sprechen.

Allgemeine Eigenwertaufgaben treten in natürlicher Weise bei der Untersuchung des dynamischen Verhaltens mechanischer Systeme mit endlich vielen Freiheitsgraden auf. Freie kleine Schwingungen eines Systems mit n Freiheitsgraden werden beschrieben durch das lineare Differentialgleichungssystem

$$\mathbf{M}\ddot{\mathbf{y}} + \mathbf{C}\dot{\mathbf{y}} + \mathbf{K}\mathbf{y} = \mathbf{0}. \tag{6.24}$$

Dabei bezeichnet \mathbf{y} den Zustandsvektor des Systems, \mathbf{M} die Massenmatrix oder Trägheitsmatrix, \mathbf{C} die Dämpfungsmatrix und \mathbf{K} die Steifigkeitsmatrix des Systems.

Mit dem Ansatz $\mathbf{y} =: \mathbf{z}e^{\omega t}$ geht die Differentialgleichung über in

$$(\omega^2 \mathbf{M} + \omega \mathbf{C} + \mathbf{K})\mathbf{z} e^{\omega t} = \mathbf{0}, \quad (6.25)$$

und man erhält eine nichttriviale Lösung \mathbf{y} des Systems (6.24), wenn $\omega \in \mathbb{C}$ so gewählt ist, dass

$$(\omega^2 \mathbf{M} + \omega \mathbf{C} + \mathbf{K})\mathbf{z} = \mathbf{0} \quad (6.26)$$

eine nichttriviale Lösung \mathbf{z} besitzt. Dieses quadratische Eigenwertproblem, kann man mit der Transformation $\mathbf{x} = \begin{pmatrix} \omega \mathbf{z} \\ \mathbf{z} \end{pmatrix}$ überführen in die äquivalente allgemeine Eigenwertaufgabe

$$\begin{pmatrix} -\mathbf{C} & -\mathbf{K} \\ \mathbf{I} & \mathbf{O} \end{pmatrix} \mathbf{x} = \omega \begin{pmatrix} \mathbf{M} & \mathbf{O} \\ \mathbf{O} & \mathbf{I} \end{pmatrix} \mathbf{x}. \quad (6.27)$$

Ist das System ungedämpft, d.h. $\mathbf{C} = \mathbf{O}$, so erhält man mit dem Ansatz $\mathbf{y} = \mathbf{z} e^{i\omega t}$ direkter die allgemeine Eigenwertaufgabe

$$\mathbf{K}\mathbf{z} = \omega^2 \mathbf{M}\mathbf{z} =: \lambda \mathbf{M}\mathbf{z}. \quad (6.28)$$

Diese hat den Vorteil, dass Symmetrie- und Definitheitseigenschaften der Matrizen \mathbf{K} und \mathbf{M} sich auf das lineare Eigenwertproblem vererben.

Die spezielle Eigenwertaufgabe besitzt stets genau n Eigenwerte, die Nullstellen des charakteristischen Polynoms $\chi(\lambda) := \det(\mathbf{A} - \lambda \mathbf{I})$. Die folgenden Beispiele zeigen, dass die Verhältnisse bei der allgemeinen Eigenwertaufgabe (6.23) verwickelter sind.

Beispiel 6.45. (1) Für

$$\mathbf{A} - \lambda \mathbf{B} = \begin{pmatrix} 0 & 1 - \lambda \\ 0 & 1 \end{pmatrix}$$

ist $\det(\mathbf{A} - \lambda \mathbf{B}) \equiv 0$.

(2) Für

$$\mathbf{A} - \lambda \mathbf{B} = \begin{pmatrix} 1 & 1 - \lambda \\ 0 & 1 \end{pmatrix}$$

ist $\det(\mathbf{A} - \lambda \mathbf{B}) \equiv 1$.

(3) Für

$$\mathbf{A} - \lambda \mathbf{B} = \begin{pmatrix} 1 & -\lambda \\ 1 & 1 \end{pmatrix}$$

ist $\det(\mathbf{A} - \lambda \mathbf{B}) = 1 + \lambda$.

Definition 6.46. Ist $\det(\mathbf{A} - \lambda\mathbf{B})$ nicht identisch Null, so heißt der Pencil $\mathbf{A} - \lambda\mathbf{B}$ **regulär**, sonst **singulär**. Ist $\mathbf{A} - \lambda\mathbf{B}$ regulär, so heißt $\chi(\lambda) := \det(\mathbf{A} - \lambda\mathbf{B})$ das **charakteristische Polynom** von $\mathbf{A} - \lambda\mathbf{B}$. Die **Eigenwerte** sind die Nullstellen von χ (mit der üblichen Definition der Vielfachheit) und ∞ , falls der Grad $\deg(\chi)$ kleiner als die Dimension n ist. Ist ∞ ein Eigenwert, so ist die Vielfachheit definiert durch $n - \deg(\chi)$.

Beispiel 6.47. (Fortsetzung von Beispiel 6.45.)

Der erste Pencil in Beispiel 6.45. ist singulär, die anderen beiden sind regulär. Der zweite Pencil hat den doppelten Eigenwert ∞ , und der letzte hat die Eigenwerte -1 und ∞ , die beide einfach sind. \square

Satz 6.48. Es sei $\mathbf{A} - \lambda\mathbf{B}$ ein regulärer Pencil. Ist $\det \mathbf{B} \neq 0$, so sind alle Eigenwerte von $\mathbf{A} - \lambda\mathbf{B}$ endlich und stimmen mit den Eigenwerten von $\mathbf{A}\mathbf{B}^{-1}$ und auch $\mathbf{B}^{-1}\mathbf{A}$ überein. Ist \mathbf{B} singulär, so ist ∞ ein Eigenwert von $\mathbf{A} - \lambda\mathbf{B}$ mit der Vielfachheit $n - \text{Rang}(\mathbf{B})$. Ist \mathbf{A} regulär, so sind die Eigenwerte von $\mathbf{A} - \lambda\mathbf{B}$ die Reziproken der Eigenwerte von $\mathbf{A}^{-1}\mathbf{B}$ und von $\mathbf{B}\mathbf{A}^{-1}$, wobei der Eigenwert 0 von $\mathbf{A}^{-1}\mathbf{B}$ mit dem Eigenwert ∞ von $\mathbf{A} - \lambda\mathbf{B}$ identifiziert wird.

Beweis: Es sei \mathbf{B} regulär. Dann gilt

$$0 = \det(\mathbf{A} - \lambda\mathbf{B}) = \det(\mathbf{B}(\mathbf{B}^{-1}\mathbf{A} - \lambda\mathbf{I})) = \det \mathbf{B} \cdot \det(\mathbf{B}^{-1}\mathbf{A} - \lambda\mathbf{I})$$

und

$$0 = \det(\mathbf{A} - \lambda\mathbf{B}) = \det((\mathbf{A}\mathbf{B}^{-1} - \lambda\mathbf{I})\mathbf{B}) = \det \mathbf{B} \cdot \det(\mathbf{A}\mathbf{B}^{-1} - \lambda\mathbf{I}),$$

und die Eigenwerte von $\mathbf{A} - \lambda\mathbf{B}$ sind genau die Eigenwerte von $\mathbf{B}^{-1}\mathbf{A}$ und $\mathbf{A}\mathbf{B}^{-1}$.

Ist \mathbf{B} singulär und $\mathbf{B} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$ die Singulärwertzerlegung von \mathbf{B} , so gilt

$$\det(\mathbf{A} - \lambda\mathbf{B}) = \det(\mathbf{U}(\mathbf{U}^T\mathbf{A}\mathbf{V} - \lambda\mathbf{\Sigma})\mathbf{V}^T) = \pm \det(\mathbf{U}^T\mathbf{A}\mathbf{V} - \lambda\mathbf{\Sigma}),$$

und dies ist offenbar ein Polynom vom Grad $\text{Rang}(\mathbf{B})$.

Ist \mathbf{A} regulär, so liest man die Behauptung aus

$$\det(\mathbf{A} - \lambda\mathbf{B}) = \det(\mathbf{A}(\mathbf{I} - \lambda\mathbf{A}^{-1}\mathbf{B})) = \det((\mathbf{I} - \lambda\mathbf{B}\mathbf{A}^{-1})\mathbf{A})$$

ab. ■

Für $\det \mathbf{B} \neq 0$ ist (6.23) äquivalent der speziellen Eigenwertaufgabe

$$\mathbf{C}\mathbf{x} := \mathbf{B}^{-1}\mathbf{A}\mathbf{x} = \lambda\mathbf{x}, \quad (6.29)$$

und es gibt genau n endliche Eigenwerte. Wir setzen dies von nun an voraus.

Ist \mathbf{B} nicht zu schlecht konditioniert, so kann man (6.29) mit einem der Verfahren aus den vorhergehenden Abschnitten behandeln. Bei großer Kondition von \mathbf{B} können erhebliche Fehler dabei auftreten.

Beispiel 6.49. Die Eigenwerte von

$$\mathbf{A}\mathbf{x} := \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \mathbf{x} = \lambda \begin{pmatrix} 1.17 & 0.8445 \\ 0.7304 & 0.5272 \end{pmatrix} \mathbf{x} =: \lambda\mathbf{B}\mathbf{x}$$

sind

$$\lambda_1 = -1.64894 \quad \text{und} \quad \lambda_2 = 1.01075e6.$$

Bei sechsstelliger Rechnung erhält man für die Eigenwerte von $\mathbf{B}^{-1}\mathbf{A}$

$$\tilde{\lambda}_1 = -6.44383 \quad \text{und} \quad \tilde{\lambda}_2 = 1.03667e6.$$

Die Eigenwerte von $\mathbf{A}^{-1}\mathbf{B}$ führen auf das korrekte Ergebnis. □

Ist die Kondition von \mathbf{B} groß, so kann man die folgende Variante des QR Verfahrens mit impliziten Shifts anwenden, den **QZ Algorithmus** von Moler und Stewart [79].

Satz 6.50. *Es seien $\mathbf{A}, \mathbf{B}, \mathbf{U}, \mathbf{V} \in \mathbb{C}^{(n,n)}$ und \mathbf{U}, \mathbf{V} nichtsingulär.*

Dann besitzen die Eigenwertaufgaben

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{B}\mathbf{x} \quad \text{und} \quad \mathbf{U}\mathbf{A}\mathbf{V}\mathbf{y} = \lambda\mathbf{U}\mathbf{B}\mathbf{V}\mathbf{y}$$

dieselben Eigenwerte.

Ist \mathbf{x} ein Eigenvektor von (6.23) so ist $\mathbf{V}^{-1}\mathbf{x}$ ein Eigenvektor von $\mathbf{U}\mathbf{A}\mathbf{V}\mathbf{y} = \lambda\mathbf{U}\mathbf{B}\mathbf{V}\mathbf{y}$.

Beweis: Es gilt

$$\det(\mathbf{U}\mathbf{A}\mathbf{V} - \lambda\mathbf{U}\mathbf{B}\mathbf{V}) = \det(\mathbf{U}) \det(\mathbf{A} - \lambda\mathbf{B}) \det(\mathbf{V}),$$

und aus $\det(\mathbf{U}) \neq 0$ und $\det(\mathbf{V}) \neq 0$ folgt die Behauptung. ■

Wir transformieren nun \mathbf{A} und \mathbf{B} simultan mit orthogonalen Transformationen auf obere Hessenberg bzw. obere Dreiecksgestalt $\tilde{\mathbf{A}}$, $\tilde{\mathbf{B}}$ und wenden auf die obere Hessenberg Matrix $\tilde{\mathbf{A}}\tilde{\mathbf{B}}^{-1}$ den Doppelschritt QR Algorithmus mit impliziten Shifts an. Dabei wird jedoch nicht $\tilde{\mathbf{A}}\tilde{\mathbf{B}}^{-1}$ berechnet, um die Kondition nicht zu verschlechtern.

Wir demonstrieren die Reduktion von \mathbf{A} bzw. \mathbf{B} auf obere Hessenberg bzw. obere Dreiecksgestalt an dem Fall $n = 4$.

Zunächst wird \mathbf{B} durch Householder Transformationen in obere Dreiecksgestalt überführt. Dann gehen \mathbf{A} bzw. \mathbf{B} über in

$$\mathbf{A} : \begin{pmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{pmatrix} \quad \mathbf{B} : \begin{pmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \\ 0 & 0 & 0 & * \end{pmatrix}.$$

Durch Multiplikation von links mit einer Drehung \mathbf{Q}_{34} wird in \mathbf{A} an der Stelle (4, 1) eine 0 erzeugt. Gleichzeitig wird dadurch in \mathbf{B} an der Stelle (4, 3) ein von 0 verschiedenes Element erzeugt.

$$\mathbf{A} : \begin{pmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ 0 & * & * & * \end{pmatrix} \quad \mathbf{B} : \begin{pmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \\ 0 & 0 & * & * \end{pmatrix}.$$

Multiplikation von rechts mit einer Drehung \mathbf{Z}_{34} annulliert das Element (4, 3) in \mathbf{B} und zerstört nicht die 0 an der Stelle (4, 1) in \mathbf{A} :

$$\mathbf{A} : \begin{pmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ 0 & * & * & * \end{pmatrix} \quad \mathbf{B} : \begin{pmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \\ 0 & 0 & 0 & * \end{pmatrix}.$$

Multipliziert man von links mit einer Drehung \mathbf{Q}_{23} , um in \mathbf{A} an der Stelle (3, 1) eine 0 zu erzeugen, und dann von rechts mit einer Drehung \mathbf{Z}_{23} um in \mathbf{B} das Element an der Stelle (3, 2) zu annullieren, so erhält man

$$\mathbf{A} : \begin{pmatrix} * & * & * & * \\ * & * & * & * \\ 0 & * & * & * \\ 0 & * & * & * \end{pmatrix} \quad \mathbf{B} : \begin{pmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & * & * & * \\ 0 & 0 & 0 & * \end{pmatrix},$$

$$\mathbf{A} : \begin{pmatrix} * & * & * & * \\ * & * & * & * \\ 0 & * & * & * \\ 0 & * & * & * \end{pmatrix} \quad \mathbf{B} : \begin{pmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \\ 0 & 0 & 0 & * \end{pmatrix}.$$

Schließlich erhält man durch Multiplikation von links mit einem \mathbf{Q}_{34} und von rechts mit einem \mathbf{Z}_{34}

$$\mathbf{A} : \begin{pmatrix} * & * & * & * \\ * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \end{pmatrix} \quad \mathbf{B} : \begin{pmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \\ 0 & 0 & * & * \end{pmatrix},$$

$$\mathbf{A} : \begin{pmatrix} * & * & * & * \\ * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \end{pmatrix} \quad \mathbf{B} : \begin{pmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \\ 0 & 0 & 0 & * \end{pmatrix},$$

und dies ist die gewünschte obere Hessenberg Matrix bzw. obere Dreiecksmatrix.

Die Elemente von \mathbf{A} werden also spaltenweise von links nach rechts von unten nach oben durch eine Drehung $\mathbf{Q}_{i,i+1}$ annulliert und das dabei entstehende von 0 verschiedene Subdiagonalelement in \mathbf{B} wird durch eine Multiplikation mit einer Drehung $\mathbf{Z}_{i,i+1}$ von rechts sofort wieder annulliert.

Es habe nun \mathbf{A} obere Hessenberg Gestalt, und es sei \mathbf{B} eine obere Dreiecksmatrix. Dann ist $\mathbf{C} := \mathbf{A}\mathbf{B}^{-1}$ eine obere Hessenberg Matrix. Ein Doppelschritt des QR Algorithmus mit Shifts für \mathbf{C} liefere die Matrix $\tilde{\mathbf{C}} := \mathbf{Q}\mathbf{C}\mathbf{Q}^H$.

Es seien $\tilde{\mathbf{Q}}$ und \mathbf{Z} unitäre Matrizen mit

- (i) $\tilde{\mathbf{Q}}$ besitzt dieselbe erste Zeile wie \mathbf{Q}
- (ii) $\tilde{\mathbf{A}} := \tilde{\mathbf{Q}}\mathbf{A}\mathbf{Z}$ ist obere Hessenberg Matrix
- (iii) $\tilde{\mathbf{B}} := \tilde{\mathbf{Q}}\mathbf{B}\mathbf{Z}$ ist obere Dreiecksmatrix.

Dann ist

$$\tilde{\mathbf{A}}\tilde{\mathbf{B}}^{-1} = \tilde{\mathbf{Q}}\mathbf{A}\mathbf{Z}\mathbf{Z}^{-1}\mathbf{B}^{-1}\tilde{\mathbf{Q}}^H = \tilde{\mathbf{Q}}\mathbf{A}\mathbf{B}^{-1}\tilde{\mathbf{Q}}^H$$

eine obere Hessenberg Matrix, und daher gilt $\tilde{\mathbf{Q}} = \mathbf{Q}$ und $\tilde{\mathbf{C}} = \tilde{\mathbf{A}}\tilde{\mathbf{B}}^{-1}$.

Wendet man diesen Algorithmus wiederholt an, so erhält man Matrizen $\mathbf{A}_1, \mathbf{A}_2, \mathbf{A}_3, \dots$ und $\mathbf{B}_1, \mathbf{B}_2, \dots$. Setzt man $\mathbf{C}_m := \mathbf{A}_m\mathbf{B}_m^{-1}$, so ist \mathbf{C}_m genau die Folge von Matrizen aus dem QR Algorithmus aus Abschnitt 6.4.

Wählt man die Shifts geeignet, so “konvergiert” \mathbf{C}_m gegen eine gestörte obere Dreiecksmatrix mit höchstens $(2, 2)$ -Blöcken in der Diagonale, und da \mathbf{B}_m obere Dreiecksmatrix ist, “konvergiert” $\mathbf{A}_m = \mathbf{C}_m\mathbf{B}_m$ gegen eine Matrix von derselben Gestalt. Aus diesen höchstens $(2, 2)$ -Diagonalblöcken von \mathbf{A}_m bzw. \mathbf{B}_m kann man die Eigenwerte von (6.23) berechnen. Man beachte, dass niemals die Matrix \mathbf{C}_m berechnet wird.

Die Schritte (ii) und (iii) kann man simultan wie oben beschrieben ausführen. Es bleibt also nur als Problem der Bestimmung der ersten Zeile von \mathbf{Q} und der Shifts.

Wie in Abschnitt 6.4 kann man die erste Zeile von \mathbf{Q} aus den ersten beiden Spalten von \mathbf{C} bestimmen. Da \mathbf{B} obere Dreiecksmatrix ist, sind diese gegeben durch

$$\begin{pmatrix} \mathbf{c}^1 & \mathbf{c}^2 \end{pmatrix} = \begin{pmatrix} \mathbf{a}^1 & \mathbf{a}^2 \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ 0 & b_{22} \end{pmatrix}^{-1}.$$

Die Shifts erhält man mit ähnlichen Überlegungen aus dem unteren $(2, 2)$ -Block von \mathbf{A} und \mathbf{B} .

Kapitel 7

Symmetrische Eigenwertaufgaben

7.1 Charakterisierung von Eigenwerten

Wir betrachten die Hermitesche Eigenwertaufgabe

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}, \quad \mathbf{A} \in \mathbb{C}^{(n,n)} \text{ Hermitesch.} \quad (7.1)$$

Wir wissen bereits, dass alle Eigenwerte reell sind und dass ein vollständiges System von Eigenvektoren existiert, wobei die Eigenvektoren \mathbf{x}^i orthonormal gewählt werden können.

Viele der Aussagen in diesem Unterabschnitt gelten auch für die allgemeine Eigenwertaufgabe

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{B}\mathbf{x}, \quad \mathbf{A}, \mathbf{B} \in \mathbb{C}^{(n,n)} \text{ Hermitesch, } \mathbf{B} \text{ positiv definit,} \quad (7.2)$$

denn ist $\mathbf{B} = \mathbf{C}\mathbf{C}^H$ die Cholesky Zerlegung von \mathbf{B} , so gilt

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{B}\mathbf{x} \quad \iff \quad \mathbf{F}\mathbf{y} := \mathbf{C}^{-1}\mathbf{A}\mathbf{C}^{-H}\mathbf{y} = \lambda\mathbf{y}, \quad \mathbf{y} := \mathbf{C}^H\mathbf{x}.$$

Auch (7.2) besitzt also reelle Eigenwerte, und sind \mathbf{y}^i und \mathbf{y}^j orthonormale Eigenvektoren von \mathbf{F} , so gilt für $\mathbf{x}^i := \mathbf{C}^{-H}\mathbf{y}^i$ und $\mathbf{x}^j := \mathbf{C}^{-H}\mathbf{y}^j$

$$(\mathbf{x}^i)^H \mathbf{B} \mathbf{x}^j = (\mathbf{y}^i)^H \mathbf{C}^{-1} \mathbf{B} \mathbf{C}^{-H} \mathbf{y}^j = (\mathbf{y}^i)^H \mathbf{y}^j = \delta_{ij}.$$

Die Eigenvektoren von (7.2) können also orthonormal bzgl. des inneren Produktes

$$\langle \mathbf{x}, \mathbf{y} \rangle_B := \mathbf{x}^H \mathbf{B} \mathbf{y}$$

gewählt werden.

Wir setzen von nun an voraus, dass die Eigenwerte von (7.1) bzw. (7.2) der Größe nach geordnet sind

$$\lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n, \quad (7.3)$$

und die zugehörigen Eigenvektoren orthonormal gewählt sind.

Für Hermitesche Matrizen kann man leicht aus dem Defekt einer Näherung für einen Eigenwert und Eigenvektor auf den Fehler für den Eigenwert schließen (vgl. LA Satz 8.88)

Satz 7.1. (Krylov, Bogoliubov) *Es sei $\lambda \in \mathbb{R}$, $\mathbf{x} \in \mathbb{C}^n \setminus \{\mathbf{0}\}$ und $\mathbf{y} := \mathbf{A}\mathbf{x} - \lambda\mathbf{x}$. Dann gilt*

$$\min_{i=1, \dots, n} |\lambda - \lambda_i| \leq \frac{\|\mathbf{y}\|_2}{\|\mathbf{x}\|_2} \quad (7.4)$$

Beweis: Es gilt $\mathbf{x} = \sum_{i=1}^n \langle \mathbf{x}, \mathbf{x}^i \rangle \mathbf{x}^i$, $\mathbf{A}\mathbf{x} = \sum_{i=1}^n \lambda_i \langle \mathbf{x}, \mathbf{x}^i \rangle \mathbf{x}^i$ und $\|\mathbf{x}\|_2^2 = \sum_{i=1}^n |\langle \mathbf{x}, \mathbf{x}^i \rangle|^2$. Damit folgt

$$\begin{aligned} \|\mathbf{A}\mathbf{x} - \lambda\mathbf{x}\|_2^2 &= \left\| \sum_{i=1}^n (\lambda_i - \lambda) \langle \mathbf{x}, \mathbf{x}^i \rangle \mathbf{x}^i \right\|_2^2 = \sum_{i=1}^n |\lambda_i - \lambda|^2 |\langle \mathbf{x}, \mathbf{x}^i \rangle|^2 \\ &\geq \min_{i=1, \dots, n} |\lambda_i - \lambda|^2 \sum_{i=1}^n |\langle \mathbf{x}, \mathbf{x}^i \rangle|^2 = \min_{i=1, \dots, n} |\lambda_i - \lambda|^2 \|\mathbf{x}\|_2^2 \end{aligned}$$

■

Dabei liefert bei gegebenem $\mathbf{x} \in \mathbb{C}^n$, $\mathbf{x} \neq \mathbf{0}$, der **Rayleigh Quotient**

$$R(\mathbf{x}) := \frac{\mathbf{x}^H \mathbf{A} \mathbf{x}}{\mathbf{x}^H \mathbf{x}}$$

von \mathbf{x} die beste Schätzung λ für einen Eigenwert.

Mit dem Rayleigh Quotienten, der für das allgemeine Eigenwertproblem gegeben ist durch

$$R(\mathbf{x}) := \frac{\mathbf{x}^H \mathbf{A} \mathbf{x}}{\mathbf{x}^H \mathbf{B} \mathbf{x}}, \quad (7.5)$$

kann man die Eigenwerte auf folgende Weise charakterisieren (vgl. LA Satz 8.42 und LA Satz 8.113)

Satz 7.2. (i) $\lambda_1 \leq R(\mathbf{x}) \leq \lambda_n$ für alle $\mathbf{x} \in \mathbb{C}^n$, $\mathbf{x} \neq \mathbf{0}$

$$(ii) \lambda_1 = \min_{\mathbf{x} \neq \mathbf{0}} R(\mathbf{x}), \quad \lambda_n = \max_{\mathbf{x} \neq \mathbf{0}} R(\mathbf{x})$$

(iii) Ist $\mathbf{x} \neq \mathbf{0}$ mit $\lambda_1 = R(\mathbf{x})$ bzw. $\lambda_n = R(\mathbf{x})$, so ist \mathbf{x} Eigenvektor zu λ_1 bzw. λ_n

(iv)

$$\begin{aligned} \lambda_i &= \min\{R(\mathbf{x}) : \mathbf{x}^H \mathbf{B} \mathbf{x}^j = 0, j = 1, \dots, i-1, \mathbf{x} \neq \mathbf{0}\} \\ &= \max\{R(\mathbf{x}) : \mathbf{x}^H \mathbf{B} \mathbf{x}^j = 0, j = i+1, \dots, n, \mathbf{x} \neq \mathbf{0}\} \end{aligned}$$

Beweis: Sei $\mathbf{x} \in \mathbb{C}^n$, $\mathbf{x} \neq \mathbf{0}$, und $c_j := \langle \mathbf{x}, \mathbf{x}^j \rangle_B$, $j = 1, \dots, n$. Dann gilt

$$\begin{aligned} \mathbf{x}^H \mathbf{A} \mathbf{x} &= \left(\sum_{j=1}^n c_j \mathbf{x}^j \right)^H \sum_{k=1}^n c_k \mathbf{A} \mathbf{x}^k = \sum_{j=1}^n \bar{c}_j (\mathbf{x}^j)^H \sum_{k=1}^n c_k \lambda_k \mathbf{B} \mathbf{x}^k \\ &= \sum_{j,k=1}^n \lambda_k \bar{c}_j c_k (\mathbf{x}^j)^H \mathbf{B} \mathbf{x}^k = \sum_{j=1}^n \lambda_j |c_j|^2, \\ \mathbf{x}^H \mathbf{B} \mathbf{x} &= \sum_{j=1}^n \bar{c}_j (\mathbf{x}^j)^H \mathbf{B} \sum_{k=1}^n c_k \mathbf{x}^k = \sum_{j,k=1}^n \bar{c}_j c_k (\mathbf{x}^j)^H \mathbf{B} \mathbf{x}^k = \sum_{j=1}^n |c_j|^2, \end{aligned}$$

und daher

$$R(\mathbf{x}) = \frac{\sum_{j=1}^n \lambda_j |c_j|^2}{\sum_{k=1}^n |c_k|^2} = \sum_{j=1}^n \alpha_j \lambda_j \quad \text{mit } \alpha_j := |c_j|^2 / \sum_{k=1}^n |c_k|^2. \quad (7.6)$$

Wegen $0 \leq \alpha_j$ und $\sum_{j=1}^n \alpha_j = 1$ sieht man nun sofort

$$\lambda_1 = \sum_{j=1}^n \alpha_j \lambda_1 \leq \sum_{j=1}^n \alpha_j \lambda_j = R(\mathbf{x}) = \sum_{j=1}^n \alpha_j \lambda_j \leq \sum_{j=1}^n \alpha_j \lambda_n = \lambda_n, \quad (7.7)$$

also die Aussage (i).

Für $\mathbf{x} = \mathbf{x}^1$ gilt in der Darstellung $\mathbf{x} = \sum_{j=1}^n c_j \mathbf{x}^j$ speziell $c_1 = 1, c_2 = \dots = c_n = 0$. Hieraus folgt für die Darstellung (7.6) von $R(\mathbf{x}^1)$, dass $\alpha_1 = 1, \alpha_2 = \dots = \alpha_n = 0$, was $R(\mathbf{x}^1) = \lambda_1$ zeigt. Analog erhält man $R(\mathbf{x}^n) = \lambda_n$. Zusammen mit (i) folgt (ii).

Ist $\mathbf{x} \neq \mathbf{0}$ mit $R(\mathbf{x}) = \lambda_1$, so muss in (7.6) gelten: $c_j = 0$ für alle j mit $\lambda_j \neq \lambda_1$. Daher ist \mathbf{x} Eigenvektor zu λ_1 . Genauso muss \mathbf{x} bei $R(\mathbf{x}) = \lambda_n$ ein Eigenvektor zu λ_n sein. Also gilt (iii).

(iv) folgt wie (ii), da $R(\mathbf{x})$ auf $V := \{\mathbf{x} \in \mathbb{C}^n : \langle \mathbf{x}, \mathbf{x}^j \rangle_B = 0, j = 1, \dots, i-1\}$ die Darstellung $R(\mathbf{x}) = \frac{\sum_{j=i}^n \lambda_j |c_j|^2}{\sum_{k=i}^n |c_k|^2}$ besitzt und auf $W := \{\mathbf{x} \in \mathbb{C}^n : \langle \mathbf{x}, \mathbf{x}^j \rangle_B = 0, j = i+1, \dots, n\}$ die Darstellung $R(\mathbf{x}) = \frac{\sum_{j=1}^i \lambda_j |c_j|^2}{\sum_{k=1}^i |c_k|^2}$. ■

Die Charakterisierung in (iv) heißt das **Rayleighsche Prinzip**. Es ist numerisch wertlos, da man zur Bestimmung von z.B. λ_2 den (exakten) Eigenvektor \mathbf{x}^1 zu λ_1 benötigt. Diesen Nachteil haben die folgenden minmax Charakterisierungen nicht. Das Minmax Prinzip von Poincaré wurde in LA Satz 8.45 (bzw. LA Satz 8.113) gezeigt.

Satz 7.3. (minmax-Prinzip von Poincaré)

$$\lambda_i = \min_{\dim V=i} \max_{\mathbf{x} \in V \setminus \{0\}} R(\mathbf{x}) = \max_{\dim V=n-i+1} \min_{\mathbf{x} \in V \setminus \{0\}} R(\mathbf{x}). \quad (7.8)$$

Beweis: Sei V ein beliebiger Teilraum des \mathbb{C}^n mit $\dim V = i$, und sei $\mathbf{y}^1, \dots, \mathbf{y}^i$ eine Basis von V . Dann besitzt das unterbestimmte lineare Gleichungssystem

$$\sum_{j=1}^i \eta_j \langle \mathbf{y}^j, \mathbf{x}^k \rangle_B = 0, \quad k = 1, \dots, i-1,$$

eine nichttriviale Lösung $(\eta_1 \ \dots \ \eta_i)^T$. Hiermit definieren wir $\tilde{\mathbf{y}} := \sum_{j=1}^i \eta_j \mathbf{y}^j \in V$. Dann gilt nach Konstruktion $\langle \tilde{\mathbf{y}}, \mathbf{x}^j \rangle_B = 0$, $j = 1, \dots, i-1$, und das Rayleighsche Prinzip liefert $R(\tilde{\mathbf{y}}) \geq \lambda_i$. Es gilt also erst recht

$$\max_{\mathbf{x} \in V \setminus \{0\}} R(\mathbf{x}) \geq R(\tilde{\mathbf{y}}) \geq \lambda_i,$$

und da dies für jeden i -dimensionalen Teilraum V des \mathbb{C}^n richtig ist, gilt $\lambda_i \leq \rho_i$.

Umgekehrt gilt für $V := \text{span}\{\mathbf{x}^1, \dots, \mathbf{x}^i\}$ nach dem Rayleighschen Prinzip $\lambda_i = \max_{\mathbf{x} \in V \setminus \{0\}} R(\mathbf{x})$, und daher ist $\lambda_i \geq \rho_i$; zusammen also die Behauptung. ■

Angenehmer ist manchmal die folgende Formulierung:

Satz 7.4. (maxmin-Prinzip von Courant-Fischer)

$$\begin{aligned} \lambda_i &= \max_{\{p^1, \dots, p^{i-1}\}} \min\{R(\mathbf{x}) : \mathbf{x} \neq \mathbf{0}, \mathbf{x}^H \mathbf{B} \mathbf{p}^j = 0, j = 1, \dots, i-1\} \\ &= \min_{\{p^1, \dots, p^{n-i}\}} \max\{R(\mathbf{x}) : \mathbf{x} \neq \mathbf{0}, \mathbf{x}^H \mathbf{B} \mathbf{p}^j = 0, j = 1, \dots, n-i\}. \end{aligned}$$

Dabei müssen die $\mathbf{p}^i \in \mathbb{R}^n$ nicht notwendig voneinander und von $\mathbf{0}$ verschieden sein.

Beweis: Wir setzen

$$h(\mathbf{p}^1, \dots, \mathbf{p}^{i-1}) := \min\{R(\mathbf{x}) : \mathbf{x} \neq \mathbf{0}, \mathbf{x}^H \mathbf{B} \mathbf{p}^j = 0, j = 1, \dots, i-1\}.$$

Dann gilt nach dem Rayleighschen Prinzip $h(\mathbf{x}^1, \dots, \mathbf{x}^{i-1}) = \lambda_i$.

Andererseits gibt es für beliebige $\mathbf{p}^1, \dots, \mathbf{p}^{i-1} \in \mathbb{C}^n$ sicher ein $\mathbf{x} \neq \mathbf{0}$, das die $n-1$ homogenen linearen Gleichungen

$$\mathbf{x}^H \mathbf{B} \mathbf{p}^j = 0, \quad j = 1, \dots, i-1, \quad \mathbf{x}^H \mathbf{B} \mathbf{x}^j = 0, \quad j = i+1, \dots, n$$

erfüllt.

Aus den letzten Gleichungen und aus dem Rayleighschen Prinzip folgt $R(\mathbf{x}) \leq \lambda_i$, und damit erst recht $h(\mathbf{p}^1, \dots, \mathbf{p}^{i-1}) \leq \lambda_i$. ■

Wir wollen nun einige Folgerungen aus Satz 7.3. und Satz 7.4. ziehen. Der folgende Satz enthält eine Verschärfung von Korollar 6.9.:

Satz 7.5. (Weyl) *Es seien $\mathbf{A}, \mathbf{E} \in \mathbb{C}^{(n,n)}$ Hermitesch. Seien $\lambda_1 \leq \dots \leq \lambda_n$ die Eigenwerte von \mathbf{A} und $\tilde{\lambda}_1 \leq \dots \leq \tilde{\lambda}_n$ die Eigenwerte von $\tilde{\mathbf{A}} := \mathbf{A} + \mathbf{E}$. Dann gilt*

$$|\lambda_j - \tilde{\lambda}_j| \leq \|\mathbf{E}\|_2. \quad (7.9)$$

Beweis: Es sei $W := \text{span}\{\mathbf{x}^1, \dots, \mathbf{x}^j\}$ der Raum, der von den Eigenvektoren zu $\lambda_1, \dots, \lambda_j$ aufgespannt wird. Dann gilt

$$\begin{aligned} \tilde{\lambda}_j &= \min_{\dim V=j} \max_{\mathbf{x} \in V \setminus \{0\}} \frac{\mathbf{x}^H (\mathbf{A} + \mathbf{E}) \mathbf{x}}{\mathbf{x}^H \mathbf{x}} \leq \max_{\mathbf{x} \in W \setminus \{0\}} \frac{\mathbf{x}^H (\mathbf{A} + \mathbf{E}) \mathbf{x}}{\mathbf{x}^H \mathbf{x}} \\ &= \max_{\mathbf{x} \in W \setminus \{0\}} \left(\frac{\mathbf{x}^H \mathbf{A} \mathbf{x}}{\mathbf{x}^H \mathbf{x}} + \frac{\mathbf{x}^H \mathbf{E} \mathbf{x}}{\mathbf{x}^H \mathbf{x}} \right) \leq \max_{\mathbf{x} \in W \setminus \{0\}} \frac{\mathbf{x}^H \mathbf{A} \mathbf{x}}{\mathbf{x}^H \mathbf{x}} + \max_{\mathbf{x} \in W \setminus \{0\}} \frac{\mathbf{x}^H \mathbf{E} \mathbf{x}}{\mathbf{x}^H \mathbf{x}} \\ &= \lambda_j + \max_{\mathbf{x} \in W \setminus \{0\}} \frac{\mathbf{x}^H \mathbf{E} \mathbf{x}}{\mathbf{x}^H \mathbf{x}} \leq \lambda_j + \|\mathbf{E}\|_2, \end{aligned}$$

und genauso gilt mit dem Raum $W := \text{span}\{\tilde{\mathbf{x}}^1, \dots, \tilde{\mathbf{x}}^j\}$, der von den Eigenvektoren von $\tilde{\mathbf{A}}$ zu den Eigenwerten $\tilde{\lambda}_1, \dots, \tilde{\lambda}_j$ aufgespannt wird, die Ungleichung

$$\lambda_j \leq \tilde{\lambda}_j + \|\mathbf{E}\|_2. \quad \blacksquare$$

Für allgemeine Eigenwertaufgaben erhält man auf ähnliche Weise

Satz 7.6. (Vergleichssatz) *Wir betrachten neben (7.2) mit den Eigenwerten $\lambda_1 \leq \dots \leq \lambda_n$ die Vergleichsaufgabe $\tilde{\mathbf{A}} \mathbf{x} = \lambda \tilde{\mathbf{B}} \mathbf{x}$ ($\tilde{\mathbf{A}}, \tilde{\mathbf{B}}$ Hermitesch, $\tilde{\mathbf{B}}$ positiv definit) mit den Eigenwerten $\tilde{\lambda}_1 \leq \dots \leq \tilde{\lambda}_n$.*

Dann folgt aus

$$\frac{\mathbf{x}^H \mathbf{A} \mathbf{x}}{\mathbf{x}^H \mathbf{B} \mathbf{x}} \leq \frac{\mathbf{x}^H \tilde{\mathbf{A}} \mathbf{x}}{\mathbf{x}^H \tilde{\mathbf{B}} \mathbf{x}} \quad \text{für alle } \mathbf{x} \neq \mathbf{0}$$

für die Eigenwerte

$$\lambda_i \leq \tilde{\lambda}_i, \quad i = 1, \dots, n \quad (7.10)$$

Beweis: Sei $\tilde{\mathbf{x}}^1, \dots, \tilde{\mathbf{x}}^n$ ein System $\tilde{\mathbf{B}}$ -orthogonaler Eigenvektoren zu $\tilde{\lambda}_1, \dots, \tilde{\lambda}_n$, und sei $W := \text{span}\{\tilde{\mathbf{x}}^1, \dots, \tilde{\mathbf{x}}^i\}$. Dann gilt

$$\lambda_i = \min_{\dim V=i} \max_{x \in V \setminus \{0\}} \frac{\mathbf{x}^H \mathbf{A} \mathbf{x}}{\mathbf{x}^H \mathbf{B} \mathbf{x}} \leq \max_{x \in W \setminus \{0\}} \frac{\mathbf{x}^H \mathbf{A} \mathbf{x}}{\mathbf{x}^H \mathbf{B} \mathbf{x}} \leq \max_{x \in W \setminus \{0\}} \frac{\mathbf{x}^H \tilde{\mathbf{A}} \mathbf{x}}{\mathbf{x}^H \tilde{\mathbf{B}} \mathbf{x}} = \tilde{\lambda}_i \quad \blacksquare$$

Der folgende Satz stellt eine Beziehung zwischen den Eigenwerten von (7.2) und den Eigenwerten von Hauptabschnittsuntermatrizen her.

Satz 7.7. (Verschachtelungssatz von Cauchy) *Seien*

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 & \mathbf{A}_2 \\ \mathbf{A}_2^H & \mathbf{A}_3 \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} \mathbf{B}_1 & \mathbf{B}_2 \\ \mathbf{B}_2^H & \mathbf{B}_3 \end{pmatrix} \quad \text{und} \quad \mathbf{A}_1, \mathbf{B}_1 \in \mathbb{C}^{(m,m)}$$

Seien $\lambda_1 \leq \dots \leq \lambda_n$ die Eigenwerte von $\mathbf{A} \mathbf{x} = \lambda \mathbf{B} \mathbf{x}$ und $\mu_1 \leq \dots \leq \mu_m$ die Eigenwerte von $\mathbf{A}_1 \mathbf{z} = \mu \mathbf{B}_1 \mathbf{z}$. Dann gilt

$$\lambda_j \leq \mu_j \leq \lambda_{j+n-m}, \quad j = 1, \dots, m \quad (7.11)$$

Beweis: Seien $\mathbf{z}^1, \dots, \mathbf{z}^m \in \mathbb{C}^m$ die Eigenvektoren von $\mathbf{A}_1 \mathbf{z} = \mu \mathbf{B}_1 \mathbf{z}$ zu μ_1, \dots, μ_m , sei $\mathbf{x}^i = \begin{pmatrix} \mathbf{z}^i \\ \mathbf{0} \end{pmatrix} \in \mathbb{C}^n, i = 1, \dots, m, W := \text{span}\{\mathbf{x}^1, \dots, \mathbf{x}^j\}$ und $Z := \text{span}\{\mathbf{z}^1, \dots, \mathbf{z}^j\}$.

Dann gilt nach Satz 7.3.

$$\lambda_j = \min_{\dim V=j} \max_{x \in V \setminus \{0\}} \frac{\mathbf{x}^H \mathbf{A} \mathbf{x}}{\mathbf{x}^H \mathbf{B} \mathbf{x}} \leq \max_{x \in W \setminus \{0\}} \frac{\mathbf{x}^H \mathbf{A} \mathbf{x}}{\mathbf{x}^H \mathbf{B} \mathbf{x}} = \max_{z \in Z \setminus \{0\}} \frac{\mathbf{z}^H \mathbf{A}_1 \mathbf{z}}{\mathbf{z}^H \mathbf{B}_1 \mathbf{z}} = \mu_j.$$

Genauso erhält man mit $\tilde{W} := \text{span}\{\mathbf{x}^j, \dots, \mathbf{x}^m\}$ und $\tilde{Z} := \text{span}\{\mathbf{z}^j, \dots, \mathbf{z}^m\}$ die Ungleichung

$$\lambda_{j+n-m} = \max_{\dim V=m-j+1} \min_{x \in V \setminus \{0\}} \frac{\mathbf{x}^H \mathbf{A} \mathbf{x}}{\mathbf{x}^H \mathbf{B} \mathbf{x}} \geq \min_{x \in \tilde{W} \setminus \{0\}} \frac{\mathbf{x}^H \mathbf{A} \mathbf{x}}{\mathbf{x}^H \mathbf{B} \mathbf{x}} = \min_{z \in \tilde{Z} \setminus \{0\}} \frac{\mathbf{z}^H \mathbf{A}_1 \mathbf{z}}{\mathbf{z}^H \mathbf{B}_1 \mathbf{z}} = \mu_j \quad \blacksquare$$

Satz 7.7. gilt natürlich für beliebige, zueinander passende Hauptuntermatrizen \mathbf{A}_1 bzw. \mathbf{B}_1 von \mathbf{A} bzw. \mathbf{B} . Streicht man insbesondere eine beliebige Zeile und zugehörige Spalte in \mathbf{A} und \mathbf{B} , so gilt für die Eigenwerte $\mu_1 \leq \dots \leq \mu_{n-1}$ der entstehenden Eigenwertaufgabe $\mathbf{A}_1 \mathbf{z} = \mu \mathbf{B}_1 \mathbf{z}$

$$\lambda_j \leq \mu_j \leq \lambda_{j+n-(n-1)} = \lambda_{j+1};$$

die Eigenwerte sind also ineinander verschachtelt.

Satz 7.8. (Monotoniesatz) Seien $\mathbf{A}_1, \mathbf{A}_2, \mathbf{B} \in \mathbb{C}^{(n,n)}$ Hermitesch, \mathbf{B} positiv definit, $\mathbf{A} := \mathbf{A}_1 + \mathbf{A}_2$. Dann gilt für die Eigenwerte $\lambda_1^{(i)} \leq \dots \leq \lambda_n^{(i)}$ von $\mathbf{A}_i \mathbf{x} = \lambda \mathbf{B} \mathbf{x}$, $i = 1, 2$,

$$\lambda_{i+j-1} \geq \lambda_i^{(1)} + \lambda_j^{(2)}, \quad i, j = 1, \dots, n, \quad i+j-1 \leq n$$

Beweis: Seien V_1 bzw. V_2 die durch die zu $\mathbf{x}_1^{(1)}, \dots, \mathbf{x}_{i-1}^{(1)}$ bzw. $\mathbf{x}_1^{(2)}, \dots, \mathbf{x}_{j-1}^{(2)}$ aufgespannten Teilräume der \mathbb{C}^n und sei $W := \text{span}\{V_1 \cup V_2\}$. Dann gilt $\dim W \leq i+j-2$, und nach Satz 7.4. folgt

$$\begin{aligned} \lambda_{i+j-1} &= \max_{\dim Z \leq i+j-2} \min \left\{ \frac{\mathbf{x}^H \mathbf{A} \mathbf{x}}{\mathbf{x}^H \mathbf{B} \mathbf{x}} : \mathbf{x}^H \mathbf{B} \mathbf{z} = 0 \quad \forall \mathbf{z} \in Z \right\} \\ &\geq \min \left\{ \frac{\mathbf{x}^H \mathbf{A}_1 \mathbf{x}}{\mathbf{x}^H \mathbf{B} \mathbf{x}} + \frac{\mathbf{x}^H \mathbf{A}_2 \mathbf{x}}{\mathbf{x}^H \mathbf{B} \mathbf{x}} : \mathbf{x}^H \mathbf{B} \mathbf{z} = 0 \quad \forall \mathbf{z} \in W \right\} \\ &\geq \min \left\{ \frac{\mathbf{x}^H \mathbf{A}_1 \mathbf{x}}{\mathbf{x}^H \mathbf{B} \mathbf{x}} : \mathbf{x}^H \mathbf{B} \mathbf{z} = 0 \quad \forall \mathbf{z} \in W \right\} + \min \left\{ \frac{\mathbf{x}^H \mathbf{A}_2 \mathbf{x}}{\mathbf{x}^H \mathbf{B} \mathbf{x}} : \mathbf{x}^H \mathbf{B} \mathbf{z} = 0 \quad \forall \mathbf{z} \in W \right\} \\ &\geq \min \left\{ \frac{\mathbf{x}^H \mathbf{A}_1 \mathbf{x}}{\mathbf{x}^H \mathbf{B} \mathbf{x}} : \mathbf{x}^H \mathbf{B} \mathbf{z} = 0 \quad \forall \mathbf{z} \in V_1 \right\} + \min \left\{ \frac{\mathbf{x}^H \mathbf{A}_2 \mathbf{x}}{\mathbf{x}^H \mathbf{B} \mathbf{x}} : \mathbf{x}^H \mathbf{B} \mathbf{z} = 0 \quad \forall \mathbf{z} \in V_2 \right\} \\ &= \lambda_i^{(1)} + \lambda_j^{(2)} \quad \blacksquare \end{aligned}$$

Ist speziell \mathbf{A}_2 positiv semidefinit, so gilt

$$\frac{\mathbf{x}^H \mathbf{A} \mathbf{x}}{\mathbf{x}^H \mathbf{B} \mathbf{x}} \geq \frac{\mathbf{x}^H \mathbf{A}_1 \mathbf{x}}{\mathbf{x}^H \mathbf{B} \mathbf{x}} \quad \text{für alle } \mathbf{x} \neq \mathbf{0},$$

und man erhält für $j = 1$ einen Spezialfall von Satz 7.6.

Satz 7.9. (Rang 1 - Modifikationen) Sei $\mathbf{A} \in \mathbb{C}^{(n,n)}$ Hermitesch, $\mathbf{c} \in \mathbb{C}^n$ und $\tau \in \mathbb{R}$. Dann gilt für die der Größe nach geordneten Eigenwerte $\lambda_i(\mathbf{B})$ von $\mathbf{B} := \mathbf{A} + \tau \mathbf{c} \mathbf{c}^H$

$$\begin{aligned} \lambda_i(\mathbf{B}) &\in [\lambda_i(\mathbf{A}), \lambda_{i+1}(\mathbf{A})] \quad , \quad \text{falls } \tau \geq 0 \quad , \quad i = 1, \dots, n \quad (\lambda_{n+1}(\mathbf{A}) = \infty) \\ \lambda_i(\mathbf{B}) &\in [\lambda_{i-1}(\mathbf{A}), \lambda_i(\mathbf{A})] \quad \text{falls } \tau \leq 0 \quad , \quad i = 1, \dots, n \quad (\lambda_0(\mathbf{A}) = -\infty) \end{aligned}$$

Beweis: Betrachte den Fall $\tau \geq 0$ ($\tau \leq 0$ folgt hieraus, wenn man \mathbf{A} als Rang 1-Modifikation von \mathbf{B} betrachtet).

$\lambda_i(\mathbf{B}) \geq \lambda_i(\mathbf{A})$ folgt aus dem Vergleichssatz wegen

$$\frac{\mathbf{x}^H \mathbf{B} \mathbf{x}}{\|\mathbf{x}\|_2^2} = \frac{\mathbf{x}^H \mathbf{A} \mathbf{x} + \tau |\mathbf{x}^H \mathbf{c}|^2}{\|\mathbf{x}\|_2^2} \geq \frac{\mathbf{x}^H \mathbf{A} \mathbf{x}}{\|\mathbf{x}\|_2^2}.$$

Sind $\mathbf{x}^1, \dots, \mathbf{x}^n$ die Eigenvektoren von \mathbf{B} , so gilt nach dem Prinzip von Courant-Fischer

$$\begin{aligned} \lambda_{i+1}(\mathbf{A}) &= \max_{\dim V \leq i} \min \left\{ \frac{\mathbf{x}^H \mathbf{A} \mathbf{x}}{\|\mathbf{x}\|_2^2} : \mathbf{x}^H \mathbf{y} = 0 \text{ für alle } \mathbf{y} \in V \right\} \\ &\geq \min \left\{ \frac{\mathbf{x}^H \mathbf{A} \mathbf{x}}{\|\mathbf{x}\|_2^2} : \mathbf{x}^H \mathbf{x}^j = 0, j = 1, \dots, i-1, \mathbf{x}^H \mathbf{c} = 0 \right\} \\ &= \min \left\{ \frac{\mathbf{x}^H \mathbf{A} \mathbf{x} + \tau |\mathbf{x}^H \mathbf{c}|^2}{\|\mathbf{x}\|_2^2} : \mathbf{x}^H \mathbf{x}^j = 0, j = 1, \dots, i-1, \mathbf{x}^H \mathbf{c} = 0 \right\} \\ &\geq \min \left\{ \frac{\mathbf{x}^H \mathbf{B} \mathbf{x}}{\|\mathbf{x}\|_2^2} : \mathbf{x}^H \mathbf{x}^j = 0, j = 1, \dots, i-1 \right\} = \lambda_i(\mathbf{B}) \quad \blacksquare \end{aligned}$$

Wir beweisen schließlich noch einen besonders einfach anwendbaren Einschließungssatz für Eigenwerte reeller, symmetrischer Matrizen:

Satz 7.10. (Quotienteneinschließungssatz, Collatz) Sei $\mathbf{A} \in \mathbb{R}^{(n,n)}$ symmetrisch, $\mathbf{x} \in \mathbb{R}^n$ mit von 0 verschiedenen Komponenten und $q_i := (\mathbf{A}\mathbf{x})_i/x_i$. Dann gibt es einen Eigenwert λ von $\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$ mit

$$q_- := \min_i q_i \leq \lambda \leq \max_i q_i =: q_+$$

Beweis: Es sei zunächst $q_- > 0$. Mit $\mathbf{Q} := \text{diag}\{q_i\}$ gilt offenbar $\mathbf{A}\mathbf{x} = \mathbf{Q}\mathbf{x}$, d.h. 1 ist Eigenwert der allgemeinen Eigenwertaufgabe $\mathbf{A}\mathbf{x} = \lambda\mathbf{Q}\mathbf{x}$ und \mathbf{Q} ist positiv definit. Es sei 1 der i -te Eigenwert. Für die Eigenwertaufgaben $\mathbf{A}\mathbf{x} = \lambda q_- \mathbf{x}$ und $\mathbf{A}\mathbf{x} = \lambda q_+ \mathbf{x}$ mit den Eigenwerten $\frac{\lambda_j}{q_-}$ bzw. $\frac{\lambda_j}{q_+}$, $j = 1, \dots, n$, gilt für alle $\mathbf{x} \neq \mathbf{0}$

$$\frac{\mathbf{x}^T \mathbf{A} \mathbf{x}}{q_+ \|\mathbf{x}\|_2^2} \leq \frac{\mathbf{x}^T \mathbf{A} \mathbf{x}}{\mathbf{x}^T \mathbf{Q} \mathbf{x}} \leq \frac{\mathbf{x}^T \mathbf{A} \mathbf{x}}{q_- \|\mathbf{x}\|_2^2},$$

also liefert der Vergleichssatz (Satz 7.6.)

$$\frac{\lambda_i}{q_+} \leq 1 \leq \frac{\lambda_i}{q_-} \quad \text{d.h.} \quad q_- \leq \lambda_i \leq q_+.$$

Sind nicht alle q_i positiv, so kann man durch eine Spektralverschiebung diesen Fall auf den obigen zurückführen: Wir wählen $\alpha > -q_-$, $\tilde{\mathbf{A}} := \mathbf{A} + \alpha \mathbf{I}$. Dann gilt

$$\tilde{q}_i = \frac{[(\mathbf{A} + \alpha \mathbf{I})\mathbf{x}]_i}{x_i} = q_i + \alpha > 0, \quad i = 1, \dots, n,$$

und daher besitzt nach dem bereits gezeigten $\tilde{\mathbf{A}}$ einen Eigenwert $\alpha + \lambda$ mit

$$\alpha + q_- \leq \alpha + \lambda \leq \alpha + q_+ \quad \blacksquare$$

Bemerkung 7.11. Dasselbe Ergebnis gilt übrigens für nicht notwendig symmetrische, positive (genauer: nichtnegative irreduzible) Matrizen. Setzt man dort $\mathbf{x} > \mathbf{0}$ voraus, so erhält man mit $q_- \leq \rho(\mathbf{A}) \leq q_+$ sogar eine Einschließung für den Spektralradius der Matrix. \square

Der Rayleighquotient hat für symmetrische Probleme sehr gute Approximationseigenschaften. Ist \mathbf{x} ein Eigenvektor von $\mathbf{Ax} = \lambda\mathbf{Bx}$ (\mathbf{A}, \mathbf{B} Hermitesch, \mathbf{B} positiv definit) mit zugehörigem Eigenwert λ , und gilt $\mathbf{y} = \mathbf{x} + O(\varepsilon)$, so gilt für den Rayleighquotienten

$$\frac{\mathbf{y}^H \mathbf{Ay}}{\mathbf{y}^H \mathbf{By}} = \lambda + O(\varepsilon^2) \quad \text{für } \varepsilon \rightarrow 0.$$

Da $R(\mathbf{x})$ Quotient zweier quadratischer Formen ist (und der Nenner nicht 0 wird) ist $R(\mathbf{x})$ beliebig oft differenzierbar. Aus $f(\mathbf{x}) := \mathbf{x}^H \mathbf{Ax} - R(\mathbf{x})\mathbf{x}^H \mathbf{Bx} = 0$ folgt

$$Df(\mathbf{x}) = 2\mathbf{x}^H \mathbf{A} - DR(\mathbf{x}) \cdot \mathbf{x}^H \mathbf{Bx} - 2R(\mathbf{x})\mathbf{x}^H \mathbf{B} = \mathbf{0}^T$$

d.h.

$$DR(\mathbf{x}) = \frac{2}{\mathbf{x}^H \mathbf{Bx}} (\mathbf{x}^H \mathbf{A} - R(\mathbf{x})\mathbf{x}^H \mathbf{B}).$$

Ist also \mathbf{x} Eigenvektor von $\mathbf{Ax} = \lambda\mathbf{Bx}$ mit Eigenwert $R(\mathbf{x})$, so gilt $DR(\mathbf{x}) = \mathbf{0}$ (d.h. der Rayleighquotient wird genau an den Eigenvektoren stationär), und die Behauptung folgt aus dem Taylorschen Satz:

$$|R(\mathbf{y}) - R(\mathbf{x})| \leq \frac{1}{2} \max_{0 \leq t \leq 1} \|D^2 R(\mathbf{x} + t(\mathbf{y} - \mathbf{x}))\|_2 \|\mathbf{y} - \mathbf{x}\|_2^2 = O(\varepsilon^2).$$

Auch wenn für nicht Hermitesche Probleme keine minmax-Charakterisierungen mit Hilfe des Rayleighquotienten mehr gelten und auch die obige Approximationseigenschaft nicht gilt, kann man für die spezielle Eigenwertaufgabe $R(\mathbf{x})$ im Sinne von Lemma 6.25. auffassen.

7.2 QR Algorithmus

In Abschnitt 6.4 haben wir gesehen, dass der QR Algorithmus für i.A. nichtsymmetrische Matrizen effizient gemacht werden kann durch vorherige Transformation auf Hessenberg Gestalt und durch implizite Shifts. Wenn wir eine Hermitesche Matrix unitär auf Hessenberg Gestalt transformieren, so ist das Ergebnis eine Tridiagonalmatrix. Auch diese Gestalt bleibt natürlich während des QR Algorithmus erhalten, so dass jeder QR Schritt hier noch billiger wird.

Es sei $\mathbf{A} \in \mathbb{R}^{(n,n)}$ symmetrisch und tridiagonal mit den Diagonalelementen $\alpha_1, \dots, \alpha_n$ und den Nebendiagonalelementen $\beta_1, \dots, \beta_{n-1}$. In dem folgenden Algorithmus werden α_i und β_i mit den entsprechenden Elementen von $\mathbf{A}' = \mathbf{Q}^T \mathbf{A} \mathbf{Q}$ überschrieben, wobei \mathbf{Q} der orthogonale Anteil der QR Zerlegung von $\mathbf{A} - \kappa \mathbf{I}$ ist. \mathbf{Q} wird als Produkt von Givens Matrizen $\mathbf{U}_{k,k+1}$ aufgebaut (bulge chasing!), und es werden die Multiplikationen mit $\mathbf{U}_{k,k+1}$ von links und rechts gleich zusammengefasst.

Wir benutzen in dem Algorithmus die Prozedur

$$[\gamma, \sigma, \nu] = \text{rot}(\pi, \rho)$$

durch die die Elemente $\gamma = \cos(\phi)$ und $\sigma = \sin(\phi)$ der Givens Rotation bestimmt werden, durch die der Vektor $(\pi, \rho)^T$ auf $(\nu, 0)^T$, $\nu = \sqrt{\pi^2 + \rho^2}$, gedreht wird.

Algorithmus 7.12. (QR Algorithmus für Tridiagonalmatrizen)

1. $\pi = \alpha_1 - \kappa$
2. $\rho = \beta_1$
3. $\tau = \alpha_1$
4. $\omega = \beta_1$
5. **for** $k = 1, \dots, n - 1$ **do**
 - $(\gamma, \sigma, \nu) = \text{rot}(\pi, \rho)$
 - if** $k \neq 1$ **then** $\beta_{k-1} = \nu$
 - $\alpha_k := \gamma^2 \tau + 2\gamma\sigma\omega + \sigma^2 \alpha_{k+1}$
 - $\pi := (\gamma^2 - \sigma^2)\omega + \gamma\sigma(\alpha_{k+1} - \tau)$
 - $\tau := \sigma^2 \tau - 2\gamma\sigma\omega + \gamma^2 \alpha_{k+1}$
 - if** $k < n - 1$ **then** $\rho = \sigma \beta_{k+1}$
 - if** $k < n - 1$ **then** $\omega = \gamma \beta_{k+1}$
6. $\beta_{n-1} := \pi$
7. $\alpha_n := \tau$.

Als Shift wählt man wie vorher $\kappa = \alpha_n$ oder auch κ als Eigenwert der Matrix

$$\begin{pmatrix} \alpha_{n-1} & \beta_{n-1} \\ \beta_{n-1} & \alpha_n \end{pmatrix},$$

der α_n am nächsten liegt. Dieser wird **Wilkinson Shift** genannt. Ferner wird wie vorher mit Hilfe der Deflationen die Größe der behandelten Eigenwertprobleme verkleinert.

Für diese Methode gilt der folgende Konvergenzsatz:

Satz 7.13. (Wilkinson) *Der QR Algorithmus mit Wilkinson Shifts für symmetrische Tridiagonalmatrizen ist global und wenigstens linear konvergent. Für fast alle Matrizen ist er asymptotisch kubisch konvergent.*

Beweis: Parlett [83], pp. 169 ff. ■

Beispiel 7.14. Wir betrachten die symmetrische Tridiagonalmatrix mit der Diagonale $\alpha := (1, 2, 3, 4, 5)$ und der Nebendiagonale $\beta = (2, 3, 4, 5)$. Die folgende Tabelle enthält in der zweiten Spalte das jeweilige letzte Nebendiagonalelement und in der dritten Spalte den Fehler des letzten Diagonalelements, wenn man den QR Algorithmus mit Wilkinson Shifts anwendet und Deflation ausführt, wenn ein Nebendiagonalelement nahezu 0 ist.

| Dimension n | $a_{n,n-1}$ | Fehler |
|---------------|-------------|------------|
| 5 | 5 | 5.75e0 |
| | 1.85e0 | 5.17e - 1 |
| | 2.44e - 2 | 1.04e - 4 |
| | 3.93e - 8 | 5.33e - 15 |
| | 2.22e - 24 | 0 |
| 4 | 6.73e - 1 | 1.08e - 1 |
| | 2.26e - 3 | 1.37e - 6 |
| | 4.62e - 11 | 8.89e - 16 |
| | 7.94e - 33 | 0 |
| 3 | 3.35e - 1 | 3.70e - 2 |
| | 2.84e - 4 | 2.82e - 8 |
| | 7.93e - 14 | 0 |

Man liest an den Fehlern ab, dass in jedem Schritt die Zahl der gültigen Stellen ungefähr verdreifacht wird. Dies zeigt die kubische Konvergenz. □

Der QR Algorithmus erfordert folgende Kosten für die Bestimmung aller Eigenwerte.

1. Die Reduktion der symmetrischen Matrix $\mathbf{A} \in \mathbb{R}^{(n,n)}$ auf Tridiagonalgestalt $\mathbf{T}_n = \mathbf{Q}_n^T \mathbf{A} \mathbf{Q}_n$ erfordert $\frac{4}{3}n^3 + O(n^2)$ flops.
2. Jede Bestimmung einer Rotationsmatrix erfordert 5 flops und eine Quadratwurzel (die in MATLAB auch als ein flop gezählt wird), und die Multiplikation

der Tridiagonalmatrix von links und rechts benötigt 17 flops. Ein Iterationsschritt des QR Algorithmus (ein “bulge chasing” mit einem Shift Parameter) für eine Tridiagonalmatrix mit k Zeilen und Spalten erfordert also $23k + O(1)$ flops.

- Um eine Näherung für einen Eigenwert durch das letzte Diagonalelement mit voller Stellenzahl zu erhalten, werden im Mittel zwei QR Schritte benötigt (vgl. Beispiel 7.14.). Die Berechnung aller Eigenwerte der Tridiagonalmatrix \mathbf{T}_n erfordert daher

$$2 \sum_{k=3}^n (23k + O(1)) = 23n^2 + O(n) \quad \text{flops.}$$

- Um für alle Eigenwerte auch die zugehörigen Eigenvektoren zu bestimmen, ist es am sinnvollsten, am Ende des QR Algorithmus mit der inversen Iteration die Eigenvektoren \mathbf{y}^j der Matrix \mathbf{T}_n zu bestimmen (hierzu genügt in der Regel ein Schritt, und es sind dazu $6n$ flops erforderlich, vgl. Abschnitt 4.3), und dann durch Rücktransformation die Eigenvektoren $\mathbf{x}^j := \mathbf{Q}_n \mathbf{y}^j$ zu bestimmen. Da die Multiplikation eines Vektors mit einer k -dimensionalen Householder Matrix $4k$ flops benötigt, erfordert die Rücktransformation eines Eigenvektors $2n^2$ flops. Will man also zusätzlich alle Eigenvektoren von \mathbf{A} bestimmen, so kostet dies zusätzlich zu den $\frac{4}{3}n^3 + O(n^2)$ flops zur Berechnung der Eigenwerte $2n^3 + O(n^2)$ flops.

7.3 Bisektion

Das Bisektionsverfahren basiert auf dem Trägheitssatz von Sylvester. Es kann verwendet werden, wenn nur wenige Eigenwerte benötigt werden, z.B. alle Eigenwerte unterhalb oder oberhalb eines gegebenen Schwellenwertes oder alle Eigenwerte in einem vorgegebenen Intervall.

Wir erinnern zunächst an die folgenden Begriffe:

Definition 7.15. *Es sei $\mathbf{A} \in \mathbb{C}^{(n,n)}$ eine Hermitesche Matrix. Existiert eine reguläre Matrix $\mathbf{X} \in \mathbb{C}^{(n,n)}$ mit $\mathbf{B} = \mathbf{X}^H \mathbf{A} \mathbf{X}$, so heißen \mathbf{A} und \mathbf{B} kongruent.*

*Besitzt \mathbf{A} den r -fachen Eigenwert 0, besitzt \mathbf{A} p positive Eigenwerte und q negative Eigenwerte, so heißt das Tripel (p, q, r) die **Signatur** von \mathbf{A} .*

Satz 7.16. (Trägheitssatz von Sylvester) *Kongruente Matrizen besitzen dieselbe Signatur.*

Beweis: Es sei $\mathbf{X} \in \mathbb{C}^{(n,n)}$ regulär mit $\mathbf{A} = \mathbf{X}^H \mathbf{B} \mathbf{X}$. Dass $\text{Rang } \mathbf{A} = \text{Rang } \mathbf{B}$ gilt, folgt bereits aus der Rangformel. Wir haben also nur noch zu zeigen, dass die Zahl der negativen Eigenwerte für beide Matrizen übereinstimmt.

\mathbf{A} besitze q negative Eigenwerte, es sei $\mathbf{u}^1, \dots, \mathbf{u}^q$ ein Orthonormalsystem von Eigenvektoren zu diesen Eigenwerten und $U = \text{span}\{\mathbf{u}^1, \dots, \mathbf{u}^q\}$. Dann gilt für den q kleinsten Eigenwert μ_q von \mathbf{B} mit $W = \{\mathbf{X}\mathbf{u} : \mathbf{u} \in U\}$ wegen $\dim W = q$

$$\begin{aligned} \mu_q &= \min_{\dim V=q} \max_{\mathbf{v} \in V \setminus \{\mathbf{0}\}} \frac{\mathbf{v}^H \mathbf{B} \mathbf{v}}{\mathbf{v}^H \mathbf{v}} \leq \max_{\mathbf{v} \in W \setminus \{\mathbf{0}\}} \frac{\mathbf{v}^H \mathbf{B} \mathbf{v}}{\mathbf{v}^H \mathbf{v}} = \max_{\mathbf{u} \in U \setminus \{\mathbf{0}\}} \frac{(\mathbf{X}\mathbf{u})^H \mathbf{B} \mathbf{X}\mathbf{u}}{\mathbf{u}^H \mathbf{X}^H \mathbf{X} \mathbf{u}} \\ &= \max_{\mathbf{u} \in U \setminus \{\mathbf{0}\}} \frac{\mathbf{u}^H \mathbf{A} \mathbf{u}}{\mathbf{u}^H \mathbf{u}} \frac{\mathbf{u}^H \mathbf{u}}{\mathbf{u}^H \mathbf{X}^H \mathbf{X} \mathbf{u}} < 0, \end{aligned}$$

da der Rayleigh Quotient von \mathbf{A} auf U negativ und der zweite Quotient stets positiv ist.

\mathbf{B} besitzt also nicht weniger negative Eigenwerte als \mathbf{A} . Vertauscht man die Rollen von \mathbf{A} und \mathbf{B} , so erhält man die Gleichheit. ■

Besitzt $\mathbf{A} - \mu \mathbf{I}$ für einen Parameter $\mu \in \mathbb{R}$ eine LDL^T Zerlegung

$$\mathbf{A} - \mu \mathbf{I} = \mathbf{L} \mathbf{D} \mathbf{L}^T$$

mit einer regulären Dreiecksmatrix \mathbf{L} und einer Diagonalmatrix \mathbf{D} , so haben $\mathbf{A} - \mu \mathbf{I}$ und \mathbf{D} dieselbe Signatur. Die Signatur von \mathbf{D} kann man aber leicht ablesen: p ist die Anzahl der positiven Elemente von \mathbf{D} , q ist die Anzahl der negativen Elemente und $r = n - q - p$. Daher besitzt $\mathbf{A} - \mu \mathbf{I}$ p positive und q negative Eigenwerte, und \mathbf{A} besitzt p Eigenwerte, die größer als μ sind, und q Eigenwerte, die kleiner als μ sind, und den r -fachen Eigenwert μ .

Ist die Matrix \mathbf{A} voll besetzt, so kann man die LDL^T Zerlegung von $\mathbf{A} - \mu \mathbf{I}$ mit dem Gaußschen Eliminationsverfahren (ohne Pivotsuche oder mit Diagonalpivotsuche) bestimmen. Dies erfordert $O(n^3)$ flops, und ist daher viel zu teuer. Ist \mathbf{A} bereits auf Tridiagonalgestalt transformiert und besitzt $\mathbf{A} - \mu \mathbf{I}$ eine LDL^T Zerlegung, so gilt (vgl. Abschnitt 4.3)

$$\mathbf{L} \mathbf{D} \mathbf{L}^T = \begin{pmatrix} 1 & & & & \\ \ell_1 & 1 & & & \\ & \ddots & \ddots & & \\ & & \ell_{n-1} & 1 & \\ & & & & 1 \end{pmatrix} \begin{pmatrix} d_1 & & & & \\ & d_2 & & & \\ & & \ddots & & \\ & & & \ddots & \\ & & & & d_n \end{pmatrix} \begin{pmatrix} 1 & \ell_1 & & & \\ & 1 & \ddots & & \\ & & \ddots & \ell_{n-1} & \\ & & & & 1 \end{pmatrix},$$

und durch Vergleich mit den Elementen von

$$\mathbf{A} - \mu \mathbf{I} =: \begin{pmatrix} \alpha_1 - \mu & \beta_1 & & & \\ \beta_1 & \alpha_2 - \mu & \beta_2 & & \\ & & \ddots & \ddots & \beta_{n-1} \\ & & & \beta_{n-1} & \alpha_n - \mu \end{pmatrix}$$

erhält man $\alpha_1 - \mu = d_1$, $\beta_1 = d_1 \ell_1$ und für die weiteren i

$$\alpha_i - \mu = \ell_{i-1}^2 d_{i-1} + d_i, \quad \beta_i = \ell_i d_i,$$

d.h.

$$d_1 = \alpha_1 - \mu, \quad d_i = \alpha_i - \mu - \frac{\beta_{i-1}^2}{d_{i-1}}, \quad i = 2, \dots, n. \quad (7.12)$$

Eine Auswertung erfordert $4n$ flops. Es kann gezeigt werden, dass die Rekursion (7.12) stabil ist (vgl. Demmel [23], p. 230).

Mit (7.12) erhält man die folgende **Bisektionsmethode** zur Bestimmung aller Eigenwerte einer (Tridiagonal-)Matrix \mathbf{A} im Intervall $[a, b]$. Dabei bezeichnen wir mit $n(\mu)$ die Anzahl der Eigenwerte von \mathbf{A} , die kleiner als μ sind.

Algorithmus 7.17. (Bisektionsverfahren)

```

Bestimme n(a) und n(b)
if n(a)==n(b) quit: Keine Eigenwerte in [a,b]; end
Lege [a,n(a),b,n(b)] auf Worklist
While Worklist not empty do
  Get [c,n(c),d,n(d)] from Worklist
  If d-c<tol
    print: [c,d] enthaelt n(d)-n(c) Eigenwerte
  else
    e=(c+d)/2;
    Berechne n(e)
    if n(e)>n(c)
      put [c,n(c),e,n(e)] onto Worklist; end
    if n(d)>n(e)
      put [e,n(e),d,n(d)] onto Worklist; end
    end
  end
end
end

```

Der Trägheitssatz von Sylvester kann auch verwendet werden, um (einen Teil der) Eigenwerte einer allgemeinen Eigenwertaufgabe

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{B}\mathbf{x}, \quad (7.13)$$

$\mathbf{A}, \mathbf{B} \in \mathbb{R}^{(n,n)}$ symmetrisch, \mathbf{B} positiv definit, zu lokalisieren und mit einem Bisektionsverfahren einzuschließen.

Sind $\lambda_1 \leq \dots \leq \lambda_n$ die Eigenwerte von (7.13), so besitzt für festes $\mu \in \mathbb{R}$ die Eigenwertaufgabe

$$(\mathbf{A} - \mu\mathbf{B})\mathbf{x} = \nu\mathbf{B}\mathbf{x}$$

die Eigenwerte $\nu_j = \lambda_j - \mu$, $j = 1, \dots, n$, und ist äquivalent der speziellen Eigenwertaufgabe

$$\mathbf{F}\mathbf{y} := \mathbf{C}^{-1}(\mathbf{A} - \mu\mathbf{B})\mathbf{C}^{-T}\mathbf{y} = \nu\mathbf{y},$$

wobei $\mathbf{B} = \mathbf{C}\mathbf{C}^T$ die Cholesky Zerlegung von \mathbf{B} ist.

Ist (p, q, r) die Signatur von $\mathbf{A} - \mu\mathbf{B}$ (die man wieder mit der LDL^T Zerlegung von $\mathbf{A} - \mu\mathbf{B}$ bestimmen kann), so besitzt \mathbf{F} q negative Eigenwerte, und da die Matrizen \mathbf{F} und $\mathbf{A} - \mu\mathbf{B}$ kongruent sind, besitzt die allgemeine Eigenwertaufgabe (7.13) q Eigenwerte, die kleiner als μ sind.

7.4 Rayleigh Quotienten Iteration

Wir betrachten (vgl. Abschnitt 6.2) die inverse Iteration, wobei wir den Shift Parameter mit Hilfe des Rayleigh Quotienten iterieren. Sei $\mathbf{A} \in \mathbb{R}^{(n,n)}$ symmetrisch und \mathbf{x}^0 ein Startvektor, den wir als normiert annehmen: $\|\mathbf{x}^0\|_2 = 1$.

Algorithmus 7.18. (Rayleigh Quotienten Iteration)

```

ρ_0=x_0'*A*x_0;
for m=1,2,... until convergence do
    L\{0}se (A-ρ_{m-1} I)y_m=x_{m-1};
    x_m=y_m/||y_m||_2;
    ρ_m=x_m'*A*x_m;
end

```

Löst man das lineare Gleichungssystem in Algorithmus 7.18. mit Hilfe des Gaußschen Eliminationsverfahrens (mit Diagonalpivotisierung), so erhält man gleichzeitig durch

die Anzahl der negativen Diagonalelemente in dem oberen Dreiecksfaktor \mathbf{R} die Information, wieviele Eigenwerte von \mathbf{A} unterhalb von ρ_{m-1} liegen (vgl. Satz 7.16.).

Um die Konvergenzgeschwindigkeit der Rayleigh Quotienten Iteration zu untersuchen, können wir ohne Beschränkung der Allgemeinheit annehmen, dass die Matrix

$$\mathbf{A} = \text{diag} \{ \lambda_1, \dots, \lambda_n \}$$

Diagonalgestalt besitzt. Denn sonst können wir mit einer orthogonalen Matrix \mathbf{Q} die Matrix \mathbf{A} auf Diagonalgestalt transformieren, $\mathbf{Q}^T \mathbf{A} \mathbf{Q} = \mathbf{\Lambda}$. Führt man dann die Rayleigh Quotienten Iteration für $\mathbf{\Lambda}$ mit dem Startwert $\tilde{\mathbf{x}}^0 := \mathbf{Q}^T \mathbf{x}^0$ aus und bestimmt hiermit die Folge $\tilde{\mathbf{x}}^m$, so gilt für $\mathbf{x}^m := \mathbf{Q} \tilde{\mathbf{x}}^m$

$$\rho_m = \frac{(\tilde{\mathbf{x}}^m)^T \mathbf{\Lambda} \tilde{\mathbf{x}}^m}{\|\tilde{\mathbf{x}}^m\|_2^2} = \frac{(\mathbf{Q}^T \mathbf{x}^m)^T \mathbf{\Lambda} \mathbf{Q}^T \mathbf{x}^m}{\|\mathbf{Q}^T \mathbf{x}^m\|_2^2} = \frac{(\mathbf{x}^m)^T \mathbf{A} \mathbf{x}^m}{\|\mathbf{x}^m\|_2^2}$$

und für $\mathbf{y}^m := \mathbf{Q} \tilde{\mathbf{y}}^m$, $\tilde{\mathbf{y}}^m := (\mathbf{\Lambda} - \rho_{m-1} \mathbf{I})^{-1} \tilde{\mathbf{x}}^{m-1}$

$$\begin{aligned} \mathbf{y}^{m+1} = \mathbf{Q} \tilde{\mathbf{y}}^{m+1} &= \mathbf{Q} (\mathbf{\Lambda} - \rho_m \mathbf{I})^{-1} \tilde{\mathbf{x}}^m = \mathbf{Q} (\mathbf{\Lambda} - \rho_m \mathbf{I})^{-1} \mathbf{Q}^T \mathbf{x}^m \\ &= (\mathbf{Q} \mathbf{\Lambda} \mathbf{Q}^T - \rho_m \mathbf{Q} \mathbf{Q}^T)^{-1} \mathbf{x}^m = (\mathbf{A} - \rho_m \mathbf{I})^{-1} \mathbf{x}^m \end{aligned}$$

Satz 7.19. *Die Rayleigh Quotienten Iteration konvergiert lokal kubisch.*

Beweis: Wir beschränken uns auf den Fall, dass der Eigenwert, in dessen Umgebung wir die Konvergenzgeschwindigkeit abschätzen, einfach ist. Für mehrfache Eigenwerte muss der Beweis modifiziert werden.

Es sei $\mathbf{A} = \text{diag} \{ \lambda_1, \dots, \lambda_n \}$ eine Diagonalmatrix, und es konvergiere die erzeugte Folge \mathbf{x}^m gegen \mathbf{e}^1 (bei Konvergenz gegen einen anderen Eigenvektor ordnen wir \mathbf{A} um). Es sei $\mathbf{x}^m =: \mathbf{e}^1 + \mathbf{d}^m$ mit $\varepsilon := \|\mathbf{d}^m\|_2 \ll 1$. Dann gilt

$$1 = \|\mathbf{x}^m\|_2^2 = (\mathbf{e}^1 + \mathbf{d}^m)^T (\mathbf{e}^1 + \mathbf{d}^m) = \|\mathbf{e}^1\|_2^2 + 2(\mathbf{e}^1)^T \mathbf{d}^m + \|\mathbf{d}^m\|_2^2 = 1 + 2d_1^m + \varepsilon^2,$$

d.h.

$$d_1^m = -\frac{1}{2} \varepsilon^2,$$

und daher

$$\begin{aligned} \rho_m &= (\mathbf{x}^m)^T \mathbf{A} \mathbf{x}^m = (\mathbf{e}^1)^T \mathbf{A} \mathbf{e}^1 + 2(\mathbf{e}^1)^T \mathbf{A} \mathbf{d}^m + (\mathbf{d}^m)^T \mathbf{A} \mathbf{d}^m \\ &= \lambda_1 + 2\lambda_1 d_1^m + (\mathbf{d}^m)^T \mathbf{A} \mathbf{d}^m = \lambda_1 - \lambda_1 \varepsilon^2 + (\mathbf{d}^m)^T \mathbf{A} \mathbf{d}^m =: \lambda_1 - \eta. \end{aligned}$$

Mit $\mu := \|\mathbf{A}\|_2 = \max\{|\lambda_j| : j = 1, \dots, m\}$ gilt

$$|\eta| \leq |\lambda_1| \varepsilon^2 + \|\mathbf{A}\|_2 \|\mathbf{d}^m\|_2^2 \leq 2\mu \varepsilon^2, \quad (7.14)$$

und es folgt

$$\begin{aligned}
 \mathbf{y}^{m+1} &= (\mathbf{A} - \rho_m \mathbf{I})^{-1} \mathbf{x}^m \\
 &= \left(\frac{x_1^m}{\lambda_1 - \rho_m}, \frac{x_2^m}{\lambda_2 - \rho_m}, \dots, \frac{x_n^m}{\lambda_n - \rho_m} \right)^T \\
 &= \left(\frac{1 + d_1^m}{\lambda_1 - \rho_m}, \frac{d_2^m}{\lambda_2 - \rho_m}, \dots, \frac{d_n^m}{\lambda_n - \rho_m} \right)^T \\
 &= \left(\frac{1 - 0.5\varepsilon^2}{\eta}, \frac{d_2^m}{\lambda_2 - \rho_m}, \dots, \frac{d_n^m}{\lambda_n - \rho_m} \right)^T.
 \end{aligned}$$

Mit $\sigma := (1 - 0.5\varepsilon^2)/\eta$ gilt also

$$\mathbf{y}^{m+1} = \sigma \left(1, \frac{d_2^m}{\sigma(\lambda_2 - \lambda_1 + \eta)}, \dots, \frac{d_n^m}{\sigma(\lambda_n - \lambda_1 + \eta)} \right)^T =: \sigma(\mathbf{e}^1 + \tilde{\mathbf{d}}^{m+1}).$$

Da λ_1 ein einfacher Eigenwert ist, gilt $\gamma := \min_{j=2, \dots, n} |\lambda_1 - \lambda_j| > 0$. Hiermit kann man jeden der Nenner in $\tilde{\mathbf{d}}^{m+1}$ dem Betrage nach abschätzen durch $|\lambda_j - \lambda_1 + \eta| \geq \gamma - |\eta|$, und man erhält

$$\|\tilde{\mathbf{d}}^{m+1}\|_2 \leq \frac{\|\mathbf{d}^m\|_2 |\eta|}{(1 - 0.5\varepsilon^2)(\gamma - |\eta|)} \leq \frac{2\mu\varepsilon^3}{(1 - 0.5\varepsilon^2)(\gamma - 2\mu\varepsilon^2)}.$$

Es gilt also $\|\tilde{\mathbf{d}}^{m+1}\|_2 = O(\varepsilon^3)$, und wegen

$$\mathbf{x}^{m+1} = \mathbf{e}^1 + \mathbf{d}^{m+1} = \frac{\mathbf{e}^1 + \tilde{\mathbf{d}}^{m+1}}{\|\mathbf{e}^1 + \tilde{\mathbf{d}}^{m+1}\|_2}$$

gilt auch $\|\mathbf{d}^{m+1}\|_2 = O(\varepsilon^3)$. ■

Bemerkung 7.20. Die Ungleichung (7.14) zeigt noch einmal die Approximationsgüte des Rayleigh Quotienten: Weicht der Vektor, an dem der Rayleigh Quotient ausgewertet wird, um ε von einem Eigenvektor ab, so weicht der Rayleigh Quotient von dem zugehörigen Eigenwert um $O(\varepsilon^2)$ ab. □

7.5 Divide-and-Conquer Verfahren

Das Divide-and-Conquer Verfahren ist (nach heutiger Kenntnis) für symmetrische Tridiagonalmatrizen, deren Dimension größer als ungefähr 25 ist (die genaue Grenze ist rechnerabhängig), die effizienteste Methode, wenn neben den Eigenwerten auch alle Eigenvektoren gesucht sind. Es wurde von Cuppen [21] 1981 eingeführt. Der

effizienten Implementierung liegt eine Arbeit von Gu und Eisenstat [52] aus dem Jahr 1995 zu Grunde.

Das Verfahren beruht auf der folgenden Überlegung: Es gilt

$$\begin{aligned}
 \mathbf{T} &= \left(\begin{array}{cccc|cccc}
 \alpha_1 & \beta_1 & & & & & & \\
 \beta_1 & \alpha_2 & \beta_2 & & & & & \\
 & \ddots & \ddots & \ddots & & & & \\
 & & \ddots & \alpha_{m-1} & \beta_{m-1} & & & \\
 & & & \beta_{m-1} & \alpha_m & \beta_m & & \\
 \hline
 & & & & \beta_m & \alpha_{m+1} & \beta_{m+1} & \\
 & & & & & \beta_{m+1} & \alpha_{m+2} & \beta_{m+2} \\
 & & & & & & \ddots & \ddots & \ddots \\
 & & & & & & & \ddots & \alpha_{n-1} & \beta_{n-1} \\
 & & & & & & & & \beta_{n-1} & \alpha_n
 \end{array} \right) \\
 &= \left(\begin{array}{cccc|cccc}
 \alpha_1 & \beta_1 & & & & & & \\
 \beta_1 & \alpha_2 & \beta_2 & & & & & \\
 & \ddots & \ddots & \ddots & & & & \\
 & & \ddots & \alpha_{m-1} & \beta_{m-1} & & & \\
 & & & \beta_{m-1} & \alpha_m - \beta_m & & & \\
 \hline
 & & & & & \alpha_{m+1} - \beta_m & \beta_{m+1} & \\
 & & & & & \beta_{m+1} & \alpha_{m+2} & \beta_{m+2} \\
 & & & & & & \ddots & \ddots & \ddots \\
 & & & & & & & \ddots & \alpha_{n-1} & \beta_{n-1} \\
 & & & & & & & & \beta_{n-1} & \alpha_n
 \end{array} \right) \\
 &+ \beta_m \mathbf{v} \mathbf{v}^T = \left(\begin{array}{c|c}
 \mathbf{T}_1 & \mathbf{O} \\
 \hline
 \mathbf{O} & \mathbf{T}_2
 \end{array} \right) + \beta_m \mathbf{v} \mathbf{v}^T
 \end{aligned}$$

mit $\mathbf{v} := (0, \dots, 0, 1, 1, 0, \dots, 0)^T$.

Sind bereits für die Matrizen \mathbf{T}_1 und \mathbf{T}_2 die Eigenwertprobleme gelöst, sind also orthogonale Matrizen \mathbf{Q}_i und Diagonalmatrizen Λ_i bekannt mit $\mathbf{T}_i = \mathbf{Q}_i \Lambda_i \mathbf{Q}_i^T$, $i = 1, 2$, so gilt

$$\begin{aligned}
 \mathbf{T} &= \left(\begin{array}{cc|cc}
 \mathbf{Q}_1 \Lambda_1 \mathbf{Q}_1^T & \mathbf{O} & & \\
 \mathbf{O} & \mathbf{Q}_2 \Lambda_2 \mathbf{Q}_2^T & & \\
 \hline
 & & &
 \end{array} \right) + \beta_m \mathbf{v} \mathbf{v}^T \\
 &= \left(\begin{array}{cc}
 \mathbf{Q}_1 & \mathbf{O} \\
 \mathbf{O} & \mathbf{Q}_2
 \end{array} \right) \left(\left(\begin{array}{cc}
 \Lambda_1 & \mathbf{O} \\
 \mathbf{O} & \Lambda_2
 \end{array} \right) + \beta_m \mathbf{u} \mathbf{u}^T \right) \left(\begin{array}{cc}
 \mathbf{Q}_1^T & \mathbf{O} \\
 \mathbf{O} & \mathbf{Q}_2^T
 \end{array} \right),
 \end{aligned}$$

mit (man beachte die Gestalt von \mathbf{v})

$$\mathbf{u} = \begin{pmatrix} \mathbf{Q}_1^T & \mathbf{O} \\ \mathbf{O} & \mathbf{Q}_2^T \end{pmatrix} \mathbf{v} = \begin{pmatrix} \text{letzte Spalte von } \mathbf{Q}_1^T \\ \text{erste Spalte von } \mathbf{Q}_2^T \end{pmatrix}.$$

Die Eigenwerte stimmen also überein mit denen einer Rang-1-Modifikation einer Diagonalmatrix. Wir untersuchen daher die Eigenwerte und Eigenvektoren von $\mathbf{D} + \rho \mathbf{u} \mathbf{u}^T$ mit einer Diagonalmatrix $\mathbf{D} := \text{diag}\{d_1, \dots, d_n\}$, $\mathbf{u} \in \mathbb{R}^n$ und $\rho \in \mathbb{R}$. Der Einfachheit halber nehmen wir dabei an, dass die Diagonalelemente von \mathbf{D} der Größe nach geordnet sind: $d_1 \leq d_2 \leq \dots \leq d_n$.

Ist λ kein Eigenwert von \mathbf{D} , so gilt

$$\det(\mathbf{D} + \rho \mathbf{u} \mathbf{u}^T - \lambda \mathbf{I}) = \det(\mathbf{D} - \lambda \mathbf{I}) \det(\mathbf{I} + \rho(\mathbf{D} - \lambda \mathbf{I})^{-1} \mathbf{u} \mathbf{u}^T). \quad (7.15)$$

Die Matrix im zweiten Faktor ist eine Rang-1-Modifikation der Identität. Daher kann man ihre Determinante mit Hilfe des folgenden Lemmas berechnen:

Lemma 7.21. Für $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ gilt

$$\det(\mathbf{I} + \mathbf{x} \mathbf{y}^T) = 1 + \mathbf{y}^T \mathbf{x}.$$

Beweis: Wir nehmen an, dass $\mathbf{x} \neq \mathbf{0}$ und $\mathbf{y} \neq \mathbf{0}$ gilt, denn sonst ist die Aussage trivial. Wir verwenden, dass die Determinante einer Matrix gleich dem Produkt ihrer Eigenwerte ist.

Ist $\mathbf{z} \in \mathbb{R}^n$ mit $\mathbf{y}^T \mathbf{z} = 0$, so gilt $(\mathbf{I} + \mathbf{x} \mathbf{y}^T) \mathbf{z} = \mathbf{z}$, und 1 ist (wenigstens) $(n-1)$ -facher Eigenwert von $\mathbf{I} + \mathbf{x} \mathbf{y}^T$.

\mathbf{x} ist wegen $(\mathbf{I} + \mathbf{x} \mathbf{y}^T) \mathbf{x} = (1 + \mathbf{y}^T \mathbf{x}) \mathbf{x}$ ebenfalls ein Eigenvektor von $\mathbf{I} + \mathbf{x} \mathbf{y}^T$ zum Eigenwert $1 + \mathbf{y}^T \mathbf{x}$.

Ist $\mathbf{y}^T \mathbf{x} \neq 0$, so sind alle Eigenwerte bestimmt, und das Produkt der Eigenwerte ist $1 + \mathbf{y}^T \mathbf{x}$. Im Fall $\mathbf{y}^T \mathbf{x} = 0$ gilt für $\mathbf{B} := \mathbf{I} + \mathbf{x} \mathbf{y}^T - 1 \cdot \mathbf{I} = \mathbf{x} \mathbf{y}^T$

$$\mathbf{B} \mathbf{y} = \|\mathbf{y}\|_2^2 \mathbf{x} \neq \mathbf{0} \quad \text{und} \quad \mathbf{B}^2 \mathbf{y} = \|\mathbf{y}\|_2^2 \mathbf{B} \mathbf{x} = \mathbf{0}.$$

Daher ist \mathbf{y} ein Hauptvektor zweiter Stufe, und 1 ist ein Eigenwert der algebraischen Vielfachheit n . Auch in diesem Fall gilt also $\det(\mathbf{I} + \mathbf{x} \mathbf{y}^T) = 1 = 1 + \mathbf{y}^T \mathbf{x}$. ■

Mit Lemma 7.21. erhalten wir

$$\begin{aligned} \det(\mathbf{I} + \rho(\mathbf{D} - \lambda \mathbf{I})^{-1} \mathbf{u} \mathbf{u}^T) &= 1 + \rho \mathbf{u}^T (\mathbf{D} - \lambda \mathbf{I})^{-1} \mathbf{u} \\ &= 1 + \rho \sum_{j=1}^n \frac{u_j^2}{d_j - \lambda} =: f(\lambda), \end{aligned} \quad (7.16)$$

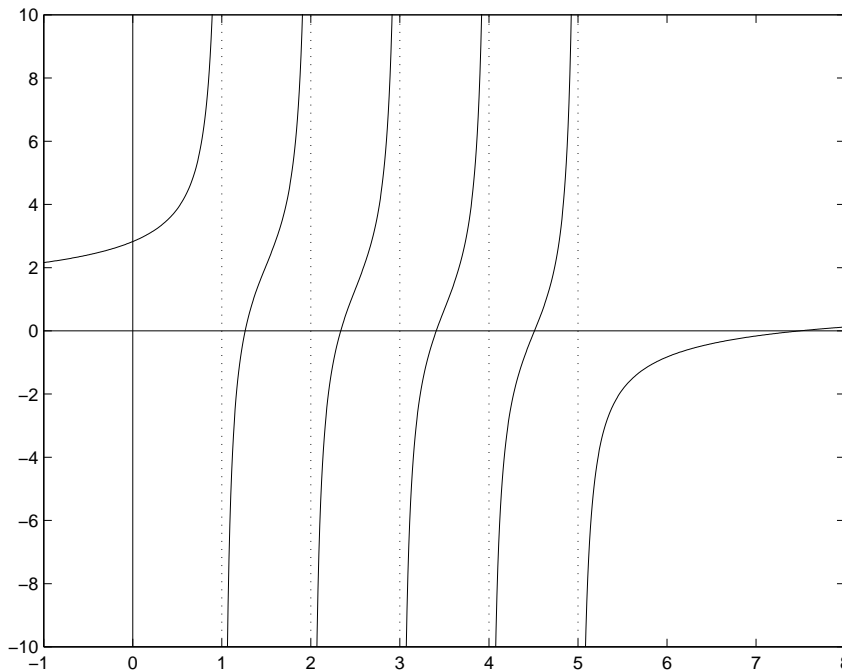


Abbildung 7.1: Sekulargleichung

Die Eigenwerte von $\mathbf{D} + \rho \mathbf{u} \mathbf{u}^T$ sind also die Nullstellen der Sekulargleichung $f(\lambda) = 0$. Sind alle d_j voneinander verschieden und alle $u_j \neq 0$ (dies ist der generische Fall), so besitzt sie einen Graphen wie die Funktion

$$f(\lambda) = 1 + \sum_{j=1}^5 \frac{0.8}{j - \lambda}$$

in Abbildung 7.1.

f besitzt in den Eigenwerten d_j von \mathbf{D} einfache Pole, ist wegen

$$f'(\lambda) = \rho \sum_{j=1}^n \frac{u_j^2}{(d_j - \lambda)^2}$$

für $\rho > 0$ überall streng monoton wachsend und für $\rho < 0$ streng monoton fallend und erfüllt $\lim_{\lambda \rightarrow \pm\infty} f(\lambda) = 1$. Daher besitzt f in jedem der Intervalle (d_{j-1}, d_j) , $j = 2, \dots, n$, genau eine Nullstelle und eine zusätzliche Nullstelle $\lambda > d_n$ im Fall $\rho > 0$ bzw. $\lambda < d_1$ im Fall $\rho < 0$.

Im Prinzip kann man die Nullstellen mit dem Newton Verfahren bestimmen. In vielen Fällen hat jedoch die Sekulargleichung einen Graphen wie in Abbildung 7.2 die Funktion

$$f(\lambda) = 1 + \sum_{j=1}^5 \frac{0.01}{j - \lambda}.$$

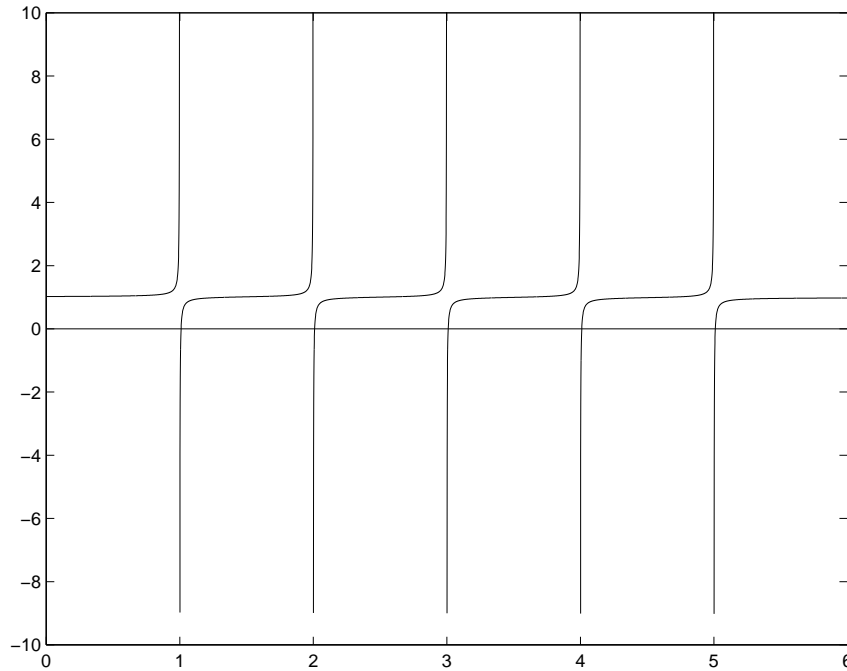


Abbildung 7.2: Sekulargleichung mit kleinen Koeffizienten

Es ist klar, dass in diesem Fall die Newton Näherung für die meisten Startwerte des Intervalls (d_{j-1}, d_j) dieses Intervall verlassen wird oder, wenn der Startwert sehr nahe an der rechten Intervallgrenze liegt, keine wesentliche Verbesserung liefert. Da die benachbarten Pole bei der Eigenwertsuche in dem Intervall (d_{j-1}, d_j) bereits bekannt sind, liegt es hier näher, die Sekulargleichung zu approximieren durch die rationale Funktion

$$g(\lambda) = \frac{c_1}{d_j - \lambda} + \frac{c_2}{d_{j+1} - \lambda} + c_3$$

mit noch zu bestimmenden Parametern c_1 , c_2 und c_3 .

Ein solches Vorgehen wurde erstmals von Bunch, Nielsen und Sorensen [15] verwendet, um die Eigenwerte von Rang-1-Modifikationen symmetrischer Matrizen zu bestimmen.

Es sei $\mu \in (d_{j-1}, d_j)$ eine Näherung für den Eigenwert in (d_{j-1}, d_j) . Um Auslöschungen zu vermeiden, zerlegen wir die Sekulargleichung in

$$f(\lambda) = 1 + \rho \sum_{k=1}^{j-1} \frac{u_k^2}{d_k - \lambda} + \rho \sum_{k=j}^n \frac{u_k^2}{d_k - \lambda} =: 1 + f_-(\lambda) + f_+(\lambda).$$

Dann werden für $\lambda \in (d_{j-1}, d_j)$ in der ersten Summe f_- nur negative Terme aufsummiert und in der zweiten Summe f_+ nur positive Terme.

Wir approximieren f_- und f_+ durch rationale Funktionen

$$g_-(\lambda) = a_- + \frac{b_-}{d_{j-1} - \lambda} \quad \text{und} \quad g_+(\lambda) = a_+ + \frac{b_+}{d_j - \lambda},$$

so dass die Graphen von g_{\pm} diejenigen von f_{\pm} in dem Punkt μ berühren, d.h.

$$g_{\pm}(\mu) = f_{\pm}(\mu) \quad \text{und} \quad g'_{\pm}(\mu) = f'_{\pm}(\mu).$$

Die Forderungen

$$g_-(\mu) = a_- + \frac{b_-}{d_{j-1} - \mu} = f_-(\mu), \quad g'_-(\mu) = \frac{b_-}{(d_{j-1} - \mu)^2} = f'_-(\mu)$$

liefern

$$b_- = (d_{j-1} - \mu)^2 f'_-(\mu), \quad a_- = f_-(\mu) - (d_{j-1} - \mu) f'_-(\mu), \quad (7.17)$$

und genauso erhält man

$$b_+ = (d_j - \mu)^2 f'_+(\mu), \quad a_+ = f_+(\mu) - (d_j - \mu) f'_+(\mu).$$

Als (hoffentlich) verbesserte Näherung für die gesuchte Nullstelle von f in (d_{j-1}, d_j) wählen wir nun die Nullstelle der approximierenden Funktion

$$g(\lambda) := 1 + g_-(\lambda) + g_+(\lambda) = 1 + a_- + a_+ + \frac{b_-}{d_{j-1} - \lambda} + \frac{b_+}{d_j - \lambda}$$

in diesem Intervall.

Wegen $f'_{\pm}(\mu) > 0$ gilt $b_{\pm} > 0$, und damit $g'_{\pm}(\lambda) > 0$ für $\lambda \in (d_{j-1}, d_j)$. Es ist also g streng monoton wachsend, und wegen $\lim_{\lambda \rightarrow d_{j-1}+0} g(\lambda) = -\infty$ und $\lim_{\lambda \rightarrow d_j-0} g(\lambda) = \infty$ besitzt g in (d_{j-1}, d_j) genau eine Nullstelle. Diese kann man leicht bestimmen, denn $g(\lambda) = 0$ ist äquivalent einer quadratischen Gleichung.

Verwendet man dieses Vorgehen iterativ, so kann man die Nullstellen der Sekulargleichung in den Intervallen (d_{j-1}, d_j) , $j = 2, \dots, n$, schnell bestimmen. Die Nullstelle in (d_n, ∞) erhält man unter Verwendung der Approximation $g(\lambda) := 1 + g_-(\lambda)$ von f , d.h. mit der Iteration

$$\mu_{k+1} = d_n + \frac{b_-}{1 + a_-},$$

wobei a_- und b_- wie in (7.17) definiert sind.

Beispiel 7.22. Wir bestimmen mit dem oben beschriebenen Verfahren die Nullstellen von

$$f(\lambda) = 1 + 0.01 \sum_{j=1}^5 \frac{1}{j - \lambda}$$

in dem Intervall $(3, 4)$ und rechts von $d_5 = 5$. Mit dem (unsinnigen) Startwert $\mu = 3.99$ erhält man die folgenden Iterierten

| k | μ_k | Fehler |
|-----|-------------------|--------------|
| 0 | 3.99 | $9.80e - 1$ |
| 1 | 3.013368340670433 | $3.37e - 3$ |
| 2 | 3.009997638254358 | $1.37e - 7$ |
| 3 | 3.009997501037015 | $2.22e - 15$ |

Für die größte Nullstelle erhält man mit dem Startwert $\mu_0 = 6$ die Näherungen

| k | μ_k | Fehler |
|-----|-------------------|-------------------------|
| 0 | 6 | $9.90e - 1$ |
| 1 | 5.014757078160140 | $4.55e - 3$. \square |
| 2 | 5.010211553344681 | $2.91e - 7$ |
| 3 | 5.010211262663904 | $8.88e - 16$ |

Das folgende Lemma gibt eine Möglichkeit zur Bestimmung der Eigenvektoren einer Rang-1-Modifikation einer Diagonalmatrix.

Lemma 7.23. *Ist λ ein Eigenwert von $\mathbf{D} + \rho\mathbf{u}\mathbf{u}^T$, so ist $\mathbf{w} = (\mathbf{D} - \lambda\mathbf{I})^{-1}\mathbf{u}$ ein zugehöriger Eigenvektor.*

Beweis: Es ist

$$\begin{aligned} (\mathbf{D} + \rho\mathbf{u}\mathbf{u}^T)(\mathbf{D} - \lambda\mathbf{I})^{-1}\mathbf{u} &= (\mathbf{D} - \lambda\mathbf{I} + \lambda\mathbf{I} + \rho\mathbf{u}\mathbf{u}^T)(\mathbf{D} - \lambda\mathbf{I})^{-1}\mathbf{u} \\ &= \mathbf{u} + \lambda(\mathbf{D} - \lambda\mathbf{I})^{-1}\mathbf{u} + \rho\mathbf{u}^T(\mathbf{D} - \lambda\mathbf{I})^{-1}\mathbf{u} \cdot \mathbf{u}, \end{aligned}$$

und wegen

$$f(\lambda) = 1 + \rho\mathbf{u}^T(\mathbf{D} - \lambda\mathbf{I})^{-1}\mathbf{u} = 0$$

erhält man

$$(\mathbf{D} + \rho\mathbf{u}\mathbf{u}^T)(\mathbf{D} - \lambda\mathbf{I})^{-1}\mathbf{u} = \lambda(\mathbf{D} - \lambda\mathbf{I})^{-1}\mathbf{u}. \quad \blacksquare$$

Die Bestimmung eines einzelnen Eigenvektors kostet, da \mathbf{D} eine Diagonalmatrix ist, $O(n)$ flops, und daher ist der Aufwand zur Bestimmung aller Eigenvektoren $O(n^2)$ flops. Nachteil dieser Formel ist aber, dass sie instabil ist: Liegen zwei Eigenwerte sehr nahe beieinander, so sind die errechneten Eigenvektoren in der Regel nicht orthogonal. Einen Ausweg bietet der folgende Satz

Satz 7.24. (Löwner) Es sei $\mathbf{D} = \text{diag}\{d_1, \dots, d_n\}$ eine Diagonalmatrix mit $d_1 < d_2 < \dots < d_n$, und es gelte für die Zahlen λ_j die Schachtelungseigenschaft

$$d_1 < \lambda_1 < d_2 < \lambda_2 < d_3 < \dots < \lambda_{n-1} < d_n < \lambda_n.$$

Es sei $\tilde{\mathbf{u}} \in \mathbb{R}^n$, und es gelte für die Komponenten

$$|\tilde{u}_j| = \sqrt{\frac{\prod_{k=1}^n (\lambda_k - d_j)}{\prod_{k=1, k \neq j}^n (d_k - d_j)}}.$$

Dann sind $\lambda_1, \dots, \lambda_n$ die Eigenwerte der Matrix $\tilde{\mathbf{D}} := \mathbf{D} + \tilde{\mathbf{u}}\tilde{\mathbf{u}}^T$.

Beweis: Es seien $\tilde{\lambda}_1 \leq \tilde{\lambda}_2 \leq \dots \leq \tilde{\lambda}_n$ die Eigenwerte von $\tilde{\mathbf{D}}$. Dann gilt für das charakteristische Polynom von $\tilde{\mathbf{D}}$

$$\chi(\lambda) = \det(\tilde{\mathbf{D}} - \lambda \mathbf{I}) = \prod_{k=1}^n (\tilde{\lambda}_k - \lambda).$$

Unter Benutzung der Sekulargleichung kann $\chi(\lambda)$ geschrieben werden als

$$\begin{aligned} \chi(\lambda) &= \left(\prod_{k=1}^n (d_k - \lambda) \right) \left(1 + \sum_{k=1}^n \frac{\tilde{u}_k^2}{d_k - \lambda} \right) \\ &= \left(\prod_{k=1}^n (d_k - \lambda) \right) \left(1 + \sum_{\substack{k=1 \\ k \neq j}}^n \frac{\tilde{u}_k^2}{d_k - \lambda} \right) + \left(\prod_{\substack{k=1 \\ k \neq j}}^n (d_k - \lambda) \right) \tilde{u}_j^2, \end{aligned}$$

und für $\lambda = d_j$ erhält man hieraus

$$\prod_{k=1}^n (\tilde{\lambda}_k - d_j) = \tilde{u}_j^2 \prod_{\substack{k=1 \\ k \neq j}}^n (d_k - d_j) = \prod_{k=1}^n (\lambda_k - d_j).$$

Damit stimmen die Polynome $\chi(\lambda)$ und $\psi(\lambda) = \prod_{k=1}^n (\lambda_k - \lambda)$ vom Grade n in den n Punkten d_j , $j = 1, \dots, n$, überein, und die eindeutige Lösbarkeit des Interpolationsproblems liefert $\chi(\lambda) = \psi(\lambda)$. ■

Bemerkung 7.25. Aus dem Beweis liest man ab, dass die Komponenten \tilde{u}_i dem Betrage nach eindeutig durch die Eigenwerte von $\mathbf{D} + \tilde{\mathbf{u}}\tilde{\mathbf{u}}^T$ festgelegt sind.

Man kann ferner zeigen: Sind λ_j die mit der Sekulargleichung bestimmten Eigenwerte von $\mathbf{D} + \rho \mathbf{u}\mathbf{u}^T$ und ist $\tilde{\mathbf{u}}$ mit dem Satz von Löwner bestimmt und gilt $\text{sign}(u_i) = \text{sign}(\tilde{u}_i)$ für alle i , so gilt

$$\|\rho \mathbf{u}\mathbf{u}^T - \tilde{\mathbf{u}}\tilde{\mathbf{u}}^T\|_2 \leq O(\varepsilon ps)(\|\mathbf{D}\|_2 + |\rho| \cdot \|\mathbf{u}\mathbf{u}^T\|_2),$$

wobei εps die Maschinengenauigkeit bezeichnet. Das bedeutet, dass die Matrizen $\mathbf{D} + \rho \mathbf{u}\mathbf{u}^T$ und $\mathbf{D} + \tilde{\mathbf{u}}\tilde{\mathbf{u}}^T$ so nahe beieinander liegen, dass die Eigenwerte und Eigenvektoren von $\mathbf{D} + \tilde{\mathbf{u}}\tilde{\mathbf{u}}^T$ stabile Approximationen der Eigenwerte und Eigenvektoren von $\mathbf{D} + \rho \mathbf{u}\mathbf{u}^T$ sind. □

Sind alle Eigenwerte von \mathbf{D} voneinander verschieden und sind alle Komponenten von \mathbf{u} von 0 verschieden, so besitzt die Summe in der Darstellung (7.16) der Sekulargleichung n Summanden, und wir können mit dem dargestellten Verfahren alle Eigenwerte und alle Eigenvektoren von $\tilde{\mathbf{D}}$ bestimmen. Gilt $d_j = d_{j+1}$ oder $u_j = 0$ für ein j , so ist die Anzahl der so berechenbaren Eigenwerte und -vektoren nur so groß wie die Anzahl der Summanden (wobei Summanden mit gleichem Nenner zusammengefasst werden). Die übrigen Eigenwerte und Eigenvektoren sind aber noch leichter zu erhalten:

Ist $u_j = 0$, so ist d_j auch Eigenwert von $\tilde{\mathbf{D}} = \mathbf{D} + \rho \mathbf{u} \mathbf{u}^T$ mit dem Eigenvektor \mathbf{e}^j , denn

$$\tilde{\mathbf{D}} \mathbf{e}^j = d_j \mathbf{e}^j + \rho \mathbf{u} \cdot u_j = d_j \mathbf{e}^j.$$

Ist $d_j = d_{j+1}$ und $u_j \neq 0$ (sonst liegt der letzte Fall vor), so gilt für $\mathbf{w} := -u_{j+1} \mathbf{e}^j + u_j \mathbf{e}^{j+1}$

$$\tilde{\mathbf{D}} \mathbf{w} = \mathbf{D} \mathbf{w} + \rho \mathbf{u} \mathbf{u}^T (-u_{j+1} \mathbf{e}^j + u_j \mathbf{e}^{j+1}) = d_j \mathbf{w}.$$

Insgesamt hat man die folgende Methode zur Bestimmung aller Eigenwerte und Eigenvektoren einer Rang-1-Modifikation $\tilde{\mathbf{D}} = \mathbf{D} + \rho \mathbf{u} \mathbf{u}^T$ einer reellen Diagonalmatrix:

1. Bestimme alle Eigenwerte λ_j mit Hilfe der Sekulargleichung

$$f(\lambda) = 1 + \rho \sum_{k=1}^n \frac{u_k^2}{d_k - \lambda} = 0.$$

2. Bestimme mit dem Satz von Löwner (Satz 7.24.) den Vektor $\tilde{\mathbf{u}}$, für den $\lambda_1, \dots, \lambda_n$ die Eigenwerte von $\mathbf{D} + \tilde{\mathbf{u}} \tilde{\mathbf{u}}^T$ sind.

3. Bestimme mit Lemma 7.23. die Eigenvektoren von $\mathbf{D} + \tilde{\mathbf{u}} \tilde{\mathbf{u}}^T$.

Die Hergeleitete Methode zur Bestimmung von Eigenwerten und Eigenvektoren von Rang-1-Modifikationen von Diagonalmatrizen kann man rekursiv nutzen, um alle Eigenwerte und Eigenvektoren einer Tridiagonalmatrix \mathbf{T} zu berechnen. Der folgende Algorithmus bestimmt eine Diagonalmatrix $\mathbf{\Lambda}$, die als Diagonalelemente die Eigenwerte von \mathbf{D} enthält, und eine orthogonale Matrix \mathbf{Q} , deren Spalten die zugehörigen Eigenvektoren sind.

Algorithmus 7.26. (Divide and Conquer Verfahren)

```
function [\Lambda, Q]=div_conq(T);
    if Dimension(T)==1
```

```

Q=1; Λ=T;
else
  Zerlege T=  $\begin{pmatrix} T_{-1} & O \\ O & T_{-2} \end{pmatrix} + \beta_{-m} vv'$ ;
  [Λ-1, Q-1]=div_conq(T-1);
  [Λ-2, Q-2]=div_conq(T-2);
  Bestimme D+ρuu' aus Λ-1, Λ-2, Q-1, Q-2
  Bestimme Eigenwerte Λ und Eigenvektoren P von D+ρuu'
  Bestimme Eigenvektoren Q=  $\begin{pmatrix} Q_{-1} & O \\ O & Q_{-2} \end{pmatrix} \cdot P$  von T;
end

```

7.6 Jacobi Verfahren

Das **Jacobi Verfahren** wurde in der Mitte des 19. Jahrhunderts von Jacobi [58] angegeben. Es löst das vollständige Eigenwertproblem für die symmetrische Matrix $\mathbf{A} \in \mathbb{R}^{(n,n)}$, d.h. es bestimmt alle Eigenwerte und Eigenvektoren von \mathbf{A} . Anders als die vorhergehenden Verfahren operiert es auf der Originalmatrix, ohne sie vorher auf Tridiagonalgestalt zu transformieren. Es wird die orthogonale Matrix \mathbf{U} mit

$$\mathbf{U}^T \mathbf{A} \mathbf{U} = \text{diag} \{ \lambda_i \} \quad (7.18)$$

als unendliches Produkt von Jacobi Rotationen $\mathbf{U}_{ik} = \mathbf{R}(i, k, \theta)$ (vgl. Kapitel 5) aufgebaut, wobei die Drehwinkel θ geeignet gewählt werden. Das Jacobi Verfahren ist wesentlich langsamer als der QR Algorithmus oder das Divide and Conquer Verfahren. Es ist dennoch von Interesse, da mit ihm sehr kleine Eigenwerte und die zugehörigen Eigenvektoren wesentlich genauer berechnet werden können als mit den beiden anderen genannten Verfahren (vgl. Demmel [23], pp. 248 ff) und da es leicht parallelisiert werden kann (vgl. Golub und Van Loan [51], pp. 431 ff).

Es sei $\mathbf{B} := \mathbf{U}_{ij}^T \mathbf{A} \mathbf{U}_{ij}$. Man rechnet leicht nach, dass dann mit $s := \sin \theta$ und $c := \cos \theta$ gilt

$$\left. \begin{aligned} b_{k\ell} &= a_{k\ell} && \text{für } k, \ell \neq i, j \\ b_{ik} &= b_{ki} = ca_{ik} - sa_{jk} && \text{für } k \neq i, j \\ b_{jk} &= b_{kj} = sa_{ik} + ca_{jk} && \text{für } k \neq i, j \\ b_{ii} &= c^2 a_{ii} - 2csa_{ij} + s^2 a_{jj} \\ b_{jj} &= s^2 a_{ii} + 2csa_{ij} + c^2 a_{jj} \\ b_{ij} &= b_{ji} = cs(a_{ii} - a_{jj}) + (c^2 - s^2)a_{ij} \end{aligned} \right\} \quad (7.19)$$

Eine Multiplikation einer Matrix von links und rechts mit einer Rotationsmatrix kostet also $O(n)$ flops.

Um den Anteil der Elemente von \mathbf{B} außerhalb der Diagonale möglichst klein zu machen, wählen wir den Winkel θ so, dass die Außennorm

$$g(\mathbf{B}) := \sqrt{\sum_{i \neq j} b_{ij}^2}$$

minimal wird (g ist keine Norm, sondern nur eine Seminorm; die Definitheit ist verletzt).

Da orthogonale Transformationen die Euklidische Länge von Vektoren erhalten, gilt für die Schur Norm

$$\|\mathbf{A}\|_S^2 = \|\mathbf{U}_{ij}^T \mathbf{A} \mathbf{U}_{ij}\|_S^2 = \|\mathbf{B}\|_S^2, \text{ d. h. } \sum_{i,j} a_{ij}^2 = \sum_{i,j} b_{ij}^2.$$

Andererseits errechnet man leicht

$$a_{ii}^2 + a_{jj}^2 + 2a_{ij}^2 = b_{ii}^2 + b_{jj}^2 + 2b_{ij}^2.$$

Zusammen mit $a_{kk} = b_{kk}$ für $k \neq i, j$ folgt daher

$$\sum_{k \neq \ell} a_{k\ell}^2 = \sum_{k \neq \ell} b_{k\ell}^2 + 2a_{ij}^2 - 2b_{ij}^2,$$

d.h.

$$g^2(\mathbf{A}) - 2a_{ij}^2 + 2b_{ij}^2 = g^2(\mathbf{B}) \quad (7.20)$$

$g(\mathbf{B})$ wird also minimal, wenn θ so bestimmt wird, dass $b_{ij} = 0$ gilt, d.h. wegen (7.19)

$$\cot(2\theta) = \frac{a_{jj} - a_{ii}}{2a_{ij}}, \quad (7.21)$$

und diese Gleichung ist in dem Intervall $\left(-\frac{\pi}{4}, \frac{\pi}{4}\right]$ eindeutig lösbar.

Das Jacobi Verfahren verläuft nun so: Ausgehend von $\mathbf{A}_0 := \mathbf{A}$ wird nach der Vorschrift

$$\mathbf{A}_{m+1} := \mathbf{V}_m^T \mathbf{A}_m \mathbf{V}_m = (a_{kl}^{(m+1)})$$

die Iterationsfolge $\mathbf{A}_0, \mathbf{A}_1, \mathbf{A}_2, \dots$ gebildet. Dabei ist \mathbf{V}_m eine Rotationsmatrix $\mathbf{R}(i, k, \theta)$. Die jeweiligen Indexpaare (i, j) , die \mathbf{V}_m bestimmen, werden nach einer Auswahlvorschrift gebildet, und θ wird so bestimmt, dass $a_{ij}^{(m+1)} = 0$ gilt.

Übliche Auswahlvorschriften sind

1. Wähle (i, j) so, dass

$$|a_{ij}^{(m)}| = \max_{k \neq \ell} |a_{k\ell}^{(m)}|.$$

Dies ist das **klassische Jacobi Verfahren**. Wegen (7.20) wird dabei die Außennorm am stärksten verkleinert. Dieses Verfahren wird aber dennoch nicht verwendet, denn es werden $\frac{1}{2}n(n-1)$ Vergleiche ausgeführt und danach nur eine Jacobi Rotation mit $O(n)$ flops Kosten.

2. (i, j) durchläuft zyklisch die Folge $(1, 2), (1, 3), \dots, (1, n), (2, 3), \dots, (n-1, n)$. Man erhält dann das **zyklische Jacobi Verfahren**.

3. Da die Auswahl nach 1. sehr aufwändig ist, andererseits bei der Auswahl nach 2. die Drehung auch ausgeführt wird, wenn $|a_{ij}^{(m)}|$ relativ klein ist, wobei $g(\mathbf{A}_m)$ nur unwesentlich reduziert wird, geht man folgendermaßen vor: Es sei ein Schwellenwert γ vorgegeben. Man durchläuft dann die Indexfolge in 2., führt aber die Rotation nur dann aus, wenn $|a_{ij}^{(m)}| \geq \gamma$ gilt.

Gilt $|a_{ij}^m| < \gamma$ für alle $i \neq j$, so wird der Schwellenwert herabgesetzt: $\gamma := \frac{\gamma}{2}$. Dieses Verfahren heißt **Threshold Jacobi Verfahren**.

Konvergenzbeweise liegen für alle drei Methoden vor. Wir behandeln nur das klassische Jacobi Verfahren.

Satz 7.27. Sei $\mathbf{A} \in \mathbb{R}^{(n,n)}$ symmetrisch, (i, j) so gewählt, dass $|a_{ij}| = \max_{k \neq \ell} |a_{k\ell}|$, $\mathbf{V} := \mathbf{V}_{ij}$ von der Gestalt (7.18), θ gemäß (7.21) bestimmt und $\mathbf{B} := \mathbf{V}^T \mathbf{A} \mathbf{V}$. Dann gilt

$$g^2(\mathbf{B}) \leq q^2 g^2(\mathbf{A}) \quad (7.22)$$

mit

$$q^2 = \frac{n^2 - n - 2}{n^2 - n} < 1.$$

Beweis: Es gibt $n^2 - n$ Nichtdiagonalelemente. Daher gilt $g^2(\mathbf{A}) \leq (n^2 - n)a_{ij}^2$, und es folgt

$$g^2(\mathbf{B}) = g^2(\mathbf{A}) - 2a_{ij}^2 \leq \left(1 - \frac{2}{n^2 - n}\right) g^2(\mathbf{A}) \quad \blacksquare$$

Satz 7.28. Das klassische Jacobi Verfahren konvergiert, d. h. es existiert eine Diagonalmatrix $\mathbf{\Lambda}$ mit

$$\lim_{m \rightarrow \infty} \mathbf{A}_m = \mathbf{\Lambda}$$

Beweis: Nach (7.22) gilt $g(\mathbf{A}_m) \rightarrow 0$, d.h. $a_{k\ell}^{(m)} \rightarrow 0$ für alle $k \neq \ell$.

Es bleibt also nur die Konvergenz der Diagonale zu zeigen. Aus (7.19) und (7.21) folgt

$$\begin{aligned} |b_{ii} - a_{ii}| &= |s^2(a_{jj} - a_{ii}) - 2csa_{ij}| \\ &= |a_{ij}| \cdot |2\sin^2\theta \cot 2\theta - 2\sin\theta \cos\theta| \\ &= |a_{ij}| \cdot \left| \frac{\sin\theta}{\cos\theta} \right| \leq |a_{ij}|, \end{aligned}$$

da $\theta \in \left(-\frac{\pi}{4}, \frac{\pi}{4}\right]$ gewählt wird.

Genauso erhält man $|b_{jj} - a_{jj}| \leq |a_{ij}|$

Ist nun (i, j) das Indexpaar bei der Behandlung von \mathbf{A}_m , so gilt für alle $k = 1, \dots, n$

$$|a_{kk}^{(m)} - a_{kk}^{(m+1)}| \leq |a_{ij}^{(m)}| \leq g(\mathbf{A}_m) \leq q^m g(\mathbf{A}),$$

und daher

$$|a_{kk}^{(m)} - a_{kk}^{(m+p)}| \leq \sum_{r=0}^{p-1} q^{m+r} g(\mathbf{A}) \leq \frac{q^m}{1-q} g(\mathbf{A}) \quad \blacksquare$$

Bemerkung 7.29. 1. Es ist nicht erforderlich, θ aus (7.21) explizit zu bestimmen, da nur $c = \cos\theta$ und $s = \sin\theta$ benötigt werden. Durch Quadrieren von (7.21) erhält man

$$\frac{1 - 4s^2 + 4s^4}{s^2(1 - s^2)} = \frac{(a_{jj} - a_{ii})^2}{a_{ij}^2},$$

also eine quadratische Gleichung in s^2 . Das Vorzeichen von s kann dann aus (7.19) bestimmt werden.

2. Sind die Eigenwerte von \mathbf{A} alle verschieden, so kann man beim klassischen Jacobi Verfahren die Konvergenz gegen die Eigenvektoren beweisen, d.h. es existiert

$$\lim_{m \rightarrow \infty} \mathbf{V}_1 \mathbf{V}_2 \cdots \mathbf{V}_m =: \mathbf{V},$$

und die Spalten von \mathbf{V} sind dann offenbar die Eigenvektoren von \mathbf{A} .

3. Alle angegebenen Verfahren sind sogar in folgendem Sinne quadratisch konvergent (vgl. [92]): Es gibt ein $C > 0$ mit

$$g(\mathbf{A}_{m+N}) \leq Cg^2(\mathbf{A}_m), \quad N := \frac{1}{2}n(n-1).$$

4. Als Abbruchkriterium beim Jacobi Verfahren eignet sich der Satz von Gerschgorin (Satz 6.6.). Die Radien der Kreise werden im Verfahren ja beliebig klein gemacht. \square

7.7 Berechnung der Singulärwertzerlegung

Die singulären Werte σ_j einer Matrix $\mathbf{A} \in \mathbb{R}^{(m,n)}$, $m \geq n$, sind (vgl. Abschnitt 5.4) die Quadratwurzeln der Eigenwerte von $\mathbf{A}^T \mathbf{A}$ bzw. $\mathbf{A} \mathbf{A}^T$, und die singulären Vektoren, d.h. die Spalten der Matrizen \mathbf{U} und \mathbf{V} in

$$\mathbf{A} = \mathbf{U} \mathbf{\Sigma} \mathbf{V}^T, \quad (7.23)$$

sind die Eigenvektoren von $\mathbf{A} \mathbf{A}^T$ und $\mathbf{A}^T \mathbf{A}$. Man kann also die in diesem Kapitel entwickelten Verfahren zur Bestimmung der Eigenwerte und Eigenvektoren einer symmetrischen Matrix verwenden, um die Singulärwertzerlegung einer Matrix zu bestimmen. Da jedoch die Kondition einer Matrix beim Übergang zu $\mathbf{A}^T \mathbf{A}$ bzw. $\mathbf{A} \mathbf{A}^T$ quadriert wird, dürfen diese Matrizen nicht ausmultipliziert werden, sondern man darf in den Verfahren nur die Ausgangsmatrix selbst verwenden.

Ein solches Verfahren auf der Basis des QR Algorithmus mit impliziten Shifts wurde von Golub und Kahan [49] 1965 erstmals vorgestellt und von Golub und Reinsch [50] 1970 in Algol60 implementiert. Wir wollen diesen Algorithmus in diesem Abschnitt beschreiben.

Varianten hiervon gelten heute als die genauesten und für kleine Dimensionen (≤ 25) als die schnellsten Verfahren zur Bestimmung der Singulärwertzerlegung einer Matrix. Die LAPACK Routine `sbdsqr` beruht auf dem LR Algorithmus, einem Vorgänger des QR Algorithmus, der für symmetrisch und positiv definite Matrizen stabil ist. Seine Grundform werden wir ebenfalls beschreiben.

Für größere Dimensionen sind Divide and Conquer Verfahren schneller als die Methoden, die auf dem QR Algorithmus beruhen. Sie haben aber den Nachteil, dass sie kleine singuläre Werte nur mit geringerer relativer Genauigkeit liefern. Ist man nur an den singulären Werten in einem vorgegebenen Intervall interessiert, so kann man Bisektion und inverse Iteration anwenden (nachdem man zunächst wie in dem hier beschriebenen Verfahren die Matrix orthogonal auf Bidiagonalgestalt transformiert hat).

Wir transformieren $\mathbf{A} \in \mathbb{R}^{(m,n)}$ auf Bidiagonalgestalt. Dazu multiplizieren wir \mathbf{A} von links mit einer Householder Matrix $\mathbf{Q}_1 \in \mathbb{R}^{(m,m)}$, so dass die erste Spalte von

\mathbf{A} auf ein Vielfaches des ersten Einheitsvektors abgebildet wird. Es sei

$$\mathbf{A}_1 := \mathbf{Q}_1 \mathbf{A} = \begin{pmatrix} a_{11}^{(1)} & a_{12}^{(1)} & a_{13}^{(1)} & \cdots & a_{1n}^{(1)} \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & \cdots & a_{2n}^{(1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & a_{m2}^{(1)} & a_{m3}^{(1)} & \cdots & a_{mn}^{(1)} \end{pmatrix}.$$

Wir wählen nun eine Householder Matrix $\tilde{\mathbf{P}}_1 \in \mathbb{R}^{(n-1, n-1)}$ mit

$$\tilde{\mathbf{P}}_1(a_{12}^{(1)}, a_{13}^{(1)}, \dots, a_{1n}^{(1)})^T = (a_{12}^{(2)}, 0, \dots, 0)^T$$

und hiermit

$$\mathbf{P}_1 := \begin{pmatrix} 1 & \mathbf{0}^T \\ \mathbf{0} & \tilde{\mathbf{P}}_1 \end{pmatrix} \in \mathbb{R}^{(n, n)}.$$

Dann bleibt in $\mathbf{A}_1 \mathbf{P}_1$ die erste Spalte von \mathbf{A}_1 erhalten, und es gilt

$$\mathbf{A}_2 := (\mathbf{Q}_1 \mathbf{A}) \mathbf{P}_1 = \begin{pmatrix} a_{11}^{(1)} & a_{12}^{(2)} & 0 & \cdots & 0 \\ 0 & a_{22}^{(2)} & a_{23}^{(2)} & \cdots & a_{2n}^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & a_{m2}^{(2)} & a_{m3}^{(2)} & \cdots & a_{mn}^{(2)} \end{pmatrix}.$$

Durch Multiplikation mit weiteren Householder Matrizen (abwechselnd von links und rechts) kann man (ähnlich wie bei der Bestimmung der QR Zerlegung einer Matrix) nacheinander die Elemente der Spalten unterhalb der Diagonale und die Elemente der Zeilen rechts von dem ersten Nebendiagonalelement annullieren. Damit wird \mathbf{A} orthogonal auf Bidiagonalgestalt transformiert:

$$\mathbf{Q}_n \mathbf{Q}_{n-1} (\dots (\mathbf{Q}_2 ((\mathbf{Q}_1 \mathbf{A}) \mathbf{P}_1)) \dots \mathbf{P}_{n-2}) = \begin{pmatrix} \mathbf{B} \\ \mathbf{O} \end{pmatrix} \quad (7.24)$$

mit

$$\mathbf{B} = \begin{pmatrix} \alpha_1 & \beta_1 & 0 & \cdots & 0 & 0 \\ 0 & \alpha_2 & \beta_2 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \alpha_{n-1} & \beta_{n-1} \\ 0 & 0 & 0 & \cdots & 0 & \alpha_n \end{pmatrix},$$

wobei \mathbf{Q}_i die Elemente $i+1, \dots, m$ der i -ten Spalte bzw. \mathbf{P}_i die Elemente $i+2, \dots, n$ der i -Zeile der Matrix annulliert, auf die \mathbf{Q}_i bzw. \mathbf{P}_i angewendet wird.

Ist dann

$$\mathbf{B} = \hat{\mathbf{U}} \hat{\mathbf{\Sigma}} \hat{\mathbf{V}}^T, \quad \hat{\mathbf{U}}, \hat{\mathbf{V}}, \hat{\mathbf{\Sigma}} \in \mathbb{R}^{(n, n)}$$

die Singulärwertzerlegung von \mathbf{B} , so erhält man mit

$$\tilde{\mathbf{Q}} := \mathbf{Q}_n \cdots \mathbf{Q}_1, \quad \tilde{\mathbf{P}} := \mathbf{P}_1 \cdots \mathbf{P}_{n-2}$$

wegen

$$\tilde{Q}^H \begin{pmatrix} \hat{U} & \mathbf{O} \\ \mathbf{O} & I_{m-n} \end{pmatrix} \begin{pmatrix} \Sigma \\ \mathbf{O} \end{pmatrix} \hat{V}^T \tilde{P}^H = \tilde{Q}^H \begin{pmatrix} \hat{U} \Sigma \hat{V}^T \\ \mathbf{O} \end{pmatrix} \tilde{P}^H = \mathbf{A}$$

die Singulärwertzerlegung von \mathbf{A} .

Es ist

$$\mathbf{B}^T \mathbf{B} = \begin{pmatrix} \alpha_1^2 & \alpha_1 \beta_1 & 0 & \dots & 0 & 0 \\ \alpha_1 \beta_1 & \alpha_2^2 + \beta_1^2 & \alpha_2 \beta_2 & \dots & 0 & 0 \\ 0 & \alpha_2 \beta_2 & \alpha_3^2 + \beta_2^2 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & \alpha_{n-1}^2 + \beta_{n-2}^2 & \alpha_{n-1} \beta_{n-1} \\ 0 & 0 & 0 & \dots & \alpha_{n-1} \beta_{n-1} & \alpha_n^2 + \beta_{n-1}^2 \end{pmatrix}$$

eine Tridiagonalmatrix. Diese ist genau dann nicht reduziert, wenn $\alpha_i \neq 0$ und $\beta_i \neq 0$ für $i = 1, \dots, n-1$ gilt.

Sind Elemente $\beta_i = 0$, so zerfällt \mathbf{B} in Blöcke, die getrennt behandelt werden können.

Sei nun $\alpha_k = 0$, $\alpha_j \neq 0$ für $j = k+1, \dots, n$ und $\beta_j \neq 0$ für $j = k, \dots, n-1$.

Dann kann man durch $n-k$ Multiplikationen von links mit Givens Rotationen \mathbf{U}_{kj} , $j = k+1, \dots, n$, die Matrix \mathbf{B} auf die Gestalt bringen

$$\mathbf{U}_{kn} \dots \mathbf{U}_{k,k+1} \mathbf{B} = \begin{pmatrix} \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \alpha_{k-1} & \beta_{k-1} & 0 & 0 & \dots & 0 & 0 \\ \dots & 0 & 0 & 0 & 0 & \dots & 0 & 0 \\ \dots & 0 & 0 & \alpha'_{k+1} & \beta'_{k+1} & \dots & 0 & 0 \\ \dots & \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \vdots \\ \dots & 0 & 0 & 0 & 0 & \dots & \alpha'_{n-1} & \beta'_{n-1} \\ \dots & 0 & 0 & 0 & 0 & \dots & 0 & \alpha'_n \end{pmatrix}$$

wobei \mathbf{U}_{kj} das Element an der Stelle (k, j) annulliert und $\alpha'_j \neq 0$, $\beta'_j \neq 0$ gilt.

Wir können uns also in jedem Fall auf den Fall beschränken, dass die Matrix $\mathbf{B}^T \mathbf{B}$ nicht reduziert ist. Wir wenden hierauf nun die folgende Variante des QR Algorithmus mit impliziten Shifts an.

Sei \mathbf{Q} der orthogonale Anteil der QR Zerlegung von $\mathbf{B}^T \mathbf{B} - \kappa \mathbf{I}$.

1. Bestimme eine orthogonale Matrix \mathbf{P}_0 , deren erste Spalte mit der ersten Spalte von \mathbf{Q} übereinstimmt.
2. Transformiere $\mathbf{A} \mathbf{P}_0$ auf die bidiagonale Matrix $\tilde{\mathbf{B}}$ mit dem Verfahren aus (7.24).

Sei $\tilde{\mathbf{P}} := \mathbf{P}_0 \mathbf{P}_1 \dots \mathbf{P}_{n-2}$, wobei $\mathbf{P}_1, \dots, \mathbf{P}_{n-2}$ die Matrizen aus (7.24) bei der Transformation von $\mathbf{B}\mathbf{P}_0$ auf Bidiagonalgestalt bezeichnen. Dann besitzt $\tilde{\mathbf{P}}$ dieselbe erste Spalte wie \mathbf{Q} und

$$\tilde{\mathbf{B}}^T \tilde{\mathbf{B}} = \tilde{\mathbf{P}}^T \mathbf{B}^T (\mathbf{Q}_{n-1} \dots \mathbf{Q}_1)^T (\mathbf{Q}_{n-1} \dots \mathbf{Q}_1) \mathbf{B} \tilde{\mathbf{P}} = \tilde{\mathbf{P}}^T \mathbf{B}^T \mathbf{B} \tilde{\mathbf{P}}.$$

Da $\tilde{\mathbf{B}}^T \tilde{\mathbf{B}}$ und $\mathbf{Q}^T \mathbf{B}^T \mathbf{B} \mathbf{Q}$ tridiagonal sind, folgt aus Satz 6.42., dass $\mathbf{Q} = \tilde{\mathbf{P}}$ und $\tilde{\mathbf{B}}^T \tilde{\mathbf{B}} = \mathbf{Q}^T \mathbf{B}^T \mathbf{B} \mathbf{Q}$ gilt. $\tilde{\mathbf{B}}^T \tilde{\mathbf{B}}$ ist also die Tridiagonalmatrix, die man mit einem QR Schritt mit dem Shift κ ausgehend von $\mathbf{B}^T \mathbf{B}$ erhält. Man beachte, dass bei der Durchführung nicht $\mathbf{B}^T \mathbf{B}$ berechnet wird.

Die erste Spalte von $\mathbf{B}^T \mathbf{B} - \kappa \mathbf{I}$ ist gegeben durch

$$\begin{pmatrix} \alpha_1^2 - \kappa & \alpha_1 \beta_1 & 0 & \dots & 0 \end{pmatrix}^T,$$

d.h. \mathbf{P}_0 kann als Drehung \mathbf{U}_{12} gewählt werden, so dass das Element $\alpha_1 \beta_1$ annulliert wird.

$\mathbf{B}\mathbf{P}_0$ hat die Gestalt

$$\begin{pmatrix} * & * & 0 & 0 & 0 & \dots \\ * & * & * & 0 & 0 & \dots \\ 0 & 0 & * & * & 0 & \dots \\ 0 & 0 & 0 & * & * & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots \end{pmatrix}$$

Wir können nun \mathbf{Q}_1 als Drehung in der (1,2)-Ebene so wählen, dass bei Multiplikation von links das Element an der Stelle (2,1) annulliert wird. Dadurch wird das Element an der Stelle (1,3) besetzt.

Danach wählen wir \mathbf{P}_1 als Drehung in der (2,3)-Ebene, so dass bei Multiplikation von rechts das Element an der Stelle (1,3) annulliert wird. Dadurch wird das Element (3,2) besetzt, usw. Das Element, das die Bidiagonalgestalt stört, wird also bei der Multiplikation von links mit der Drehung $\mathbf{U}_{i,i+1}$ von der Position $(i+1, i)$ in die Position $(i, i+2)$ "gejagt", danach wird es durch eine Multiplikation von rechts mit einer Drehung $\mathbf{U}_{i+1,i+2}$ an die Position $(i+2, i+1)$ weitertransportiert.

Als Shift κ wählt man wieder den Eigenwert von

$$\begin{pmatrix} \alpha_{n-1}^2 + \beta_{n-2}^2 & , & \alpha_{n-1} \beta_{n-1} \\ \alpha_{n-1} \beta_{n-1} & , & \alpha_n^2 + \beta_{n-1}^2 \end{pmatrix},$$

der $\alpha_n^2 + \beta_{n-1}^2$ am nächsten ist.

Die Matrizen \mathbf{U} und \mathbf{V} kann man aus den \mathbf{Q}_i und \mathbf{P}_i als Produkt mitberechnen.

Wir beschreiben nun das auf der LR Iteration beruhende **dqds Verfahren** (der Name ist eine Abkürzung für “differential quotient-difference algorithm with shifts” und hat historische Gründe). Für positiv definite Matrizen hat der **LR Algorithmus** die folgende Gestalt: Es sei \mathbf{T}_0 eine symmetrische, positiv definite Matrix

Algorithmus 7.30. (LR Algorithmus)

```

For  $i = 1, 2, \dots$  until convergence do
  Wähle einen Shift  $\tau_i^2$ , der kleiner
    als der kleinste Eigenwert von  $T_i$  ist
  Berechne die Cholesky Zerlegung  $T_i - \tau_i^2 I = B_i^T B_i$ 
   $T_{i+1} = B_i B_i^T + \tau_i I$ 

```

Dabei bezeichnet \mathbf{B}_i abweichend von der bisherigen Notation für die Cholesky Zerlegung eine obere Dreiecksmatrix.

Wie für den QR Algorithmus sieht man, dass die Matrizen \mathbf{T}_i einander ähnlich sind:

$$\mathbf{T}_{i+1} = \mathbf{B}_i \mathbf{B}_i^T + \tau_i^2 \mathbf{I} = \mathbf{B}_i^{-T} \mathbf{B}_i^T \mathbf{B}_i \mathbf{B}_i^T + \tau_i^2 \mathbf{I} = \mathbf{B}_i^{-T} (\mathbf{T}_i - \tau_i^2 \mathbf{I}) \mathbf{B}_i^T + \tau_i^2 \mathbf{I} = \mathbf{B}_i^{-T} \mathbf{T}_i \mathbf{B}_i^T.$$

Der LR Algorithmus hat eine ähnliche Struktur wie der QR Algorithmus. Es wird eine Zerlegung von $\mathbf{T}_i + \tau_i^2 \mathbf{I}$ berechnet, und die nächste Iterierte erhält man, indem man die beiden Faktoren in umgekehrter Reihenfolge multipliziert und die Spektralverschiebung wieder rückgängig macht. Ist $\tau_i = 0$ für alle i , so kann man sogar zeigen, dass zwei Schritte des LR Algorithmus dasselbe Ergebnis wie ein Schritt der Grundform des QR Algorithmus liefert.

Satz 7.31. *Es sei \mathbf{T}_2 die Matrix, die ausgehend von der positiv definiten Matrix \mathbf{T}_0 durch zwei Schritte des LR Algorithmus ohne Shifts erzeugt wird, und $\tilde{\mathbf{T}}$ das Ergebnis eines QR Schrittes. Dann gilt $\mathbf{T}_2 = \tilde{\mathbf{T}}$.*

Beweis: Es sei $\mathbf{T}_0 = \mathbf{QR}$ die QR Zerlegung von \mathbf{T}_0 , wobei die Diagonalelemente von \mathbf{R} positiv gewählt sind. Dann gilt

$$\mathbf{T}_0^2 = \mathbf{T}_0^T \mathbf{T}_0 = (\mathbf{QR})^T \mathbf{QR} = \mathbf{R}^T \mathbf{R},$$

und daher ist $\mathbf{T}_0^2 = \mathbf{R}^T \mathbf{R}$ die Cholesky Zerlegung von \mathbf{T}_0^2 . Ferner gilt mit den Matrizen \mathbf{B}_0 und \mathbf{B}_1 aus Algorithmus 7.30.

$$\mathbf{T}_0^2 = \mathbf{B}_0^T \mathbf{B}_0 \mathbf{B}_0^T \mathbf{B}_0 = \mathbf{B}_0^T (\mathbf{B}_1^T \mathbf{B}_1) \mathbf{B}_0 = (\mathbf{B}_1 \mathbf{B}_0)^T (\mathbf{B}_1 \mathbf{B}_0),$$

und da auch $\mathbf{B}_1\mathbf{B}_0$ eine obere Dreiecksmatrix ist, ist auch dies die Cholesky Zerlegung von \mathbf{T}_0^2 , und die Eindeutigkeit liefert $\mathbf{R} = \mathbf{B}_1\mathbf{B}_0$. Hiermit folgt

$$\begin{aligned}\tilde{\mathbf{T}} &= \mathbf{R}\mathbf{Q} = \mathbf{R}\mathbf{Q}(\mathbf{R}\mathbf{R}^{-1}) = \mathbf{R}(\mathbf{Q}\mathbf{R})\mathbf{R}^{-1} = \mathbf{R}\mathbf{T}_0\mathbf{R}^{-1} \\ &= (\mathbf{B}_1\mathbf{B}_0)(\mathbf{B}_0^T\mathbf{B}_0)(\mathbf{B}_1\mathbf{B}_0)^{-1} = \mathbf{B}_1\mathbf{B}_0\mathbf{B}_0^T\mathbf{B}_0\mathbf{B}_0^{-1}\mathbf{B}_1^{-1} = \mathbf{B}_1(\mathbf{B}_0\mathbf{B}_0^T)\mathbf{B}_1^{-1} \\ &= \mathbf{B}_1(\mathbf{B}_1^T\mathbf{B}_1)\mathbf{B}_1^{-1} = \mathbf{B}_1\mathbf{B}_1^T = \mathbf{T}_2 \quad \blacksquare\end{aligned}$$

Um die Singulärwertzerlegung einer Matrix \mathbf{A} zu bestimmen, transformieren wir sie wieder zunächst orthogonal auf Bidiagonalgestalt \mathbf{B}_0 und wenden dann auf $\mathbf{B}_0^T\mathbf{B}_0$ Algorithmus 7.30. an. Dabei berechnen wir wie vorher die Matrix $\mathbf{B}_0^T\mathbf{B}_0$ nicht explizit (da dies die Kondition wieder quadrieren würde), sondern wir bestimmen aus \mathbf{B}_i direkt die nächste Iterierte \mathbf{B}_{i+1} .

Es seien a_1, \dots, a_n die Diagonalelemente und b_1, \dots, b_{n-1} die Superdiagonalelemente von \mathbf{B}_i und $\tilde{a}_1, \dots, \tilde{a}_n$ und $\tilde{b}_1, \dots, \tilde{b}_{n-1}$ die entsprechenden Elemente von \mathbf{B}_{i+1} . Dann folgt aus der Identität

$$\mathbf{B}_{i+1}^T\mathbf{B}_{i+1} + \tau_{i+1}^2\mathbf{I} = \mathbf{T}_{i+1} = \mathbf{B}_i\mathbf{B}_i^T + \tau_i^2\mathbf{I} \quad (7.25)$$

durch Vergleich der Diagonalelemente mit den Bezeichnungen $b_0 = \tilde{b}_0 = b_n = \tilde{b}_n = 0$

$$\tilde{a}_j^2 + \tilde{b}_{j-1}^2 + \tau_{i+1}^2 = a_j^2 + b_j^2 + \tau_i^2,$$

d.h.

$$\tilde{a}_j^2 = a_j^2 + b_j^2 - \tilde{b}_{j-1}^2 - \delta \quad (7.26)$$

mit $\delta := \tau_{i+1}^2 - \tau_i^2$, und durch Vergleich der Superdiagonalelemente $\tilde{a}_j\tilde{b}_j = a_{j+1}b_j$, d.h.

$$\tilde{b}_j^2 = a_{j+1}^2 b_j^2 / \tilde{a}_j^2. \quad (7.27)$$

Kombiniert man die Gleichungen (7.26) und (7.27), so erhält man den vorläufigen Algorithmus

Algorithmus 7.32. (qds Algorithmus)

```
for j=1 to n-1
   $\tilde{a}_j^2 = a_j^2 + b_j^2 - \tilde{b}_{j-1}^2 - \delta$ 
   $\tilde{b}_j^2 = b_j^2(a_{j+1}^2 / \tilde{a}_j^2)$ 
end
 $\tilde{a}_n^2 = a_n^2 - \tilde{b}_{n-1}^2 - \delta$ 
```

Dieser Algorithmus bestimmt sofort aus den Quadraten der Elemente $q_j := a_j^2$ und $e_j := b_j^2$ von \mathbf{B}_i die Quadrate der Elemente von \mathbf{B}_{i+1} . Die Elemente selbst werden nicht im Laufe der Iteration benötigt und erst am Ende bestimmt. Eine stabilere Version erhält man, wenn man den Ausdruck

$$d_j := q_j - \tilde{e}_{j-1} - \delta = a_j^2 - \tilde{b}_{j-1}^2 - \delta$$

umrechnet gemäß

$$\begin{aligned} d_j &= q_j - \tilde{e}_{j-1} - \delta = q_j - \frac{q_j e_{j-1}}{\tilde{q}_{j-1}} - \delta = q_j \cdot \frac{\tilde{q}_{j-1} - e_{j-1}}{\tilde{q}_{j-1}} - \delta \\ &= q_j \cdot \frac{q_{j-1} - \tilde{e}_{j-2} - \delta}{\tilde{q}_{j-1}} - \delta = \frac{q_j}{\tilde{q}_{j-1}} \cdot d_{j-1} - \delta. \end{aligned}$$

Hiermit erhält die innere Schleife die Gestalt

$$\begin{aligned} \tilde{q}_j &= d_j + e_j \\ \tilde{e}_j &= e_j \cdot (q_{j+1}/\tilde{q}_j) \\ d_{j+1} &= d_j \cdot (q_{j+1}/\tilde{q}_j) - \delta. \end{aligned}$$

Überschreibt man schließlich noch d_j mit d_{j+1} und setzt $t := q_{j+1}/\tilde{q}_j$, so erhält man

Algorithmus 7.33. (dqds Algorithmus)

```

d = q1 - δ
for j = 1 to n - 1
    q̃j = d + ej
    t = qj+1/q̃j
    ẽj = ej · t
    d = d · t - δ
end
q̃n = d

```

Der dqds Algorithmus benötigt dieselbe Zahl von flops wie der qds Algorithmus. Es wird allerdings eine Subtraktion durch eine Multiplikation ersetzt. Dies erhöht die Stabilität (vgl. Demmel [23], p. 245 ff). Wir haben hier nicht das Abbruchkriterium behandelt und die Wahl des Shifts τ_i^2 . Beide werden in einer Arbeit von Fernando und Parlett [39] ausführlich diskutiert.

7.8 Allgemeine Eigenwertaufgaben

Wir betrachten erneut die allgemeine Eigenwertaufgabe

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{B}\mathbf{x}. \quad (7.28)$$

Dabei seien $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{(n,n)}$ symmetrisch und \mathbf{B} positiv definit. Ist $\mathbf{B} = \mathbf{C}\mathbf{C}^T$ die Cholesky Zerlegung von \mathbf{B} , so ist (7.28) äquivalent der speziellen symmetrischen Eigenwertaufgabe

$$\mathbf{F}\mathbf{y} := \mathbf{C}^{-1}\mathbf{A}\mathbf{C}^{-T}\mathbf{y} = \lambda\mathbf{y}, \quad \mathbf{y} = \mathbf{C}^T\mathbf{x}. \quad (7.29)$$

Es liegt nun nahe, mit einem der Verfahren aus Abschnitt 7.2 bis Abschnitt 7.6 das Eigenwertproblem (7.29) zu lösen, d.h. eine orthogonale Matrix \mathbf{Y} und eine Diagonalmatrix $\mathbf{\Lambda}$ zu bestimmen mit

$$\mathbf{Y}^T\mathbf{F}\mathbf{Y} = \mathbf{\Lambda}.$$

Dann gilt für $\mathbf{X} := \mathbf{C}^{-T}\mathbf{Y}$

$$\mathbf{X}^T\mathbf{A}\mathbf{X} = \mathbf{Y}^T\mathbf{C}^{-1}\mathbf{A}\mathbf{C}^{-T}\mathbf{Y} = \mathbf{Y}^T\mathbf{F}\mathbf{Y} = \mathbf{\Lambda}, \quad (7.30)$$

$$\mathbf{X}^T\mathbf{B}\mathbf{X} = \mathbf{Y}^T\mathbf{C}^{-1}(\mathbf{C}\mathbf{C}^T)\mathbf{C}^{-T}\mathbf{Y} = \mathbf{Y}^T\mathbf{Y} = \mathbf{I}, \quad (7.31)$$

und daher

$$\mathbf{A}\mathbf{X} = \mathbf{X}^{-T}\mathbf{\Lambda} = \mathbf{B}\mathbf{X}\mathbf{\Lambda}. \quad (7.32)$$

Die Diagonale von $\mathbf{\Lambda}$ enthält also die Eigenwerte von (7.28) und die Spalten von \mathbf{X} bilden ein \mathbf{B} -orthogonales System von zugehörigen Eigenvektoren. Dieser Algorithmus wurde erstmals von Wilkinson [121] angegeben.

Die Matrix \mathbf{F} kann man rekursiv auf folgende Weise berechnen: Es seien

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_{n-1} & \mathbf{a} \\ \mathbf{a}^T & \alpha \end{pmatrix}, \quad \mathbf{C} = \begin{pmatrix} \mathbf{C}_{n-1} & \mathbf{0} \\ \mathbf{c}^T & \gamma \end{pmatrix}, \quad \mathbf{F} = \begin{pmatrix} \mathbf{F}_{n-1} & \mathbf{f} \\ \mathbf{f}^T & \phi \end{pmatrix},$$

und es sei die symmetrische Matrix \mathbf{F}_{n-1} bereits bestimmt mit

$$\mathbf{F}_{n-1} = \mathbf{C}_{n-1}^{-1}\mathbf{A}_{n-1}\mathbf{C}_{n-1}^{-T}.$$

Wir berechnen \mathbf{f} und ϕ mit

$$\begin{pmatrix} \mathbf{F}_{n-1} & \mathbf{f} \\ \mathbf{f}^T & \phi \end{pmatrix} = \begin{pmatrix} \mathbf{C}_{n-1} & \mathbf{0} \\ \mathbf{c}^T & \gamma \end{pmatrix}^{-1} \begin{pmatrix} \mathbf{A}_{n-1} & \mathbf{a} \\ \mathbf{a}^T & \alpha \end{pmatrix} \begin{pmatrix} \mathbf{C}_{n-1}^T & \mathbf{c} \\ \mathbf{0}^T & \gamma \end{pmatrix}^{-1},$$

d.h.

$$\begin{pmatrix} \mathbf{C}_{n-1} & \mathbf{0} \\ \mathbf{c}^T & \gamma \end{pmatrix} \begin{pmatrix} \mathbf{F}_{n-1} & \mathbf{f} \\ \mathbf{f}^T & \phi \end{pmatrix} \begin{pmatrix} \mathbf{C}_{n-1}^T & \mathbf{c} \\ \mathbf{0}^T & \gamma \end{pmatrix} = \begin{pmatrix} \mathbf{A}_{n-1} & \mathbf{a} \\ \mathbf{a}^T & \alpha \end{pmatrix},$$

und daher

$$\begin{pmatrix} \mathbf{C}_{n-1}\mathbf{F}_{n-1}\mathbf{C}_{n-1}^T & \mathbf{C}_{n-1}\mathbf{F}_{n-1}\mathbf{c} + \gamma\mathbf{C}_{n-1}\mathbf{f} \\ \mathbf{c}^T\mathbf{F}_{n-1}\mathbf{C}_{n-1}^T + \gamma\mathbf{f}^T\mathbf{C}_{n-1}^T & \mathbf{c}^T\mathbf{F}_{n-1}\mathbf{c} + 2\gamma\mathbf{c}^T\mathbf{f} + \gamma^2\phi \end{pmatrix} = \begin{pmatrix} \mathbf{A}_{n-1} & \mathbf{a} \\ \mathbf{a}^T & \alpha \end{pmatrix}.$$

Dies zeigt, dass \mathbf{f} und ϕ berechnet werden können gemäß

$$\begin{aligned} \mathbf{f} &= \frac{1}{\gamma}(\mathbf{C}_{n-1}^{-1}\mathbf{a} - \mathbf{F}_{n-1}\mathbf{c}), \\ \phi &= \frac{1}{\gamma^2}(\alpha - \mathbf{c}^T\mathbf{F}_{n-1}\mathbf{c} - 2\gamma\mathbf{c}^T\mathbf{f}). \end{aligned}$$

Kapitel 8

Numerische Lösung nichtlinearer Gleichungssysteme

In den vorhergehenden Abschnitten haben wir lineare Probleme betrachtet. Wir untersuchen nun die numerische Bestimmung der Lösungen von nichtlinearen Gleichungen oder Gleichungssystemen. Es sei $f : \mathbb{R}^n \supset D \rightarrow \mathbb{R}^n$ gegeben. Wir betrachten das **Nullstellenproblem**

$$f(\mathbf{x}) = \mathbf{0}. \quad (8.1)$$

In Abschnitt 8.1 betrachten wir (8.1) in der Gestalt eines äquivalenten **Fixpunktproblems**

$$\phi(\mathbf{x}) = \mathbf{x} \quad (8.2)$$

und erinnern an den Fixpunktsatz für kontrahierende Abbildungen.

8.1 Fixpunktsatz für kontrahierende Abbildungen

Wir betrachten das Fixpunktproblem

$$\phi(\mathbf{x}) = \mathbf{x}, \quad (8.3)$$

mit einer gegebenen Funktion $\phi : \mathbb{R}^n \supset D \rightarrow \mathbb{R}^n$. Dieses ist äquivalent dem Nullstellenproblem (8.1), wenn wir z.B. $\phi(\mathbf{x}) = \mathbf{x} - \mathbf{A}f(\mathbf{x})$ mit einer regulären Matrix $\mathbf{A} \in \mathbb{R}^{(n,n)}$ wählen.

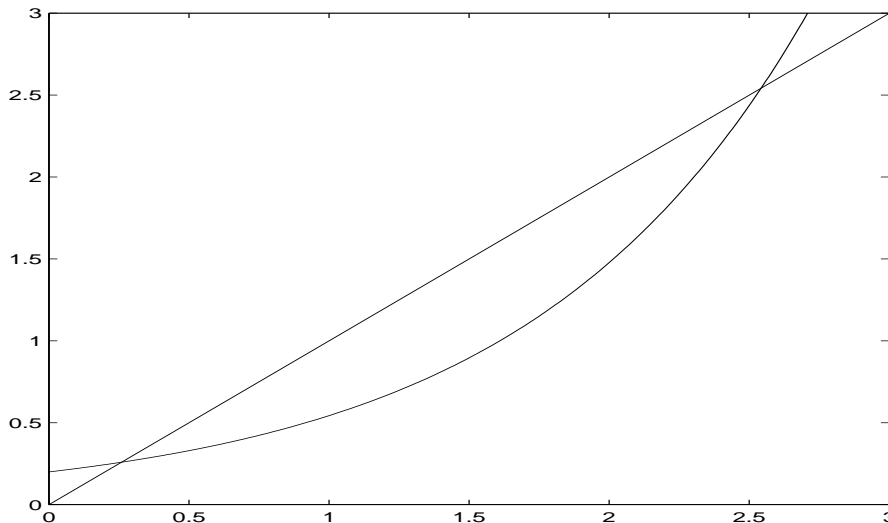


Abbildung 8.1 : Zu Beispiel 8.1.

Obwohl die meisten Aussagen auch in Banachräumen oder vollständigen metrischen Räumen richtig sind, beschränken wir uns auf den \mathbb{R}^n . Es bezeichne $\|\cdot\|$ irgendeine Vektornorm im \mathbb{R}^n oder eine passende Matrixnorm.

Für das Problem (8.3) liegt es nahe, eine Näherung \mathbf{x}^0 für eine Lösung zu wählen und ausgehend hiervon die Folge

$$\mathbf{x}^{m+1} = \phi(\mathbf{x}^m) \quad (8.4)$$

iterativ zu bestimmen.

Beispiel 8.1. Wir betrachten das reelle Fixpunktproblem

$$x = 0.2 \exp(x). \quad (8.5)$$

Der Graph von $\phi(x) := 0.2 \exp(x)$ ist in Abbildung 8.1 dargestellt, und man liest ab, dass ϕ (wenigstens) zwei Fixpunkte besitzt, einen in der Nähe von 0.25 und einen in der Nähe von 2.5 (und wenn man genauer zeichnen würde, würde man erkennen, dass der zweite Fixpunkt in der Nähe von 2.5426 liegt).

Tabelle 8.1 enthält die Ergebnisse der Iteration (8.4) für die Startwerte $x_0 = 0$, $x_0 = 2.5426$ und $x_0 = 2.5427$. Dieses Beispiel vermittelt den Eindruck, dass nicht alle Fixpunkte einer Funktion ϕ mit der Iteration (8.4) erreichbar sind, auch dann nicht, wenn man sehr nahe bei dem Fixpunkt startet. \square

Eine hinreichende Bedingung dafür, dass die Iteration (8.4) gegen einen Fixpunkt von ϕ konvergiert, liefert der Fixpunktsatz für kontrahierende Abbildungen. Aus ihm folgt zugleich die Existenz eines Fixpunktes.

Tabelle 8.1: Fixpunktiteration; Beispiel 8.1.

| m | x_m | x_m | x_m |
|----------|----------|----------|----------|
| 0 | 0.000000 | 2.542600 | 2.542700 |
| 1 | 0.200000 | 2.542536 | 2.542790 |
| 2 | 0.244281 | 2.542374 | 2.543021 |
| 3 | 0.255340 | 2.541962 | 2.543606 |
| 4 | 0.258180 | 2.540914 | 2.545094 |
| 5 | 0.258914 | 2.538252 | 2.548886 |
| \vdots | \vdots | \vdots | \vdots |
| 10 | 0.259171 | 2.138065 | 3.378282 |
| 11 | 0.259171 | 1.696602 | 5.864073 |

Definition 8.2. $\phi : D \rightarrow \mathbb{R}^n$ heißt **Lipschitz stetig** in D , wenn es eine Konstante $q \geq 0$ gibt mit

$$\|\phi(\mathbf{x}) - \phi(\mathbf{y})\| \leq q\|\mathbf{x} - \mathbf{y}\| \quad \text{für alle } \mathbf{x}, \mathbf{y} \in D. \quad (8.6)$$

Eine Lipschitz stetige Abbildung $\phi : D \rightarrow \mathbb{R}^n$ heißt **kontrahierend** (bzgl. der Vektornorm $\|\cdot\|$), wenn ϕ eine Lipschitz Konstante $q < 1$ besitzt. q heißt dann die **Kontraktionskonstante** von ϕ .

Bemerkung 8.3. Aus der Äquivalenz aller Vektornormen auf dem \mathbb{R}^n folgt, dass aus der Lipschitz Stetigkeit von ϕ bzgl. einer Norm folgt, dass ϕ auch bzgl. jeder anderen Vektornorm auf \mathbb{R}^n Lipschitz stetig ist. Für die Kontraktivität einer Abbildung ist dies nicht richtig; sie kann von der gewählten Norm abhängen. Man mache sich dies an einer Drehstreckung des \mathbb{R}^2 klar, die bzgl. der Euklidischen Norm sicher kontrahierend ist, wenn der Streckungsfaktor kleiner als 1 ist, bzgl. der Maximumnorm aber nicht, wenn der Streckungsfaktor genügend nahe bei 1 liegt und der Drehwinkel klein genug ist. \square

Die Lipschitz Stetigkeit und die Kontraktivität einer Funktion kann man häufig mit Hilfe des Mittelwertsatzes der Differentialrechnung nachweisen.

Satz 8.4. (i) Es sei $\phi : [a, b] \rightarrow \mathbb{R}$ stetig differenzierbar. ϕ ist genau dann kontrahierend mit der Kontraktionskonstante q , wenn gilt

$$q := \max_{x \in [a, b]} |\phi'(x)| < 1.$$

(ii) Es sei $D \subset \mathbb{R}^n$ konvex und $\phi : D \rightarrow \mathbb{R}^n$ stetig differenzierbar. Es bezeichne $\|\cdot\|$ eine Vektornorm und eine passende Matrixnorm, und es gelte

$$\|\phi'(\mathbf{x})\| \leq q < 1 \quad \text{für alle } \mathbf{x} \in D.$$

Dann ist ϕ kontrahierend in D bzgl. $\|\cdot\|$ mit der Kontraktionskonstante q .

Beweis: (i): Ist ϕ kontrahierend auf $[a, b]$ mit der Konstante $q < 1$, so ist

$$|\phi(x) - \phi(y)| \leq q|x - y| \quad \text{für alle } x, y \in [a, b].$$

Es folgt

$$\left| \frac{\phi(x) - \phi(y)}{x - y} \right| \leq q \quad \text{für alle } x, y \in [a, b], x \neq y,$$

und daher

$$\lim_{y \rightarrow x} \left| \frac{\phi(x) - \phi(y)}{x - y} \right| = |\phi'(x)| \leq q \quad \text{für alle } x \in [a, b].$$

Ist umgekehrt $|\phi'(x)| \leq q < 1$ für alle $x \in [a, b]$, so gilt nach dem Mittelwertsatz mit einem $\xi \in (a, b)$

$$|\phi(x) - \phi(y)| = |\phi'(\xi)| \cdot |x - y|,$$

und daher

$$|\phi(x) - \phi(y)| \leq q|x - y| \quad \text{für alle } x, y \in [a, b].$$

(ii): Da D konvex ist, liegt mit $\mathbf{x}, \mathbf{y} \in D$ auch die Verbindungsgerade in D , und nach dem Mittelwertsatz gilt

$$\|\phi(\mathbf{x}) - \phi(\mathbf{y})\| \leq \sup_{t \in [0,1]} \|\phi'(t\mathbf{x} + (1-t)\mathbf{y})\| \cdot \|\mathbf{x} - \mathbf{y}\| \leq q\|\mathbf{x} - \mathbf{y}\|. \quad \blacksquare$$

Satz 8.5. (Fixpunktsatz für kontrahierende Abbildungen)

Es sei $D \subset \mathbb{R}^n$ eine abgeschlossene Menge, $\phi : D \rightarrow \mathbb{R}^n$ eine kontrahierende Abbildung mit der Kontraktionskonstante q , und es gelte $\phi(D) \subset D$.

Dann gilt

(i) ϕ hat genau einen Fixpunkt $\bar{\mathbf{x}} \in D$.

(ii) Für jeden Startwert $\mathbf{x}^0 \in D$ konvergiert die durch $\mathbf{x}^{m+1} := \phi(\mathbf{x}^m)$ definierte Folge gegen $\bar{\mathbf{x}}$.

(iii) Es gelten die Fehlerabschätzungen

$$\|\bar{\mathbf{x}} - \mathbf{x}^m\| \leq \frac{q^m}{1-q} \|\mathbf{x}^1 - \mathbf{x}^0\| \quad (8.7)$$

$$\|\bar{\mathbf{x}} - \mathbf{x}^m\| \leq \frac{q}{1-q} \|\mathbf{x}^m - \mathbf{x}^{m-1}\|. \quad (8.8)$$

Beweis: Die Existenz eines Fixpunktes zeigen wir konstruktiv. Sei $\mathbf{x}^0 \in D$ beliebig gewählt. Wegen $\phi(D) \subset D$ ist dann die Folge $\{\mathbf{x}^m\}$, $\mathbf{x}^{m+1} := \phi(\mathbf{x}^m)$, definiert und $\{\mathbf{x}^m\} \subset D$.

Aus

$$\|\mathbf{x}^{m+1} - \mathbf{x}^m\| = \|\phi(\mathbf{x}^m) - \phi(\mathbf{x}^{m-1})\| \leq q \|\mathbf{x}^m - \mathbf{x}^{m-1}\|$$

erhält man durch Induktion für alle $k \in \mathbb{N}$

$$\|\mathbf{x}^{m+k} - \mathbf{x}^{m+k-1}\| \leq q^k \|\mathbf{x}^m - \mathbf{x}^{m-1}\|,$$

und daher

$$\begin{aligned} \|\mathbf{x}^{m+p} - \mathbf{x}^m\| &= \left\| \sum_{k=1}^p (\mathbf{x}^{m+k} - \mathbf{x}^{m+k-1}) \right\| \\ &\leq \sum_{k=1}^p \|\mathbf{x}^{m+k} - \mathbf{x}^{m+k-1}\| \leq \sum_{k=1}^p q^k \|\mathbf{x}^m - \mathbf{x}^{m-1}\| \\ &\leq q \|\mathbf{x}^m - \mathbf{x}^{m-1}\| \sum_{k=0}^{p-1} q^k \leq \frac{q}{1-q} \|\mathbf{x}^m - \mathbf{x}^{m-1}\| \\ &\leq \frac{q^m}{1-q} \|\mathbf{x}^1 - \mathbf{x}^0\| \longrightarrow 0. \end{aligned} \quad (8.9)$$

Nach dem Cauchyschen Konvergenzkriterium ist $\{\mathbf{x}^m\}$ konvergent gegen ein $\bar{\mathbf{x}}$, da D abgeschlossen ist, gilt $\bar{\mathbf{x}} \in D$, und wegen der Stetigkeit von ϕ ist $\bar{\mathbf{x}}$ ein Fixpunkt von ϕ .

$\bar{\mathbf{x}}$ ist der einzige Fixpunkt von ϕ in D , denn ist $\hat{\mathbf{x}}$ ein beliebiger Fixpunkt von ϕ in D , so gilt

$$\|\bar{\mathbf{x}} - \hat{\mathbf{x}}\| = \|\phi(\bar{\mathbf{x}}) - \phi(\hat{\mathbf{x}})\| \leq q \|\bar{\mathbf{x}} - \hat{\mathbf{x}}\|,$$

d.h. $0 \leq (1-q) \|\bar{\mathbf{x}} - \hat{\mathbf{x}}\| \leq 0$, und wegen $q < 1$ folgt $\bar{\mathbf{x}} = \hat{\mathbf{x}}$.

Die Fehlerabschätzungen (8.7) und (8.8) erhält man unmittelbar aus (8.9) mit $p \rightarrow \infty$. ■

Bemerkung 8.6. Die Ungleichung (8.9) zeigt, dass Abschätzung (8.8) bessere Fehlerschranken liefert als die Abschätzung (8.7).

Der Vorteil der Abschätzung (8.7) liegt darin, dass man mit ihr schon zu Beginn der Rechnung den Fehler einer Iterierten \mathbf{x}^m abschätzen kann, ohne \mathbf{x}^m berechnet zu haben. Man kann mit ihr also abschätzen, wie hoch der Aufwand sein wird, um eine vorgegebene Genauigkeit ε zu erreichen. Es gilt ja

$$\|\bar{\mathbf{x}} - \mathbf{x}^m\| \leq \frac{q^m}{1-q} \|\mathbf{x}^1 - \mathbf{x}^0\| \leq \varepsilon, \quad \text{falls } m \geq \frac{\ln\left(\frac{\varepsilon(1-q)}{\|\mathbf{x}^1 - \mathbf{x}^0\|}\right)}{\ln q}.$$

Tabelle 8.2: Fixpunktiteration mit Abschätzung

| m | $x(n)$ | a priori | a posteriori | Fehler |
|-----|--------------------|--------------|--------------|--------------|
| 0 | 0.5000000000000000 | | | |
| 1 | 0.877582561890373 | $2.00E + 00$ | $2.00E + 00$ | $1.38E - 01$ |
| 2 | 0.639012494165259 | $1.69E + 00$ | $1.27E + 00$ | $1.00E - 01$ |
| 3 | 0.802685100682335 | $1.42E + 00$ | $8.69E - 01$ | $6.36E - 02$ |
| 4 | 0.694778026788006 | $1.19E + 00$ | $5.73E - 01$ | $4.43E - 02$ |
| 5 | 0.768195831282016 | $1.00E + 00$ | $3.90E - 01$ | $2.91E - 02$ |
| 6 | 0.719165445942419 | $8.46E - 01$ | $2.60E - 01$ | $1.99E - 02$ |
| 7 | 0.752355759421527 | $7.12E - 01$ | $1.76E - 01$ | $1.33E - 02$ |
| 8 | 0.730081063137823 | $5.99E - 01$ | $1.18E - 01$ | $9.00E - 03$ |
| 9 | 0.745120341351440 | $5.04E - 01$ | $7.98E - 02$ | $6.04E - 03$ |
| 10 | 0.735006309014843 | $4.24E - 01$ | $5.37E - 02$ | $4.08E - 03$ |
| 20 | 0.739006779780813 | $7.55E - 02$ | $1.03E - 03$ | $7.84E - 05$ |
| 30 | 0.739083626103480 | $1.34E - 02$ | $1.99E - 05$ | $1.51E - 06$ |
| 40 | 0.739085104225471 | $2.39E - 03$ | $3.82E - 07$ | $2.90E - 08$ |
| 50 | 0.739085132657536 | $4.25E - 04$ | $7.35E - 09$ | $5.58E - 10$ |

(8.7) heißt daher eine **a priori Abschätzung**. Mit (8.8) kann man den Fehler von \boldsymbol{x}^m erst abschätzen, nachdem man \boldsymbol{x}^m berechnet hat. (8.8) ist eine **a posteriori Abschätzung**. Sie kann benutzt werden, um während der Rechnung die Güte der Näherung zu kontrollieren und bei ausreichender Genauigkeit die Iteration abzubrechen. \square

Beispiel 8.7. Wir betrachten das Fixpunktproblem

$$x = \phi(x) := \cos x, \quad x \in I := [0, 1].$$

Da ϕ monoton fallend in $[0, 1]$ ist, folgt aus $\phi(0) = 1 \in I$ und $\phi(1) = 0.54 \in I$, dass $\phi(I) \subset I$ gilt.

ϕ ist kontrahierend auf I mit der Kontraktionskonstante $q = 0.85$, denn

$$\max_{x \in I} |\phi'(x)| = \max_{x \in I} |\sin x| = \sin 1 \leq 0.85 =: q.$$

Man erhält die Näherungen und Fehlerabschätzungen in Tabelle 8.2. Während die a posteriori Abschätzung ziemlich realistische Werte für den Fehler liefert, wird der Fehler durch die a priori Abschätzung sehr stark überschätzt. \square

Oft ist es schwierig, insbesondere bei der Anwendung auf Funktionen von mehreren Veränderlichen, eine Menge D zu bestimmen, die durch ϕ in sich abgebildet wird. Hilfreich hierbei ist

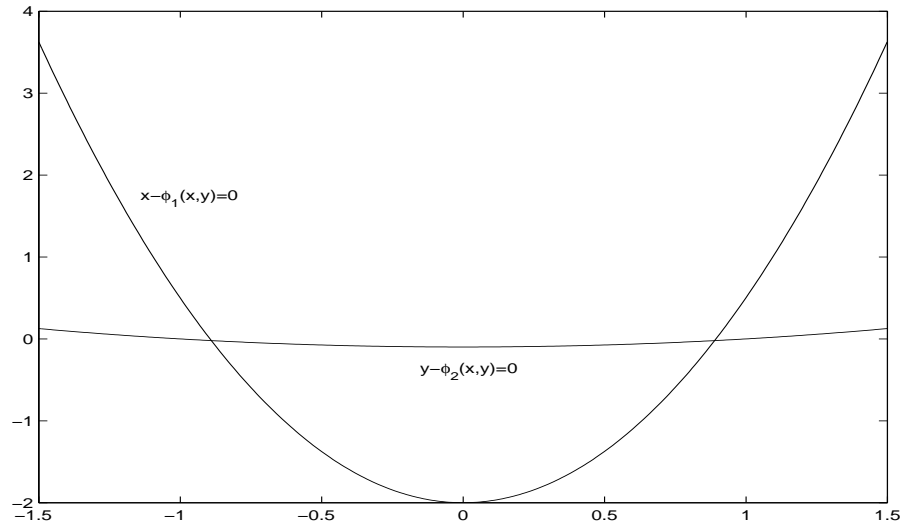


Abbildung 8.2 : Zu Beispiel 8.9.

Korollar 8.8. *Es sei $D \subset \mathbb{R}^n$ und $\phi : D \rightarrow \mathbb{R}^n$. Es sei ϕ kontrahierend in der Kugel $K := \{\mathbf{y} \in \mathbb{R}^n : \|\mathbf{y} - \mathbf{z}\| \leq r\}$ mit der Kontraktionskonstante q , und es gelte die Kugelbedingung*

$$\|\phi(\mathbf{z}) - \mathbf{z}\| \leq (1 - q)r. \quad (8.10)$$

Dann gelten die Aussagen (i) - (iii) von Satz 8.5. mit K an Stelle von D .

Beweis: Wir haben nur zu zeigen, dass $\phi(K) \subset K$ gilt. Für $\mathbf{y} \in K$ ist

$$\|\phi(\mathbf{y}) - \mathbf{z}\| \leq \|\phi(\mathbf{y}) - \phi(\mathbf{z})\| + \|\phi(\mathbf{z}) - \mathbf{z}\| \leq q\|\mathbf{y} - \mathbf{z}\| + (1 - q)r \leq r. \quad \blacksquare$$

Beispiel 8.9. Wir betrachten das Fixpunktproblem

$$\begin{pmatrix} x \\ y \end{pmatrix} = \phi(x, y) := \begin{pmatrix} 0.4 - 0.5x^2 + x + 0.2y \\ 0.1(x^2 + y^2 - 1) \end{pmatrix}. \quad (8.11)$$

Abbildung 8.2 zeigt, dass in der Nähe des Punktes $(x_0, y_0)^T := (1, 0)^T$ eine Lösung liegt.

Es gilt

$$\left\| \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} - \phi(x_0, y_0) \right\|_{\infty} = \left\| \begin{pmatrix} 1 \\ 0 \end{pmatrix} - \begin{pmatrix} 0.9 \\ 0 \end{pmatrix} \right\|_{\infty} = 0.1$$

und wegen

$$\phi'(x, y) = \begin{pmatrix} 1 - x & 0.2 \\ 0.2x & 0.2y \end{pmatrix}$$

gilt in der Kugel (bzgl. der ∞ -Norm)

$$K_r(x_0, y_0) = [1 - r, 1 + r] \times [-r, r]$$

(im Falle $r \leq 1$)

$$\begin{aligned} \|\phi'(x, y)\|_\infty &= \max\{|1 - x| + 0.2, 0.2|x| + 0.2|y|\} \\ &\leq \max(r + 0.2, 0.2(1 + r) + 0.2r) = 0.2 + r. \end{aligned}$$

Die Kugelbedingung (8.10) lautet also

$$0.1 \leq r(1 - (r + 0.2)) = 0.8r - r^2,$$

und diese ist erfüllt für

$$r \in [0.4 - \sqrt{0.06}, 0.4 + \sqrt{0.06}] \subset [0.16, 0.64].$$

Daher besitzt ϕ genau eine Lösung in der Kugel $K_{0.16}(x_0, y_0)$. Mit der Fixpunktiteration erhält man hierfür die Näherung $(0.88977, -0.02079)^T$. \square

Wir haben schon in Beispiel 8.1. gesehen, dass nicht jeder Fixpunkt $\bar{\mathbf{x}}$ einer Abbildung ϕ mit der Iteration $\mathbf{x}^{m+1} := \phi(\mathbf{x}^m)$ erreicht werden kann.

Definition 8.10. Ein Fixpunkt $\bar{\mathbf{x}}$ von $\phi : D \rightarrow \mathbb{R}^n$, $D \subset \mathbb{R}^n$, heißt **anziehend**, wenn die Fixpunktiteration für jeden genügend nahe bei $\bar{\mathbf{x}}$ liegenden Startwert \mathbf{x}^0 gegen $\bar{\mathbf{x}}$ konvergiert (d.h. es gibt ein $\varepsilon > 0$, so dass $\|\mathbf{x}^0 - \bar{\mathbf{x}}\| \leq \varepsilon$, $\mathbf{x}^{m+1} := \phi(\mathbf{x}^m)$, zur Folge hat $\lim_{m \rightarrow \infty} \mathbf{x}^m = \bar{\mathbf{x}}$); er heißt **abstoßend**, wenn es eine Kugel gibt mit dem Mittelpunkt $\bar{\mathbf{x}}$, so dass für jeden Startwert $\mathbf{x}^0 \neq \bar{\mathbf{x}}$ aus dieser Kugel die Fixpunktiteration aus der Kugel herausführt (d.h. es gibt ein $\varepsilon > 0$, so dass zu jedem $\mathbf{x}^0 \in \mathbb{R}^n$ mit $\|\mathbf{x}^0 - \bar{\mathbf{x}}\| < \varepsilon$, $\mathbf{x}^0 \neq \bar{\mathbf{x}}$, ein $m \in \mathbb{N}$ existiert mit $\|\mathbf{x}^m - \bar{\mathbf{x}}\| \geq \varepsilon$, wobei $\mathbf{x}^{m+1} := \phi(\mathbf{x}^m)$).

Im Falle einer stetig differenzierbaren reellen Funktion $\phi : I \rightarrow \mathbb{R}$ ist ein Fixpunkt \bar{x} anziehend, falls $|\phi'(\bar{x})| < 1$ gilt, denn zu $\varepsilon > 0$ mit $|\phi'(\bar{x})| + \varepsilon =: q < 1$ gibt es ein $\delta > 0$ mit

$$|\phi'(x)| \leq |\phi'(\bar{x})| + |\phi'(x) - \phi'(\bar{x})| \leq |\phi'(\bar{x})| + \varepsilon = q < 1$$

für alle $x \in [\bar{x} - \delta, \bar{x} + \delta] =: J$, d.h. ϕ ist kontrahierend auf J , und wegen $|\bar{x} - \phi(\bar{x})| = 0 \leq (1 - q)\delta$ wird J nach Korollar 8.8. durch ϕ in sich abgebildet.

Ist $|\phi'(\bar{x})| > 1$, so gibt es ein Intervall $J := [\bar{x} - r, \bar{x} + r]$ mit

$$|\phi'(x)| \geq |\phi'(\bar{x})| - |\phi'(x) - \phi'(\bar{x})| \geq q > 1$$

für alle $x \in J$, und daher gilt für $x \in J$ mit einem $\xi = \bar{x} + \theta(x - \bar{x})$

$$|\bar{x} - \phi(x)| = |\phi(\bar{x}) - \phi(x)| = |\phi'(\xi)| \cdot |\bar{x} - x| \geq q |\bar{x} - x|,$$

und es folgt, dass \bar{x} abstoßend ist.

Ist $|\phi'(\bar{x})| = 1$, so kann \bar{x} anziehend oder abstoßend oder keines von beiden sein. Dies zeigen die Beispiele $\phi(x) := x - x^3$, $\phi(x) := x + x^3$ und $\phi(x) := x - x^2$ für den Fixpunkt $\bar{x} = 0$.

8.2 Nullstellen reeller Funktionen

Wir betrachten für das Nullstellenproblem

$$f(x) = 0 \tag{8.12}$$

für die reelle Funktion $f : [a, b] \rightarrow \mathbb{R}$ zwei Typen von Verfahren. Zunächst untersuchen wir Methoden, die sich durch Linearisierung von f ergeben, danach Verfahren, die auf dem Zwischenwertsatz beruhen und Einschließungen der Nullstelle liefern. Verbreiteter ist der erste Typ von Methoden (er ist auch der Ausgangspunkt für die Entwicklung von Methoden für nichtlineare Systeme von Gleichungen), effizienter der zweite Typ.

8.2.1 Newton Verfahren

Wir betrachten die nichtlineare Gleichung (8.12) und setzen voraus, dass $f : [a, b] \rightarrow \mathbb{R}$ differenzierbar ist.

Um eine gegebene Näherung x_0 für eine Nullstelle $\bar{x} \in (a, b)$ von f zu verbessern, ersetzen wir f durch ihre Linearisierung

$$\tilde{f}(x) = f(x_0) + f'(x_0)(x - x_0)$$

in x_0 . Gilt $f'(x_0) \neq 0$, so besitzt die Ersatzfunktion \tilde{f} genau eine Nullstelle

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)},$$

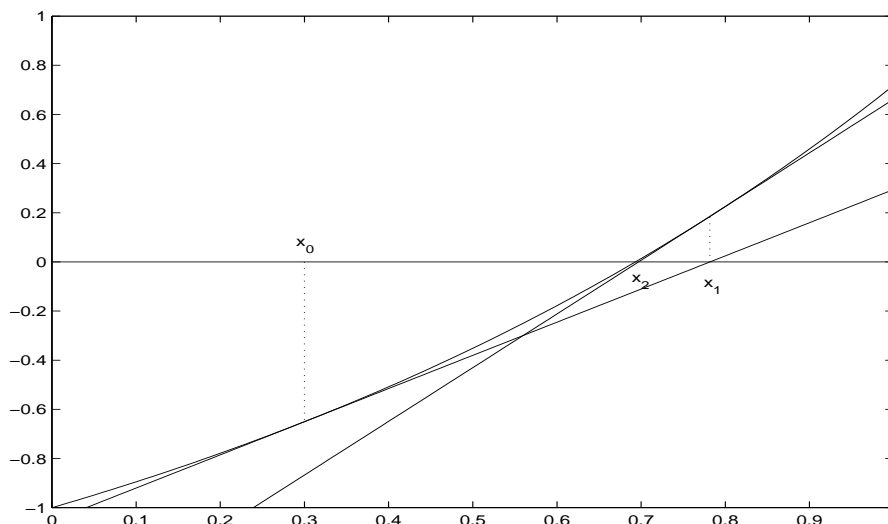


Abbildung 8.3: Newton Verfahren

Tabelle 8.3: Newton Verfahren; Beispiel 8.11.

| m | x_m | Fehler |
|-----|-----------------|--------------|
| 0 | 2.5 | $4.26e - 02$ |
| 1 | 2.5443 | $1.56e - 03$ |
| 2 | 2.5426434 | $2.01e - 06$ |
| 3 | 2.5426413577769 | $3.34e - 12$ |

die wir als neue Näherung für \bar{x} auffassen.

Wiederholt man diesen Schritt iterativ mit x_n an Stelle von x_0 , so erhält man das **Newton Verfahren**

$$x_{n+1} := x_n - \frac{f(x_n)}{f'(x_n)}. \quad (8.13)$$

Beispiel 8.11. Wir bestimmen den größeren der beiden Fixpunkte von $\phi(x) = 0.2 \exp(x)$. Aus Abbildung 8.1 liest man die Anfangsnäherung $x_0 = 2.5$ ab, und mit $f(x) := x - 0.2 \exp(x)$ lautet das Newton Verfahren

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} = x_n - \frac{x_n - 0.2 \exp(x_n)}{1 - 0.2 \exp(x_n)}.$$

Hiermit erhält man die Näherungen und Fehler in Tabelle 8.3. □

Das Newton Verfahren kann geschrieben werden als

$$x_{n+1} = \phi(x_n), \quad n \in \mathbb{N}_0,$$

mit

$$\phi(x) := x - \frac{f(x)}{f'(x)}.$$

ϕ ist auf $D := \{x \in [a, b] : f'(x) \neq 0\}$ definiert, und die Nullstellen von f in D sind genau die Fixpunkte von ϕ in D . Wendet man den Fixpunktsatz für kontrahierende Abbildungen (Satz 8.5.) auf dieses ϕ an, so erhält man

Satz 8.12. *Es sei $f : I \rightarrow \mathbb{R}$ zweimal stetig differenzierbar und \bar{x} eine einfache Nullstelle von f im Innern von I . Dann gibt es ein $r > 0$, so dass das Newton Verfahren für alle Startwerte $x_0 \in I$ mit $|\bar{x} - x_0| \leq r$ gegen \bar{x} konvergiert.*

Beweis: \bar{x} ist Fixpunkt von

$$\phi(x) := x - \frac{f(x)}{f'(x)}.$$

Wegen der Stetigkeit von f' und $f'(\bar{x}) \neq 0$ gilt $f'(x) \neq 0$ für alle $x \in J := [\bar{x} - \rho, \bar{x} + \rho]$ mit einem geeigneten $\rho > 0$. Daher ist ϕ in J definiert und stetig differenzierbar mit

$$\phi'(x) = \frac{f(x) f''(x)}{(f'(x))^2},$$

d.h. $\phi'(\bar{x}) = 0$, und \bar{x} ist ein anziehender Fixpunkt von ϕ . ■

Satz 8.12. ist ein typischer lokaler Konvergenzsatz: Wenn der Startwert x_0 nur genügend nahe an der Lösung liegt ($|x - x_0| \leq r$), so konvergiert das Verfahren. Über die Größe von r wird nichts ausgesagt.

Aus den Abschätzungen von Satz 8.5. folgt, dass der Fehler beim Newton Verfahren wie eine geometrische Folge gegen 0 geht. Tatsächlich gilt nach dem Taylorschen Satz

$$\begin{aligned} |x_{n+1} - \bar{x}| &= \left| x_n - \frac{f(x_n)}{f'(x_n)} - \bar{x} \right| = \left| \frac{f(\bar{x}) - f(x_n) - f'(x_n)(\bar{x} - x_n)}{f'(x_n)} \right| \\ &= \frac{1}{2} \left| \frac{f''(\xi_n)}{f'(x_n)} (\bar{x} - x_n)^2 \right|, \quad \xi_n = \bar{x} + \theta_n (\bar{x} - x_n). \end{aligned}$$

Da zu $\varepsilon > 0$ ein $r > 0$ existiert mit

$$|f'(x)| \geq \frac{1}{2}|f'(\bar{x})| \quad \text{und} \quad |f''(x)| \leq \varepsilon + |f''(\bar{x})|$$

für alle x mit $|\bar{x} - x| \leq r$ und da nach Satz 8.12. x_n gegen \bar{x} konvergiert, erhält man

$$|x_{n+1} - \bar{x}| \leq \frac{|f''(\bar{x})| + \varepsilon}{|f'(\bar{x})|} |\bar{x} - x_n|^2 =: C |\bar{x} - x_n|^2.$$

Bis auf die multiplikative Konstante C wird also der Fehler beim Newton Verfahren von Schritt zu Schritt quadriert, was die rasche Konvergenz des Verfahrens in Beispiel 8.11. erklärt.

Um die Konvergenzgeschwindigkeit von Folgen zu vergleichen, führen wir die folgenden Begriffe ein:

Definition 8.13. Sei $\{\mathbf{x}^m\} \subset \mathbb{R}^n$ eine konvergente Folge mit $\lim_{m \rightarrow \infty} \mathbf{x}^m = \bar{\mathbf{x}}$. Die Folge $\{\mathbf{x}^m\}$ besitzt wenigstens die **Q-Ordnung** $p \in [1, \infty)$, wenn es eine Konstante $C > 0$ und ein $m_0 \in \mathbb{N}$ gibt mit

$$\frac{\|\bar{\mathbf{x}} - \mathbf{x}^{m+1}\|}{\|\bar{\mathbf{x}} - \mathbf{x}^m\|^p} \leq C \quad \text{für alle } m \geq m_0.$$

Ist $p = 1$, so fordern wir zusätzlich $C < 1$, damit die lokale Konvergenz überhaupt gesichert ist.

Das maximale p mit dieser Eigenschaft heißt die **Q-Konvergenzordnung** der Folge.

Ein Iterationsverfahren hat die Q-Konvergenzordnung p , wenn jede mit ihm erzeugte, konvergente Folge die Q-Konvergenzordnung p hat.

Eine Folge (ein Verfahren) **konvergiert Q-linear**, falls $p = 1$ ist, **Q-quadratisch** im Falle $p = 2$ und **Q-superlinear**, falls

$$\limsup_{m \rightarrow \infty} \frac{\|\bar{\mathbf{x}} - \mathbf{x}^{m+1}\|}{\|\bar{\mathbf{x}} - \mathbf{x}^m\|} = 0$$

gilt.

Wir werden später noch den Begriff der Konvergenzordnung modifizieren. Zur Unterscheidung verwenden wir für die hier eingeführten Konvergenzbegriffe das Präfix Q, da sich die Ordnung auf Quotienten von aufeinanderfolgenden Fehlern bezieht.

Die durch Satz 8.5. erfassten Verfahren konvergieren wenigstens Q-linear, das Newton Verfahren konvergiert lokal Q-quadratisch gegen einfache Nullstellen von f , wenn f zweimal stetig differenzierbar in einer Umgebung der Nullstelle ist.

Allgemeiner erhält man

Satz 8.14. Sei $I := [a, b] \subset \mathbb{R}$, $p > 1$ und $\bar{x} \in (a, b)$ ein Fixpunkt von $\phi \in C^p[a, b]$ mit $\phi^{(j)}(\bar{x}) = 0$, $j = 1, 2, \dots, p - 1$.

Dann gibt es eine Umgebung $U(\bar{x}) \subset (a, b)$ von \bar{x} , so dass für alle Startwerte $x_0 \in U(\bar{x})$ die Folge $x_{n+1} = \phi(x_n)$ von mindestens der Q-Ordnung p gegen \bar{x} konvergiert.

Beweis: Dass das Verfahren lokal konvergent ist, erhält man wieder aus Korollar 8.8.

Nach dem Taylorsche Satz gilt für $x_n \in (a, b)$,

$$|x_{n+1} - \bar{x}| = |\phi(x_n) - \phi(\bar{x})| = \left| \frac{\phi^{(p)}(\xi)}{p!} (\bar{x} - x_n)^p \right|$$

für ein $\xi \in (a, b)$, mit

$$C := \max \left\{ \frac{1}{p!} |\phi^{(p)}(\xi)| : \xi \in [a, b] \right\}$$

also die Abschätzung

$$|x_{n+1} - \bar{x}| \leq C|x_n - \bar{x}|^p. \quad \blacksquare$$

Bemerkung 8.15. Mit

$$\phi(x) := x - \frac{f(x)}{f'(x)}$$

erhält man aus Satz 8.14. wegen

$$\phi'(x) = \frac{f(x) f''(x)}{(f'(x))^2}$$

wieder die Q-quadratische Konvergenz des Newton Verfahrens gegen einfache Nullstellen von f (allerdings unter der stärkeren Voraussetzung $f \in C^3[a, b]$). \square

Bemerkung 8.16. Man kann mit Hilfe von Satz 8.14. leicht (bei genügender Glattheit von f) Verfahren höherer Ordnung als 2 konstruieren. Diese werden in der Praxis jedoch kaum angewendet. Wir geben einen Weg an. Weitere Möglichkeiten findet man z.B. in Werner [118].

Sei \bar{x} einfache Nullstelle von f . Wir machen den Ansatz

$$\phi(x) = x - g(x)f(x) - h(x)f^2(x),$$

wobei g und h in einer Umgebung von \bar{x} genügend oft differenzierbar sind und $g(\bar{x}) \neq 0$, $h(\bar{x}) \neq 0$ gilt. Es ist

$$\phi'(x) = 1 - g'(x)f(x) - g(x)f'(x) - h'(x)f^2(x) - 2h(x)f(x)f'(x),$$

d.h. $\phi'(\bar{x}) = 0$ gilt genau dann, wenn $g(\bar{x}) = \frac{1}{f'(\bar{x})}$. Damit also das Verfahren wenigstens quadratisch konvergiert, wählen wir $g(x) = \frac{1}{f'(x)}$.

Weiter ist

$$\phi''(x) = -g''(x)f(x) - 2g'(x)f'(x) - g(x)f''(x) - 2h(x)(f'(x))^2 - f(x)\{\dots\},$$

Tabelle 8.4: $f(x) = x^2 - 3$

| n | Newton | verb. Newton |
|-----|-------------------|-------------------|
| 1 | 1.750000000000000 | 1.734375000000000 |
| 2 | 1.73214285714286 | 1.73205080965507 |
| 3 | 1.73205081001473 | 1.73205080756888 |
| 4 | 1.73205080756888 | |

d.h. wegen $g'(x) = -\frac{f''(x)}{(f'(x))^2}$,

$$\phi''(\bar{x}) = -2g'(\bar{x})f'(\bar{x}) - g(\bar{x})f''(\bar{x}) - 2h(\bar{x})(f'(\bar{x}))^2 = \frac{f''(\bar{x})}{f'(\bar{x})} - 2h(\bar{x})(f'(\bar{x}))^2,$$

und daher gilt $\phi''(\bar{x}) = 0$ genau dann, wenn

$$h(\bar{x}) = \frac{f''(\bar{x})}{2(f'(\bar{x}))^3}.$$

Wählt man also

$$\phi(x) = x - \frac{f(x)}{f'(x)} - \frac{f''(x)(f(x))^2}{2(f'(x))^3}, \quad (8.14)$$

so ist das Verfahren $x_{n+1} = \phi(x_n)$ lokal konvergent von mindestens der Q-Ordnung 3. (8.14) Es heißt **verbessertes Newton Verfahren**. \square

Beispiel 8.17. Wir berechnen $\sqrt{3}$ mit dem Newton Verfahren bzw. dem verbesserten Newton Verfahren mit dem Startwert $x_0 = 2$. Die Näherungen enthält Tabelle 8.4. \square

Global lohnt sich der Mehraufwand nicht. Man kann aber, nachdem man mit dem Newton Verfahren bereits eine gute Näherung erhalten hat, mit einem zusätzlichen Schritt mit dem verbesserten Newton Verfahren die Genauigkeit wesentlich steigern (vgl. Werner [118], p. 87).

Ist \bar{x} eine mehrfache Nullstelle von f , so konvergiert das Newton Verfahren immer noch lokal, aber nur linear, also wie vom Fixpunktsatz für kontrahierende Abbildungen (Satz 8.5.) vorhergesagt. Genauer gilt Satz 8.18.

Satz 8.18. *Ist $f : [a, b] \rightarrow \mathbb{R}$ $(k+3)$ -mal stetig differenzierbar, und besitzt f eine Nullstelle $\bar{x} \in (a, b)$ der Vielfachheit $k \geq 2$, so konvergiert das Verfahren*

$$x_{n+1} := x_n - \alpha \frac{f(x_n)}{f'(x_n)} \quad (8.15)$$

für alle $\alpha \in (0, 2k)$ lokal Q -linear gegen \bar{x} . Für $\alpha = k$ ist die Konvergenz sogar Q -quadratisch.

Tabelle 8.5: Newton Verfahren

| n | Newton für f | (8.15) mit $\alpha = 3$ | Newton für g |
|-----|----------------|-------------------------|----------------|
| 0 | $1.00e + 00$ | $1.00e + 00$ | $1.00e + 00$ |
| 1 | $6.55e - 01$ | $3.46e - 02$ | $6.48e - 02$ |
| 2 | $4.34e - 01$ | $1.38e - 06$ | $1.81e - 05$ |
| 3 | $2.88e - 01$ | $2.33e - 13$ | $3.40e - 14$ |

Beweis: Es ist $f(x) = (x - \bar{x})^k g(x)$, wobei $g(\bar{x}) \neq 0$ gilt und g dreimal stetig differenzierbar ist. Hiermit kann man die Iterationsvorschrift (8.15) schreiben als

$$x_{n+1} = \phi(x_n), \quad \phi(x) = x - \alpha \frac{(x - \bar{x}) g(x)}{k g(x) + (x - \bar{x}) g'(x)}.$$

ϕ ist sicher in einer Umgebung von \bar{x} definiert und differenzierbar, und es gilt mit $h(x) := k g(x) + (x - \bar{x}) g'(x)$

$$\phi'(x) = 1 - \alpha \frac{h(x) (g(x) + (x - \bar{x}) g'(x)) - (x - \bar{x}) g(x) h'(x)}{(h(x))^2},$$

d.h.

$$\phi'(\bar{x}) = 1 - \alpha \frac{h(\bar{x}) g(\bar{x})}{(h(\bar{x}))^2} = 1 - \frac{\alpha}{k}.$$

Für $\alpha \in (0, 2k)$ gilt $|\phi'(\bar{x})| < 1$, und \bar{x} ist ein anziehender Fixpunkt von ϕ . Im Falle $\alpha = k$ gilt $\phi'(\bar{x}) = 0$, und die quadratische Konvergenz folgt aus Satz 8.14. ■

Im allgemeinen ist die Vielfachheit einer Nullstelle nicht bekannt. Ist \bar{x} eine k -fache Nullstelle von f , so ist \bar{x} eine $(k - 1)$ -fache Nullstelle von f' , und daher hat

$$g(x) := \frac{f(x)}{f'(x)}$$

die einfache Nullstelle \bar{x} . Das Newton Verfahren für g konvergiert also Q-quadratisch.

Beispiel 8.19. Für die dreifache Nullstelle $\bar{x} = 0$ von

$$f(x) = \sin x - x$$

erhält man die Fehler in Tabelle 8.5. □

Da die Berechnung der Ableitung $f'(x)$ oft aufwändig ist, ersetzt man im Newton Verfahren $f'(x_n)$ durch $f'(x_0)$, berechnet also die Ableitung nur im Startwert x_0 . Man erhält dann das **vereinfachte Newton Verfahren**

$$x_{n+1} := x_n - \frac{f(x_n)}{f'(x_0)}. \quad (8.16)$$

Hierfür gilt der folgende Konvergenzsatz

Satz 8.20. Ist $f : [a, b] \rightarrow \mathbb{R}$ stetig differenzierbar und $\bar{x} \in (a, b)$ eine einfache Nullstelle von f , so gibt es ein $r > 0$, so dass das vereinfachte Newton Verfahren (8.16) für alle $x_0 \in [\bar{x} - r, \bar{x} + r]$ gegen \bar{x} konvergiert. Die Konvergenz ist Q-linear.

Beweis: Wir wenden Korollar 8.8. für festes $x_0 \in I_r := [\bar{x} - r, \bar{x} + r]$ und geeignetes $r > 0$ auf die Iterationsfunktion

$$\phi(x; x_0) := x - \frac{f(x)}{f'(x_0)}$$

an.

Wegen $\phi(\bar{x}; x_0) = \bar{x}$ muss das Intervall I_r nur so klein gewählt werden, dass für jedes $x_0 \in I_r$ die Funktion $\phi(\cdot; x_0) : I_r \rightarrow \mathbb{R}$ kontrahierend auf I_r ist, d.h. so dass für ein $q \in [0, 1)$

$$\max_{x \in I_r} \left| 1 - \frac{f'(x)}{f'(x_0)} \right| \leq q$$

gilt. Dass diese Wahl von r möglich ist, folgt aus der Stetigkeit der Funktion

$$\psi(x, x_0) := 1 - \frac{f'(x)}{f'(x_0)}$$

in einer Umgebung des Punktes (\bar{x}, \bar{x}) und $\psi(\bar{x}, \bar{x}) = 0$. ■

Eine weitere Möglichkeit zur Gewinnung eines ableitungsfreien Verfahrens ist, die Ableitung $f'(x_n)$ im Newton Verfahren durch den Differenzenquotienten

$$\frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}}$$

(die Funktion f also durch ihre Sekante zu den Stützstellen x_n und x_{n-1}) zu ersetzen. Man erhält dann das **Sekantenverfahren** (manchmal auch nicht ganz korrekt **regula falsi Verfahren**)

$$x_{n+1} := x_n - \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})} f(x_n).$$

Wie das Newton Verfahren konvergiert das Sekantenverfahren lokal, d.h. wenn die Startwerte x_0 und x_1 genügend nahe bei der Nullstelle \bar{x} gewählt werden, falls \bar{x} eine einfache Nullstelle ist.

Für das Sekantenverfahren verwenden wir einen schwächeren Konvergenzordnungsbegriff als den der Q-Ordnung.

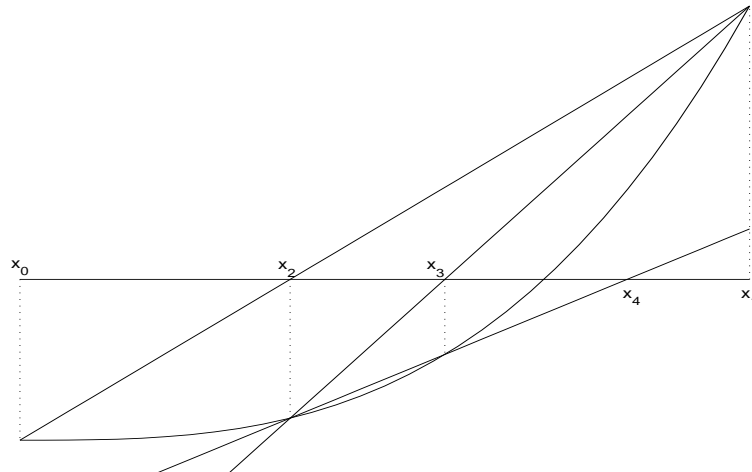


Abbildung 8.4: Sekantenverfahren

Definition 8.21. Eine Folge $\{\mathbf{x}^m\} \subset \mathbb{R}^n$ konvergiert gegen $\bar{\mathbf{x}}$ von wenigstens der **R-Ordnung** p , wenn es eine reelle Nullfolge $\{r_m\}$ mit

$$\|\mathbf{x}^m - \bar{\mathbf{x}}\| \leq r_m$$

gibt, die wenigstens von der Q -Ordnung p konvergiert.

Die R-Ordnung wird für Mehrpunktverfahren (die \mathbf{x}^{m+1} unter Benutzung von mehreren Vorgängern $\mathbf{x}^m, \dots, \mathbf{x}^{m-k}$ bestimmen) wie das Sekantenverfahren häufig mit Hilfe der positiven Nullstelle (engl.: root) eines Polynoms bestimmt. Dies erklärt das Präfix R.

Satz 8.22. Sei f zweimal stetig differenzierbar in der Umgebung einer einfachen Nullstelle \bar{x} von f . Dann konvergiert das Sekantenverfahren lokal von wenigstens der R-Ordnung $p = \frac{1}{2}(1 + \sqrt{5})$.

Beweis: Es gilt

$$\begin{aligned} x_{n+1} - \bar{x} &= x_n - \bar{x} - \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})} f(x_n) \\ &= x_n - \bar{x} - \frac{f(x_n)}{[x_n, x_{n-1}]}, \end{aligned}$$

und wegen

$$f(x_n) = \frac{f(x_n) - f(\bar{x})}{x_n - \bar{x}} (x_n - \bar{x}) = [x_n, \bar{x}] (x_n - \bar{x})$$

erhält man weiter

$$\begin{aligned} x_{n+1} - \bar{x} &= (x_n - \bar{x}) \left(1 - \frac{[x_n, \bar{x}]}{[x_n, x_{n-1}]} \right) \\ &= (x_n - \bar{x})(x_{n-1} - \bar{x}) \frac{[x_{n-1}, x_n, \bar{x}]}{[x_n, x_{n-1}]} \\ &= (x_n - \bar{x})(x_{n-1} - \bar{x}) \frac{0.5f''(\xi_2)}{f'(\xi_1)} \end{aligned}$$

mit $\xi_1 \in I(x_{n-1}, x_n)$ und $\xi_2 \in I(x_{n-1}, x_n, \bar{x})$.

Wegen $f'(\bar{x}) \neq 0$ und der Stetigkeit von f' und f'' gibt es $\varepsilon > 0$ und $M > 0$, so dass

$$\left| \frac{1}{2} \frac{f''(\xi_2)}{f'(\xi_1)} \right| \leq M \quad \text{für alle } \xi_1, \xi_2 \text{ mit } |\bar{x} - \xi_j| \leq \varepsilon, \quad j = 1, 2.$$

Sei $e_n := M|\bar{x} - x_n|$ und $e_0, e_1 < \min\{1, \varepsilon M\}$. Dann folgt

$$e_{n+1} \leq e_n e_{n-1} \quad \text{für alle } n \in \mathbb{N}.$$

Aus dieser Ungleichung erhält man nun, dass das Sekantenverfahren konvergiert mit der R-Konvergenzordnung $p := \frac{1}{2}(1 + \sqrt{5}) = 1.618\dots$

Es sei nämlich $k := \max\{e_0, e_1^{1/p}\} < 1$. Dann gilt

$$e_n \leq k^{(p^n)}, \quad \text{für alle } n \geq 0,$$

denn für $n = 0$ und $n = 1$ ist diese Ungleichung nach Wahl von k richtig, und aus ihrer Gültigkeit für $0, 1, \dots, n$ folgt wegen $p^2 = p + 1$

$$e_{n+1} \leq e_n e_{n-1} \leq k^{(p^n)} k^{(p^{n-1})} = k^{(p^{n-1})(p+1)} = k^{(p^{n+1})}. \quad \blacksquare$$

Da das Sekantenverfahren nur eine Funktionsauswertung in jedem Schritt benötigt (aus dem vorhergehenden Schritt ist $f(x_{n-1})$ ja bekannt) und keine Auswertung der Ableitung, ist es effizienter als das Newton Verfahren. Es gilt nämlich

$$k^{(p^{n+2})} = \left(k^{(p^n)}\right)^{p^2} = \left(k^{(p^n)}\right)^{p+1},$$

und daher wird der Fehler e_{2n} der Folge x_{2n} (in der nur jeder zweite Sekantenschritt gezählt wird) durch eine Folge majorisiert, die von der Ordnung $p + 1 = 2.618\dots$ konvergiert. Bei gleichem Aufwand konvergiert das Sekantenverfahren also schneller als das Newton Verfahren.

8.2.2 Einschließende Verfahren

Ist $f : [a, b] \rightarrow \mathbb{R}$ stetig und gilt $f(a)f(b) \leq 0$, so besitzt f in $[a, b]$ eine Nullstelle \bar{x} . Bewiesen wird dieser Satz mit Hilfe der **Bisektion**, die zugleich ein numerisches Verfahren liefert:

Algorithmus 8.23. (Bisektion)

```

fa=f(a); fb=f(b);
repeat
  c=0.5(a+b); fc=f(c);
  if fa*fc < 0
    b=c; fb=fc;
  else
    a=c; fa=fc; end
until abs(a-b) < eps

```

Dieses Verfahren konvergiert R-linear, denn die Länge des Intervalls, das die Nullstelle von f enthält wird in jedem Schritt halbiert. Um eine Dezimalstelle zu gewinnen, sind also (etwas mehr als) drei Funktionsauswertungen erforderlich.

Es konvergiert aber i.A. nicht Q-linear, wie das folgende Beispiel zeigt. Damit ist der Begriff der Q-Konvergenzordnung echt stärker als der der R-Konvergenzordnung.

Beispiel 8.24. Die stetige Funktion f besitze in

$$\bar{x} = \frac{1}{7} = \sum_{\nu=1}^{\infty} \left(\frac{1}{8}\right)^{\nu}$$

eine Nullstelle, und es gelte $f(x) \neq 0$ für $x \neq \frac{1}{7}$.

Führt man für diese Funktion das Bisektionsverfahren mit dem Startwert $a_0 = 0$ und $b_0 = 1$ durch, so erhält man die Iterierten

$$\begin{aligned}
 x_1 &= \frac{1}{2}, \quad x_2 = \frac{1}{2} - \frac{1}{4}, \quad x_3 = \frac{1}{2} - \frac{1}{4} - \frac{1}{8} \\
 x_4 &= \left(\frac{1}{2} - \frac{1}{4} - \frac{1}{8}\right) + \frac{1}{16}, \quad x_5 = \left(\frac{1}{2} - \frac{1}{4} - \frac{1}{8}\right) + \frac{1}{16} - \frac{1}{32} \\
 x_6 &= \left(\frac{1}{2} - \frac{1}{4} - \frac{1}{8}\right) + \frac{1}{16} - \frac{1}{32} - \frac{1}{64} \\
 x_7 &= \left(\frac{1}{2} - \frac{1}{4} - \frac{1}{8}\right) + \left(\frac{1}{16} - \frac{1}{32} - \frac{1}{64}\right) + \frac{1}{128}
 \end{aligned}$$

d.h.

$$x_{3n} = \sum_{\nu=1}^n (4-2-1) \left(\frac{1}{8}\right)^\nu, \quad x_{3n+1} = x_{3n} + 4 \left(\frac{1}{8}\right)^{n+1}, \quad x_{3n+2} = x_{3n} + (4-2) \left(\frac{1}{8}\right)^{n+1}.$$

Damit gilt für den Fehler

$$\begin{aligned} |\bar{x} - x_{3n}| &= \left| \sum_{\nu=n+1}^{\infty} \left(\frac{1}{8}\right)^\nu \right| = \frac{1}{7} \cdot \left(\frac{1}{8}\right)^n \\ |\bar{x} - x_{3n+1}| &= \left| \bar{x} - x_{3n} - 4 \left(\frac{1}{8}\right)^{n+1} \right| = \frac{5}{14} \cdot \left(\frac{1}{8}\right)^n \\ |\bar{x} - x_{3n+2}| &= \left| \bar{x} - x_{3n} - 2 \left(\frac{1}{8}\right)^{n+1} \right| = \frac{3}{28} \cdot \left(\frac{1}{8}\right)^n. \end{aligned}$$

Der Fehler fällt also nicht monoton, sondern wird im Schritt von x_{3n} zu x_{3n+1} für alle $n \in \mathbb{N}$ sogar wieder vergrößert, und damit liegt keine Q-lineare Konvergenz vor.

□

Da die Funktionswerte $f(a)$ und $f(b)$ bekannt sind, kann man an Stelle des Intervallmittelpunktes als neuen Punkt (wie beim Sekantenverfahren) die Nullstelle der Sekante durch die Punkte $(a, f(a))$ und $(b, f(b))$ wählen. Man erhält die **regula falsi**:

Algorithmus 8.25. (regula falsi)

```

fa=f(a); fb=f(b);
repeat
  c=a-(b-a)*fa/(fb-fa); fc=f(c);
  if fa*fc < 0
    b=c; fb=fc;
  else
    a=c; fa=fc; end
until abs(a-b) < eps

```

Nachteil der regula falsi ist, dass in vielen Fällen einer der beiden Intervallendpunkte "hängen bleibt", nämlich dann, wenn ein Intervall erreicht ist, in dem f konvex oder konkav ist. Ist f (wie in Abbildung 8.5) konvex und monoton wachsend in $[a, b]$, so liegt die Nullstelle der Sekante links von der Nullstelle von f , und der rechte Intervallendpunkt bleibt unverändert. Dies gilt auch für die folgenden Schritte.

Das regula falsi Verfahren wird beschleunigt, indem man den Funktionswert auf der hängen bleibenden Seite modifiziert, wenn zwei aufeinanderfolgende Schritte auf

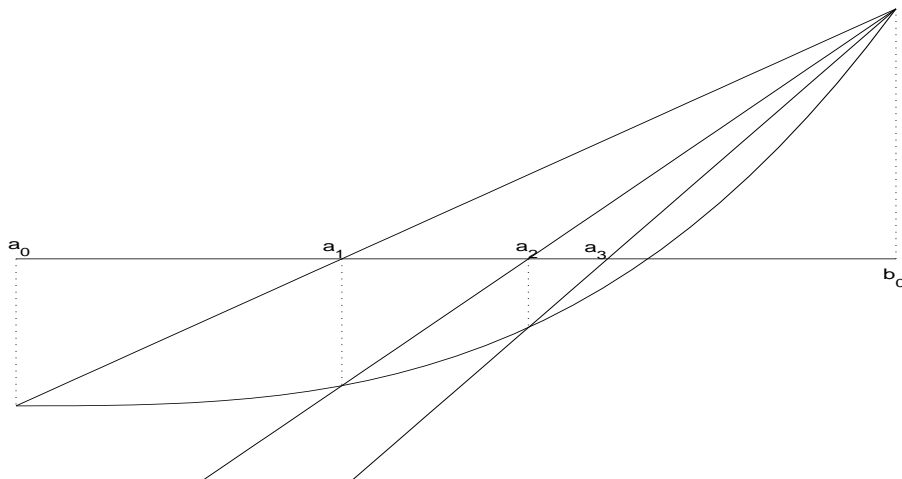


Abbildung 8.5 : regula falsi

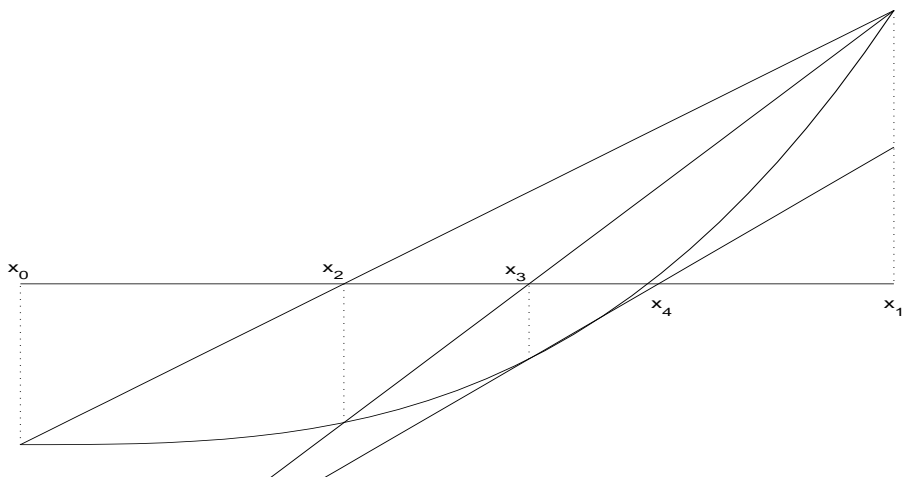


Abbildung 8.6 : Illinois Verfahren

derselben Seite der Nullstelle liegen (tatsächlich: wenn die Funktionswerte gleiches Vorzeichen haben).

Ein erstes Verfahren dieses Typs ist das **Illinois Verfahren**. Es wurde vermutlich ca. 1950 am Rechenzentrum der University of Illinois eingeführt. In Dowell und Jarrett [33] wurde gezeigt, dass die R-Konvergenzordnung $\sqrt[3]{3} \approx 1.442$ ist. Beim Illinois Verfahren wird der Funktionswert auf der hängenden Seite stets halbiert. Abbildung 8.6 zeigt die ersten Schritte des Illinois Verfahrens. Nach der Bestimmung von x_3 wird dem rechts liegenden Punkt x_1 als neuer Funktionswert $0.5f(x_1)$ für den Sekantenschritt zugeordnet. Dadurch wird x_4 größer als die Nullstelle von f .

Algorithmus 8.26. (Illinois Verfahren)

```

f1=f(x1); f2=f(x2);
repeat
  x3=x2-(x2-x1)*f2/(f2-f1); f3=f(x3);
  if f2*f3 < 0
    x1=x2; f1=f2; x2=x3; f2=f3;
  else
    x2=x3; f2=f3; f1=0.5*f1; end
until abs(x1-x2) < eps

```

Etwas bessere Konvergenzeigenschaften hat das **Pegasus Verfahren**, dessen Ursprung ebenfalls im Dunkeln liegt. Bei ihm wird im Falle $f_2 * f_3 > 0$ (mit den Bezeichnungen aus Algorithmus 8.26.) der Funktionswert f_1 multipliziert mit $f_2 / (f_2 + f_3) \in (0, 1)$. Dieses Verfahren hat (vgl. Dowell und Jarrett [34]) die R-Konvergenzordnung $\sqrt[4]{7.275} \approx 1.642$, ist also sogar noch etwas schneller als das Sekantenverfahren und liefert zusätzlich eine Fehlerabschätzung in Form einer Einschließung.

Algorithmus 8.27. (Pegasus Verfahren)

```

f1=f(x1); f2=f(x2);
repeat
  x3=x2-(x2-x1)*f2/(f2-f1); f3=f(x3);
  if f2*f3 < 0
    x1=x2; f1=f2; x2=x3; f2=f3;
  else
    f1=f1*f2/(f2+f3); x2=x3; f2=f3; end
until abs(x1-x2) < eps

```

Eine weitere Verbesserung wurde von Anderson und Björck [5] eingeführt. Im Falle $f_2 * f_3 > 0$ wird die folgende Modifikation vorgenommen: Es wird eine Parabel durch (x_1, f_1) , (x_2, f_2) und (x_3, f_3) gelegt und die Tangente im mittleren Punkt (x_3, f_3) an die Parabel konstruiert. Schneidet diese Tangente die x -Achse zwischen den Punkten x_1 und x_3 , so wird dieser Punkt als die nächste Näherung x_4 für die Nullstelle gewählt. Ist dies nicht der Fall, so wird ein Illinois Schritt ausgeführt. Die Konstruktion des Punktes x_4 ist in Abbildung 8.7 demonstriert. q bezeichnet den Graphen der interpolierenden quadratischen Funktion.

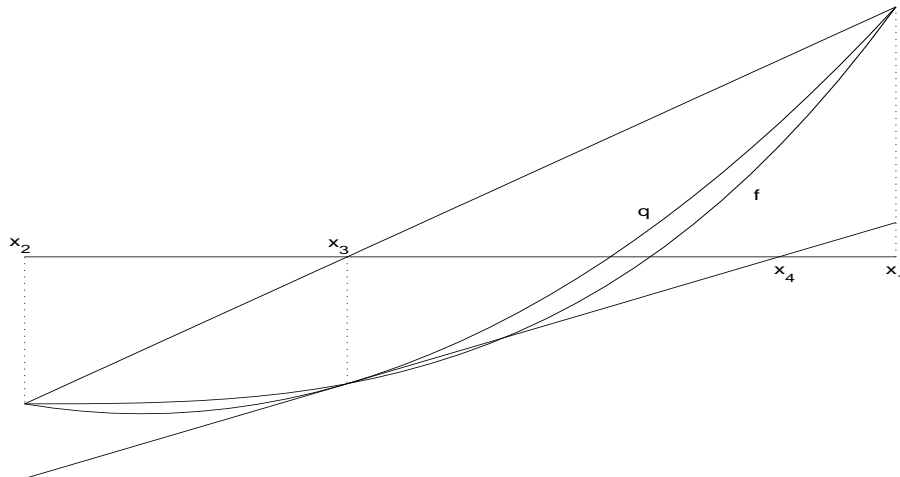


Abbildung 8.7 : Anderson Björck Verfahren

Die Lagrangesche Interpolationsformel liefert für die interpolierende quadratische Funktion

$$q(x) = f_1 \frac{(x - x_2)(x - x_3)}{(x_1 - x_2)(x_1 - x_3)} + f_2 \frac{(x - x_1)(x - x_3)}{(x_2 - x_1)(x_2 - x_3)} + f_3 \frac{(x - x_1)(x - x_2)}{(x_3 - x_1)(x_3 - x_2)},$$

mit der Ableitung in x_3

$$m = f_1 \frac{x_3 - x_2}{(x_1 - x_2)(x_1 - x_3)} + f_2 \frac{x_3 - x_1}{(x_2 - x_1)(x_2 - x_3)} + f_3 \frac{2x_3 - x_1 - x_2}{(x_3 - x_1)(x_3 - x_2)},$$

und unter Berücksichtigung von

$$x_3 = x_2 - \frac{x_1 - x_2}{f_1 - f_2} f_2 = x_1 - \frac{x_1 - x_2}{f_1 - f_2} f_1$$

rechnet man leicht nach, dass gilt

$$f_1^* := f_3 + m(x_1 - x_3) = f_1 \left(1 - \frac{f_3}{f_2} \right).$$

Man kann x_4 also durch einen Sekantenschritt mit den Punkten (x_3, f_3) und (x_1, f_1^*) berechnen.

Ist $|f_2| < |f_3|$, so haben f_1 und f_1^* entgegengesetztes Vorzeichen, und x_4 liegt nicht zwischen x_1 und x_3 . In diesem Fall wird ein Illinois Schritt ausgeführt. Man erhält das folgende Verfahren:

Algorithmus 8.28. (Anderson Björck Verfahren)

```
f1=f(x1); f2=f(x2);
repeat
```

```

x3=x2-(x2-x1)*f2/(f2-f1); f3=f(x3);
if f2*f3 < 0
  x1=x2; f1=f2; x2=x3; f2=f3;
else
  if |f2| < |f3|
    g=0.5;
  else
    g=1-f3/f2; end
  f1=g*f1;
  x2=x3; f2=f3; end
until abs(x1-x2) < eps

```

Anders als beim Illinois und beim Pegasus Verfahren ist nicht klar, in welcher Folge asymptotisch Sekantenschritte und modifizierte Sekantenschritte im Anderson Björck Verfahren auftreten. Man erhält daher nicht eine feste Konvergenzordnung, sondern je nach der Folge der Schritte ergibt sich eine R-Konvergenzordnung von 1.682 oder 1.710 (vgl. Anderson und Björck [5]).

Eine weitere Modifikation wurde von King [63] für das Pegasus Verfahren vorgeschlagen. Es werden niemals nacheinander zwei Sekantenschritte zugelassen, sondern auf jeden Sekantenschritt muss ein modifizierter Schritt folgen.

Man geht aus von zwei Punkten x_0 und x_1 mit $f_0 f_1 < 0$, wobei $f_j := f(x_j)$ gesetzt ist.

Algorithmus 8.29. (King Verfahren)

```

repeat
  x2=x1-(x0-x1)*f1/(f0-f1); f2=f(x2);
  if f1*f2 < 0
    vertausche (x0,f0) mit (x1,f1); end
repeat
  f0=f0*f1/(f1+f2); x1=x2; f1=f2;
  x2=x1-(x0-x1)*f1/(f0-f1); f2=f(x2);
  until f1*f2 <= 0
x0=x1; f0=f1; x1=x2; f1=f2;
until abs(x0-x1) < eps

```

Tabelle 8.6: Einschließende Verfahren

| m | reg. falsi | Illinois | Pegasus | And. Björck | King | ABK |
|-----|------------|------------|------------|-------------|------------|------------|
| 1 | $1.17e-01$ | $1.17e-01$ | $1.17e-01$ | $1.17e-01$ | $1.17e-01$ | $1.17e-01$ |
| 2 | $5.02e-02$ | $5.02e-02$ | $5.02e-02$ | $5.02e-02$ | $2.02e-02$ | $8.48e-03$ |
| 3 | $2.10e-02$ | $6.00e-03$ | $8.56e-03$ | $1.49e-03$ | $6.61e-04$ | $8.35e-04$ |
| 4 | $8.76e-03$ | $2.45e-04$ | $2.34e-05$ | $6.11e-05$ | $1.07e-05$ | $3.63e-07$ |
| 5 | $3.62e-03$ | $1.16e-06$ | $1.59e-07$ | $7.24e-08$ | $5.73e-11$ | $2.41e-10$ |
| 6 | $1.50e-03$ | $1.15e-06$ | $2.95e-12$ | $2.66e-15$ | 0 | 0 |
| 7 | $6.18e-04$ | $1.06e-12$ | 0 | | | |
| 8 | $2.55e-04$ | 0 | | | | |

Es wurde von King gezeigt, dass für dieses **King Verfahren** die R-Konvergenzgeschwindigkeit auf 1.710 bzw. 1.732 gehoben wird. Dieselbe Konvergenzgeschwindigkeit erreicht man, wenn man die Modifikation von King in das Verfahren von Anderson und Björck einbaut. Das entstehende Verfahren heißt **Anderson Björck King Verfahren**.

Beispiel 8.30. Wir wenden die einschließenden Verfahren auf das Problem an, die dritte Wurzel von 2 zu bestimmen. Als Startwerte verwenden wir $x_0 = 1$ und $x_1 = 2$. Tabelle 8.6 enthält die Fehler für die verschiedenen Verfahren. \square

Im letzten Beispiel erreicht man die volle Genauigkeit von 16 Stellen mit dem Newton Verfahren mit dem Startwert $x_0 = 1$ nach 5 Schritten. Man beachte aber, dass jeder Newton Schritt zwei Funktionsauswertungen benötigt, die 5 Newton Schritte also vergleichbar mit 10 Pegasus Schritten sind. Um die Wirksamkeit von Verfahren zu vergleichen wurde von Traub [111] der Begriff der **Effizienz** eingeführt. Erfordert ein Verfahren in jedem Schritt H Funktionsauswertungen (diese Zahl heißt auch die **Hornerzahl**) und ist seine Konvergenzordnung p , so ist die Effizienz definiert durch

$$E = p^{1/H}.$$

Für die einschließenden Verfahren stimmt also die Effizienz mit der angegebenen Konvergenzordnung überein, für das Newton Verfahren ist sie jedoch nur $\sqrt{2} = 1.414$. Die einschließenden Verfahren sind also vorzuziehen.

8.3 Newton Verfahren für Systeme

Wir betrachten das Fixpunktproblem

$$\phi(\mathbf{x}) = \mathbf{x} \tag{8.17}$$

mit $\phi : \mathbb{R}^n \supset D \rightarrow \mathbb{R}^n$.

Ähnlich wie im reellen Fall gilt die folgende lokale Konvergenzaussage für die Fixpunktiteration

$$\mathbf{x}^{m+1} := \phi(\mathbf{x}^m). \quad (8.18)$$

Satz 8.31. (Satz von Ostrowski)

$\phi : \mathbb{R}^n \supset D \rightarrow \mathbb{R}^n$ habe einen Fixpunkt $\bar{\mathbf{x}} \in \overset{\circ}{D}$ (dem Inneren von D) und sei differenzierbar in $\bar{\mathbf{x}}$. Gilt dann für den Spektralradius $\rho(\phi'(\bar{\mathbf{x}})) = \sigma < 1$, so existiert eine Umgebung $U(\bar{\mathbf{x}}) \subset D$, so dass die Iteration (8.18) für jeden Startwert $\mathbf{x}^0 \in U(\bar{\mathbf{x}})$ (Q -linear) gegen $\bar{\mathbf{x}}$ konvergiert.

Zum Beweis benötigen wir zunächst das folgende Lemma:

Lemma 8.32. Es sei $\mathbf{A} \in \mathbb{R}^{(n,n)}$ mit dem Spektralradius $\rho(\mathbf{A})$. Dann gibt es zu jedem $\varepsilon > 0$ eine Vektornorm $\|\cdot\|$, so dass für die zugehörige Matrixnorm gilt:

$$\|\mathbf{A}\| \leq \rho(\mathbf{A}) + \varepsilon.$$

Beweis: Ist $\|\cdot\|$ eine beliebige Vektornorm auf \mathbb{R}^n und $\mathbf{T} \in \mathbb{R}^{(n,n)}$ eine reguläre Matrix, so ist offenbar auch $\|\mathbf{x}\|' := \|\mathbf{T}\mathbf{x}\|$ eine Vektornorm auf \mathbb{R}^n , und die zugehörige Matrixnorm ist

$$\|\mathbf{B}\|' = \max_{\mathbf{x} \neq \mathbf{0}} \frac{\|\mathbf{B}\mathbf{x}\|'}{\|\mathbf{x}\|'} = \max_{\mathbf{x} \neq \mathbf{0}} \frac{\|\mathbf{T}\mathbf{B}\mathbf{x}\|}{\|\mathbf{T}\mathbf{x}\|} = \max_{\mathbf{T}\mathbf{x} \neq \mathbf{0}} \frac{\|(\mathbf{T}\mathbf{B}\mathbf{T}^{-1})\mathbf{T}\mathbf{x}\|}{\|\mathbf{T}\mathbf{x}\|} = \|\mathbf{T}\mathbf{B}\mathbf{T}^{-1}\|.$$

Ist $\mathbf{J} = \mathbf{X}^{-1}\mathbf{A}\mathbf{X}$ die Jordansche Normalform von \mathbf{A} und $\mathbf{D} := \text{diag}\{1, \varepsilon, \varepsilon^2, \dots, \varepsilon^{n-1}\}$, so stimmen die Diagonalen von \mathbf{J} und $\tilde{\mathbf{J}} := \mathbf{D}^{-1}\mathbf{J}\mathbf{D}$ überein und die Einsen in der Nebendiagonale von \mathbf{J} gehen in $\tilde{\mathbf{J}}$ über in ε . Damit gilt

$$\|\mathbf{D}^{-1}\mathbf{X}^{-1}\mathbf{A}\mathbf{X}\mathbf{D}\|_{\infty} = \|\tilde{\mathbf{J}}\|_{\infty} = \rho(\mathbf{A}) + \varepsilon,$$

und man kann als Vektornorm

$$\|\mathbf{x}\| := \|(\mathbf{X}\mathbf{D})^{-1}\mathbf{x}\|_{\infty}$$

wählen. ■

Beweis: (von Satz 8.31.)

Sei $\varepsilon > 0$ vorgegeben, so dass $q := \sigma + 2\varepsilon < 1$. Dann gibt es eine Vektornorm $\|\cdot\|$ im \mathbb{R}^n , so dass für die induzierte Matrixnorm gilt $\|\phi'(\bar{\mathbf{x}})\| \leq \sigma + \varepsilon$.

Ferner existiert zu ε ein $\delta > 0$ mit $U(\bar{\mathbf{x}}) := \{\mathbf{x} : \|\mathbf{x} - \bar{\mathbf{x}}\| \leq \delta\} \subset D$ und

$$\|\phi(\mathbf{x}) - \phi(\bar{\mathbf{x}}) - \phi'(\bar{\mathbf{x}})(\mathbf{x} - \bar{\mathbf{x}})\| \leq \varepsilon \|\mathbf{x} - \bar{\mathbf{x}}\| \quad \text{für alle } \mathbf{x} \in U(\bar{\mathbf{x}}).$$

Für $\mathbf{x} \in U(\bar{\mathbf{x}})$ gilt

$$\begin{aligned} \|\phi(\mathbf{x}) - \bar{\mathbf{x}}\| &= \|\phi(\mathbf{x}) - \phi(\bar{\mathbf{x}})\| \\ &\leq \|\phi(\mathbf{x}) - \phi(\bar{\mathbf{x}}) - \phi'(\bar{\mathbf{x}})(\mathbf{x} - \bar{\mathbf{x}})\| + \|\phi'(\bar{\mathbf{x}})(\mathbf{x} - \bar{\mathbf{x}})\| \\ &\leq \varepsilon \|\mathbf{x} - \bar{\mathbf{x}}\| + (\sigma + \varepsilon) \|\mathbf{x} - \bar{\mathbf{x}}\| = q \|\mathbf{x} - \bar{\mathbf{x}}\|, \end{aligned}$$

d.h. $\phi(U(\bar{\mathbf{x}})) \subset U(\bar{\mathbf{x}})$. Also ist für $\mathbf{x}^0 \in U(\bar{\mathbf{x}})$ die Folge in (8.18) definiert und

$$\|\mathbf{x}^m - \bar{\mathbf{x}}\| \leq q \|\mathbf{x}^{m-1} - \bar{\mathbf{x}}\| \leq \dots \leq q^m \delta \rightarrow 0. \quad \blacksquare$$

Bemerkung 8.33. Ist ϕ stetig differenzierbar in $\bar{\mathbf{x}}$, so erhält man das Ergebnis schneller aus Korollar 8.8. Es gibt nämlich zu $\varepsilon > 0$ ein $\delta > 0$, so dass

$$\|\phi'(\mathbf{x})\| \leq \sigma + 2\varepsilon = \alpha \quad \text{für alle } \mathbf{x} \text{ mit } \|\mathbf{x} - \bar{\mathbf{x}}\| \leq \delta.$$

Damit ist nach Satz 8.4. ϕ kontrahierend auf $U(\bar{\mathbf{x}})$, und wegen

$$\|\bar{\mathbf{x}} - \phi(\bar{\mathbf{x}})\| = 0 \leq (1 - q)\delta$$

sind nach Korollar 8.8. die Voraussetzungen des Fixpunktsatzes für kontrahierende Abbildungen in $U(\bar{\mathbf{x}})$ erfüllt. \square

Ist $\phi'(\bar{\mathbf{x}}) = \mathbf{O}$, so erhält man wieder quadratische Konvergenz.

Satz 8.34. Die Voraussetzungen von Satz 8.31. seien erfüllt, es sei ϕ zweimal stetig differenzierbar in einer Umgebung von $\bar{\mathbf{x}}$, und es gelte $\phi'(\bar{\mathbf{x}}) = \mathbf{O}$. Dann konvergiert das Iterationsverfahren (8.18) lokal Q -quadratisch gegen $\bar{\mathbf{x}}$.

Beweis: Nach Satz 8.31. konvergiert die Folge (8.18) lokal gegen $\bar{\mathbf{x}}$. Durch Taylorentwicklung erhält man für $\mathbf{x} \in U(\bar{\mathbf{x}})$

$$\phi_i(\mathbf{x}) - \phi_i(\bar{\mathbf{x}}) = \frac{1}{2} \sum_{j,k=1}^n \frac{\partial^2 \phi_i}{\partial x_j \partial x_k}(\boldsymbol{\xi}^i)(x_j - \bar{x}_j)(x_k - \bar{x}_k),$$

und hieraus folgt mit

$$M := \frac{1}{2} \max_{i,j,k} \max_{\mathbf{x} \in U(\bar{\mathbf{x}})} \left| \frac{\partial^2 \phi_i}{\partial x_j \partial x_k}(\mathbf{x}) \right|$$

die quadratische Konvergenz:

$$\|\mathbf{x}^{m+1} - \bar{\mathbf{x}}\|_\infty \leq Mn^2 \|\mathbf{x}^m - \bar{\mathbf{x}}\|_\infty^2. \quad \blacksquare$$

Wir betrachten nun das Nullstellenproblem

$$f(\mathbf{x}) = \mathbf{0} \tag{8.19}$$

mit $f : \mathbb{R}^n \supset D \rightarrow \mathbb{R}^n$.

Für den Fall $n = 1$ ist das folgende Prinzip zur Gewinnung von Iterationsverfahren bekannt: Die nichtlineare Funktion f wird im m -ten Schritt durch eine einfachere Funktion $\tilde{f}_m : \mathbb{R}^n \supset D_m \rightarrow \mathbb{R}^n$ ersetzt, die f in einer Umgebung der letzten Iterierten \mathbf{x}^m approximiert und für die die Ersatzaufgabe $\tilde{f}_m(\mathbf{x}) = \mathbf{0}$ leicht lösbar ist. Die Lösung von $\tilde{f}_m(\mathbf{x}) = \mathbf{0}$ wird dann als neue Iterierte gewählt.

Wir betrachten nur affin lineare Approximationen

$$\tilde{f}_m(\mathbf{x}) = \mathbf{a}^m + \mathbf{A}_m(\mathbf{x} - \mathbf{x}^m), \quad \mathbf{x} \in \mathbb{R}^n, \tag{8.20}$$

mit $\mathbf{a}^m \in \mathbb{R}^n$, $\mathbf{A}_m \in \mathbb{R}^{(n,n)}$.

Ist f differenzierbar, so erhält man lokal (in einer Umgebung von \mathbf{x}^m) mit den Ansatz (8.20) die beste Approximation durch

$$\tilde{f}_m(\mathbf{x}) = f(\mathbf{x}^m) + f'(\mathbf{x}^m)(\mathbf{x} - \mathbf{x}^m) \tag{8.21}$$

nach Definition der Ableitung.

Ist $f'(\mathbf{x}^m)$ regulär, so ist die $(m+1)$ -te Iterierte die Lösung des linearen Gleichungssystems

$$f(\mathbf{x}^m) + f'(\mathbf{x}^m)(\mathbf{x} - \mathbf{x}^m) = \mathbf{0} \tag{8.22}$$

oder (obwohl man bei der numerischen Ausführung des Verfahrens niemals die Inverse berechnen wird)

$$\mathbf{x}^{m+1} = \mathbf{x}^m - f'(\mathbf{x}^m)^{-1} f(\mathbf{x}^m). \tag{8.23}$$

Das so definierte Verfahren heißt **Newton Verfahren**.

Setzt man genügend hohe Differenzierbarkeit für f voraus, so liefert Satz 8.34. die quadratische Konvergenz des Newton Verfahrens. Wir führen die genauere Analyse mit den minimalen Glattheitsvoraussetzungen aus.

Definition 8.35. *E sei $f : \mathbb{R}^n \supset D \rightarrow \mathbb{R}^n$ und $\bar{\mathbf{x}} \in D$ mit $f(\bar{\mathbf{x}}) = \mathbf{0}$. $\bar{\mathbf{x}}$ heißt reguläre Nullstelle von f , falls*

(i) $\bar{\mathbf{x}}$ ein innerer Punkt von D ist, d.h. es existiert $\delta^* > 0$, so dass $S^* := \{\mathbf{x} : \|\mathbf{x} - \bar{\mathbf{x}}\| \leq \delta^*\} \subset D$,

(ii) f differenzierbar in S^* ist mit Lipschitz stetiger Ableitung, d.h. es gibt ein $q > 0$ mit

$$\|f'(\mathbf{x}) - f'(\mathbf{y})\| \leq q\|\mathbf{x} - \mathbf{y}\| \quad \text{für alle } \mathbf{x}, \mathbf{y} \in S^*,$$

(iii) $f'(\bar{\mathbf{x}})$ eine reguläre Matrix ist.

Lemma 8.36. *Sei $f : \mathbb{R}^n \supset D \rightarrow \mathbb{R}^n$ in der konvexen Menge $B \subset D$ differenzierbar und sei f' Lipschitz stetig mit der Konstante q in B . Dann gilt für alle $\mathbf{x}, \mathbf{y}, \mathbf{z} \in B$*

$$\|f(\mathbf{y}) - f(\mathbf{x}) - f'(\mathbf{z})(\mathbf{y} - \mathbf{x})\| \leq \frac{q}{2}\|\mathbf{y} - \mathbf{x}\|\{\|\mathbf{y} - \mathbf{z}\| + \|\mathbf{x} - \mathbf{z}\|\} \quad (8.24)$$

Für $\mathbf{x} = \mathbf{z}$ ergibt sich insbesondere

$$\|f(\mathbf{y}) - f(\mathbf{x}) - f'(\mathbf{x})(\mathbf{y} - \mathbf{x})\| \leq \frac{q}{2}\|\mathbf{y} - \mathbf{x}\|^2 \quad \text{für alle } \mathbf{x}, \mathbf{y} \in B. \quad (8.25)$$

Beweis: Nach dem Mittelwertsatz der Differentialrechnung gilt

$$\begin{aligned} & \|f(\mathbf{y}) - f(\mathbf{x}) - f'(\mathbf{z})(\mathbf{y} - \mathbf{x})\| \\ &= \left\| \int_0^1 (f'(\mathbf{x} + t(\mathbf{y} - \mathbf{x})) - f'(\mathbf{z}))(\mathbf{y} - \mathbf{x}) dt \right\| \\ &\leq \int_0^1 \|f'(\mathbf{x} + t(\mathbf{y} - \mathbf{x})) - f'(\mathbf{z})\| dt \cdot \|\mathbf{y} - \mathbf{x}\| \\ &\leq q\|\mathbf{y} - \mathbf{x}\| \int_0^1 \|(1-t)(\mathbf{x} - \mathbf{z}) + t(\mathbf{y} - \mathbf{z})\| dt \\ &\leq \frac{1}{2}q\|\mathbf{y} - \mathbf{x}\|\{\|\mathbf{x} - \mathbf{z}\| + \|\mathbf{y} - \mathbf{z}\|\}. \quad \blacksquare \end{aligned}$$

Satz 8.37. *$f : \mathbb{R}^n \supset D \rightarrow \mathbb{R}^n$ besitze die reguläre Nullstelle $\bar{\mathbf{x}} \in D$. Dann existieren Zahlen $\delta > 0$ und $Q > 0$ mit $Q\delta < 1$, so dass das Newton Verfahren (8.23) für jeden Startwert $\mathbf{x}^0 \in S := \{\mathbf{x} : \|\mathbf{x} - \bar{\mathbf{x}}\| \leq \delta\}$ durchführbar ist, die Iterierten in S verbleiben und gegen $\bar{\mathbf{x}}$ konvergieren. Zudem gilt*

$$\|\mathbf{x}^{m+1} - \bar{\mathbf{x}}\| \leq Q\|\mathbf{x}^m - \bar{\mathbf{x}}\|^2 \quad \text{für alle } m \geq 0. \quad (8.26)$$

Beweis: Wir zeigen zunächst, dass für genügend kleines $\delta > 0$ und jedes $\mathbf{x}^m \in S$ die Inverse $f'(\mathbf{x}^m)^{-1}$ existiert, also \mathbf{x}^{m+1} definiert ist, und dass (8.26) gilt.

Sei dazu $\alpha := \|f'(\bar{\mathbf{x}})^{-1}\|$. Zu vorgegebenem $\kappa \in (0, 1)$ wählen wir $\delta > 0$ mit $\delta < \min\left(\delta^*, \frac{\kappa}{\alpha q}\right)$. Dann gilt

$$\|f'(\bar{\mathbf{x}})^{-1}\| \cdot \|f'(\mathbf{x}^m) - f'(\bar{\mathbf{x}})\| \leq \alpha q \|\bar{\mathbf{x}} - \mathbf{x}^m\| \leq \alpha q \delta \leq \kappa < 1$$

für alle $\mathbf{x}^m \in S$. Nach dem Störungslemma existiert $f'(\mathbf{x}^m)^{-1}$ und

$$\|f'(\mathbf{x}^m)^{-1}\| \leq \frac{\alpha}{1 - \kappa} =: M \quad \text{für alle } \mathbf{x}^m \in S.$$

Daher existiert \mathbf{x}^{m+1} , und aus der Identität

$$\mathbf{x}^{m+1} - \bar{\mathbf{x}} = -f'(\mathbf{x}^m)^{-1}(f(\mathbf{x}^m) - f(\bar{\mathbf{x}}) - f'(\mathbf{x}^m)(\mathbf{x}^m - \bar{\mathbf{x}}))$$

folgt mit (8.25)

$$\begin{aligned} \|\mathbf{x}^{m+1} - \bar{\mathbf{x}}\| &\leq \|f'(\mathbf{x}^m)^{-1}\| \cdot \|f(\mathbf{x}^m) - f(\bar{\mathbf{x}}) - f'(\mathbf{x}^m)(\mathbf{x}^m - \bar{\mathbf{x}})\| \\ &\leq M \cdot \frac{q}{2} \|\mathbf{x}^m - \bar{\mathbf{x}}\|^2 =: Q \|\mathbf{x}^m - \bar{\mathbf{x}}\|^2. \end{aligned}$$

Durch weiteres Abschätzen ergibt sich

$$\|\mathbf{x}^{m+1} - \bar{\mathbf{x}}\| \leq (Q \|\mathbf{x}^m - \bar{\mathbf{x}}\|) \|\mathbf{x}^m - \bar{\mathbf{x}}\| \leq Q \delta \|\mathbf{x}^m - \bar{\mathbf{x}}\|.$$

Unter etwaiger Verkleinerung von δ kann man $L := \delta Q < 1$ erreichen. Dann gilt

$$\|\mathbf{x}^{m+1} - \bar{\mathbf{x}}\| \leq L \|\mathbf{x}^m - \bar{\mathbf{x}}\| \leq \delta,$$

d.h. $\{\mathbf{x}^m\} \subset S$ ist wohldefiniert, genügt der Abschätzung (8.26) und ist wegen $\|\mathbf{x}^m - \bar{\mathbf{x}}\| \leq q^m \delta \rightarrow 0$ gegen $\bar{\mathbf{x}}$ konvergent. \blacksquare

Bemerkung 8.38. Unter den Voraussetzungen von Satz 8.37. ist $\bar{\mathbf{x}}$ die einzige Nullstelle von f in S , denn ist $\bar{\mathbf{y}} \in S$ eine weitere Nullstelle, so wähle man $\mathbf{x}^0 := \bar{\mathbf{y}}$. Dann gilt $\mathbf{x}^m = \bar{\mathbf{y}}$ für alle m und daher $\bar{\mathbf{x}} = \bar{\mathbf{y}}$. Diese Überlegung zeigt, dass reguläre Nullstellen im anschaulichen Sinne isoliert sind. \square

Bemerkung 8.39. Existiert $f''(\bar{\mathbf{x}})$ und ist $f''(\bar{\mathbf{x}})$ definit, d.h. $\mathbf{h}^T f''(\bar{\mathbf{x}}) \mathbf{h} \neq 0$ für alle $\mathbf{h} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$, so kann man sogar zeigen, dass das Newton Verfahren von genau der Q-Ordnung 2 konvergiert, d.h. zusätzlich zu (8.26) gilt mit einem $\tilde{Q} > 0$ (vgl. Schwetlick [94], p. 98)

$$\|\mathbf{x}^{m+1} - \bar{\mathbf{x}}\| \geq \tilde{Q} \|\mathbf{x}^m - \bar{\mathbf{x}}\|^2 \quad \text{für alle } m \geq 0. \quad \square$$

Satz 8.37. ist ein typischer lokaler Konvergenzsatz: Unter geeigneten Glattheits- und Regularitätsvoraussetzungen wird die Konvergenz eines Verfahrens für genügend gute Startwerte gesichert. Da in den Voraussetzungen die Lösung $\bar{\mathbf{x}}$ explizit vorkommt, lassen sie sich für eine konkrete Aufgabenstellung jedoch nicht überprüfen. Falls dies für Existenzaussagen oder Fehlerabschätzungen erforderlich ist, muss man auf semilokale Konvergenzsätze zurückgreifen, bei denen unter überprüfbaren Voraussetzungen sowohl die Existenz einer Lösung als auch die Konvergenz des Verfahrens gegen die Lösung bewiesen wird. Ein typischer Vertreter ist

Satz 8.40. (Kantorowitsch)

$f : \mathbb{R}^n \supset D \rightarrow \mathbb{R}^n$ sei in der konvexen Menge $B \subset D$ differenzierbar, und es gelte mit einem $q > 0$

$$\|f'(\mathbf{x}) - f'(\mathbf{y})\| \leq q\|\mathbf{x} - \mathbf{y}\| \quad \text{für alle } \mathbf{x}, \mathbf{y} \in B.$$

Für ein $\mathbf{x}^0 \in B$ existiere $f'(\mathbf{x}^0)^{-1}$, und es gelte mit $\alpha > 0, \eta \geq 0$

$$\|f'(\mathbf{x}^0)^{-1}\| \leq \alpha, \quad \|f'(\mathbf{x}^0)^{-1}f(\mathbf{x}^0)\| \leq \eta.$$

Gilt dann

$$h := \alpha q \eta \leq \frac{1}{2}$$

und ist die Kugelbedingung

$$S_1 := \{\mathbf{x} \in \mathbb{R}^n : \|\mathbf{x} - \mathbf{x}^1\| \leq \xi_1\} \subset B$$

mit

$$\mathbf{x}^1 := \mathbf{x}^0 - f'(\mathbf{x}^0)^{-1}f(\mathbf{x}^0), \quad \xi_1 := \xi_0 - \eta, \quad \xi_0 := \frac{1}{\alpha q}(1 - \sqrt{1 - 2h})$$

erfüllt, so gelten die folgenden Aussagen:

- (i) Das Newton Verfahren mit dem Startwert \mathbf{x}^0 ist unbeschränkt ausführbar, es gilt $\mathbf{x}^m \in S_1$ für alle $m \geq 1$, die Folge $\{\mathbf{x}^m\}$ konvergiert gegen eine Nullstelle $\bar{\mathbf{x}} \in S_1$ von f , und diese ist in

$$\left\{ \mathbf{x} \in \mathbb{R}^n : \|\mathbf{x} - \mathbf{x}^0\| \leq \frac{1}{\alpha q}(1 + \sqrt{1 - 2h}) \right\} \cap B$$

eindeutig.

- (ii) Es gilt die Fehlerabschätzung

$$\|\mathbf{x}^m - \bar{\mathbf{x}}\| \leq \frac{1}{\alpha q} \frac{\kappa^{2^m}}{2^m} \quad \text{für alle } m \geq 0$$

mit $\kappa := \alpha q \xi_0 = 1 - \sqrt{1 - 2h}$.

Beweis: Schwetlick [94], pp. 99 ff. ■

Der Einzugsbereich einer Nullstelle $\bar{\mathbf{x}}$ von f für das Newton Verfahren (d.h. die Menge aller Startwerte \mathbf{x}^0 , für die das Newton Verfahren gegen $\bar{\mathbf{x}}$ konvergiert) ist häufig sehr klein. Er kann manchmal durch Einführung einer Dämpfung vergrößert werden.

Es sei $\mathbf{h}^m := f'(\mathbf{x}^m)^{-1}f(\mathbf{x}^m)$ die Verbesserung nach dem Newton Verfahren ausgehend von \mathbf{x}^m . Wir setzen

$$\mathbf{x}^{m+1} := \mathbf{x}^m - \lambda_m \mathbf{h}^m, \quad (8.27)$$

wobei der Dämpfungsparameter $\lambda_m \in (0, 1]$ so gewählt wird, dass “die Größe des Funktionswerts f mit jedem Schritt verkleinert wird”.

Die Größe von $f(\mathbf{x})$ messen wir mit der Testfunktion

$$t(\mathbf{x}) := \|f(\mathbf{x})\|_2^2.$$

Wir wählen also $\lambda_m \in (0, 1]$ so, dass $t(\mathbf{x}^{m+1}) < t(\mathbf{x}^m)$ für alle $m \in \mathbb{N}_0$ gilt. Dass dies immer möglich ist, wenn die Nullstelle noch nicht erreicht ist, zeigt Satz 8.41.

Satz 8.41. *Es sei $f : \mathbb{R}^n \supset D \rightarrow \mathbb{R}^n$ stetig differenzierbar, $\tilde{\mathbf{x}} \in D$ mit $f(\tilde{\mathbf{x}}) \neq \mathbf{0}$, $f'(\tilde{\mathbf{x}})$ regulär und $\mathbf{h} := f'(\tilde{\mathbf{x}})^{-1}f(\tilde{\mathbf{x}})$ die Verbesserung nach dem Newton Verfahren.*

Dann existiert ein $\mu > 0$, so dass

$$t(\tilde{\mathbf{x}} - \lambda \mathbf{h}) < t(\tilde{\mathbf{x}}) \quad \text{für alle } \lambda \in (0, \mu].$$

Beweis: Es ist $t(\mathbf{x}) = f(\mathbf{x})^T f(\mathbf{x})$, und daher

$$\text{grad } t(\mathbf{x}) = 2f(\mathbf{x})^T f'(\mathbf{x}).$$

Es sei $\phi(\lambda) := t(\tilde{\mathbf{x}} - \lambda \mathbf{h})$. Dann ist ϕ in einer Umgebung von $\lambda = 0$ stetig differenzierbar mit

$$\phi(0) = t(\tilde{\mathbf{x}})$$

und

$$\phi'(\lambda) = -\text{grad } t(\tilde{\mathbf{x}} - \lambda \mathbf{h}) \mathbf{h} = -2f(\tilde{\mathbf{x}} - \lambda \mathbf{h})^T f'(\tilde{\mathbf{x}} - \lambda \mathbf{h}) \mathbf{h},$$

d.h.

$$\phi'(0) = -2f(\tilde{\mathbf{x}})^T f'(\tilde{\mathbf{x}}) f'(\tilde{\mathbf{x}})^{-1} f(\tilde{\mathbf{x}}) = -2\|f(\tilde{\mathbf{x}})\|_2^2 < 0,$$

d.h. ϕ ist in einer Umgebung von 0 streng monoton fallend. ■

Einen geeigneten Dämpfungsparameter λ_m kann man durch fortgesetztes Halbieren bestimmen. Man erhält dann das **gedämpfte Newton Verfahren**:

Bestimme $\mathbf{h}^m \in \mathbb{R}^n$ mit $f'(\mathbf{x}^m)\mathbf{h}^m = f(\mathbf{x}^m)$
 bestimme $\ell := \min\{k \in \mathbb{N}_0 : t(\mathbf{x}^m - 2^{-k}\mathbf{h}^m) < t(\mathbf{x}^m)\}$
 setze $\mathbf{x}^{m+1} := \mathbf{x}^m - 2^{-\ell}\mathbf{h}^m$

Beispiel 8.42.

$$f(\mathbf{x}) = \begin{pmatrix} (x_1^2 + x_2^2)(1 + 0.8x_1 + 0.6x_2) - 1 \\ (x_1^2 + x_2^2)(1 - 0.6x_1 + 0.8x_2) - 2x_1 \end{pmatrix} = \mathbf{0}.$$

Mit dem Startwert $\mathbf{x}^0 = (0.55, -1)^T$ erhält man mit dem Newton Verfahren die Näherungen in Tabelle 8.7. Man entfernt sich also sehr weit von der Nullstelle von f und wird zufällig im zwölften Schritt in den näheren Einzugsbereich der Nullstelle getragen.

Mit dem gedämpften Newton Verfahren erhält man die Werte aus Tabelle 8.8. Mit dem geänderten Startwert $\mathbf{x}^0 = (0.54, -1)^T$ für das gedämpften Newton Verfahren erhält man die Näherungen in Tabelle 8.9.

Das gedämpfte Newton Verfahren führt also nicht notwendig in eine Nullstelle von f , also in ein globales Minimum von t , sondern es kann auch (wie im obigen Fall) in einem lokalen Minimum stecken bleiben.

Das Newton Verfahren findet (nach einigem Herumirren) nach dreißig Iterationen die Nullstelle von f . □

Das Newton Verfahren ist sehr aufwändig. In jedem Schritt hat man die Jacobimatrix $f'(\mathbf{x}^m)$ und die n -Vektorfunktion $f(\mathbf{x}^m)$, also $n(n+1)$ reelle Funktionen, auszuwerten. Der Aufwand wird wesentlich kleiner, wenn man iteriert nach

$$\mathbf{x}^{m+1} = \mathbf{x}^m - f'(\mathbf{x}^0)^{-1}f(\mathbf{x}^m). \quad (8.28)$$

Dies ist das **vereinfachte Newton Verfahren**. Die Jacobimatrix wird also nur im ersten Schritt berechnet und in den folgenden Schritten unverändert übernommen. Man hat in (8.28) eine Folge von linearen Gleichungssystemen mit sich ändernden rechten Seiten, aber derselben Koeffizientenmatrix $f'(\mathbf{x}^0)$ zu lösen, was man sehr effizient mit der LR Zerlegung oder QR Zerlegung tun kann.

Ist f stetig differenzierbar in einer Umgebung einer Nullstelle $\bar{\mathbf{x}}$ von f und ist $f'(\bar{\mathbf{x}})$ regulär, so ist $f'(\mathbf{x}^0)$ auch für alle genügend nahe bei $\bar{\mathbf{x}}$ liegenden \mathbf{x}^0 regulär, und die Iterationsfunktion des vereinfachten Newton Verfahrens

$$\phi(\mathbf{x}) = \mathbf{x} - f'(\mathbf{x}^0)^{-1}f(\mathbf{x})$$

ist definiert. Wörtlich wie im eindimensionalen Fall kann man hiermit die lokale Konvergenz des vereinfachten Newton Verfahrens zeigen. Diese ist jedoch nur linear.

Tabelle 8.7: Newton Verfahren, ungedämpft

| m | x_1^m | x_2^m | $t(\mathbf{x}^m)$ |
|-----|-----------------------|-----------------------|-------------------|
| 0 | 0.5500000000000000 | -1.0000000000000000 | $1.62E + 0000$ |
| 1 | -14.43530223571625730 | -33.68447879289723070 | $2.24E + 0009$ |
| 2 | -9.55846899009341918 | -22.65588840979738190 | $1.97E + 0008$ |
| 3 | -6.30263512634718873 | -15.30801418517524050 | $1.73E + 0007$ |
| 4 | -4.12540490095915977 | -10.41578314254021010 | $1.52E + 0006$ |
| 5 | -2.66453732230278943 | -7.16283531774792886 | $1.33E + 0005$ |
| 6 | -1.67817129098230139 | -5.00476389670139821 | $1.16E + 0004$ |
| 7 | -1.00582442516326456 | -3.57701132447667355 | $1.02E + 0003$ |
| 8 | -0.54362763832473274 | -2.63207040180621126 | $9.00E + 0001$ |
| 9 | -0.22705037437102526 | -1.99656005864760327 | $8.40E + 0000$ |
| 10 | -0.01136811408793452 | -1.53983061247833520 | $9.68E - 0001$ |
| 11 | 0.16037483148821532 | -1.11595852255304619 | $2.68E - 0001$ |
| 12 | 1.29720182573823523 | 2.35416946826005801 | $7.32E + 0002$ |
| 13 | 0.83216662389917077 | 1.52437646280603078 | $5.84E + 0001$ |
| 14 | 0.53744240258463048 | 1.02770844932461125 | $3.96E + 0000$ |
| 15 | 0.40668900211956326 | 0.76355080012724792 | $1.56E - 0001$ |
| 16 | 0.38264871526056001 | 0.67356137691691342 | $1.11E - 0003$ |
| 17 | 0.38202607642813437 | 0.66409468893962469 | $1.00E - 0007$ |
| 18 | 0.38203126706979271 | 0.66400128301154753 | $8.77E - 0016$ |
| 19 | 0.38203126811166926 | 0.66400127421998055 | $6.75E - 0032$ |
| 20 | 0.38203126811166927 | 0.66400127421998048 | $5.88E - 0039$ |

Tabelle 8.8: Newton Verfahren, gedämpft, 1

| m | x_1^m | x_2^m | $t(\mathbf{x}^m)$ |
|-----|---------------------|----------------------|-------------------|
| 0 | 0.5500000000000000 | -1.0000000000000000 | $1.62E + 0000$ |
| 1 | 0.53536591578543334 | -1.03191843632118870 | $1.62E + 0000$ |
| 2 | 0.56110003914930182 | -0.97315152261970942 | $1.62E + 0000$ |
| 3 | 0.52522028267090552 | -1.04907142845895020 | $1.60E + 0000$ |
| 4 | 0.58022724802414461 | -0.91988676253078696 | $1.59E + 0000$ |
| 5 | 0.44186642606308465 | -1.20117318695284988 | $1.57E + 0000$ |
| 6 | 0.78715923029833133 | 0.06828325649427959 | $1.47E + 0000$ |
| 7 | 0.64691727552902532 | 0.76081952209075180 | $9.44E - 0001$ |
| 8 | 0.42036801216744384 | 0.70268466773197415 | $3.33E - 0002$ |
| 9 | 0.38320847227569939 | 0.66655934000194405 | $9.38E - 0005$ |
| 10 | 0.38203275158923288 | 0.66400946937820756 | $8.35E - 0010$ |
| 11 | 0.38203126811259582 | 0.66400127429259763 | $6.18E - 0020$ |
| 12 | 0.38203126811166927 | 0.66400127421998048 | $1.47E - 0038$ |

8.4 Bairstow Verfahren für komplexe Nullstellen von Polynomen

Das Newton Verfahren für Systeme kann verwendet werden, um komplexe Nullstellen von Polynomen mit reellen Koeffizienten zu bestimmen. Wir betrachten

$$p_n(x) = \sum_{j=0}^n a_j x^{n-j} \quad (8.29)$$

Ist $p_n(z) = 0$ für ein $z \in \mathbb{C} \setminus \mathbb{R}$, so gilt auch $p_n(\bar{z}) = 0$ und daher

$$p_n(x) = (x - z)(x - \bar{z})q_{n-2}(x),$$

Tabelle 8.9: Newton Verfahren, gedämpft, 2

| m | x_1^m | x_2^m | $t(\mathbf{x}^m)$ |
|-----|---------------------|----------------------|-------------------|
| 0 | 0.5400000000000000 | -1.0000000000000000 | 1.544E + 0000 |
| 1 | 0.52226320324077548 | -1.03837727037179463 | 1.541E + 0000 |
| 2 | 0.54674729383789015 | -0.98233280952907319 | 1.536E + 0000 |
| 3 | 0.51049823387024609 | -1.05912654060680413 | 1.526E + 0000 |
| 4 | 0.55730345511559661 | -0.94800525500289117 | 1.509E + 0000 |
| 5 | 0.47836282966206385 | -1.10979544203828296 | 1.471E + 0000 |
| 6 | 0.61791730287807408 | -0.73956641616568136 | 1.443E + 0000 |
| 7 | 0.34190731543662328 | -1.31095184668199709 | 1.333E + 0000 |
| 8 | 0.47291974083537622 | -0.57110672040690209 | 8.306E - 0001 |
| 9 | 0.23567800090906323 | -1.21822240324259060 | 5.097E - 0001 |
| 10 | 0.32376926330177793 | -0.93765575143268807 | 4.505E - 0001 |
| 11 | 0.24244040787039246 | -1.13335030418307687 | 4.038E - 0001 |
| 12 | 0.27401919832012331 | -1.04385654817449567 | 3.955E - 0001 |
| 13 | 0.25802477791587531 | -1.08563557440662883 | 3.925E - 0001 |
| 14 | 0.26512517724780759 | -1.06641164648454920 | 3.925E - 0001 |
| 15 | 0.25920807677376168 | -1.08216418620516801 | 3.922E - 0001 |
| 16 | 0.26221776870739162 | -1.07404006558494690 | 3.921E - 0001 |
| 17 | 0.25966889237959624 | -1.08087102592776657 | 3.921E - 0001 |

d.h. im Falle komplexer Nullstellen kann man einen quadratischen Faktor abspalten

$$(x - z)(x - \bar{z}) = x^2 - 2sx + s^2 + t^2, \quad z := s + it.$$

Die Idee des Bairstow Verfahrens ist folgende:

1. $x^2 - ux - v$ sei eine Näherung für einen quadratischen Faktor von p_n . Dividiere dann p_n durch diese Näherung, d.h. bestimme Konstanten b_0, b_1, \dots, b_n , so dass

$$p_n(x) = (x^2 - ux - v)q_{n-2}(x) + b_{n-1}(x - u) + b_n, \quad (8.30)$$

$$q_{n-2}(x) = b_0x^{n-2} + b_1x^{n-3} + \dots + b_{n-2}. \quad (8.31)$$

2. Fasse b_{n-1} und b_n als Funktionen von u und v auf:

$$b_{n-1} = b_{n-1}(u, v), \quad b_n = b_n(u, v)$$

($x^2 - ux - v$ ist genau dann ein quadratischer Faktor von p_n , wenn $b_{n-1}(u, v) = b_n(u, v) = 0$ gilt), und wende auf das Gleichungssystem

$$\begin{aligned} b_{n-1}(u, v) &= 0 \\ b_n(u, v) &= 0 \end{aligned} \quad (8.32)$$

das Newton Verfahren an.

Zunächst benötigen wir einen Algorithmus, der 1. löst:

Satz 8.43. Seien $a_0, \dots, a_n, u, v \in \mathbb{R}$ gegeben. Bestimme b_0, \dots, b_n aus der Rekursionsformel

$$\begin{aligned} b_{-2} &= b_{-1} = 0 \\ b_i &= a_i + ub_{i-1} + vb_{i-2}, \quad i = 0, \dots, n. \end{aligned} \quad (8.33)$$

Dann gilt mit (8.31) die Identität (8.30).

Beweis: Durch Nachrechnen. ■

Um auf (8.32) das Newton Verfahren anwenden zu können, benötigen wir die Elemente der Jacobi-Matrix

$$\begin{pmatrix} \frac{\partial b_{n-1}}{\partial u}(u, v) & \frac{\partial b_{n-1}}{\partial v}(u, v) \\ \frac{\partial b_n}{\partial u}(u, v) & \frac{\partial b_n}{\partial v}(u, v) \end{pmatrix}$$

Sei $c_i := \frac{\partial b_{i+1}}{\partial u}$. Dann erhält man aus (8.33)

$$\begin{aligned} (b_1 &= a_1 + b_0u + 0v) &\Rightarrow c_0 &= b_0 \\ (b_2 &= a_2 + b_1u + b_0v) &\Rightarrow c_1 &= b_1 + c_0u \\ (b_{i+1} &= a_{i+1} + b_iu + b_{i-1}v) &\Rightarrow c_i &= b_i + c_{i-1}u + c_{i-2}v; \end{aligned}$$

die letzte Formel gilt also für alle i , wenn wir setzen:

$$c_{-1} = c_{-2} = 0.$$

Sei $d_i := \frac{\partial b_{i+2}}{\partial v}$. Dann erhält man genauso

$$d_i = b_i + ud_{i-1} + vd_{i-2}, \quad i = 0, 1, \dots, n-2, \quad d_{-1} = d_{-2} = 0.$$

Offensichtlich stimmen die Rekursionsformeln für die c_i und d_i und die Anfangsbedingungen überein. Daher gilt

$$d_i = c_i, \quad i = 0, \dots, n-2.$$

Sind $(u + \delta, v + \varepsilon)^T$ die durch das Newton Verfahren verbesserten Näherungen, so erfüllt $(\delta, \varepsilon)^T$ das folgende Gleichungssystem:

$$\begin{pmatrix} c_{n-2} & c_{n-3} \\ c_{n-1} & c_{n-2} \end{pmatrix} \begin{pmatrix} \delta \\ \varepsilon \end{pmatrix} = - \begin{pmatrix} b_{n-1} \\ b_n \end{pmatrix}$$

mit der Lösung

$$\delta = \frac{b_n c_{n-3} - b_{n-1} c_{n-2}}{c_{n-2}^2 - c_{n-1} c_{n-3}}, \quad \varepsilon = \frac{b_{n-1} c_{n-1} - b_n c_{n-2}}{c_{n-2}^2 - c_{n-1} c_{n-3}} \quad (8.34)$$

Insgesamt besteht das Bairstow Verfahren aus dem folgenden Algorithmus: Gegeben sei das Polynom

$$p_n(x) = \sum_{j=0}^n a_j x^{n-j}, \quad a_j \in \mathbb{R},$$

und eine Näherung $x^2 - u_0x - v_0$ für einen quadratischen Faktor.

Bestimme eine Folge $\{x^2 - u_mx - v_m\}$ nach der Vorschrift:

Für $m = 0, 1, 2, \dots$ bestimme $b_i^{(m)}$ und $c_i^{(m)}$ gemäß

$$\begin{aligned} b_{-2}^{(m)} &= b_{-1}^{(m)} = 0, & b_i^{(m)} &= a_i + u_m b_{i-1}^{(m)} + v_m b_{i-2}^{(m)}, \quad i = 0, \dots, n \\ c_{-2}^{(m)} &= c_{-1}^{(m)} = 0, & c_i^{(m)} &= b_i^{(m)} + u_m c_{i-1}^{(m)} + v_m c_{i-2}^{(m)}, \quad i = 0, \dots, n-1. \end{aligned}$$

Setze dann $u_{m+1} = u_m + \delta$, $v_{m+1} = v_m + \varepsilon$, wobei δ, ε aus (8.34) berechnet werden.

Nach Beendigung der Iteration berechne man aus dem quadratischen Faktor $x^2 - ux - v$ die zugehörigen Nullstellen

$$\frac{u}{2} \pm i \sqrt{-\frac{u^2}{4} - v}.$$

Man kann zeigen, dass die Funktionalmatrix an der Stelle eines quadratischen Faktors, der zu einer einfachen Nullstelle $s \pm it$ von p_n gehört, nichtsingulär ist. Also konvergiert das Bairstow Verfahren lokal quadratisch nach Satz 8.37.

8.5 Newton ähnliche Verfahren

Wir fragen nun, wie man superlinear konvergente Verfahren entwickeln kann, die keine partiellen Ableitungen und möglichst wenig Funktionsauswertungen benötigen.

Wir nehmen an, dass durch ein Iterationsverfahren I für alle genügend guten Startwerte \mathbf{x}^0 eine gegen $\bar{\mathbf{x}}$ (eine reguläre Nullstelle von f) konvergente Folge $\{\mathbf{x}^m\}$ erzeugt wird, die wir in der Form

$$\mathbf{x}^{m+1} = \mathbf{x}^m - \mathbf{q}^m, \quad m \geq 0 \tag{8.35}$$

mit den Korrekturen $-\mathbf{q}^m$ schreiben.

Man kann erwarten, dass die Folge sich etwa wie die Newton Folge verhält, wenn die \mathbf{q}^m genügend gute Approximationen für die Verbesserungen

$$\mathbf{p}^m := f'(\mathbf{x}^m)^{-1} f(\mathbf{x}^m) \tag{8.36}$$

nach dem Newton Verfahren (ausgehend von \mathbf{x}^m) sind. Da im Konvergenzfall \mathbf{q}^m und \mathbf{p}^m gegen $\mathbf{0}$ konvergieren, ist die Forderung $\|\mathbf{p}^m - \mathbf{q}^m\| \rightarrow \mathbf{0}$ sicher zu schwach.

Definition 8.44. $f : \mathbb{R}^n \supset D \rightarrow \mathbb{R}^n$ besitze die reguläre Nullstelle $\bar{\mathbf{x}}$. Das Iterationsverfahren I heißt **Newton ähnlich** bzgl. $\bar{\mathbf{x}}$, wenn für jede durch I erzeugte, gegen $\bar{\mathbf{x}}$ konvergente, nichttriviale Folge $\{\mathbf{x}^m\}$ gilt:

$$\lim_{m \rightarrow \infty} \frac{\|\mathbf{q}^m - \mathbf{p}^m\|}{\|\mathbf{p}^m\|} = 0. \quad (8.37)$$

Dabei heißt $\{\mathbf{x}^m\}$ nichttrivial, falls $\mathbf{x}^m \neq \bar{\mathbf{x}}$ für alle m gilt.

Satz 8.45. $f : \mathbb{R}^n \supset D \rightarrow \mathbb{R}^n$ besitze die reguläre Nullstelle $\bar{\mathbf{x}} \in D$. Dann ist das Verfahren I genau dann Newton ähnlich bzgl. $\bar{\mathbf{x}}$, wenn es bzgl. $\bar{\mathbf{x}}$ Q -superlinear konvergent ist.

Beweis: Sei $S := \{\mathbf{x} : \|\mathbf{x} - \bar{\mathbf{x}}\| \leq \delta\}$ die in Satz 8.37. konstruierte Kugel, in der das Newton Verfahren für jeden Startwert verbleibt. Dann existiert $m_0 \in \mathbb{N}$, so dass $\mathbf{x}^m \in S$ und $\mathbf{x}^m \neq \bar{\mathbf{x}}$ für alle $m \geq m_0$ gilt. Insbesondere ist daher für $m \geq m_0$ die Newton Korrektur $-\mathbf{p}^m$ definiert und von $\mathbf{0}$ verschieden, so dass der Quotient in (8.37) definiert ist.

Sei $\mathbf{y}^m := \mathbf{x}^m - \mathbf{p}^m$ der zu \mathbf{x}^m gehörende Newton Punkt. Dann gilt nach Satz 8.37.

$$\|\mathbf{y}^m - \bar{\mathbf{x}}\| \leq Q\|\mathbf{x}^m - \bar{\mathbf{x}}\|^2 \leq Q\delta\|\mathbf{x}^m - \bar{\mathbf{x}}\| =: q\|\mathbf{x}^m - \bar{\mathbf{x}}\|.$$

Es folgt

$$|\|\mathbf{x}^m - \bar{\mathbf{x}}\| - \|\mathbf{p}^m\|| \leq \|\mathbf{x}^m - \bar{\mathbf{x}} - \mathbf{p}^m\| = \|\mathbf{y}^m - \bar{\mathbf{x}}\| \leq q\|\mathbf{x}^m - \bar{\mathbf{x}}\|$$

und daher

$$(1 - q)\|\mathbf{x}^m - \bar{\mathbf{x}}\| \leq \|\mathbf{p}^m\| \leq (1 + q)\|\mathbf{x}^m - \bar{\mathbf{x}}\|. \quad (8.38)$$

Genauso erhält man aus

$$\begin{aligned} |\|\mathbf{x}^{m+1} - \bar{\mathbf{x}}\| - \|\mathbf{p}^m - \mathbf{q}^m\|| &\leq \|\mathbf{x}^{m+1} - \bar{\mathbf{x}} - \mathbf{p}^m + \mathbf{q}^m\| \\ &= \|\mathbf{y}^m - \bar{\mathbf{x}}\| \leq Q\|\mathbf{x}^m - \bar{\mathbf{x}}\|^2 \end{aligned}$$

die Abschätzungen

$$\|\mathbf{x}^{m+1} - \bar{\mathbf{x}}\| \leq \|\mathbf{p}^m - \mathbf{q}^m\| + Q\|\mathbf{x}^m - \bar{\mathbf{x}}\|^2 \quad (8.39)$$

und

$$\|\mathbf{p}^m - \mathbf{q}^m\| \leq \|\mathbf{x}^{m+1} - \bar{\mathbf{x}}\| + Q\|\mathbf{x}^m - \bar{\mathbf{x}}\|^2. \quad (8.40)$$

Wegen (8.38) und (8.39) ist somit

$$\begin{aligned} \frac{\|\mathbf{x}^{m+1} - \bar{\mathbf{x}}\|}{\|\mathbf{x}^m - \bar{\mathbf{x}}\|} &\leq \frac{\|\mathbf{p}^m - \mathbf{q}^m\| + Q\|\mathbf{x}^m - \bar{\mathbf{x}}\|^2}{\|\mathbf{x}^m - \bar{\mathbf{x}}\|} \\ &\leq (1+q) \frac{\|\mathbf{p}^m - \mathbf{q}^m\|}{\|\mathbf{p}^m\|} + Q\|\mathbf{x}^m - \bar{\mathbf{x}}\|, \end{aligned}$$

d.h. (8.37) impliziert die Q-superlineare Konvergenz von \mathbf{x}^m

$$\lim_{m \rightarrow \infty} \frac{\|\mathbf{x}^{m+1} - \bar{\mathbf{x}}\|}{\|\mathbf{x}^m - \bar{\mathbf{x}}\|} = 0. \quad (8.41)$$

Umgekehrt folgt aus (8.41) mit (8.38) und (8.40)

$$\begin{aligned} \frac{\|\mathbf{p}^m - \mathbf{q}^m\|}{\|\mathbf{p}^m\|} &\leq \frac{\|\mathbf{x}^{m+1} - \bar{\mathbf{x}}\| + Q\|\mathbf{x}^m - \bar{\mathbf{x}}\|^2}{(1-q)\|\mathbf{x}^m - \bar{\mathbf{x}}\|} \\ &\leq \frac{1}{1-q} \frac{\|\mathbf{x}^{m+1} - \bar{\mathbf{x}}\|}{\|\mathbf{x}^m - \bar{\mathbf{x}}\|} + \frac{Q}{1-q} \|\mathbf{x}^m - \bar{\mathbf{x}}\| \rightarrow 0 \end{aligned}$$

also (8.37) ■

Satz 8.45. besagt, dass sich ein superlinear konvergentes Verfahren nur durch genügend genaue Approximation der Newton Korrektur gewinnen lässt. Dies unterstreicht die fundamentale Bedeutung des Newton Verfahrens.

Wir wollen nun die Bedingung (8.37) durch Bedingungen ersetzen, die nicht mehr \mathbf{p}^m enthalten. Dazu benötigen wir zunächst

Lemma 8.46. *f* besitze die reguläre Nullstelle $\bar{\mathbf{x}} \in D$. Dann existieren $\delta > 0$ und Konstanten $M_1, M_2 > 0$, so dass

$$M_1\|\mathbf{x} - \mathbf{y}\| \leq \|f(\mathbf{x}) - f(\mathbf{y})\| \leq M_2\|\mathbf{x} - \mathbf{y}\| \quad (8.42)$$

für alle $\mathbf{x}, \mathbf{y} \in U := \{\mathbf{z} : \|\mathbf{z} - \bar{\mathbf{x}}\| \leq \delta\}$.

Beweis: Die rechte Ungleichung folgt sofort aus dem Mittelwertsatz, die linke ist eine Folgerung aus dem Satz über implizite Funktionen.

Betrachte $g(\mathbf{x}, \mathbf{y}) := f(\mathbf{x}) - \mathbf{y} = \mathbf{0}$. Dann gilt

$$\frac{\partial}{\partial \mathbf{x}} g(\bar{\mathbf{x}}, \mathbf{0}) = f'(\bar{\mathbf{x}}), \quad g(\bar{\mathbf{x}}, \mathbf{0}) = \mathbf{0},$$

und daher existiert φ mit

$$g(\varphi(\mathbf{y}), \mathbf{y}) = \mathbf{0} \iff f(\varphi(\mathbf{y})) = \mathbf{y} \quad \text{für alle } \|\mathbf{y}\| \leq \eta$$

Wegen der Lipschitz Stetigkeit von φ gilt mit einer Konstanten $q > 0$

$$\|\mathbf{x}^1 - \mathbf{x}^2\| = \|\varphi(\mathbf{y}^1) - \varphi(\mathbf{y}^2)\| \leq q\|\mathbf{y}^1 - \mathbf{y}^2\| = q\|f(\mathbf{x}^1) - f(\mathbf{x}^2)\|. \quad \blacksquare$$

Satz 8.47. f besitze die reguläre Nullstelle $\bar{\mathbf{x}} \in D$ und $\{\mathbf{x}^m\}$ sei eine nichttriviale, gegen $\bar{\mathbf{x}}$ konvergente Folge. $\{\mathbf{x}^m\}$ konvergiert genau dann Q -superlinear gegen $\bar{\mathbf{x}}$, wenn

$$\lim_{m \rightarrow \infty} \frac{\|f(\mathbf{x}^{m+1})\|}{\|\mathbf{x}^{m+1} - \mathbf{x}^m\|} = 0 \quad (8.43)$$

gilt.

Beweis: Sei $\{\mathbf{x}^m\}$ Q -superlinear konvergent gegen $\bar{\mathbf{x}}$. Dann gilt für genügend große $m \in \mathbb{N}$

$$Q_m := \frac{\|\mathbf{x}^{m+1} - \bar{\mathbf{x}}\|}{\|\mathbf{x}^m - \bar{\mathbf{x}}\|} < 1,$$

und aus (8.42) folgt

$$\frac{\|f(\mathbf{x}^{m+1})\|}{\|\mathbf{x}^{m+1} - \mathbf{x}^m\|} \leq \frac{M_2 \|\mathbf{x}^{m+1} - \bar{\mathbf{x}}\|}{\|\mathbf{x}^{m+1} - \mathbf{x}^m\|} \leq \frac{M_2 \|\mathbf{x}^{m+1} - \bar{\mathbf{x}}\|}{\|\mathbf{x}^m - \bar{\mathbf{x}}\| - \|\mathbf{x}^{m+1} - \bar{\mathbf{x}}\|} = \frac{M_2 Q_m}{1 - Q_m} \rightarrow 0$$

wegen $Q_m \rightarrow 0$, d.h. (8.43).

Gilt umgekehrt (8.43), so liefert die linke Ungleichung von (8.42)

$$\frac{\|f(\mathbf{x}^{m+1})\|}{\|\mathbf{x}^{m+1} - \mathbf{x}^m\|} \geq \frac{M_1 \|\mathbf{x}^{m+1} - \bar{\mathbf{x}}\|}{\|\mathbf{x}^{m+1} - \bar{\mathbf{x}}\| + \|\mathbf{x}^m - \bar{\mathbf{x}}\|} = M_1 \frac{Q_m}{1 + Q_m},$$

und es folgt $\lim_{m \rightarrow \infty} Q_m = 0$. ■

Soll \mathbf{q}^m die Newton Korrektur \mathbf{p}^m approximieren, so ist der Ansatz

$$\mathbf{q}^m := \mathbf{A}_m^{-1} f(\mathbf{x}^m)$$

mit einer noch zu wählenden, nichtsingulären Matrix $\mathbf{A}_m \in \mathbb{R}^{(n,n)}$ naheliegend. Für Korrekturen dieses Typs lässt sich eine weitere äquivalente Beziehung angeben.

Satz 8.48. f besitze die reguläre Nullstelle $\bar{\mathbf{x}} \in D$ und sei $\{\mathbf{x}^m\}$ eine nichttriviale, gegen $\bar{\mathbf{x}}$ konvergente Folge der Form

$$\mathbf{x}^{m+1} = \mathbf{x}^m - \mathbf{A}_m^{-1} f(\mathbf{x}^m), \quad m \geq 0. \quad (8.44)$$

Dann ist $\{\mathbf{x}^m\}$ genau dann Q -superlinear konvergent gegen $\bar{\mathbf{x}}$, wenn

$$\lim_{m \rightarrow \infty} \frac{\|(\mathbf{A}_m - f'(\bar{\mathbf{x}}))(\mathbf{x}^{m+1} - \mathbf{x}^m)\|}{\|\mathbf{x}^{m+1} - \mathbf{x}^m\|} = 0. \quad (8.45)$$

Beweis: Wegen der Lipschitz Stetigkeit von f' und wegen

$$f(\mathbf{x}^m) = -\mathbf{A}_m(\mathbf{x}^{m+1} - \mathbf{x}^m)$$

ist für genügend großes m

$$\begin{aligned} & \|(\mathbf{A}_m - f'(\bar{\mathbf{x}}))(\mathbf{x}^{m+1} - \mathbf{x}^m) + f(\mathbf{x}^{m+1})\| \\ &= \|[f(\mathbf{x}^{m+1}) - f(\mathbf{x}^m) - f'(\mathbf{x}^m)(\mathbf{x}^{m+1} - \mathbf{x}^m)] + [f'(\mathbf{x}^m) - f'(\bar{\mathbf{x}})](\mathbf{x}^{m+1} - \mathbf{x}^m)\| \\ &\leq \frac{q}{2}\|\mathbf{x}^{m+1} - \mathbf{x}^m\|^2 + q\|\mathbf{x}^m - \bar{\mathbf{x}}\| \cdot \|\mathbf{x}^{m+1} - \mathbf{x}^m\|. \end{aligned}$$

Daher folgt

$$\begin{aligned} & \left| \frac{\|(\mathbf{A}_m - f'(\bar{\mathbf{x}}))(\mathbf{x}^{m+1} - \mathbf{x}^m)\|}{\|\mathbf{x}^{m+1} - \mathbf{x}^m\|} - \frac{\|f(\mathbf{x}^{m+1})\|}{\|\mathbf{x}^{m+1} - \mathbf{x}^m\|} \right| \\ & \leq q \left(\frac{1}{2}\|\mathbf{x}^{m+1} - \mathbf{x}^m\| + \|\mathbf{x}^m - \bar{\mathbf{x}}\| \right), \end{aligned}$$

d.h. (8.45) und (8.43) sind äquivalent, und die Behauptung folgt aus Satz 8.47. ■

Hinreichend für das Erfülltsein von (8.45) ist die Bedingung

$$\lim_{m \rightarrow \infty} \|\mathbf{A}_m - f'(\bar{\mathbf{x}})\| = \mathbf{O}, \quad (8.46)$$

d.h. das Iterationsverfahren ist sicher dann Newton ähnlich, wenn die \mathbf{A}_m die Jacobi-Matrix $f'(\bar{\mathbf{x}})$ beliebig gut approximieren.

Da (8.46) noch das unbekanntes $\bar{\mathbf{x}}$ explizit enthält, ist es sinnvoll, (8.46) durch die nachprüfbarere (aber einschneidendere) Bedingung

$$\lim_{m \rightarrow \infty} \|\mathbf{A}_m - f'(\mathbf{x}^m)\| = 0 \quad (8.47)$$

zu ersetzen. In der Tat gilt

$$\begin{aligned} & \frac{1}{\|\mathbf{x}^{m+1} - \mathbf{x}^m\|} \|(\mathbf{A}_m - f'(\bar{\mathbf{x}}))(\mathbf{x}^{m+1} - \mathbf{x}^m)\| \leq \|\mathbf{A}_m - f'(\bar{\mathbf{x}})\| \\ & \leq \|\mathbf{A}_m - f'(\mathbf{x}^m)\| + \|f'(\mathbf{x}^m) - f'(\bar{\mathbf{x}})\|, \end{aligned}$$

d.h. aus (8.47) folgt (8.46), und hieraus folgt (8.45) für $\lim_{m \rightarrow \infty} \mathbf{x}^m = \bar{\mathbf{x}}$.

Die einfachste und nächstliegende Möglichkeit, ein Verfahren der Gestalt (8.44) zu konstruieren, das (8.46) erfüllt, besteht darin, die partiellen Ableitungen $\frac{\partial f_i}{\partial x_j}$ durch Differenzenquotienten zu ersetzen:

$$\mathbf{A}_m = \frac{1}{h_m} (f_i(\mathbf{x}^m + h_m \mathbf{e}^j) - f_i(\mathbf{x}^m))_{i,j=1,\dots,n}, \quad h_m \rightarrow 0 \text{ für } m \rightarrow \infty.$$

Setzt man $\mathbf{u}^{m,j} := \mathbf{x}^m + h_m \mathbf{e}^j$, $\mathbf{v}^{m,j} := \mathbf{x}^m$, so ist \mathbf{A}_m bestimmt durch die n Gleichungen

$$\mathbf{A}_m(\mathbf{u}^{m,j} - \mathbf{v}^{m,j}) = f(\mathbf{u}^{m,j}) - f(\mathbf{v}^{m,j}), \quad j = 1, \dots, n. \quad (8.48)$$

Verallgemeinerungen dieser Vorgehensweise sind möglich, indem man in jeder Gleichung j von (8.48)

- eine andere Schrittweite zulässt, d.h. die Differenzen $\mathbf{u}^{m,j} - \mathbf{v}^{m,j}$ brauchen nicht mehr dieselbe Länge zu haben,
- die Richtungen $\mathbf{u}^{m,j} - \mathbf{v}^{m,j}$ nicht mehr parallel zu den Koordinatenachsen \mathbf{e}^j wählt
- die Punkte $\mathbf{v}^{m,j}$ nicht notwendig mit \mathbf{x}^m identifiziert.

In Schwetlick [94] werden eine Reihe von Newton ähnlichen Verfahren diskutiert, die jedoch in der Praxis keine große Bedeutung haben, da sie entweder denselben Aufwand wie das Newton Verfahren oder die obige Modifikation erfordern oder degenerieren können, d.h. \mathbf{A}_m ist durch (8.48) nicht mehr eindeutig bestimmt.

8.6 Quasi-Newton Verfahren

Der größte Nachteil des Newton Verfahrens und der in Abschnitt 8.5 angesprochenen Newton ähnlichen Verfahren ist, dass in jedem Schritt $n(n+1)$ reelle Funktionen ausgewertet werden müssen. Wir wollen in diesem Abschnitt Verfahren besprechen, die mit n Funktionsauswertungen auskommen (mit weniger wird es kaum gehen!) und ebenfalls superlinear konvergieren. Bemerkenswert ist, dass hierfür die hinreichende Bedingung (8.46) für die superlineare Konvergenz nicht erfüllt ist.

Es seien eine Näherung \mathbf{x}^m mit $f(\mathbf{x}^m) \neq \mathbf{0}$ für eine reguläre Nullstelle $\bar{\mathbf{x}}$ von f und eine reguläre Approximation \mathbf{B}_m für $f'(\mathbf{x}^m)$ bekannt. Wir bestimmen dann eine neue Näherung für $\bar{\mathbf{x}}$ durch

$$\mathbf{x}^{m+1} = \mathbf{x}^m - \mathbf{B}_m^{-1} f(\mathbf{x}^m). \quad (8.49)$$

Da \mathbf{B}_m regulär ist und $f(\mathbf{x}^m) \neq \mathbf{0}$, ist die Änderung

$$\mathbf{s}^m := \mathbf{x}^{m+1} - \mathbf{x}^m \quad (8.50)$$

von $\mathbf{0}$ verschieden. Die neue Approximation \mathbf{B}_{m+1} für $f'(\mathbf{x}^{m+1})$ soll dann so bestimmt werden, dass gilt

$$\mathbf{B}_{m+1}(\mathbf{x}^{m+1} - \mathbf{x}^m) = f(\mathbf{x}^{m+1}) - f(\mathbf{x}^m) \quad (8.51)$$

(vgl. (8.48)). Dabei soll nur \mathbf{B}_m , die im gerade ausgeführten Schritt eingetretene Änderung \mathbf{s}^m der Argumente und die zugehörige Änderung der Funktionswerte

$$f(\mathbf{x}^{m+1}) - f(\mathbf{x}^m) =: \mathbf{y}^m \quad (8.52)$$

verwendet werden. Insbesondere sollen keine zusätzlichen Funktionsauswertungen benutzt werden.

Wir suchen also \mathbf{B}_{m+1} in der Form

$$\mathbf{B}_{m+1} = \Phi(\mathbf{B}_m, \mathbf{s}^m, \mathbf{y}^m) \quad (8.53)$$

mit einer **Aufdatierungsfunktion**

$$\Phi : \mathbb{R}^{(n,n)} \times \mathbb{R}^n \times \mathbb{R}^n \supset D_\Phi \rightarrow \mathbb{R}^{(n,n)}.$$

Definition 8.49. Die Vorschrift (8.53) heißt **Aufdatierungsformel**. Verfahren der Gestalt (8.49), (8.53), die der Bedingung (8.51) genügen, heißen **Quasi-Newton Verfahren**.

\mathbf{B}_{m+1} ist durch die Bedingung (8.51) (außer im Falle $n = 1$, in dem man das Sekantenverfahren erhält) nicht eindeutig festgelegt. Die zu (8.51) äquivalente Bedingung

$$(\mathbf{B}_{m+1} - \mathbf{B}_m)\mathbf{s}^m = \mathbf{y}^m - \mathbf{B}_m\mathbf{s}^m \quad (= f(\mathbf{x}^{m+1}) \neq \mathbf{0}) \quad (8.54)$$

zeigt, dass durch sie die Änderung von \mathbf{B}_m nur in bezug auf die Richtung \mathbf{s}^m bestimmt wird.

Wir fordern zusätzlich, dass die Matrizen \mathbf{B}_{m+1} hinsichtlich ihrer Wirkung auf zu \mathbf{s}^m orthogonale Richtungen unverändert bleiben:

$$(\mathbf{B}_{m+1} - \mathbf{B}_m)\mathbf{s} = \mathbf{0} \quad \text{für alle } \mathbf{s} \in \mathbb{R}^n \text{ mit } \mathbf{s}^T \mathbf{s}^m = 0 \quad (8.55)$$

Wegen (8.55) besitzt die Matrix $\mathbf{B}_{m+1} - \mathbf{B}_m$ den Rang 1, ist also von der Gestalt

$$\mathbf{B}_{m+1} - \mathbf{B}_m = \mathbf{u}^m(\mathbf{v}^m)^T, \quad \mathbf{u}^m \neq \mathbf{0}, \mathbf{v}^m \neq \mathbf{0}.$$

Aus (8.55) folgt

$$(\mathbf{B}_{m+1} - \mathbf{B}_m)\mathbf{s} = \mathbf{u}^m(\mathbf{v}^m)^T \mathbf{s} = \mathbf{0} \quad \text{für alle } \mathbf{s} \text{ mit } \mathbf{s} \perp \mathbf{s}^m$$

d.h.

$$(\mathbf{v}^m)^T \mathbf{s} = 0 \quad \text{für alle } \mathbf{s} \text{ mit } \mathbf{s} \perp \mathbf{s}^m$$

und daher ohne Beschränkung der Allgemeinheit $\mathbf{v}^m = \mathbf{s}^m$.

Die Quasi-Newton Gleichung (8.54) liefert

$$(\mathbf{B}_{m+1} - \mathbf{B}_m) \mathbf{s}^m = \mathbf{u}^m (\mathbf{s}^m)^T \mathbf{s}^m = \mathbf{y}^m - \mathbf{B}_m \mathbf{s}^m,$$

d.h.

$$\mathbf{u}^m = \frac{1}{\|\mathbf{s}^m\|_2^2} (\mathbf{y}^m - \mathbf{B}_m \mathbf{s}^m),$$

und es ist

$$\mathbf{B}_{m+1} = \mathbf{B}_m + \frac{1}{\|\mathbf{s}^m\|_2^2} (\mathbf{y}^m - \mathbf{B}_m \mathbf{s}^m) (\mathbf{s}^m)^T. \quad (8.56)$$

Die zu (8.56) gehörige Aufdatierungsfunktion

$$\Phi(\mathbf{B}, \mathbf{s}, \mathbf{y}) = \mathbf{B} + \frac{1}{\|\mathbf{s}\|_2^2} (\mathbf{y} - \mathbf{B}\mathbf{s}) \mathbf{s}^T$$

ist auf der Menge

$$D_\Phi := \{(\mathbf{A}, \mathbf{s}, \mathbf{y}) : \mathbf{A} \in \mathbb{R}^{(n,n)}, \mathbf{y} \in \mathbb{R}^n, \mathbf{s} \in \mathbb{R}^n \setminus \{\mathbf{0}\}\}$$

definiert.

Definition 8.50. *Das durch (8.56) gegebene Quasi-Newton Verfahren heißt **Broyden Rang-1-Verfahren**.*

Eine andere Motivation für das Broyden Rang-1-Verfahren liefert

Satz 8.51. *Es seien $\mathbf{B} \in \mathbb{R}^{(n,n)}$, $\mathbf{y} \in \mathbb{R}^n$ und $\mathbf{s} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$, gegeben. Dann ist*

$$\mathbf{B}' := \mathbf{B} + \frac{1}{\|\mathbf{s}\|_2^2} (\mathbf{y} - \mathbf{B}\mathbf{s}) \mathbf{s}^T$$

die eindeutige Lösung des Minimierungsproblems

$$\|\mathbf{B} - \bar{\mathbf{B}}\|_s = \min!, \quad \bar{\mathbf{B}}\mathbf{s} = \mathbf{y},$$

wobei $\|\cdot\|_s$ die Schur-Norm bezeichnet.

Beweis: Für alle $\mathbf{C}, \mathbf{D} \in \mathbb{R}^{(n,n)}$ gilt $\|\mathbf{CD}\|_s \leq \|\mathbf{C}\|_s \cdot \|\mathbf{D}\|_2$. Daher folgt für alle $\bar{\mathbf{B}} \in \mathbb{R}^{(n,n)}$ mit $\mathbf{y} = \bar{\mathbf{B}}\mathbf{s}$

$$\begin{aligned} \|\mathbf{B}' - \mathbf{B}\|_s &= \left\| \frac{(\mathbf{y} - \mathbf{B}\mathbf{s})\mathbf{s}^T}{\|\mathbf{s}\|_2^2} \right\|_s = \left\| (\bar{\mathbf{B}} - \mathbf{B}) \frac{\mathbf{s}\mathbf{s}^T}{\|\mathbf{s}\|_2^2} \right\|_s \\ &\leq \|\bar{\mathbf{B}} - \mathbf{B}\|_s \cdot \left\| \frac{\mathbf{s}\mathbf{s}^T}{\|\mathbf{s}\|_2^2} \right\|_2 = \|\bar{\mathbf{B}} - \mathbf{B}\|_s, \end{aligned}$$

denn die symmetrische Matrix $\mathbf{s}\mathbf{s}^T \in \mathbb{R}^{(n,n)}$ besitzt den $(n-1)$ -fachen Eigenwert 0 und den einfachen Eigenwert $\|\mathbf{s}\|_2^2$ (mit dem Eigenvektor \mathbf{s}), d.h. es ist

$$\left\| \frac{\mathbf{s}\mathbf{s}^T}{\|\mathbf{s}\|_2^2} \right\|_2 = 1.$$

Die obige Rechnung zeigt, dass \mathbf{B}' das angegebene Minimierungsproblem löst. Da das Funktional $f(\mathbf{A}) := \|\mathbf{B} - \mathbf{A}\|_s$ strikt konvex ist und die zulässige Menge ein affiner Raum, also konvex ist, ist die Lösung eindeutig. ■

\mathbf{B}_{m+1} ist also diejenige Matrix, die die Quasi-Newton Bedingung (8.51) erfüllt und bzgl. der Schur-Norm der Matrix \mathbf{B}_m des letzten Schrittes am nächsten liegt.

Die Lösung des linearen Gleichungssystems $\mathbf{B}_m \mathbf{s}^m = -f(\mathbf{x}^m)$ in m -ten Schritt des Broyden Rang-1-Verfahrens kann vermieden werden, denn es gilt

Lemma 8.52. Seien $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ und $\mathbf{A} \in \mathbb{R}^{(n,n)}$ regulär. Dann ist die Rang 1 Modifikation $\mathbf{A} + \mathbf{u}\mathbf{v}^T$ von \mathbf{A} genau dann regulär, wenn $\sigma := 1 + \mathbf{v}^T \mathbf{A}^{-1} \mathbf{u} \neq 0$.

Ist $\sigma \neq 0$, so gilt

$$(\mathbf{A} + \mathbf{u}\mathbf{v}^T)^{-1} = \mathbf{A}^{-1} - \frac{1}{\sigma} \mathbf{A}^{-1} \mathbf{u}\mathbf{v}^T \mathbf{A}^{-1}. \quad (8.57)$$

Definition 8.53. Die Aufdatierungsformel (8.57) für die Inverse einer Rang-1-Modifikation einer regulären Matrix heißt **Sherman Morrison Formel**.

Beweis: Für $\sigma \neq 0$ ist die rechte Seite von (8.57) definiert und man rechnet leicht nach, dass

$$(\mathbf{A} + \mathbf{u}\mathbf{v}^T) \left(\mathbf{A}^{-1} - \frac{1}{\sigma} \mathbf{A}^{-1} \mathbf{u}\mathbf{v}^T \mathbf{A}^{-1} \right) = \mathbf{I}$$

gilt.

Für $\sigma \neq 0$ ist $\mathbf{z} := \mathbf{A}^{-1} \mathbf{u} \neq \mathbf{0}$ und $(\mathbf{A} + \mathbf{u}\mathbf{v}^T) \mathbf{z} = \mathbf{0}$, d.h. $\mathbf{A} + \mathbf{u}\mathbf{v}^T$ ist singulär. ■

Es sei $\mathbf{H}_m := \mathbf{B}_m^{-1}$ und \mathbf{B}_{m+1} die mit dem Broyden Rang-1-Verfahren aufdatierte Matrix. Dann ist $\mathbf{H}_{m+1} := \mathbf{B}_{m+1}^{-1}$ nach Lemma 8.52. genau dann definiert, wenn $(\mathbf{s}^m)^T \mathbf{H}_m \mathbf{y}^m \neq 0$ gilt, und

$$\mathbf{H}_{m+1} = \mathbf{H}_m + \frac{1}{(\mathbf{s}^m)^T \mathbf{H}_m \mathbf{y}^m} \cdot (\mathbf{s}^m - \mathbf{H}_m \mathbf{y}^m)(\mathbf{s}^m)^T \mathbf{H}_m \quad (8.58)$$

(wir wählen nämlich $\mathbf{u} = \mathbf{y}^m - \mathbf{B}_m \mathbf{s}^m$ und $\mathbf{v} = \mathbf{s}^m / \|\mathbf{s}^m\|_2^2$).

Hiermit kann man das Broyden Verfahren schreiben als

$$\mathbf{x}^{m+1} = \mathbf{x}^m - \mathbf{H}_m f(\mathbf{x}^m), \quad (8.59)$$

und man benötigt offenbar in jedem Schritt sowohl für die Aufdatierung von \mathbf{H}_{m+1} als auch für die Berechnung der neuen Näherung \mathbf{x}^{m+1} nur $O(n^2)$ Operationen neben den n Funktionsauswertungen.

Mit demselben Aufwand kann man das Broyden Rang-1-Verfahren in seiner ursprünglichen Gestalt (8.49) behandeln, wenn man die spezielle Gestalt der Matrix \mathbf{B}_{m+1} in (8.56) beachtet. Wir leiten dazu eine Aufdatierung für die QR Zerlegung für die Rang 1-Modifikation einer Matrix her.

Es sei für die reguläre Matrix \mathbf{A} eine QR Zerlegung bekannt:

$$\mathbf{P}\mathbf{A} = \mathbf{R}, \quad \mathbf{P} \text{ orthogonal, } \mathbf{R} \text{ obere Dreiecksmatrix,}$$

und es sei $\bar{\mathbf{A}} = \mathbf{A} + \mathbf{u}\mathbf{v}^T$ eine Rang-1-Modifikation von \mathbf{A}

Dann gilt

$$\mathbf{P}\bar{\mathbf{A}} = \mathbf{R} + \mathbf{w}\mathbf{v}^T, \quad \mathbf{w} := \mathbf{P}\mathbf{u}. \quad (8.60)$$

Wir annullieren zunächst die Komponenten $n, n-1, \dots, 2$ von \mathbf{w} durch Multiplikation von \mathbf{w} mit Rotationsmatrizen $\mathbf{J}_{n-1,n}, \mathbf{J}_{n-2,n-1}, \dots, \mathbf{J}_{12}$ von links, so dass

$$\tilde{\mathbf{w}} = k\mathbf{e}^1 := \mathbf{J}_{12}\mathbf{J}_{23}\dots\mathbf{J}_{n-1,n}\mathbf{w} =: \mathbf{J}\mathbf{w}, \quad k = \pm\|\mathbf{w}\|_2 = \pm\|\mathbf{u}\|_2.$$

Multipliziert man (8.60) mit \mathbf{J} , so erhält man

$$\tilde{\mathbf{P}}\bar{\mathbf{A}} = \mathbf{R}' + k\mathbf{e}^1\mathbf{v}^T, \quad \tilde{\mathbf{P}} := \mathbf{J}\mathbf{P}, \quad \mathbf{R}' := \mathbf{J}\mathbf{R}.$$

Dabei ist $\tilde{\mathbf{P}}$ mit \mathbf{P} und \mathbf{J} eine orthogonale Matrix, und \mathbf{R}' und damit auch $\tilde{\mathbf{R}} = \mathbf{R} + k\mathbf{e}^1\mathbf{v}^T$ ist eine obere Hessenberg Matrix. Natürlich hätte man \mathbf{w} auch durch eine Householder Transformation \mathbf{H} in $k\mathbf{e}^1$ abbilden können, dann wäre aber $\mathbf{H}\mathbf{R}$ voll besetzt.

Im 2. Teilschritt annullieren wir nun der Reihe nach die Subdiagonalelemente $\tilde{r}_{i+1,i}$, $i = 1, \dots, n-1$, von $\tilde{\mathbf{R}}$ durch geeignete Rotationen $\tilde{\mathbf{J}}_{12}, \tilde{\mathbf{J}}_{23}, \dots, \tilde{\mathbf{J}}_{n-1,n}$, so dass schließlich

$$\bar{\mathbf{P}}\bar{\mathbf{A}} := \tilde{\mathbf{J}}_{n-1,n} \dots \tilde{\mathbf{J}}_{12} \tilde{\mathbf{J}} \mathbf{P} \bar{\mathbf{A}} = \bar{\mathbf{R}}$$

eine obere Dreiecksmatrix ist.

Wendet man diese Aufdatierungstechnik auf den Übergang von \mathbf{B}_m zu \mathbf{B}_{m+1} an, so kann man offenbar den m -ten Schritt des Broyden Rang-1-Verfahrens mit $O(n^2)$ Operationen ausführen. Da diese Aufdatierung häufig stabiler ist als die Anwendung der Sherman Morrison Formel (8.58) bzw. (8.59), wird sie vorgezogen.

Bevor wir einen lokalen Konvergenzsatz für das Broyden Rang-1-Verfahren beweisen können, benötigen wir zunächst eine Abschätzung für die aufdatierte Matrix.

Lemma 8.54. *Sei $\bar{\mathbf{x}}$ eine reguläre Nullstelle von f , $\mathbf{x} \in S^*$ mit $f(\mathbf{x}) \neq \mathbf{0}$ und $\mathbf{B} \in \mathbb{R}^{(n,n)}$ eine regulär Matrix, so dass $\mathbf{x}' := \mathbf{x} - \mathbf{B}^{-1}f(\mathbf{x}) \in S^*$. Es sei*

$$\mathbf{B}' = \mathbf{B} + \frac{1}{\|\mathbf{x}' - \mathbf{x}\|_2^2} \cdot (f(\mathbf{x}') - f(\mathbf{x}) - \mathbf{B}(\mathbf{x}' - \mathbf{x}))(\mathbf{x}' - \mathbf{x})^T.$$

Dann gilt mit der Lipschitz Konstante q von f' in S^*

$$\|\mathbf{B}' - f'(\bar{\mathbf{x}})\|_2 \leq \|\mathbf{B} - f'(\bar{\mathbf{x}})\|_2 + \frac{q}{2\|\mathbf{x}' - \mathbf{x}\|_2} (\|\mathbf{x}' - \bar{\mathbf{x}}\|_2^2 + \|\mathbf{x} - \bar{\mathbf{x}}\|_2^2).$$

Beweis: Es ist

$$\begin{aligned} \mathbf{B}' - f'(\bar{\mathbf{x}}) &= \mathbf{B} - f'(\bar{\mathbf{x}}) - \frac{1}{\|\mathbf{x}' - \mathbf{x}\|_2^2} (\mathbf{B} - f'(\bar{\mathbf{x}}))(\mathbf{x}' - \mathbf{x})(\mathbf{x}' - \mathbf{x})^T \\ &\quad + \frac{1}{\|\mathbf{x}' - \mathbf{x}\|_2^2} (f(\mathbf{x}') - f(\bar{\mathbf{x}}) - f'(\bar{\mathbf{x}})(\mathbf{x}' - \bar{\mathbf{x}}))(\mathbf{x}' - \mathbf{x})^T \\ &\quad - \frac{1}{\|\mathbf{x}' - \mathbf{x}\|_2^2} (f(\mathbf{x}) - f(\bar{\mathbf{x}}) - f'(\bar{\mathbf{x}})(\mathbf{x} - \bar{\mathbf{x}}))(\mathbf{x}' - \mathbf{x})^T. \end{aligned}$$

Die symmetrische Matrix $\mathbf{I} - \mathbf{u}\mathbf{u}^T$, $\|\mathbf{u}\|_2 = 1$, besitzt offenbar den $(n-1)$ -fachen Eigenwert 1 und den einfachen Eigenwert 0, und für die Rang-1-Matrix $\mathbf{C} := \mathbf{v}\mathbf{w}^T$ gilt $\mathbf{C}^T\mathbf{C} = \|\mathbf{v}\|_2^2\mathbf{w}\mathbf{w}^T$ mit dem einfachen Eigenwert $\|\mathbf{v}\|_2^2\|\mathbf{w}\|_2^2$ und dem $(n-1)$ -fachen Eigenwert 0. Daher gilt $\|\mathbf{I} - \mathbf{u}\mathbf{u}^T\|_2 = 1$ und $\|\mathbf{v}\mathbf{w}^T\|_2 = \|\mathbf{v}\|_2\|\mathbf{w}\|_2$, und es folgt aus Lemma 8.36.

$$\begin{aligned} \|\mathbf{B}' - f'(\bar{\mathbf{x}})\|_2 &\leq \|\mathbf{B} - f'(\bar{\mathbf{x}})\|_2 \cdot \left\| \mathbf{I} - \frac{(\mathbf{x}' - \mathbf{x})(\mathbf{x}' - \mathbf{x})^T}{\|\mathbf{x}' - \mathbf{x}\|_2^2} \right\|_2 \\ &\quad + \frac{q}{2} \frac{\|\mathbf{x}' - \bar{\mathbf{x}}\|_2^2}{\|\mathbf{x}' - \mathbf{x}\|_2} + \frac{q}{2} \frac{\|\mathbf{x} - \bar{\mathbf{x}}\|_2^2}{\|\mathbf{x}' - \mathbf{x}\|_2}. \quad \blacksquare \end{aligned}$$

Wir zeigen nun zunächst die lineare Konvergenz des Broyden Rang-1-Verfahrens und dann in einem zweiten Schritt in Satz 8.56. die superlineare Konvergenz.

Satz 8.55. *Es sei $\bar{\mathbf{x}}$ reguläre Nullstelle von f . Dann existieren $\varepsilon > 0$ und $\delta > 0$, so dass das Broyden Rang-1-Verfahren für alle Startwerte \mathbf{x}^0 und \mathbf{B}_0 mit $\|\mathbf{x}^0 - \bar{\mathbf{x}}\|_2 < \varepsilon$ und $\|\mathbf{B}_0 - f'(\bar{\mathbf{x}})\|_2 < \delta$ gegen $\bar{\mathbf{x}}$ Q -linear konvergiert.*

Beweis: Wir bezeichnen in diesem Beweis den Fehler im k -ten Schritt des Verfahrens mit $e_k := \|\bar{\mathbf{x}} - \mathbf{x}^k\|_2$.

Es sei q die Lipschitz Konstante von f' in der Umgebung S^* von $\bar{\mathbf{x}}$ und $\|f'(\bar{\mathbf{x}})^{-1}\| \leq \beta$. Wir wählen dann $\delta \leq \frac{1}{6\beta}$ und $\varepsilon \leq \frac{2\delta}{5q}$, so dass $S := \{\mathbf{x} : \|\mathbf{x} - \bar{\mathbf{x}}\|_2 \leq \varepsilon\} \subset D$ gilt.

Nach dem Störungslemma existiert $\mathbf{H}_0 := \mathbf{B}_0^{-1}$, und es gilt

$$\|\mathbf{H}_0\|_2 = \frac{\|f'(\bar{\mathbf{x}})^{-1}\|_2}{1 - \|f'(\bar{\mathbf{x}})^{-1}\|_2 \|\mathbf{B}_0 - f'(\bar{\mathbf{x}})\|_2} \leq \frac{\beta}{1 - \beta\delta}.$$

Also ist \mathbf{x}^1 definiert, und es gilt

$$\begin{aligned} e_1 &:= \|\mathbf{x}^0 - \bar{\mathbf{x}} - \mathbf{H}_0(f(\mathbf{x}^0) - f(\bar{\mathbf{x}}))\|_2 \\ &\leq \|\mathbf{H}_0\|_2 \|f(\mathbf{x}^0) - f(\bar{\mathbf{x}}) - \mathbf{B}_0(\mathbf{x}^0 - \bar{\mathbf{x}})\|_2 \\ &\leq \|\mathbf{H}_0\|_2 (\|f(\mathbf{x}^0) - f(\bar{\mathbf{x}}) - f'(\bar{\mathbf{x}})(\mathbf{x}^0 - \bar{\mathbf{x}})\|_2 + \|f'(\bar{\mathbf{x}}) - \mathbf{B}_0\|_2 \|\mathbf{x}^0 - \bar{\mathbf{x}}\|_2) \\ &\leq \frac{\beta}{1 - \beta\delta} \left(\frac{1}{2} q e_0^2 + \delta e_0 \right) = \frac{\beta}{1 - \beta\delta} \left(\frac{1}{2} q e_0 + \delta \right) e_0 \leq \frac{6}{5} \frac{\beta\delta}{1 - \beta\delta} e_0 < \frac{1}{2} e_0. \end{aligned}$$

Daher gilt $\mathbf{x}^1 \in S$ und $f(\mathbf{x}^1)$ und damit \mathbf{B}_1 sind definiert.

Ferner ist

$$\|\mathbf{x}^1 - \mathbf{x}^0\|_2 \geq \|\mathbf{x}^0 - \bar{\mathbf{x}}\|_2 - \|\mathbf{x}^1 - \bar{\mathbf{x}}\|_2 \geq e_0 - \frac{1}{2} e_0 = \frac{1}{2} e_0,$$

und wegen

$$\mathbf{B}_1 = \mathbf{B}_0 + \frac{1}{\|\mathbf{x}^1 - \mathbf{x}^0\|_2^2} (f(\mathbf{x}^1) - f(\mathbf{x}^0 - \mathbf{B}_0(\mathbf{x}^1 - \mathbf{x}^0)))(\mathbf{x}^1 - \mathbf{x}^0)$$

folgt aus Lemma 8.54.

$$\begin{aligned} \|\mathbf{B}_1 - f'(\bar{\mathbf{x}})\|_2 &\leq \delta + \frac{q}{2\|\mathbf{x}^1 - \mathbf{x}^0\|_2} \left(\frac{1}{4} e_0^2 + e_0^2 \right) \\ &= \delta + \frac{5}{4} q e_0 \frac{1}{2} e_0 \|\mathbf{x}^1 - \mathbf{x}^0\|_2^{-1} \leq \delta + \frac{5}{4} q e_0 \leq \frac{3}{2} \delta < 2\delta. \end{aligned}$$

Es gilt also $\|f'(\bar{\mathbf{x}})^{-1}\|_2 \cdot \|\mathbf{B}_1 - f'(\bar{\mathbf{x}})\|_2 \leq 2\beta\delta < 1$. Nach dem Störungslemma existiert $\mathbf{H}_1 = \mathbf{B}_1^{-1}$, und es gilt die Abschätzung

$$\|\mathbf{H}_1\|_2 \leq \frac{\beta}{1 - 2\beta\delta}.$$

Es sei die Existenz von $\mathbf{x}^1, \dots, \mathbf{x}^m$ und von $\mathbf{H}_1, \dots, \mathbf{H}_{m-1}$ bereits bewiesen, und es gelte für $k = 1, \dots, m$

$$e_k \leq 0.5e_{k-1}, \quad \|\mathbf{B}_k - f'(\bar{\mathbf{x}})\|_2 \leq (2 - 0.5^k) \delta.$$

Dann folgt

$$\|f'(\bar{\mathbf{x}})^{-1}\| \cdot \|\mathbf{B}_m - f'(\bar{\mathbf{x}})\|_2 \leq (2 - 0.5^m) \delta \beta < 2\delta\beta \leq \frac{1}{3},$$

also existiert \mathbf{H}_m und $\|\mathbf{H}_m\|_2 \leq \beta/(1 - 2\beta\delta)$. Ferner ist \mathbf{x}^{m+1} ist definiert und es gilt

$$\begin{aligned} e_{m+1} &\leq \|\mathbf{H}_m\|_2 (\|f(\mathbf{x}^m) - f(\bar{\mathbf{x}}) - f'(\bar{\mathbf{x}})(\mathbf{x}^m - \bar{\mathbf{x}})\|_2 + \|f'(\bar{\mathbf{x}}) - \mathbf{B}_m\|_2 e_m) \\ &\leq \frac{\beta}{1 - 2\beta\delta} (0.5qe_m^2 + (2 - 0.5^m)\delta e_m) \\ &\leq \frac{\beta}{1 - 2\beta\delta} \left(\frac{1}{5} \left(\frac{1}{2} \right)^m \delta + \left(2 - \left(\frac{1}{2} \right)^m \right) \delta \right) e_m \\ &\leq \frac{\beta}{1 - 2\beta\delta} 2\delta e_m \leq \frac{1}{2} e_m \end{aligned}$$

sowie wiederum nach Lemma 8.54.

$$\begin{aligned} \|\mathbf{B}_{m+1} - f'(\bar{\mathbf{x}})\|_2 &\leq \|\mathbf{B}_m - f'(\bar{\mathbf{x}})\|_2 + \frac{1}{2\|\mathbf{x}^{m+1} - \mathbf{x}^m\|_2} q (e_{m+1}^2 + e_m^2) \\ &\leq (2 - 0.5^m) \delta + \frac{5}{4} q e_m \\ &\leq \left(2 - 0.5^m + \frac{5}{4} 0.5^m \frac{2}{5} \right) \delta = (2 - 0.5^{m+1}) \delta, \end{aligned}$$

wobei

$$\frac{1}{2} e_m \leq \|\mathbf{x}^{m+1} - \mathbf{x}^m\|_2$$

benutzt wurde.

Damit ist der Induktionsbeweis geführt. Das Broyden Verfahren ist also unbeschränkt ausführbar und konvergiert linear. ■

Satz 8.56. *Unter den Voraussetzungen von Satz 8.55. konvergiert das Broyden-Verfahren lokal Q -superlinear gegen $\bar{\mathbf{x}}$.*

Beweis: Es sei

$$\psi_m := \frac{1}{\|\mathbf{s}^m\|_2} \cdot \|(\mathbf{B}_m - f'(\bar{\mathbf{x}})) \mathbf{s}^m\|_2.$$

Wegen Satz 8.48. genügt es, $\lim_{m \rightarrow \infty} \psi_m = 0$ zu zeigen.

Mit den Bezeichnungen aus Lemma 8.54. gilt

$$\begin{aligned} \mathbf{B}' - f'(\bar{\mathbf{x}}) &= (\mathbf{B} - f'(\bar{\mathbf{x}})) \left(\mathbf{I} - \frac{(\mathbf{x}' - \mathbf{x})(\mathbf{x}' - \mathbf{x})^T}{\|\mathbf{x}' - \mathbf{x}\|_2^2} \right) \\ &\quad + \frac{1}{\|\mathbf{x}' - \mathbf{x}\|_2^2} (f(\mathbf{x}') - f(\mathbf{x}) - f'(\bar{\mathbf{x}})(\mathbf{x}' - \mathbf{x}))(\mathbf{x}' - \mathbf{x})^T, \end{aligned}$$

also wegen Lemma 8.36.

$$\begin{aligned} \|\mathbf{B}' - f'(\bar{\mathbf{x}})\|_s &\leq \left\| (\mathbf{B} - f'(\bar{\mathbf{x}})) \left(\mathbf{I} - \frac{(\mathbf{x}' - \mathbf{x})(\mathbf{x}' - \mathbf{x})^T}{\|\mathbf{x}' - \mathbf{x}\|_2^2} \right) \right\|_s \\ &\quad + \frac{q}{2} (\|\mathbf{x}' - \bar{\mathbf{x}}\|_2 + \|\mathbf{x} - \bar{\mathbf{x}}\|_2) \end{aligned} \quad (8.61)$$

Ferner ist für jedes $\mathbf{C} = (c_{ij})_{i,j=1,\dots,n} \in \mathbb{R}^{(n,n)}$ und $\mathbf{s} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$

$$\begin{aligned} \left\| \mathbf{C} \left(\mathbf{I} - \frac{\mathbf{s}\mathbf{s}^T}{\|\mathbf{s}\|_2^2} \right) \right\|_s^2 &= \left\| \left(c_{ij} - \frac{(\mathbf{C}\mathbf{s})_i s_j}{\|\mathbf{s}\|_2^2} \right) \right\|_s^2 \\ &= \sum_{i,j=1}^n c_{ij}^2 - 2 \sum_{i,j=1}^n c_{ij} \frac{(\mathbf{C}\mathbf{s})_i s_j}{\|\mathbf{s}\|_2^2} + \sum_{i,j=1}^n \frac{(\mathbf{C}\mathbf{s})_i^2 s_j^2}{\|\mathbf{s}\|_2^4} \\ &= \|\mathbf{C}\|_s^2 - \left(\frac{\|\mathbf{C}\mathbf{s}\|_2}{\|\mathbf{s}\|_2} \right)^2, \end{aligned}$$

und wegen $(\alpha^2 - \beta^2)^{1/2} \leq \alpha - (2\alpha)^{-1}\beta^2$ für $0 \leq \beta \leq \alpha$, $\alpha \neq 0$ folgt

$$\left\| \mathbf{C} \left(\mathbf{I} - \frac{\mathbf{s}\mathbf{s}^T}{\|\mathbf{s}\|_2^2} \right) \right\|_s \leq \|\mathbf{C}\|_s - \frac{1}{2\|\mathbf{C}\|_s} \cdot \left(\frac{\|\mathbf{C}\mathbf{s}\|_2}{\|\mathbf{s}\|_2} \right)^2. \quad (8.62)$$

Sei nun

$$\eta_m := \|\mathbf{B}_m - f'(\bar{\mathbf{x}})\|_s.$$

Dann folgt aus (8.61) und (8.62)

$$\eta_{m+1} \leq \eta_m - \frac{1}{2\eta_m} \Psi_m^2 + \frac{q}{2}(e_{m+1} + e_m),$$

und wegen $\eta_m \leq \eta$ (vgl. Beweis von Satz 8.55.)

$$\psi_m^2 \leq 2\eta \left(\eta_m - \eta_{m+1} + \frac{q}{2}(e_{m+1} + e_m) \right).$$

Summiert man diese Ungleichung über m , so erhält man schließlich

$$\sum_{m=0}^{\infty} \psi_m^2 \leq 2\eta \left(\eta_0 + q \sum_{m=0}^{\infty} e_m \right) \leq 2\eta \left(\eta_0 + qe_0 \sum_{m=0}^{\infty} \left(\frac{1}{2} \right)^m \right) \leq 2\eta(\eta_0 + 2qe_0),$$

Tabelle 8.10: Broyden Verfahren

| m | x_1^m | x_2^m | $t(\mathbf{x}^m)$ |
|-----|--------------------------|---------------------------|-------------------|
| 0 | 0.5500000000000000e + 00 | -1.0000000000000000e + 00 | 1.62E + 00 |
| 1 | 4.5590000000000002e - 01 | 2.6932500000000003e - 01 | 7.47e - 01 |
| 2 | 1.543981538116330e + 00 | 1.501304221151375e + 00 | 1.91e + 02 |
| 3 | -1.025397030327286e - 01 | 1.001738027581978e + 00 | 4.68e + 00 |
| 4 | 1.155418539286380e + 00 | -4.749815441829222e - 01 | 8.31e + 00 |
| 5 | 3.598779288346202e - 01 | 2.769192432357461e - 01 | 7.53e - 01 |
| 6 | 3.059435797761894e - 01 | 5.139683793110996e - 01 | 2.27e - 01 |
| 7 | 3.168501618408199e - 01 | 6.429750746281397e - 01 | 2.70e - 02 |
| 8 | 3.428381587159949e - 01 | 6.199750293193117e - 01 | 3.16e - 02 |
| 9 | 3.593134438436643e - 01 | 6.428635461736542e - 01 | 8.76e - 03 |
| 10 | 3.823317197048445e - 01 | 6.654991281621061e - 01 | 2.82e - 05 |
| 11 | 3.819657986428381e - 01 | 6.638948931722465e - 01 | 1.76e - 07 |
| 12 | 3.820297255805328e - 01 | 6.639992049514174e - 01 | 7.24e - 11 |
| 13 | 3.820312595808607e - 01 | 6.640012640813945e - 01 | 1.85e - 15 |
| 14 | 3.820312680827534e - 01 | 6.640012741848831e - 01 | 2.19e - 20 |
| 15 | 3.820312681116868e - 01 | 6.640012742200018e - 01 | 8.07e - 27 |
| 16 | 3.820312681116693e - 01 | 6.640012742199805e - 01 | 1.23e - 32 |

und hieraus folgt

$$\lim_{m \rightarrow \infty} \Psi_m = 0. \quad \blacksquare$$

Beispiel 8.57. Wir betrachten erneut Beispiel 8.42.

$$f(\mathbf{x}) = \begin{pmatrix} (x_1^2 + x_2^2)(1 + 0.8x_1 + 0.6x_2) - 1 \\ (x_1^2 + x_2^2)(1 - 0.6x_1 + 0.8x_2) - 2x_1 \end{pmatrix} = \mathbf{0}.$$

Mit dem Startwert $\mathbf{x}^0 = (0.55, -1)^T$ und $\mathbf{B} = \mathbf{E}_2$ erhält man die Näherungen in Tabelle 8.10. $t(\mathbf{x}^m)$ bezeichnet wie vorher $t(\mathbf{x}^m) = \|f(\mathbf{x}^m)\|_2^2$. \square

Bemerkung 8.58. Die superlineare Konvergenz gilt, ohne dass die hinreichende Bedingung

$$\lim_{m \rightarrow \infty} \|\mathbf{B}_m - f'(\bar{\mathbf{x}})\| = 0,$$

aus Abschnitt 8.5 gilt, denn sei

$$f : \mathbb{R}^2 \rightarrow \mathbb{R}^2, \quad f(\mathbf{x}) = \begin{pmatrix} x_1 \\ x_2 + x_2^3 \end{pmatrix}.$$

Wir wählen die Anfangswerte

$$\mathbf{x}^0 := \begin{pmatrix} 0 \\ \varepsilon \end{pmatrix} \quad \text{und} \quad \mathbf{B}_0 = \begin{pmatrix} 1 + \delta & 0 \\ 0 & 1 \end{pmatrix}.$$

Dann zeigt man leicht, dass

$$\mathbf{B}_m = \begin{pmatrix} 1 + \delta & 0 \\ 0 & * \end{pmatrix} \quad \text{für alle } m \in \mathbb{N}$$

gilt, d.h. $\mathbf{B}_m \not\rightarrow f'(\mathbf{0})$. \square

8.7 Homotopieverfahren

Wir haben bereits erwähnt, dass in vielen Fällen der Einzugsbereich einer Lösung $\bar{\mathbf{x}}$ eines nichtlinearen Gleichungssystems $f(\mathbf{x}) = \mathbf{0}$ für das Newton Verfahren (oder ein anderes numerisches Verfahren) sehr klein ist. Die folgende Vorgehensweise gibt häufig eine Möglichkeit, in den Einzugsbereich vorzudringen:

Wir führen zu den Variablen x_1, \dots, x_n eine zusätzliche Variable λ künstlich ein und betten das zu behandelnde Problem

$$f(\mathbf{x}) = \mathbf{0} \quad (8.63)$$

mit $f : \mathbb{R}^n \supset D \rightarrow \mathbb{R}^n$ in ein Nullstellenproblem

$$h(\mathbf{x}, \lambda) = \mathbf{0} \quad (8.64)$$

mit $h : \mathbb{R}^{n+1} \supset D \times [0, 1] \rightarrow \mathbb{R}^n$ ein, wobei $h(\mathbf{x}, 1) = f(\mathbf{x})$ für alle $\mathbf{x} \in D$ gilt und für $h(\mathbf{x}, 0) = \mathbf{0}$ eine Lösung $\mathbf{x}(0)$ bekannt ist (bzw. $h(\mathbf{x}, 0) = \mathbf{0}$ leicht lösbar ist).

Ist z.B. \mathbf{x}^0 eine Näherungslösung für $\bar{\mathbf{x}}$, so kann man

$$h(\mathbf{x}, \lambda) = f(\mathbf{x}) - (1 - \lambda)f(\mathbf{x}^0)$$

wählen.

Enthält die Lösungsmenge von (8.64) eine Kurve γ mit $\mathbf{x}(0) \in \gamma$, so kann man versuchen, ausgehend von $\mathbf{x}(0)$ mit einem numerischen Verfahren der Kurve γ zu folgen. Erreicht man dabei auf der Kurve einen Punkt $(\bar{\mathbf{x}}, 1)$, so gilt $h(\bar{\mathbf{x}}, 1) = f(\bar{\mathbf{x}}) = \mathbf{0}$, und man hat eine Lösung von (8.63) gefunden.

Definition 8.59. *Jede Methode, die das Problem (8.63) in ein Problem (8.64) einbettet und Lösungskurven von (8.64) verfolgt, um Lösungen von (8.63) zu approximieren, heißt **Homotopieverfahren** oder **Einbettungsverfahren** oder **Fortsetzungsverfahren** oder (in der Ingenieurliteratur) **Inkremental-Lastmethode**.*

In den Anwendungen hat der Parameter λ häufig eine anschauliche Bedeutung: Es sei \mathbf{x} der Vektor der Verschiebungen der Knoten in einem belasteten Stabwerk gegenüber dem unbelasteten Zustand, und es sei λ ein Maß für eine von außen aufgebraachte Last (vgl. Abbildung 8.8).

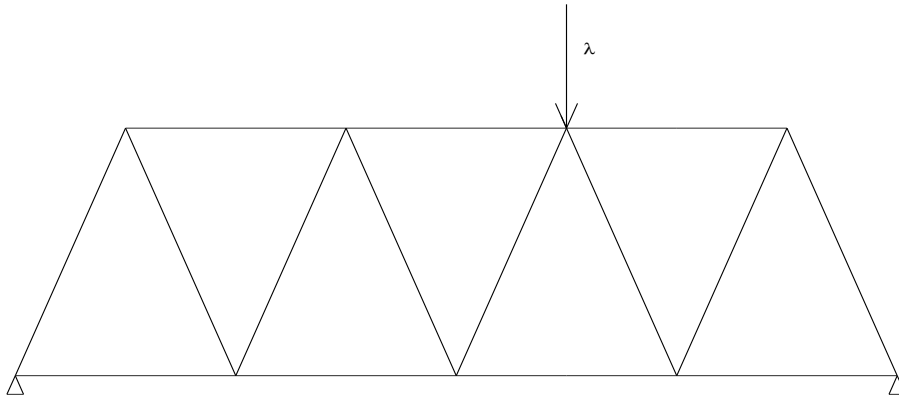


Abbildung 8.8 : Stabwerk

Der Zusammenhang zwischen der Last und dem Verschiebungsvektor werde beschrieben durch das Gleichungssystem

$$h(\mathbf{x}, \lambda) = \mathbf{0}$$

Gesucht ist die Verschiebung $\bar{\mathbf{x}}$ für eine gegebene positive Last $\bar{\lambda}$.

Erhöht man die Last λ ausgehend von $\lambda = 0$, so werden sich die zugehörigen Verschiebungen $\mathbf{x}(\lambda)$ stetig ändern, solange in dem Stabwerk keine Stäbe geknickt werden oder durchschlagen. Die folgende Vorgehensweise zur Ermittlung von $\bar{\mathbf{x}} = \mathbf{x}(\bar{\lambda})$ ist daher naheliegend:

Man zerlege das Intervall $[0, \bar{\lambda}]$ in

$$0 = \lambda_0 < \lambda_1 < \lambda_2 < \dots < \lambda_m := \bar{\lambda}.$$

Ist für $i = 1, \dots, m$ bereits eine gute Näherung \mathbf{x}^{i-1} für eine Lösung von $h(\mathbf{x}, \lambda_{i-1}) = \mathbf{0}$ bekannt (für $i = 1$ wählt man natürlich $\mathbf{x}^0 = \mathbf{x}(0) = \mathbf{0}$), so bestimme man mit dem Newton Verfahren (oder einem anderen Verfahren zur Lösung von nichtlinearen Systemen) für das Gleichungssystem $h(\mathbf{x}, \lambda_i) = \mathbf{0}$ mit dem Startwert \mathbf{x}^{i-1} eine gute Näherung für eine Lösung von $h(\mathbf{x}, \lambda_i) = \mathbf{0}$. Im m -ten Schritt erhält man damit eine Näherung für eine Lösung von $h(\mathbf{x}, \bar{\lambda}) = \mathbf{0}$.

Beispiel 8.60. Das von Mises Stabwerk der Abbildung 8.9, links, wird beschrieben durch

$$h(x, \lambda) = x^3 - \alpha_0^2 x + \lambda = 0.$$

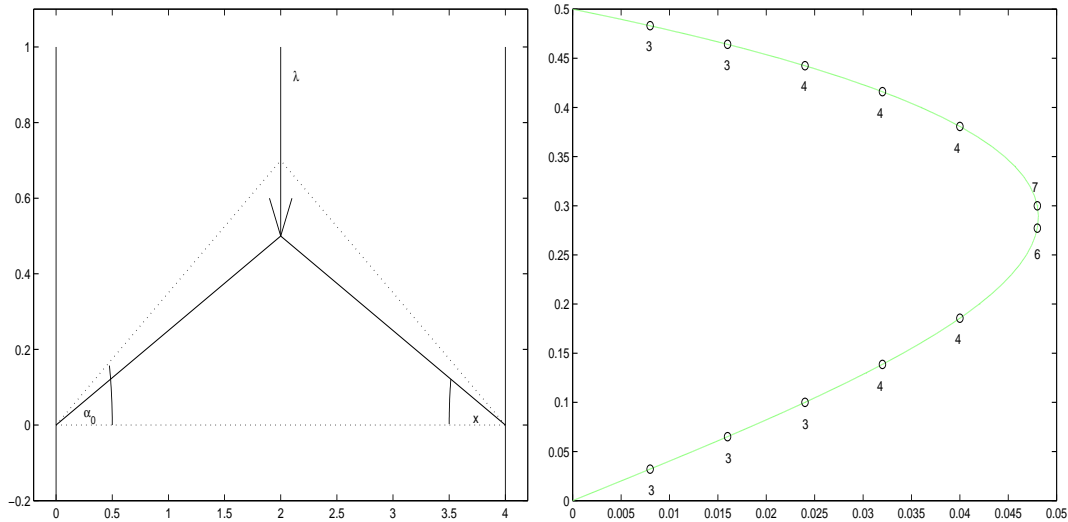


Abbildung 8.9 : Von Mises Stabwerk

Dabei bezeichnet x den Auslenkungswinkel, α_0 den Auslenkungswinkel im unbelasteten Zustand, und der Parameter λ ist proportional zur aufgebrachten Last.

Für $\alpha_0 = 0.5$ ist die kritische Last, bei der das Stabwerk durchschlägt,

$$\lambda^* = \frac{1}{12\sqrt{3}} \approx 0.0481.$$

Mit der Schrittweite $\lambda_i - \lambda_{i-1} = 0.008$, $i = 1, \dots, 6$, und dem Startwert $\mathbf{x}(0) = 0.5$ benötigt man die in Abbildung 8.9, rechts, angegebenen Anzahlen von Newton Iterationen, um $\mathbf{x}^i \approx \mathbf{x}(0.08i)$ mit einem relativen Fehler von 10^{-5} ausgehend von \mathbf{x}^{i-1} zu bestimmen.

Genauso kann man ausgehend von $\mathbf{x}(0) = \mathbf{0}$ die (instabilen) kleinen positiven Lösungen von $h(\mathbf{x}, \lambda) = \mathbf{0}$ berechnen. \square

Sind bereits Näherungen $\mathbf{x}^{i-1} \approx \mathbf{x}(\lambda_{i-1})$ und $\mathbf{x}^i \approx \mathbf{x}(\lambda_i)$ bekannt, so kann mit Hilfe der Sekante eine i.A. bessere Startnäherung als \mathbf{x}^i für $\mathbf{x}(\lambda_{i+1})$ erhalten:

$$\mathbf{x}(\lambda_{i+1}) \approx \mathbf{x}^i + \frac{\lambda_{i+1} - \lambda_i}{\lambda_i - \lambda_{i-1}} (\mathbf{x}^i - \mathbf{x}^{i-1}).$$

Hiermit benötigt man in dem Beispiel 8.60. auf dem oberen Ast 2, 2, 2, 2, 2 und 5 Iterationen und auf dem unteren Ast 2, 1, 1, 2, 2 und 5 Iterationen.

Ist man nicht an der Lösungskurve $\mathbf{x}(\lambda)$ von $h(\mathbf{x}, \lambda) = \mathbf{0}$ interessiert, sondern nur an der Lösung für einen festen Parameter $\bar{\lambda}$, z.B. an der von $f(\mathbf{x}) = h(\mathbf{x}, 1) = \mathbf{0}$, so wird man nicht mit einer vorgegebenen Zerlegung des Parameterintervalls rechnen, sondern die Zerlegung der Lösungskurve anpassen.

Der folgende Algorithmus zur Bestimmung einer Lösung von $h(\mathbf{x}, 1) = \mathbf{0}$ enthält eine einfache **Schrittweitensteuerung** :

Gegeben seien die beiden letzten Parameterwerte λ_0 und λ_1 ($\lambda_0 < \lambda_1$), Näherungen \mathbf{x}^0 und \mathbf{x}^1 für die zugehörigen Lösungen $\mathbf{x}(\lambda_0)$ und $\mathbf{x}(\lambda_1)$ von $\mathbf{h}(\mathbf{x}, \lambda) = \mathbf{0}$, eine Testschrittweite τ und $\delta_1 := |\det \frac{\partial}{\partial \mathbf{x}} \mathbf{h}(\mathbf{x}^1, \lambda_1)|$.

Für die Steuerung des Algorithmus seien ferner die folgenden Größen vor Beginn der Rechnung bereitgestellt:

- τ_0 , die minimal zugelassene Schrittweite,
- τ_∞ , die maximal zugelassene Schrittweite,
- κ_0 , die minimale Länge eines Newton Schritts,
- κ_∞ , die maximale Länge eines Newton Schritts,
- μ_∞ , der maximale Defekt in einem Newton Schritt,
- α , der Vergrößerungsfaktor für die Schrittlänge ($\alpha > 1$),
- β , der Verkleinerungsfaktor für die Schrittlänge ($0 < \beta < 1$),
- m_∞ , die Maximalzahl der Newton Schritte,
- δ_0 , der minimale Faktor bei der Veränderung der Determinante.

Wir beschreiben den Algorithmus mit einem Pseudocode:

```
repeat
  boole:=true;
  {Mit boole merken wir uns, ob die Testschrittweite  $\tau$ 
  richtig gewählt war}
   $\sigma := \lambda_1 + \tau$ ;
   $m := 1$ ;
  { $m$  zählt die Newton Schritte für den Parameter  $\sigma$ }
   $\delta_2 := \delta_1$ ;
  { $\delta_1$  wird gesichert, falls  $\tau$  zu groß gewählt war}
   $\mathbf{y} := \mathbf{x}^1 + \tau(\mathbf{x}^1 - \mathbf{x}^0)/(\lambda_1 - \lambda_0)$ ;
  {Schätzung einer Näherung für  $\mathbf{x}(\sigma)$  mit einem
  Sekantenschritt}
  repeat
     $\mathbf{z} := \mathbf{y} - \frac{\partial}{\partial \mathbf{x}} \mathbf{h}(\mathbf{y}, \sigma)^{-1} \mathbf{h}(\mathbf{y}, \sigma)$ ;
    {Newton Schritt für  $\mathbf{h}(\mathbf{x}, \sigma) = \mathbf{0}$  mit dem
    Startwert  $\mathbf{y}$ }
     $\delta_z := |\det \frac{\partial}{\partial \mathbf{x}} \mathbf{h}(\mathbf{z}, \sigma)|$ ;
```

```

m := m + 1;
If ( $\|\mathbf{y} - \mathbf{z}\| > \kappa_\infty$ ) or ( $\delta_z < \delta_0\delta_1$ ) or ( $m > m_\infty$ ) then
    {Die Schrittweite wird verkürzt,
     wenn ein Newton Schritt zu lang ist,
     wenn sich die Funktionaldeterminante zu stark ändert
     oder wenn zu viele Newton Schritte benötigt werden}
begin
     $\tau := \beta\tau$ ;
    boole:=false;
    {Wurde die Schrittweite verkürzt, so ist es nicht
     sinnvoll, sie sofort wieder zu verlängern}
     $\delta_1 := \delta_2$ ;
    if  $\tau < \tau_0$  then write ( $\lambda = 1$  wurde nicht erreicht) STOP;
    {Unterschreitet die Schrittweite  $\tau_0$ , so ist der
     Algorithmus stecken geblieben}
     $\mathbf{y} := \mathbf{x}^1 + \tau(\mathbf{x}^1 - \mathbf{x}^0)/(\lambda_1 - \lambda_0)$ ;
    {Schätzung einer Näherung für  $\mathbf{x}(\sigma)$  mit einem
     Sekantenschritt für die verkürzte Schrittweite}
end
else
begin
     $\mathbf{y} := \mathbf{z}$ ;
     $\delta_1 := \delta_z$ 
    {Startwert und Funktionaldeterminante werden für
     einen neuen Newton Schritt bereitgestellt}
end
until ( $\|\mathbf{y} - \mathbf{z}\| < \kappa_0$ ) and ( $\|\mathbf{h}(\mathbf{z}, \sigma)\| < \mu_\infty$ );
    {Das Newton Verfahren für  $\mathbf{h}(\mathbf{x}, \sigma) = \mathbf{0}$  war erfolgreich,
     falls der letzte Newton Schritt kleiner als  $\kappa_0$  war
     und die Norm von  $\mathbf{h}(\mathbf{z}, \sigma)$  kleiner als  $\mu_\infty$  ist}
If  $\sigma < 1$  then
begin
     $\lambda_0 = \lambda_1$ ;  $\lambda_1 = \sigma$ ;
     $\mathbf{x}^0 = \mathbf{x}^1$ ;  $\mathbf{x}^1 = \mathbf{z}$ ;
     $\delta_1 = \delta_z$ 
end;

```

Tabelle 8.11: Homotopieverfahren

| m | x_1^m | x_2^m | λ_m | Newton Schritte |
|-----|-----------|------------|-------------|-----------------|
| 0 | 0.5500000 | -1.0000000 | 0.0000000 | |
| 1 | 0.5475886 | -1.0052771 | 0.0001221 | 5 |
| 2 | 0.5464193 | -1.0078491 | 0.0001526 | 4 |
| 3 | 0.5458752 | -1.0090490 | 0.0001602 | 4 |
| 4 | 0.5456517 | -1.0095422 | 0.0001621 | 4 |
| 5 | 0.5454854 | -1.0099096 | 0.0001631 | 4 |
| 6 | 0.5453518 | -1.0102050 | 0.0001636 | 4 |
| 7 | 0.5452979 | -1.0103242 | 0.0001637 | 4 |
| 8 | 0.5452591 | -1.0104098 | 0.0001637 | 3 |

if boole then $\tau := \min(\alpha\tau, \tau_\infty)$;

{Wenn im letzten Parameterschritt keine Verkürzungen
erforderlich waren, wird die Schrittweite verlängert}

$\tau := \min(\tau, 1 - \sigma)$;

until $\tau = 0$;

Beispiel 8.61. Für das Beispiel 8.42.

$$\mathbf{f}(\mathbf{x}) = \begin{pmatrix} (x_1^2 + x_2^2)(1 + 0.8x_1 + 0.6x_2) - 1 \\ (x_1^2 + x_2^2)(1 - 0.6x_1 + 0.8x_2) - 2x_1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

erhält man mit der Homotopie

$$\mathbf{h}(\mathbf{x}, \lambda) = \mathbf{f}(\mathbf{x}) - (1 - \lambda)\mathbf{f}(\mathbf{x}^0)$$

und $\mathbf{x}^0 = (0.55, -1)^T$ die Punkte auf der Lösungskurve $\mathbf{x}(\lambda)$ in Tabelle 8.11

Das Verfahren wird gestoppt, da $\frac{\partial}{\partial \mathbf{x}} \mathbf{h}(\mathbf{x}^1, \lambda_1)$ (numerisch) singular wird, der Satz über implizite Funktionen also nicht mehr angewendet werden kann, um $\mathbf{h}(\mathbf{x}, \lambda) = \mathbf{0}$ nach \mathbf{x} aufzulösen. \square

Ist $(\mathbf{x}^1, \lambda_1)$ ein Verzweigungspunkt, so hat man grundsätzliche Schwierigkeiten bei der Kurvenverfolgung zu erwarten. Ist $(\mathbf{x}^1, \lambda_1)$ (wie in unserem Beispiel 8.61.) ein Umkehrpunkt, so kann man durch Umparametrisierung der Lösungsmenge, d.h. indem man den Parameter λ nicht mehr auszeichnet, den Umkehrpunkt überwinden.

Wir beschreiben, wie man ausgehend von einem Punkt $\mathbf{u}^0 := (\mathbf{x}, \lambda)$ mit $\mathbf{h}(\mathbf{u}^0) = \mathbf{0}$ und $\text{Rang } \mathbf{h}'(\mathbf{u}^0) = n$ mit der Suchrichtung \mathbf{v} , ($\|\mathbf{v}\|_2 = 1$) (z.B. der Sekantenrichtung) und der Schrittlänge τ einen neuen Punkt \mathbf{u}^1 mit $\mathbf{h}(\mathbf{u}^1) = \mathbf{0}$ konstruiert:

Es sei $\mathbf{X} := \{\mathbf{z} \in \mathbb{R}^{n+1} : (\mathbf{z} - \mathbf{u}^0)^T \mathbf{v} = \tau\}$ der affine Unterraum des \mathbb{R}^{n+1} , der senkrecht auf \mathbf{v} steht und von \mathbf{u}^0 den Abstand τ hat. Wir bestimmen dann

Tabelle 8.12: modifiziertes Homotopieverfahren

| m | x_1^m | x_2^m | λ_m | Newton Schritte |
|-----|-----------|------------|-------------|-----------------|
| 0 | 0.5500000 | -1.0000000 | 0.0000000 | |
| 1 | 0.5824505 | -0.9315693 | -0.0078125 | 6 |
| 2 | 0.5858292 | -0.9246311 | -0.0091438 | 3 |
| 3 | 0.5959513 | -0.9039555 | -0.0135687 | 3 |
| 4 | 0.6261830 | -0.8425103 | -0.0298007 | 3 |
| 5 | 0.7113176 | -0.6576314 | -0.0859551 | 3 |
| 6 | 0.8054892 | 0.0532877 | 0.0179935 | 5 |
| 7 | 0.5731184 | 0.5100110 | 0.5904037 | 5 |
| 8 | 0.3820313 | 0.6640013 | 1.0000000 | 5 |

$\mathbf{u}^1 \in \mathbb{R}^{n+1}$ als Lösung von

$$\mathbf{g}(\mathbf{u}) := \begin{pmatrix} \mathbf{h}(\mathbf{u}) \\ (\mathbf{u} - \mathbf{u}^0)^T \mathbf{v} - \tau \end{pmatrix} = \mathbf{0}$$

mit dem Newton Verfahren für \mathbf{g} .

Es ist klar, dass bei geeigneter Wahl von \mathbf{v} (z.B. der Sekantenrichtung zu den letzten beiden berechneten Kurvenpunkten \mathbf{u}^0 und \mathbf{u}^{-1} , falls diese genügend nahe beieinander liegen) und genügend kleinem $\tau > 0$ der Raum \mathbf{X} die Lösungsmenge von $\mathbf{h}(\mathbf{u}) = \mathbf{0}$ schneidet, das Gleichungssystem $\mathbf{g}(\mathbf{u}) = \mathbf{0}$ also lösbar ist.

Ist $\mathbf{v} = (0, \dots, 0, 1)^T \in \mathbb{R}^{n+1}$, so entspricht der obige Schritt dem Vorgehen in Beispiel 8.61.

Beispiel 8.62. Mit diesem Verfahren kann der Lösungsast in dem Beispiel 8.61. verfolgt werden. Man erhält die Punkte auf dem Lösungsast in Tabelle 8.12. \square

8.8 Nichtlineare Ausgleichsprobleme

Wir betrachten das **nichtlineare Ausgleichsproblem**:

Es seien $f : \mathbb{R}^n \supset D \rightarrow \mathbb{R}^m$ und $\mathbf{y} \in \mathbb{R}^m$ gegeben mit $m \geq n$. Man bestimme $\mathbf{x} \in \mathbb{R}^n$, so dass

$$\|f(\mathbf{x}) - \mathbf{y}\|_2 \tag{8.65}$$

minimal wird.

Es sei \mathbf{x}^0 eine Näherung für ein Minimum. Wir linearisieren f an der Stelle \mathbf{x}^0 , ersetzen also $f(\mathbf{x})$ durch

$$\tilde{f}(\mathbf{x}) = f(\mathbf{x}^0) + \mathbf{J}(\mathbf{x} - \mathbf{x}^0),$$

wobei $\mathbf{J} := f'(\mathbf{x}^0)$ die Jacobi Matrix von f an der Stelle \mathbf{x}^0 bezeichnet.

Setzt man \tilde{f} für f ein, so erhält man das lineare Ausgleichsproblem

Bestimme $\boldsymbol{\xi} \in \mathbb{R}^n$, so dass

$$\|\mathbf{J}\boldsymbol{\xi} - \mathbf{r}\|_2 \tag{8.66}$$

minimal ist. Dabei bezeichnet $\mathbf{r} := \mathbf{y} - f(\mathbf{x}^0)$ das Residuum im Punkt \mathbf{x}^0 .

Dann ist $\mathbf{x}^1 := \mathbf{x}^0 + \boldsymbol{\xi}$ i.A. keine bessere Näherung für eine Lösung des nichtlinearen Ausgleichsproblems als \mathbf{x}^0 , aber es gilt Satz 8.63.

Satz 8.63. *Es seien $\mathbf{x}^0 \in \mathbb{R}^n$ mit $\text{Rang } \mathbf{J} = n$, $\mathbf{r} \neq \mathbf{0}$ und $\boldsymbol{\xi} \in \mathbb{R}^n$ die Lösung des linearen Ausgleichsproblems*

$$\|\mathbf{J}\boldsymbol{\xi} - \mathbf{r}\|_2 = \min!. \tag{8.67}$$

Ist $\boldsymbol{\xi} \neq \mathbf{0}$, so existiert $\bar{t} > 0$, so dass

$$\phi(t) := \|\mathbf{y} - f(\mathbf{x}^0 + t\boldsymbol{\xi})\|_2^2, \quad t \in (0, \bar{t})$$

streng monoton fällt, d.h. $\boldsymbol{\xi}$ ist eine Abstiegsrichtung für $\|f(\mathbf{x}) - \mathbf{y}\|_2$ in \mathbf{x}^0 .

Beweis: ϕ ist stetig differenzierbar und

$$\begin{aligned} \phi'(0) &= \left. \frac{d}{dt} \left\{ (\mathbf{y} - f(\mathbf{x}^0 + t\boldsymbol{\xi}))^T (\mathbf{y} - f(\mathbf{x}^0 + t\boldsymbol{\xi})) \right\} \right|_{t=0} \\ &= -2(\mathbf{J}\boldsymbol{\xi})^T (\mathbf{y} - f(\mathbf{x}^0)). \end{aligned}$$

Da $\boldsymbol{\xi}$ das lineare Ausgleichsproblem löst, ist $\boldsymbol{\xi}$ auch Lösung der zugehörigen Normalgleichungen

$$\mathbf{J}^T \mathbf{J} \boldsymbol{\xi} = \mathbf{J}^T \mathbf{r},$$

d.h.

$$\phi'(0) = -2\boldsymbol{\xi}^T \mathbf{J}^T \mathbf{r} = -2\boldsymbol{\xi}^T \mathbf{J}^T \mathbf{J} \boldsymbol{\xi} = -2\|\mathbf{J}\boldsymbol{\xi}\|_2^2 < 0. \quad \blacksquare$$

Tabelle 8.13: Gauß Newton Verfahren

| m | x_0^m | y_0^m | r^m |
|-----|--------------------|-------------------|-------------------|
| 0 | 10.000000000000000 | 0.000000000000000 | 5.000000000000000 |
| 1 | 6.14876214033877 | 2.15725426511744 | 8.04036508586118 |
| 2 | 3.90540362558127 | 3.86325842696234 | 1.23334135895049 |
| 3 | 3.30542254249197 | 3.27561123666543 | 1.00518648399760 |
| 4 | 2.01445622380190 | 1.88326627386490 | 2.37615193167457 |
| 5 | 1.28012769560372 | 1.20757631375102 | 3.63420894517193 |
| 6 | 1.05146620440749 | 1.00051076514201 | 3.94851196633348 |
| 7 | 1.03357181946508 | 0.98450202132343 | 3.97248530883056 |
| 8 | 1.03339414968563 | 0.98435519042950 | 3.97270001636676 |
| 9 | 1.03339325417899 | 0.98435449459529 | 3.97270097982488 |
| 10 | 1.03339324956150 | 0.98435449093551 | 3.97270098483765 |
| 11 | 1.03339324955561 | 0.98435449093087 | 3.97270098484402 |
| 12 | 1.03339324955506 | 0.98435449093043 | 3.97270098484462 |
| 13 | 1.03339324955506 | 0.98435449093043 | 3.97270098484462 |

Dieses Ergebnis legt nun nahe, in Richtung der Lösung des linearisierten Ausgleichsproblems fortzuschreiten und ähnlich wie beim gedämpften Newton Verfahren die Schrittweite durch fortgesetzte Halbierung so klein zu wählen, dass die Norm in (8.65) verkleinert wird. Wiederholt man diesen Schritt, so erhält man das **Gauß Newton Verfahren**, genauer das **gedämpfte Gauß Newton Verfahren**.

Algorithmus 8.64. (Gedämpftes Gauß Newton Verfahren)

For $i = 1, 2, \dots$ until convergence do

Berechne die Lösung ξ^i des linearen Ausgleichsproblems

$$\|f'(\mathbf{x}^{i-1})\xi - (\mathbf{y} - f(\mathbf{x}^{i-1}))\|_2 = \min!$$

Bestimme das minimale $\ell \in \mathbb{N}_0$ mit

$$\|\mathbf{y} - \mathbf{f}(\mathbf{x}^{i-1} + 2^{-\ell}\xi^i)\|_2^2 < \|\mathbf{y} - \mathbf{f}(\mathbf{x}^{i-1})\|_2^2$$

Setze $\mathbf{x}^i := \mathbf{x}^{i-1} + 2^{-\ell}\xi^i$.

Ohne Dämpfung, d.h. für $\ell = 0$ in jedem Schritt, wurde dieses Verfahren von Gauß benutzt, um Bahnen von Planetoiden vorherzusagen.

Beispiel 8.65. Gegeben seien die Punkte

| | | | | | | | | | |
|-------|---|-----|-----|-----|-----|-----|-----|-----|---|
| x_j | 1 | 1.5 | 2 | 2.5 | 3 | 3.5 | 4 | 4.5 | 5 |
| y_j | 5 | 4.9 | 4.8 | 4.7 | 4.4 | 4.1 | 3.7 | 2.9 | 1 |

in der Ebene. Man bestimme durch Ausgleich einen Kreis

$$K = \{(x, y)^T : \sqrt{(x - x_0)^2 + (y - y_0)^2} = r\},$$

der diesen Punkten möglichst nahe ist, d.h. mit

$$f_i(x_0, y_0, r) := \sqrt{(x_i - x_0)^2 + (y_i - y_0)^2} - r, \quad i = 1, \dots, 9$$

löse man das nichtlineare Ausgleichsproblem

$$\|(f_i(x_0, y_0, r))_{i=1, \dots, 9}\|_2 = \min!$$

Mit dem Gauß Newton Verfahren und den (unsinnigen) Startwerten $x_0^0 := 10, y_0^0 = 0, r_0^0 := 5$ erhält man die Näherungen in Tabelle 8.13. \square

Das Gauß Newton Verfahren ist eine Liniensuchmethode. Zur Näherung \mathbf{x}^0 wird mit dem linearen Ausgleichsproblem (8.67) eine Abstiegsrichtung $\boldsymbol{\xi}$ für $\|\mathbf{y} - f(\mathbf{x}^0 + t\boldsymbol{\xi})\|_2$ berechnet und dann hierzu durch fortgesetzte Halbierung eine geeignete Suchlänge bestimmt, so dass die Norm in (8.65) verkleinert wird.

Neben diesen Liniensuchmethoden werden in der nichtlinearen Optimierung Vertrauensbereichsmethoden (engl. trust region methods) betrachtet, bei denen die Suchrichtung und Suchlänge simultan bestimmt werden. Für nichtlineare Ausgleichsprobleme haben sie die folgende Gestalt:

Zur Näherungslösung \mathbf{x}^0 und zum Radius $\Delta > 0$ des Vertrauensbereichs bestimme man $\mathbf{x} := \mathbf{x}^0 + \boldsymbol{\xi}$ mit

$$\|\boldsymbol{\xi}\|_2 \leq \Delta \quad \text{und} \quad \|\mathbf{J}\boldsymbol{\xi} - \mathbf{r}\|_2 = \min! \quad (8.68)$$

Ist dann

$$\|f(\mathbf{x}) - \mathbf{y}\|_2 < \|f(\mathbf{x}^0) - \mathbf{y}\|_2, \quad (8.69)$$

so war der Schritt erfolgreich. \mathbf{x} wird als neue Näherung akzeptiert, und der Radius Δ des Vertrauensbereichs vergrößert (z.B. verdoppelt), wenn die Abnahme in (8.69) sehr stark war. Ist (8.69) nicht erfüllt, wird der Schritt mit einem verkleinerten (z.B. halbierten) Δ wiederholt. Dieses sog. **Levenberg-Marquardt-Verfahren** wurde erstmals von Levenberg [71] und Marquardt [77] (allerdings mit einer anderen Motivation) zur Lösung von nichtlinearen Ausgleichsproblemen vorgeschlagen.

Die Kuhn-Tucker Bedingungen sagen, dass (8.68) äquivalent ist dem Problem

Bestimme $\boldsymbol{\xi}$ und $\lambda \geq 0$ mit

$$(\mathbf{J}^T \mathbf{J} + \lambda \mathbf{I}) \boldsymbol{\xi} = \mathbf{J}^T \mathbf{r}, \quad (8.70)$$

$$\lambda(\Delta - \|\boldsymbol{\xi}\|_2) = 0. \quad (8.71)$$

(8.69) sind die Normalgleichungen des linearen Ausgleichsproblems

$$\left\| \begin{pmatrix} \mathbf{J} \\ \sqrt{\lambda} \mathbf{I} \end{pmatrix} \boldsymbol{\xi} - \begin{pmatrix} \mathbf{r} \\ \mathbf{0} \end{pmatrix} \right\|_2 = \min! \quad (8.72)$$

so dass man das System (8.70) lösen kann, ohne die Matrix $\mathbf{J}^T \mathbf{J}$ zu berechnen. Hierzu verwendet man die QR-Zerlegung

$$\begin{pmatrix} \mathbf{J} \\ \sqrt{\lambda} \mathbf{I} \end{pmatrix} = \mathbf{Q}_\lambda \begin{pmatrix} \mathbf{R}_\lambda \\ \mathbf{O} \end{pmatrix}. \quad (8.73)$$

Um die Bedingung (8.71) (wenigstens näherungsweise) zu erfüllen oder wenn ein Schritt des Vertrauensbereichs Verfahrens verworfen wird, muss das Ausgleichsproblem (8.72) für verschiedene λ gelöst werden. Dies kann auf folgende Weise effizient durchgeführt werden:

Wir bestimmen zunächst mit Householder Transformationen die QR-Zerlegung

$$\mathbf{J} = \mathbf{Q} \begin{pmatrix} \mathbf{R} \\ \mathbf{O} \end{pmatrix}.$$

Dann gilt

$$\begin{pmatrix} \mathbf{R} \\ \mathbf{O} \\ \sqrt{\lambda} \mathbf{I} \end{pmatrix} = \begin{pmatrix} \mathbf{Q}^T & \\ & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{J} \\ \sqrt{\lambda} \mathbf{I} \end{pmatrix}. \quad (8.74)$$

Die Matrix auf der linken Seite ist schon fast eine obere Dreiecksmatrix. Man muss nur noch die Elemente der Diagonale von $\sqrt{\lambda} \mathbf{I}$ annullieren. Dies kann man mit Givens Rotationen ausführen:

Wir bezeichnen die ersten n Zeilen der linken Seite von (8.74) immer mit \mathbf{R} und die letzten n -Zeilen (die zunächst mit $\sqrt{\lambda} \mathbf{I}$ besetzt sind) mit \mathbf{N} . Durch Kombination der n -ten Zeile von \mathbf{R} und der n -ten Zeile von \mathbf{N} kann man das n -te Diagonalelement von \mathbf{N} eliminieren. Hat man bereits die Zeilen $n, n-1, \dots, i+1$ von \mathbf{N} behandelt, so kombiniert man die Zeilen i von \mathbf{R} und von \mathbf{N} und erzeugt eine 0 in der Position (i, i) von \mathbf{N} . Hierdurch werden die Elemente an den Stellen $(i, i+1), \dots, (i, n)$ in

\mathbf{N} aufgefüllt. Man kann sie aber nach einander durch Kombination der i -ten Zeile von \mathbf{N} mit den Zeilen $i + 1, i + 2, \dots, n$ von \mathbf{R} annullieren.

Damit kann man für jedes λ ausgehend von (8.74) durch $n(n + 1)/2$ Givens Rotationen die QR-Zerlegung (8.73) bestimmen. Dies kann insbesondere im Fall $n \ll m$ sehr viel Arbeit ersparen.

Ausgleichsprobleme sind häufig sehr schlecht skaliert. Es kommt vor, dass einige der zu berechnenden x_j die Größenordnung 10^4 haben und andere 10^{-6} . In diesem Fall kann das Levenberg-Marquardt Verfahren ein sehr schlechtes Verhalten haben. Einige Möglichkeit, der schlechten Skalierung zu begegnen, besteht darin, statt der kugelförmigen Vertrauensbereiche ellipsoidförmige zu verwenden. Der k -te Schritt des Verfahrens erhält dann die Gestalt: Bestimme $\boldsymbol{\xi}$, so dass

$$\|\mathbf{D}_k \boldsymbol{\xi}\|_2 \leq \Delta_k \quad \text{und} \quad \|\mathbf{J}_k \boldsymbol{\xi} - \mathbf{r}_k\|_2 = \min! \quad (8.75)$$

wobei \mathbf{D}_k eine Diagonalmatrix mit positiven Diagonalelementen bezeichnet. In den Kuhn-Tucker Bedingungen ist dann (8.70) durch

$$(\mathbf{J}_k^T \mathbf{J}_k + \lambda \mathbf{D}_k) \boldsymbol{\xi} = \mathbf{J}_k^T \mathbf{r}_k \quad (8.76)$$

zu ersetzen, und dies ist äquivalent dem linearen Ausgleichsproblem

$$\left\| \begin{pmatrix} \mathbf{J}_k \\ \sqrt{\lambda} \mathbf{D}_k \end{pmatrix} \boldsymbol{\xi} - \begin{pmatrix} \mathbf{r}_k \\ \mathbf{0} \end{pmatrix} \right\|_2 = \min! \quad (8.77)$$

Die Diagonalelemente von \mathbf{D}_k können dabei an die Größenordnungen der Komponenten der Näherungslösung von Schritt zu Schritt angepasst werden. An der Technik, die QR-Zerlegung der Koeffizientenmatrix von (8.77) für verschiedene λ aus der von \mathbf{J}_k mit Givens Rotationen zu berechnen, ändert sich nichts.

Literaturverzeichnis

- [1] *M. Abramowitz, I.A. Stegun (eds.): Handbook of Mathematical Functions.* Dover Publications, New York, 1971
- [2] *E.L. Allgower, K. Georg: Numerical Continuation Methods.* Springer, Berlin, 1990
- [3] *G. Ammar, W.B. Gragg: Superfast solution of real positive definite Toeplitz systems.* SIAM J. Matrix Anal. Appl. 9, 61 – 76 (1988)
- [4] *E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, D. Sorensen: LAPACK Users' Guide.* Third Edition, SIAM, Philadelphia, 2000
- [5] *N. Anderson, A. Björck: A new high order method of regula falsi type for computing a root of a equation.* BIT 13, 253 – 264 (1973)
- [6] *Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, H. van der Vorst (eds.), Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide.* SIAM, Philadelphia, 2000
- [7] *R.E. Bank: PLTMG: A Software Package for Solving Elliptic Partial Differential Equations.* Users' Guide 7.0. SIAM, Philadelphia, 1994
- [8] *F.L. Bauer, C.F. Fike: Norms and exclusion theorems.* Numer. Math. 2, 123 – 144 (1960)
- [9] *A. Björck: Least squares methods.* In P.G. Ciarlet, J.L. Lions (eds.), *Handbook of Numerical Analysis.* Vol. I, pp. 465 – 652, North Holland, Amsterdam, 1990
- [10] *A. Björck: Numerical Methods for Least Squares Problems.* SIAM, Philadelphia, 1996

- [11] *A. Björk, V. Pereyra*: Solution of Vandermonde systems of equations. *Math. Comp.* 24,893 – 903 (1970)
- [12] *L.S. Blackford, J. Choi, A. Cleary, E. D’Azevedo, J. Demmel, I. Dhillon, J. Dongarra, S. Hammarling, G. Henry, A. Petitet, K. Stanley, D. Walker, R.C. Whaley*: ScaLAPACK User’s Guide. SIAM, Philadelphia, 1997
- [13] *D. Braess*: Nonlinear Approximation Theory. Springer, Berlin, 1986
- [14] *R. Bulirsch*: Bemerkungen zur Romberg Integration. *Numer. Math.* 6, 6 – 16 (1964)
- [15] *J.R. Bunch, C.P. Nielsen, D.S. Sorensen*: Rank-one modification of the symmetric eigenproblem. *Numer. Math.* 31, 31 – 48 (1978)
- [16] *F. Chaitin – Chatelin, V. Fraysse*: Lectures on Finite Precision Computations. SIAM, Philadelphia, 1996
- [17] *F. Chatelin*: Eigenvalues of Matrices. Wiley, Chichester, 1993
- [18] *R. Clint Whaley, A. Petitet, J.J. Dongarra*: Automated empirical optimization of software and the ATLAS project. <http://www.netlib.org/atlas/>
- [19] *T.F. Coleman, C.F. Van Loan*: Handbook of Matrix Computations. SIAM, Philadelphia, 1988
- [20] *L. Collatz*: Eigenwertaufgaben mit technischen Anwendungen. Akademische Verlagsgesellschaft, Leipzig, 1963
- [21] *J.J.M. Cuppen*: A divide and conquer method for the symmetric tridiagonal eigenproblem. *Numer.Math.* 36, 177 – 195 (1981)
- [22] *C. de Boor*: A Practical Guide to Splines. Springer, New York, 1978
- [23] *J.W. Demmel*: Applied Numerical Linear Algebra. SIAM, Philadelphia, 1997
- [24] *J.E. Dennis, Jr., R.B. Schnabel*: Numerical Methods for Unconstrained Optimization and Nonlinear Equations. SIAM, Philadelphia, 1996
- [25] *P. Deuffhard, A. Hohmann*: Numerische Mathematik. Eine algorithmisch orientierte Einführung. de Gruyter, Berlin, 1991
- [26] *A New $O(n^2)$ Algorithm for the Symmetric Tridiagonal Eigenvalue/Eigenvector Problem*. Ph.D. Thesis, University of California, Berkeley, 1997

- [27] *J. Dongarra, J. Du Croz, I. Duff, S. Hammarling*: A set of level 3 basic linear algebra subroutines. *ACM Trans. Math. Software* 16, 1 – 17 (1990)
- [28] *J. Dongarra, J. Du Croz, S. Hammarling, R.J. Hanson*: An extended set of FORTRAN basic linear algebra subroutines. *ACM Trans. Math. Software* 14, 1 – 17 (1988)
- [29] *J.J. Dongarra, I.S. Duff, D.C. Sorensen, H.A. van der Vorst*: Solving Linear Systems on Vector and Shared Memory Computers. SIAM, Philadelphia, 1991
- [30] *J.J. Dongarra, I.S. Duff, D.C. Sorensen, H.A. van der Vorst*: Numerical Linear Algebra for High-Performance Computers. SIAM, Philadelphia, 1998
- [31] *J. Dongarra, D.S. Sorensen*: A fully parallel algorithm for the symmetric eigenproblem. *SIAM J. Sci. Stat. Comput.* 8, 139 – 154 (1987)
- [32] *K. Dowd*: High Performance Computing. O'Reilly & Associates, Sewastopol, 1993
- [33] *M. Dowell, P. Jarrat*: A modified regula falsi method for computing the root of an equation. *BIT* 11, 168 – 174 (1971)
- [34] *M. Dowell, P. Jarrat*: The “Pegasus” method for computing the root of an equation. *BIT* 12, 503 – 508 (1972)
- [35] *J. Durbin*: The fitting of time series models. *Rev. Inst. Int. Stat.* 28, 233 – 243 (1960)
- [36] *L. Eldén*: Algorithms for the regularization of ill-conditioned least squares problems. *BIT* 17, 134 – 145 (1977)
- [37] *G. Engeln-Müllges, F. Reutter*: Numerik-Algorithmen. Entscheidungshilfe zur Auswahl und Nutzung. 8. Auflage, VDI Verlag, Düsseldorf, 1996
- [38] *H.W. Engl*: Integralgleichungen. Springer, Wien, 1997
- [39] *K. Fernando, B. Parlett*: Accurate singular values and differential qd algorithms. *Numer.Math.* 67, 191 – 229 (1994)
- [40] *J.G.F. Francis*: The QR transformation. A unitary analogue to the LR transformation. Part 1. *Computer J.* 4, 256 – 272 (1961)

- [41] *J.G.F. Francis*: The QR transformation. A unitary analogue to the LR transformation. Part 2 Computer J. 4, 332 – 345 (1961)
- [42] *K. Frühauf, J. Ludewig, H. Sandmayr*: Software-Prüfung. Eine Anleitung zum Test und zur Inspektion. Teubner, Stuttgart, 1991
- [43] *F.R. Gantmacher*: Matrizen-theorie. Springer, Berlin, 1986
- [44] *W.M. Gentleman*: Least squares computations by Givens transformations without square roots. J. Inst. Maths. Applic. 12, 329 – 336 (1973)
- [45] *S.A. Gerschgorin*: Über die Abgrenzung der Eigenwerte einer Matrix. Izvestia Akademii Nauk SSSR, Mathematics and Natural Sciences 6, 749 – 754 (1931)
- [46] *P.E. Gill, W. Murray, M.H. Wright*: Practical Optimization. Academic Press, London, 1981
- [47] *W. Givens*: Computation of plane unitary rotations transforming a general matrix to triangular form. SIAM J. Appl. Math. 6, 26 – 50 (1958)
- [48] *G.H. Golub*: Numerical methods for solving linear least squares problems. Numer. Math. 7, 206 – 216 (1965)
- [49] *G.H. Golub, W. Kahan*: Calculating the singular values and pseudo-inverse of a matrix. SIAM J. Num. Anal. Ser. B 2, 205 – 224 (1965)
- [50] *G.H. Golub, C. Reinsch*: Singular value decomposition and least squares problems. Numer. Math. 14, 403 – 420 (1970)
- [51] *G.H. Golub, C.F. Van Loan*: Matrix Computations. 3rd ed., The John Hopkins University Press, Baltimore, 1996
- [52] *M. Gu, S.C. Eisenstat*: A divide-and-conquer algorithm for the symmetric tridiagonal eigenproblem. SIAM J. Matr. Anal. Appl. 16, 172 – 191 (1995)
- [53] *S. Hammarling*: A note on modifications of the Givens plane rotation. J. Inst. Maths. Applic. 13, 215 – 218 (1974)
- [54] *P.C. Hansen*: Rank-Deficient and Discrete Ill-Posed Problems: Numerical Aspects of Linear Inversion. SIAM, Philadelphia, 1998
- [55] *P.C. Hansen*: The L-curve and its use in the numerical treatment of inverse problems. In P. Johnston (ed): Computational Inverse Problems in Electrocardiography. WIT Press, Southampton 2000

- [56] *N.J. Higham*: Accuracy and Stability of Numerical Algorithms. SIAM, Philadelphia, 1996
- [57] *I.C.F. Ipsen*: Computing an eigenvector with inverse iteration. SIAM Review 39, 254 – 291 (1997)
- [58] *C.G.J. Jacobi*: Über ein leichtes Verfahren, die in der Theorie der Sekularstörungen vorkommenden Gleichungen numerisch aufzulösen. Crelle Journal für die reine und angewandte Mathematik 30, 51 – 94 (1846)
- [59] *C.G.J. Jacobi*: Über eine elementare Transformation eines in Bezug jedes von zwei Variablen-Systemen linearen und homogenen Ausdrucks. Crelle Journal für die reine und angewandte Mathematik 53, 265 – 270 (1857) (posthum)
- [60] *D. Kahaner, C. Moler, S. Nash*: Numerical Methods and Software. Prentice Hall, Englewood Cliffs, 1989
- [61] *C.T. Kelley*: Iterative Methods for Linear and Nonlinear Equations. SIAM, Philadelphia, 1995
- [62] *A. Kielbasinski, H. Schwetlick*: Numerische Lineare Algebra. VEB Deutscher Verlag der Wissenschaften, Berlin, 1988
- [63] *R.F. King*: An improved Pegasus-method for root finding. BIT 13, 423 – 427 (1973)
- [64] *N. Köckler*: Numerical Methods and Scientific Computing Using Software Libraries for Problem Solving. Clarendon Press, Oxford, 1994
- [65] *A.R. Krommer, C.W. Ueberhuber*: Computational Integration. SIAM, Philadelphia, 1998
- [66] *A.S. Kronrod*: Integration with control of accuracy. Soviet Physics Dokl. 9, 17 – 19 (1964)
- [67] *A.S. Kronrod*: Nodes and weights of quadrature formulas. Consultants Bureau, New York, 1965
- [68] *V.N. Kublanowskaya*: On some algorithms for the solution of the complete eigenvalue problem. USSR Comp. Math. Phys. 3, 637 – 657 (1961)
- [69] *C.L. Lawson, R.J. Hanson*: Solving Least Squares Problems. Prentice – Hall, Englewood Cliffs, 1974

- [70] *C. Lawson, R. Hanson, D. Kincaid, F. Krogh*: Basic linear algebra subprograms for FORTRAN usage. *ACM Trans. Math. Software* 5, 308 – 323 (1979)
- [71] *K. Levenberg*: A method for the solution of certain non-linear problems in least squares. *Quart. Appl. Math.* 2, 164 – 168 (1944)
- [72] *N. Levinson*: The Weiner RMS error criterion in filter design and prediction. *J. Math. Phys.* 25, 261 – 278 (1947)
- [73] *A.K. Louis*: Inverse und schlecht gestellte Probleme. Teubner, Stuttgart, 1989
- [74] *A.K. Louis, P. Maaß und K. Rieder*: Wavelets. Teubner, Stuttgart, 1994
- [75] *W. Mackens, H. Voß*: Mathematik I für Studierende der Ingenieurwissenschaften. HECO-Verlag, Alsdorf, 1993
- [76] *W. Mackens, H. Voß*: Aufgaben und Lösungen zur Mathematik I für Studierende der Ingenieurwissenschaften. HECO-Verlag, Alsdorf, 1994
- [77] *D.W. Marquardt*: An algorithm for least squares estimation of non-linear parameters. *SIAM J.* 11, 431 – 441 (1963)
- [78] *G. Meinardus, G. März*: Praktische Mathematik I. Bibliographisches Institut, Mannheim, 1979
- [79] *C.B. Moler, G.W. Stewart*: An algorithm for generalized matrix eigenvalue problems. *SIAM J. Numer. Anal.* 10, 241 – 256 (1973)
- [80] *J. Moré, S.J. Wright*: Optimization Software Guide. SIAM, Philadelphia, 1993
- [81] *J. Nocedal, S.J. Wright*: Numerical Optimization. Springer, New York, 1999
- [82] *J.M. Ortega, W. Rheinboldt*: Iterative Solution of Nonlinear Equations in Several Variables. Academic Press, New York – London, 1970
- [83] *B.N. Parlett*: The Symmetric Eigenvalue Problem. SIAM, Classics in Applied Mathematics 20, Philadelphia, 1998
- [84] *D.L. Philips*: A technique for the numerical solution of certain integral equations of the first kind. *J. Assoc. Comput. Mach.* 9, 84 – 97 (1962)
- [85] *R. Piessens, E. de Doncker-Kapenga, C.W. Überhuber, D.K. Kahaner*: QUADPACK. A Subroutine Package for Automatic Integration. Springer Verlag, Berlin, 1983

- [86] *W.H. Press, B.P. Lannery, S.A. Teukolsky, W.T. Vetterling*: Numerical Recipes in FORTRAN – The Art of Scientific Computing. 2nd edition. Cambridge University Press, Cambridge, 1992
- [87] *W.H. Press, B.P. Lannery, S.A. Teukolsky, W.T. Vetterling*: Numerical Recipes in C – The Art of Scientific Computing. 2nd edition. Cambridge University Press, Cambridge, 1992
- [88] *W.H. Press, B.P. Lannery, S.A. Teukolsky, W.T. Vetterling*: Numerical Recipes in FORTRAN – Example Book. 2nd edition. Cambridge University Press, 1992
- [89] *W.H. Press, B.P. Lannery, S.A. Teukolsky, W.T. Vetterling*: Numerical Recipes in C – Example Book. 2nd edition. Cambridge University Press, 1992
- [90] *W. Romberg*: Vereinfachte numerische Integration. Det Kongelige Norske Videnskabers Forhandling, Bind 28 (7), 1955
- [91] *Y. Saad*: Numerical Methods for Large Eigenvalue Problems. Manchester University Press, Manchester, 1992
- [92] *A. Schönhage*: On the quadratic convergence of the Jacobi process. Numer.Math. 6, 410 – 412 (1964)
- [93] *H.R. Schwarz*: Numerische Mathematik. Teubner, Stuttgart, 1988
- [94] *H. Schwetlick*: Numerische Lösung nichtlinearer Gleichungen. Oldenbourg, München – Wien, 1979
- [95] *H. Schwetlick, H. Kretzschmar*: Numerische Verfahren für Naturwissenschaftler und Ingenieure. Fachbuchverlag Leipzig, Leipzig, 1991
- [96] *P. Spelucci, W. Törnig*: Eigenwertberechnung in den Ingenieurwissenschaften. Teubner, Stuttgart, 1985
- [97] *G.W. Stewart*: Introduction to Matrix Computations. Academic Press, New York, 1973.
- [98] *G.W. Stewart, J.-G. Sun*: Matrix Perturbation Theory. Academic Press, New York, 1990.
- [99] *G.W. Stewart*: Matrix Algorithms. Vol. 1: Basic Decompositions. SIAM, Philadelphia, 1998

- [100] *G.W. Stewart*: Matrix Algorithms. Vol. 2: Eigensystems. SIAM, Philadelphia, 2001
- [101] *J. Stoer*: Einführung in die Numerische Mathematik I. 6. Auflage. Springer Verlag, Berlin, 1993
- [102] *J. Stoer, R. Bulirsch*: Einführung in die Numerische Mathematik II. 3. Auflage, Springer, Berlin, 1990
- [103] *J. Stoer, R. Bulirsch*: Introduction to Numerical Analysis. Springer, New York, 1980
- [104] *G. Strang*: Introduction to Linear Algebra. Wellesley – Cambridge Press, Wellesley, 1993
- [105] *G. Strang, T. Nguyen*: Wavelets and Filter Banks. Wellesley – Cambridge Press, Wellesley, 1996
- [106] *A. Tarantola*: Inverse Problem Theory. Elsevier, 1987
- [107] *A.N. Tichonov*: Solution of incorrectly formulated problems and the regularization method. Soviet. Math. Dokl. 4, 1035 – 1038 (1963). Englische Übersetzung von Dokl. Akad. Nauk. SSSR 151, 501 – 504 (1963)
- [108] *F. Tisseur, J. Dongarra*: A parallel divide and conquer algorithm for the symmetric eigenvalue problem on distributed memory architectures. SIAM J. Sci. Comput. 20, 2223 – 2236 (1999)
- [109] *W. Törnig, P. Spelucci*: Numerische Mathematik für Ingenieure und Physiker. Band 1: Numerische Methoden der Algebra, 2. Auflage, Springer, Berlin, 1988
- [110] *W. Törnig, P. Spelucci*: Numerische Mathematik für Ingenieure und Physiker. Band 2: Numerische Methoden der Analysis, 2. Auflage, Springer, Berlin, 1990
- [111] *J.F. Traub*: Iterative Methods for the Solution of Equations. 2nd Ed., Prentice Hall, Englewood Cliffs, 1981
- [112] *L.N. Trefethen, D. Bau, III*: Numerical Linear Algebra. SIAM, Philadelphia, 1997
- [113] *C. Überhuber*: Computer Numerik 1. Springer, Berlin, 1995
- [114] *C. Überhuber*: Computer Numerik 2. Springer, Berlin, 1995

- [115] *C.F. Van Loan*: Computational Frameworks for the Fast Fourier Transform. SIAM, Philadelphia, 1992
- [116] *D.S. Watkins*: Understanding the QR-algorithm. SIAM Review 24, 427 – 440 (1982)
- [117] *D.S. Watkins*: Fundamentals of Matrix Computations. 2nd Edition. Wiley, New York, 2002
- [118] *H. Werner*: Praktische Mathematik I. 3. Auflage, Springer Verlag, Berlin, 1982
- [119] *H. Werner, R. Schaback*: Praktische Mathematik II. Springer Verlag, Berlin, 1972
- [120] *H. Wielandt*: Beiträge zur mathematischen Behandlung komplexer Eigenwertprobleme, Teil V: Bestimmung höherer Eigenwerte durch gebrochene Iteration. Bericht B 44/J/37, Aerodynamische Versuchsanstalt Göttingen, 1944
- [121] *J.H. Wilkinson*: The Algebraic Eigenvalue Problem. Clarendon Press, Oxford, 1965.
- [122] *J.H. Wilkinson*: Note on matrices with a very ill-conditioned eigenproblem. Numer. Math. 19, 176 – 178 (1972)
- [123] *R. Zurmühl*: Matrizen und ihre technischen Anwendungen. Springer Verlag, Berlin, 1964
- [124] *R. Zurmühl, S. Falk*: Matrizen und ihre Anwendungen für Ingenieure, Physiker und Angewandte Mathematiker. Teil 1: Grundlagen, 6. vollst. neu bearb. Auflage, Springer, Berlin, 1992
- [125] *R. Zurmühl, S. Falk*: Matrizen und ihre Anwendungen für Ingenieure, Physiker und Angewandte Mathematiker. Teil 2: Numerische Methoden, 5. überarb. und erw. Auflage, Springer, Berlin, 1986

Index

- Ähnlichkeitstransformation, 160
- ähnliche Matrizen, 160
- a posteriori Abschätzung, 250
- a priori Abschätzung, 250
- abgeschlossene Newton–Cotes Formeln, 47
- abstoßender Fixpunkt, 252
- Abweichung von Normalität, 163
- Aitken Lemma, 14
- algebraische Vielfachheit, 160
- allgemeine Eigenwertaufgabe, 201
- Anderson Björck King Verfahren, 268
- anziehender Fixpunkt, 252
- Aufdatierungsformel, 287
- Aufdatierungsfunktion, 287
- Ausgleichsproblem
 - lineares, 130
 - nichtlineares, 302
- B-Splines, 38
- Bandmatrix, 108
- Bauer, Fike, Satz von, 166
- Bernoullische Zahlen, 58
- Bernstein Polynom, 41
- Bezierkurve, 41
- Bezierpolygon, 41
- Bezierpunkte, 41
- Bisektion, 262
- Bisektionsmethode, 220
- BLAS, 102
- BLAS1, 103
- BLAS2, 104
- BLAS3, 105
- Broyden Rang-1-Verfahren, 288
- Bulirsch Folge, 60
- Casteljau Algorithmus, 43
- Cauchy, Verschachtelungssatz von, 212
- charakteristisches Polynom, 160, 202
- Chebyshev Knoten, 20
- Chebyshev Polynom, 20
- Cholesky Zerlegung, 98
- CLAPACK, 129
- Clenshaw–Curtis Formel, 49
- Collatz, Quotienteneinschließungssatz von, 214
- Dachfunktion, 28
- Datenfehler, 1
- Deflation durch Ähnlichkeitstransformation, 180
- diskrete Fourier Transformation, 122
- dividierte Differenzen, 16
- dominanter Eigenwert, 172
- dqds Verfahren, 240
- Durbin, Algorithmus von, 126
- effektive Pseudoinverse, 155
- Effizienz, 269
- Eigenvektor, 159
- Eigenwert, 159
- Eigenwertaufgabe, 159
- Einbettungsverfahren, 297

- Eliminationsverfahren von Gauß, 94
 Erhard Schmidt Norm, 114
 Euler Maclaurinsche Summenformel, 56

 Faber, Satz von, 23
 fast Fourier transform, 123
 Fehlerkonstante, 53
 Fehlerordnung, 51, 89
 Festpunktdarstellung, 4
 FFT, 123
 Fixpunktproblem, 245
 Fixpunktsatz für kontrahierende Abbildungen, 248
 flops, 95
 Fortsetzungsverfahren, 297
 Fourier Transformation
 diskrete, 122
 inverse diskrete, 122
 Frobenius Norm, 114

 Gaußsches Eliminationsverfahren, 94
 Gauß Newton Verfahren, 304
 Gauß–Hermite Quadraturformel, 76
 Gauß–Laguerre Quadraturformel, 76
 Gauß Quadraturformeln, 63
 gedämpftes Gauß Newton Verfahren, 304
 gedämpftes Newton Verfahren, 276
 geometrische Vielfachheit, 160
 Gerschgorin, Satz von, 164
 Givens Reflexion, 137
 Givens Rotation, 137
 Gleitpunktdarstellung, 5
 Gleitpunktoperation, 6
 Gram–Schmidt Verfahren
 klassisches, 132
 modifiziertes, 132

 Hauptuntermatrix, 94
 Hermite Birkhoff Interpolationsproblem, 9
 Hermite Interpolation, 8
 Hermite Polynome, 76
 Hessenberg Matrix, 191
 Hilbert Matrix, 154
 Homotopieverfahren, 297
 Hornerzahl, 269
 Hotelling Deflation, 179
 Householder Matrix, 133

 Illinois Verfahren, 264
 Inkremental-Lastmethode, 297
 Interpolationsproblem, 8
 inverse diskrete Fourier Transformation, 122
 inverse Iteration, 176

 Jacobi Rotation, 137
 Jacobi Verfahren, 232
 Jordansche Normalform, 161

 Kantorowitsch, Satz von, 275
 Keplersche Fassregel, 47
 King Verfahren, 268
 klassisches Gram–Schmidt Verfahren, 132
 klassisches Jacobi Verfahren, 234
 Knoten, 8
 Kondition, 110, 116, 152
 Kondition eines Eigenwerts, 170
 kongruent, 218
 kontrahierend, 247
 Kontraktionskonstante, 247
 Kontraktionssatz, 248
 Kontrollpolygon, 41
 Kontrollpunkt, 41

- Konvergenzordnung, 256, 260
- Kronrod Formel, 83
- Krylov, Bogoliubov, Satz von, 208
- kubischer Hermite Spline, 26
- kubischer Spline, 26
- Kugelbedingung, 251

- L-Kurven Methode, 157
- Löwner, Satz von, 230
- Lagrange Interpolation, 8
- Lagrangesche Interpolationsformel, 11
- Lagrangesche Interpolationsproblem, 10
- Laguerre Polynome, 76
- LAPACK, 129
- LAPACK++, 129
- LAPACK90, 129
- LDL^T Zerlegung, 97
- Legendre Polynom, 70
- Level 1 BLAS, 103
- Level 2 BLAS, 104
- Level 3 BLAS, 105
- Levenberg-Marquardt-Verfahren, 306
- Levinson, Algorithmus von, 127
- lineare Konvergenz, 256
- lineares Ausgleichsproblem, 130
- lineares Interpolationsproblem, 8
- Linkseigenvektor, 160
- Lipschitz stetig, 247
- Lobatto Formel, 76
- LR Algorithmus, 240
- LR Zerlegung, 94

- Matrix
 - Hessenberg-, 191
 - Hilbert, 154
 - Householder, 133
 - kongruent, 218
 - normale, 162
 - persymmetrische, 124
 - Quasidreiecks-, 163
 - Signatur, 218
 - Toeplitz, 124
 - zirkulante, 128
- matrix pencil, 201
- Matrixnorm, 111
 - Erhard Schmidt Norm, 114
 - Frobenius Norm, 114
 - Schur Norm, 114
 - Spaltensummennorm, 113
 - Spektralnorm, 112
 - Zeilensummennorm, 112
- maxmin Prinzip von Courant-Fischer, 210
- Milne Regel, 47
- minmax Prinzip von Poincaré, 210
- Mittelpunktregel, 46
- modifiziertes Gram-Schmidt Verfahren, 132
- Monotoniesatz für Eigenwerte, 213
- Moore-Penrose Bedingungen, 150
- Moore-Penrose Inverse, 148

- natürlicher Spline, 31
- Neville und Aitken Algorithmus, 15
- Newton ähnliches Verfahren, 282
- Newton Verfahren, 254, 272
- Newtonsche Interpolationsformel, 16
- nichtlineares Ausgleichsproblem, 302
- nichtreduziert, 196
- normale Matrix, 162
- Normalgleichungen, 130
- not-a-knot Bedingung, 31
- Nullstellenproblem, 245

- obere Bandbreite, 108
- offene Newton–Cotes Formeln, 48
- orthogonale Iteration, 180
- Ostrowski, Satz von, 269

- Peano Kern, 52
- Pegasus Verfahren, 266
- pencil, 201
- persymmetrisch, 124
- Pivotsuch
 - vollständige, 97
- Pivotsuche
 - Spalten-, 96
- Polynominterpolation, 9
- Potenzmethode, 173
- Pseudoinverse, 148
 - effektive, 155
- Pseudonormallösung, 148

- Q-Konvergenzordnung, 256
- QR Zerlegung, 131
- quadratische Konvergenz, 256
- Quadraturformel
 - abgeschlossene Newton–Cotes, 47
 - Clenshaw–Curtis, 49
 - offene Newton–Cotes, 48
 - zusammengesetzte Newton–Cotes, 49
- Quadraturformel
 - von Gauß, 63
 - von Gauß–Hermite, 76
 - von Gauß–Laguerre, 76
 - von Kronrod, 83
 - von Lobatto, 76
 - von Radau, 75
 - von Romberg, 56
- Quasi-Newton Verfahren, 287
- Quasidreiecksmatrix, 163
- Quotienteneinschließungssatz von Col-
latz, 214
- QZ Algorithmus, 204

- R-Konvergenzordnung, 260
- Rückwärtseinsetzen, 94
- rückwärtsgenommener Differenzenquo-
tient, 88
- Radau Formel, 75
- rationale Interpolation, 9
- Rayleigh Quotient, 178, 208
- Rayleigh Quotienten Shift, 191
- Rayleighsches Prinzip, 210
- Rechteckregel, 46
- Rechtseigenvektor, 159
- reelle Schursche Normalform, 163
- Reflexion
 - Givens, 137
- reguläre Nullstelle, 272
- regulärer Pencil, 202
- regula falsi, 260, 264
- Regularisierung, 155
- Romberg Verfahren, 56, 60
- Rundungsfehler, 1

- Satz
 - von Bauer, Fike, 166
 - von Faber, 23
 - von Gerschgorin, 164
 - von Kantorowitsch, 275
 - von Krylov, Bogoliubov, 208
 - von Löwner, 230
 - von Ostrowski, 269
 - von Weyl, 211
- ScaLAPACK, 129
- schnelle Givens Transformation, 138

- Schrittweitensteuerung, 299
- Schur Norm, 114
- Schursche Normalform, 162
- Sekantenverfahren, 260
- Sherman Morrison Formel, 289
- Shift
- Rayleigh Quotienten, 191
 - Wilkinson, 198
- Signatur, 218
- Simpson Regel, 47
- singulärer Pencil, 202
- Singulärwerte, 141
- Singulärwertzerlegung, 140
- Spaltenpivotsuche, 96
- Spaltensummennorm, 113
- Spektralnorm, 112
- Spektrum, 159
- spezielle Eigenwertaufgabe, 159
- Spline, 26
- Spline Interpolation, 9
- Störungslemma, 115
- Stützstelle, 8
- Stetigkeitsmodul, 28
- submultiplikativ, 111
- summierte Newton–Cotes Formel, 49
- summierte Rechteckregel, 49
- summierte Simpson Regel, 49
- summierte Trapezregel, 49
- superlineare Konvergenz, 256
- SVD, 140
- Sylvester, Trägheitssatz von, 219
- Threshold Jacobi Verfahren, 234
- Tichonov Regularisierung, 155
- Toeplitz Matrix, 124
- Trägheitssatz von Sylvester, 219
- Trapezregel, 47
- trigonometrische Interpolation, 9
- trust region, 305
- untere Bandbreite, 108
- Unterraum Iteration, 180
- verbessertes Newton Verfahren, 258
- vereinfachtes Newton Verfahren, 259, 278
- Verfahrensfehler, 1
- Vergleichssatz für Eigenwerte, 211
- Verschachtelungssatz von Cauchy, 212
- Vertrauensbereich, 305
- Vielfachheit
- algebraische, 160
 - geometrische, 160
- Vielfachheit eines Knotens, 8
- vollständige Pivotsuche, 97
- von Mises Verfahren, 173
- Vorwärtseinsetzen, 94
- vorwärtsgenommener Differenzenquotient, 88
- Weyl, Satz von, 211
- Wielandt Deflation, 179
- Wilkinson Shift, 198, 216
- Wilkinson Verfahren, 243
- Yule–Walker System, 125
- Zeilensummennorm, 112
- zentraler Differenzenquotient, 88
- Zerlegung
- Cholesky, 98
 - LDL^T, 97
 - LR, 94
 - QR, 131
- zirkulante Matrix, 128

zusammengesetzte Newton–Cotes Formel, 49

zyklisches Jacobi Verfahren, 234