Numerik–Skripte 1994 Uni Marburg

B. Schmitt

15. März 1996

Ausdruck: Frank Wübbeling
Datum: 15. März 1996

Numerik I

K. Böhmer, B. Schmitt

Sommer-Semester 1994

Inhaltsverzeichnis

1 Einleitung **3**

1.1 Beispiel: Physikalisches Pendel 3

2 Polynom-Interpolation **5**

2.1 Problemstellung 5

2.2 Lagrange-Interpolation für Polynome 6

2.3 Newton-Interpolation 10

2.4 Entwicklung nach Orthogonalpolynomen, dreistufige Rekursionen 15

2.5 Vergleich der Verfahren 20

2.6 Interpolationsfehler, optimale Knoten 20

3 Numerische Integration und Differentiation **24**

3.1 Quadratur 25

3.2 Newton-Cotes-Formeln 26

3.3 Gauß-Quadratur 28

3.4 Adaptive Integration 31

3.5 Richardson-Extrapolation, Romberg-Verfahren 33

Praktische Durchführung 36

3.6 Numerische Differentiation 38

4 Lineare Gleichungssysteme **40**

4.1 Beispiele 40
4.2 Hilfsmittel 42
4.3 Der Gauß-Algorithmus 45
LR-Zerlegung 47
Pivotisierung 48
4.4 Fehlerausbreitung, Kondition 52
Schranken für $\kappa(A)$ 54
4.5 Iterationsverfahren für lineare Gleichungssysteme 55
Gesamt- und Einzelschrittverfahren 58
Relaxationsverfahren 61

5 Nichtlineare Gleichungssysteme **63**
5.1 Nullstellen einer skalaren Funktion 63
Newton-Verfahren 65
Sekantenverfahren, Regula falsi 67
Interpolationsverfahren höherer Ordnung 70
5.2 Newtonverfahren im $\mathbf{R}^n$ 70

6 Analyse der Rundungsfehler **74**
6.1 Zahldarstellung und Rundungsfehler 74
6.2 Rundungsfehleranalyse einiger Algorithmen 78
Approximationsverfahren 78
Gauß-Algorithmus 79
Fixpunkt-Iteration 80

1 Einleitung

Bei der Modellierung realer Situationen tauchen in den meisten Fällen mathematische Probleme auf, deren Lösung nur näherungsweise, mit Computereinsatz, möglich ist. Die Modellierung besteht aus mehreren Schritten, eventuell mit verschiedenen Alternativen:

Modellierungs-Schema:

Reale Situation, Beschreibung durch physikalische Gesetze, Experiment
↓
Entwicklung mathematischer Modelle, meist mit Vereinfachungen
↓
Mathematische Analyse, Existenz, Eindeutigkeit, qualitative Aussagen

↙	↘
Explizite Lösung (für Spezialfälle), Reihenentwicklungen, Eigenschaften ($\forall$ Lösungen,..)	Entwicklung numerischer Verfahren mit Fehleranalyse
↓	↓
Lösungsdiskussion, asymptotische Aussagen, Parameterabhängigkeit (Optimierung?)	Implementierung, Lösung vieler Einzelfälle, Simulation, Optimierung

Beispiel Pendel

$\varphi'' + \omega^2 \sin\varphi = 0$, ohne Reibung etc.

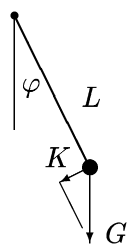
AWP eindeutig lösbar, RWP?

$\varphi'' + \omega^2 \varphi = 0$, 'Frequenz'

Approx. φ'', Fehler $\sim h^2$

Subprobleme, Gleichungssysteme

1.1 Beispiel: Physikalisches Pendel



Kräftegleichgewicht: Kraft $K = -G \sin\varphi$, Gewicht $G = mg$.

Newtonsches Gesetz: $mL\varphi''(t) = -G \sin\varphi$. Mit $\omega^2 := g/L$ folgt eine Gleichung für den Winkel $\varphi(t)$:

$$\varphi''(t) + \omega^2 \sin\varphi(t) = 0. \qquad (1.1.1)$$

Bei dieser Modellierung wurden Luftwiderstand, Reibung, etc. vernachlässigt. Die Gleichung (1.1.1) ist eine *Differentialgleichung* und beschreibt die Menge *aller* möglichen Bewegungen des Pendels. Einzelne Lösungen werden daraus ausgewählt durch zusätzliche Bedingungen. Beim

Anfangswertproblem (AWP) sind Start-Winkel und -Geschwindigkeit vorgegeben:

$$\varphi''(t) + \omega^2 \sin\varphi(t) = 0,\ t > 0, \quad \varphi(0) = \alpha,\ \varphi'(0) = \beta. \tag{1.1.2}$$

Randwertproblem (RWP) werden Start- und End-Winkel vorgegeben:

$$\varphi''(t) + \omega^2 \sin\varphi(t) = 0,\ 0 < t < T, \quad \varphi(0) = \alpha_0,\ \varphi(T) = \alpha_1, \tag{1.1.3}$$

z.B. $\varphi(0) = \varphi(T) = 0$, Schwingung mit Halb-Periode T.

$\Longrightarrow$ *Keines der Probleme (1.1.2), (1.1.3) ist explizit lösbar!* $\Longleftarrow$

Konsequenz: Approximationsverfahren für die Lösung sind erforderlich. Mögliche Ansätze:

- **mathematisch**: Für 'kleine' Winkel φ gilt $\sin\varphi \cong \varphi$. Statt (1.1.1) wird die Differentialgleichung

  $$\varphi''(t) + \omega^2\varphi(t) = 0 \quad \Rightarrow \quad \varphi(t) = a\cos\omega t + b\sin\omega t, \tag{1.1.4}$$

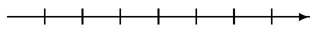
  mit bekannter allgemeiner Lösung verwendet. Damit ergibt sich beim

  AWP: $a = \alpha,\ b = \beta/\omega$, AWP eindeutig lösbar für beliebige α, β.

  RWP: Nichttriviale Lösungen zu $\alpha_0 = \alpha_1 = 0$ existieren nur für $\sin\omega T = 0$, dann aber unendlich viele, $a = 0,\ b \in \mathbf{R}$!

- **numerisch**: Die 'Differentiale' werden ersetzt durch endliche Differenzen. Analysis:

  $$\frac{d\varphi}{dt}(t) = \varphi'(t) \cong \frac{\varphi(t+h) - \varphi(t)}{h}, \quad \varphi''(t) \cong \frac{\varphi(t+h) - 2\varphi(t) + \varphi(t-h)}{h^2}.$$

  Dabei wird h 'klein', aber fest, gewählt, und Werte nur in endlich vielen Punkten $t_j = jh,\ j = 0, \dots, m$ mit $h = T/m$ verwendet: 

  Dies führt für die Näherungen $y_j \cong \varphi(t_j)$ auf

  $$0 = \varphi''(t_j) + \omega^2\varphi(t_j) \cong \frac{y_{j-1} - 2y_j + y_{j+1}}{h^2} + \omega^2 \sin y_j \stackrel{!}{=} 0.$$

  Mit diesem *Differenzenverfahren* wird, z.B., das RWP (1.1.3) approximiert durch das Problem

  $$\begin{cases} y_{j-1} - 2y_j + y_{j+1} + h^2\omega^2 \sin y_j = 0, & j = 1, \dots, m-1, \\ y_0 = \alpha_0,\ y_m = \alpha_1. \end{cases} \tag{1.1.5}$$

  Dies ist ein nichtlineares Gleichungssystem im $\mathbf{R}^{m-1}$. Im Zusammenhang mit dessen Lösung (in einem Computer-Programm) und den Beziehungen zum Ausgangsproblem (1.1.3) ergeben sich folgende Fragen:

  - Auflösung nichtlinearer Gleichungssysteme ($\to$ §5), dabei:

    Auflösung linearer Gleichungssysteme ($\to$ §4),
  - Wie genau sind die Näherungen $y_k \cong \varphi(t_k)$ theoretisch? ($\to$ Numerik IIB)

- Kann $\varphi(t)$ überall in $[0,T]$ approximiert werden? ($\rightarrow$ §2)
- Berechnung von $\sin y$? ($\rightarrow$ §2)
- Reelle Zahlen werden im Computer approximiert, wie wirken sich Rundungsfehler und ihre Fortpflanzung auf die Ergebnisse aus? ($\rightarrow$ §6).

Da überall in der Numerik mit Approximationen gearbeitet wird, spielt die Analyse von Fehlern und der Versuch, sie zu beherrschen, eine zentrale Rolle. Dabei gibt es zwei Zugänge:

- theoretische Analyse der Fehlerquellen und ihrer Auswirkungen, damit Konstruktion und Verwendung von Fehler*schätzungen* in den Verfahren.
- in Teilbereichen (nicht-/ lineare Algebra) sind exakte Fehler*einschließungen* auf Computern durchführbar durch sichere Arithmetiken (Programmbibliothek ACRITH, Sprache PASCAL-SC). Aufwendig, Einsatzbereich begrenzt.

Wird man in der Praxis mit einem speziellen mathematischen Problem konfrontiert, für das numerische Lösungen gesucht sind, sollte man außer den eigenen Kenntnissen das allgemein vorhandene Wissen nutzen. Dazu können Informationsquellen unterschiedlicher Art herangezogen werden.

- In klassischer Weise kann man sich in Bibliotheken (der Universität) durch nachlesen in Lehrbüchern und Zeitschriften informieren.
- Kurzreferate zu mathematischen Veröffentlichungen erscheinen schon lange gedruckt im 'Zentralblatt für Mathematik' und den 'Mathematical Reviews'. Mittlerweile gibt es diese Informationen als elektronische Literatur-Datenbanken (z.B. auf CD-ROM), in denen mit Computerunterstützung gesucht werden kann ($\rightarrow$ Bibliothek des Fachbereichs).
- Für viele numerische Standardprobleme gibt es fertige Lösungsalgorithmen als Unterprogramme in allgemeinen, kommerziellen Software-Bibliotheken (NAG,IMSL).
- Für speziellere Probleme kann man eventuell Software aus Bibliotheken im Internet (z.B. elib) finden.

Der Einsatz von Standard-Programmen erfordert aber im allgemeinen Kenntnisse aus der Numerik, um den Einsatzbereich der zur Verfügung stehenden Programme abgrenzen zu können.

2 Polynom-Interpolation

2.1 Problemstellung

In vielen Situationen sind Datenwerte für bestimmte Größen eines Parameters, z.B., zu verschiedenen Zeitpunkten, bekannt.

Beispiel 2.1.1

a) Umsatz (Gewinn) einer Firma vierteljährlich.

b) Kontostände einer Privatperson jeweils zum Monatsende.

c) Weg oder Geschwindigkeit eines Fahrzeuges von Minute zu Minute.

d) Gewicht eines Säuglings am Beginn jeder Woche.

e) Temperatur von Patienten, $3\times$ täglich.

f) Werte einer (komplizierten) Funktion für bestimmte Argumentwerte.

Hier erhebt sich die Frage, ob Aussagen über Zwischenwerte in den angegebenen Beispielen aufgrund der vorliegenden Information möglich und sinnvoll sind. Für die o.g. Beispiele a) bis f) ergibt sich folgendes Bild:

a) Zwischenwerte kaum sinnvoll.

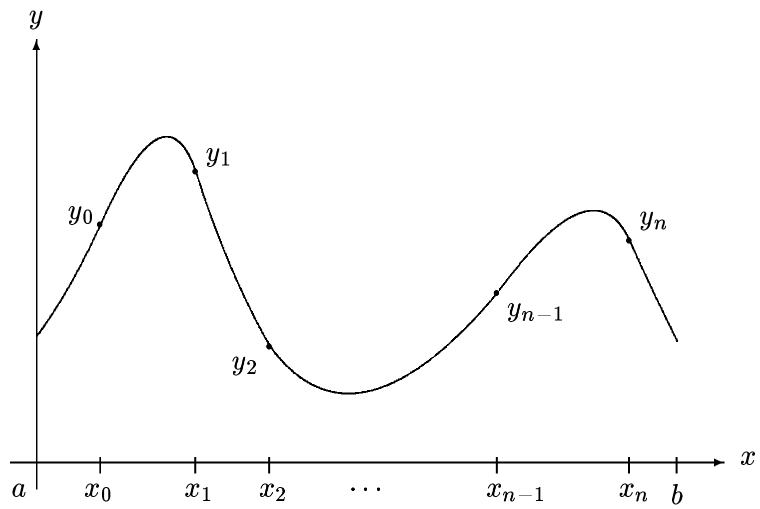
b) Zwischenwerte beinahe beliebig, hängen ab von den Konto-Konditionen.

c) Zwischenwerte abhängig von maximalen Beschleunigung und Bremswerten des Fahrzeugs.

d), e) Zwischenwerte hoffentlich (?) nahe bei Testwerten.

f) Zwischenwerte abhängig von höheren Ableitungen der Funktion.

Zur Berechnung von Zwischenwerten ist es sinnvoll, einfache Ersatzfunktionen zu definieren, die die vorgegebene Information in geeigneter Weise ausnützen. Wir werden uns hier darauf beschränken, als Ersatzfunktionen Polynome zu wählen, die in bestimmten Parameterwerten ('Stützstellen') vorgegebene Datenwerte annehmen, d.h., interpolieren.



2.2 Lagrange-Interpolation für Polynome

Definition 2.2.1 *Zum Gitter* $\{x_0, \ldots, x_n\}$ *mit*

$$a \leq x_0, \ldots, x_n \leq b, \quad x_i \neq x_j \text{ für } i \neq j, \tag{2.2.1}$$

und dem Datenvektor $y := (y_0, \dots, y_n)^{\mathsf{T}}$ *sei* p *ein Polynom mit*

$$p(x_i) = y_i, \quad i = 0, 1, \dots, n. \tag{2.2.2}$$

Dann heißt p Interpolationspolynom. *Ein Polynom, das (2.2.2) erfüllt, nennt man genauer auch* Lagrange-Interpolationspolynom. *Falls (2.2.2) ersetzt wird durch*

$$p^{(j)}(x_i) = y_{ij}, \quad j = 0, \dots, m_i - 1, \; i = 0, \dots, n, \tag{2.2.3}$$

heißt p Hermite-Interpolationspolynom.

Die Bedingungen (2.2.2) führen auf ein lineares Gleichungssystem für die Koeffizienten a_j eines Polynoms vom Grad n, wenn letzteres in der Form

$$p_n(x) = \sum_{j=0}^{n} a_j x^j, \quad \Pi_n := \{p_n : \; a_j \in \mathbf{R}, j = 0, \dots, n\}, \tag{2.2.4}$$

geschrieben wird. Es gibt aber auch eine direkte konstruktive Methode, das Lagrange-Interpolationspolynom zu bestimmen. Es gilt nämlich

Satz 2.2.2 *Zu beliebigen Wertepaaren* (x_i, y_i), $i = 0, \dots, n$, *mit paarweise verschiedenen Stützstellen* $x_i \neq x_j$ *(für* $i \neq j$*) existiert genau ein Interpolationspolynom vom Grad* n *zu (2.2.2). Es hat die Form*

$$p := p_y := \sum_{i=0}^{n} y_i L_i, \quad L_i(x) := \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j}. \tag{2.2.5}$$

p_y *hängt linear von* y *ab.*

Offensichtlich gilt

$$L_i \in \Pi_n, \quad L_i(x_j) = \delta_{ij} = \begin{cases} 1 & \text{für} \quad i = j, \\ 0 & \text{für} \quad i \neq j \end{cases}$$

Eines von mehreren Kriterien zur Beurteilung von numerischen Algorithmen ist der Rechenaufwand. Bei der Bestimmung des Aufwands werden sämtliche Operationen wie Additionen, Multiplikationen und Divisionen gezählt, d.h., alle Gleitpunktoperationen, englisch 'Floating Point Operations' = FLOPS. Diese Operationsanzahlen werden vielfach in Abhängigkeit von einem Dimensionsparameter, hier des Polynomgrads n aus (2.2.4), angegeben. Wenn die Differenzen $x - x_j$ getrennt berechnet werden, sind zur Auswertung von (2.2.5) jeweils $(n+1)^2 + n$ Aditionen, $n(n+1)$ Multiplikationen und $n(n+1)$ Divisionen nötig. Zur Vereinfachung betrachtet man meist nur den am stärksten mit n wachsen Anteil, hier die n^2-Terme, alle anderen werden als $\mathcal{O}(n)$-Ausdruck zusammengefaßt. Dies ergibt den

<u>Rechenaufwand</u> zur Auswertung von (2.2.5): $3n^2 + \mathcal{O}(n)$ Operationen.

Beweis zu Satz 2.2.2 Wegen der genannten Eigenschaften der L_i gilt für das Polynom p aus (2.2.5)

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

Zur Eindeutigkeit: wenn zwei Polynome $p, \bar{p} \in \Pi_n$ beide (2.2.2) erfüllen, so folgt

$$(p - \bar{p})(x_i) = 0, \quad i = 0, \dots, n.$$

Damit sind nach dem Fundamentalsatz der Algebra p und $\bar{p}$ identisch. ■

Als nachteilig bei der Darstellung (2.2.5) erweist sich der hohe Rechenaufwand, eine gewisse Rundungsfehleranfälligkeit, und der große Zusatzaufwand bei Auswertung in einer neuen Stelle x, bzw. bei der Hinzunahme neuer Interpolationspunkte $x_{n+1}, \dots$. Diese Nachteile lassen sich im wesentlichen vermeiden, wenn man für die L_i aus (2.2.5) eine andere Darstellung wählt. Mit

$$w_i^{(n)} := w_i := \frac{1}{\prod\limits_{\substack{j=0 \\ j \neq i}}^{n} (x_i - x_j)}, \quad w(x) := \prod_{j=0}^{n} (x - x_j) \tag{2.2.6}$$

ist

$$L_i(x) = \frac{w(x)}{x - x_i} w_i \quad \text{für } x \neq x_i.$$

Damit erhält man schließlich für das Polynom p aus (2.2.5) die Form

$$p_n(x) = w(x) \sum_{i=0}^{n} \frac{w_i}{x - x_i} y_i \quad \text{für } x \neq x_i.$$

Speziell ergibt sich mit der Eigenschaft

$$p_n(x) \equiv 1, \text{ wenn } y_i \equiv 1 \; \forall i$$

aus der letzten Gleichung die Relation

$$1 \equiv w(x) \sum_{i=0}^{n} \frac{w_i}{x - x_i}, \quad \text{also} \quad w(x) = \frac{1}{\sum\limits_{i=0}^{n} \frac{w_i}{x - x_i}} \quad \text{für } x \neq x_j \; \forall j. \tag{2.2.7}$$

Dies zeigt folgenden

Satz 2.2.3 *Das Lagrange-Interpolationspolynom ist darstellbar in der* baryzentrischen Form *(als gewichtetes Mittel)*

$$p_n(x) = \begin{cases} \dfrac{\sum\limits_{i=0}^{n} \dfrac{w_i}{x - x_i} y_i}{\sum\limits_{i=0}^{n} \dfrac{w_i}{x - x_i}} & \textit{für} \quad x \neq x_j \; \forall j, \\[2ex] y_j & \textit{wenn} \quad x = x_j \end{cases} \tag{2.2.8}$$

Zur Auswertung von (2.2.8) berechnet man zweckmäßigerweise zunächst die vom Punkt x und den y_i unabhängigen Gewichte w_i. Die Auswertung von (2.2.8) ist 'numerisch stabiler' ($\rightarrow$ §6) als (2.2.5).

Bei Hinzunahme einer Stützstelle ergibt sich der Übergang von den $w_i^{(n)}$ in (2.2.6) zu den ersten $n+1$ Koeffizienten $w_i^{(n+1)}$ folgendermaßen:

$$w_i^{(n+1)} := w_i^{(n)}/(x_i - x_{n+1}), \quad i = 0, 1, \ldots, n \tag{2.2.9}$$

Ferner wird (2.2.7) mit $n+1$ umgeformt

$$1 \equiv \sum_{i=0}^{n+1} w_i^{(n+1)} \prod_{\substack{j=0 \\ j \neq i}}^{n+1} (x - x_j) = x^{n+1} \sum_{i=0}^{n+1} w_i^{(n+1)} + \ldots \tag{2.2.10}$$

Durch Koeffizientenvergleich folgt also

$$\sum_{i=0}^{n+1} w_i^{(n+1)} = 0 \quad \text{für } n \geq 0. \tag{2.2.11}$$

Daraus kann auch noch $w_{n+1}^{(n+1)}$ bestimmt werden und es ergibt sich

Algorithmus 2.2.4 Rekursive Berechnung der Gewichtsfaktoren $w_i^{(n)}$ in (2.2.6):

$$\boxed{\begin{array}{l} w_0^{(0)} := 1; \\ \text{für } k := 1 \text{ bis } n: \\ \quad \text{für } i := 0 \text{ bis } k-1: \\ \qquad w_i^{(k)} := w_i^{(k-1)}/(x_i - x_k); \\ \quad w_k^{(k)} := -\sum_{i=0}^{k-1} w_i^{(k)}; \end{array}} \tag{2.2.12}$$

(Werner: Math. Comp. 43, 1984) Dieser Algorithmus (2.2.12) erfordert

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

Divisionen und doppelt soviel Additionen, also ist der

<u>**Rechenaufwand**</u> Baryzentrische Darstellung:

$$\begin{array}{rl} \frac{3}{2}n^2 + \mathcal{O}(n) & \text{Operationen zur Berechnung der } \{w_i^{(n)}\} \text{ in (2.2.12)}, \\ 5n + \mathcal{O}(1) & \text{Operationen zur Auswertung von } p(x) \text{ in (2.2.8).} \end{array}$$

Man beachte, daß im Dreieck-Schema (2.2.12) die $w_i^{(k)}$ die $w_i^{(k-1)}$ überschreiben dürfen, es wird also nur ein lineares Feld $[0, \ldots, n]$ benötigt.

Beispiel 2.2.5 a) Interpolation auf einem *äquidistanten Gitter*: Mit

$$x_i := x_0 + ih, \; i = 0, \ldots, n$$

erhält man

$$\frac{1}{w_i} = \prod_{\substack{j=0 \\ j \neq i}}^{n} (x_i - x_j) = h^n \prod_{\substack{j=0 \\ j \neq i}}^{n} (i - j) = h^n (-1)^{n-i} i!(n-i)! \,. \tag{2.2.13}$$

Dies ergibt

$$w_i = \frac{(-1)^n}{h^n n!} \cdot \frac{n!(-1)^i}{i!(n-i)!} = \frac{(-1)^n}{h^n n!} \cdot (-1)^i \binom{n}{i}.$$

Kürzt man in der Darstellung (2.2.8) hierbei den von i unabhängigen Vorfaktor heraus und führt die Größen

$$\tilde{w}_i := (-1)^i \binom{n}{i}$$

ein, so ergibt sich für den Fall der äquidistanten Knoten die folgende vereinfachte Formel für das Interpolationspolynom p,

$$p_n(x) = \sum_{i=0}^{n} \frac{\tilde{w}_i}{x - x_i} y_i \Big/ \sum_{i=0}^{n} \frac{\tilde{w}_i}{x - x_i} \qquad \text{für } x \neq x_j \, \forall j. \tag{2.2.14}$$

b) Zahlenbeispiel: n=3,

$$\begin{array}{rcccc} i = & 0 & 1 & 2 & 3 \\ x_i = & 0 & 1.5 & 2.5 & 4.5 \end{array}$$

Man erhält das Dreieckschema für die $w_i^{(k)}$:

k	x_k	$i=0$	1	2	3
0	0	1			
1	1.5	−0.66667	0.66667		
2	2.5	0.26667	−0.66667	0.40000	
3	4.5	−0.05926	0.22222	−0.20000	0.03704

2.3 Newton-Interpolation

Zum rekursiven Aufbau der Interpolationspolynome nach wachsender Stützstellenzahl ist folgender Ansatz zweckmäßig

$$p_n(x) = p_{n-1}(x) + (x - x_0) \cdots (x - x_{n-1}) a_n.$$

Man nennt diese Darstellung die Newton-Darstellung des Interpolationspolynoms. In dieser Darstellung verschwindet der zweite Summand in den Punkten $x_0, \ldots, x_{n-1}$. Man setzt also induktiv voraus, daß p_{n-1} die Interpolationsaufgabe in den ersten n Punkten erfüllt, $p_{n-1}(x_i) = y_i,\ i = 0, \ldots, n-1$. Den Wert a_n berechnet man dann aus der zusätzlichen Interpolationsbedingung $p_n(x_n) = y_n$ oder zweckmäßiger aus einem weiter unten formulierten Algorithmus. Beginnend mit $p_0(x) \equiv y_0 = a_0$ wird also für p_n die folgende Form angenommen:

$$\begin{aligned} p_n(x) &= a_0 + (x - x_0) a_1 + \ldots + (x - x_0) \cdots (x - x_{n-1}) a_n \qquad (2.3.1) \\ &= a_0 + (x - x_0)[\cdots [a_{n-2} + (x - x_{n-2})[a_{n-1} + (x - x_{n-1}) a_n]] \cdots] \end{aligned}$$

Diese Klammerung deutet bereits an, wie $p_n(x)$ am zweckmäßigsten zu berechnen ist:

Algorithmus 2.3.1 Horner-Schema zur Auswertung von $p_n(x)$ bei bekannten a_i.

$$
\boxed{
\begin{array}{l}
q_n := a_n; \\
\text{für } k = n-1, n-2, \ldots, 0: \\
\quad q_k := a_k + (x - x_k) q_{k+1}; \\
p_n(x) = q_0.
\end{array}
}
$$

Rechenaufwand Hornerschema: $3n$ Operationen:

Für den Spezialfall $x_i = 0 \ \forall i$ erhält man $p_n(x) = \sum_{i=0}^{n} a_i x^i$ und braucht für die Auswertung dieser Polynomform nach dem Hornerschema nur $2n$ Operationen.

Somit bleibt die wesentliche Frage der Bestimmung der a_i in (2.3.1). Das gelingt in der folgenden Definition und im Algorithmus 2.

Definition 2.3.2 (und Algorithmus) *Man erhält die l-ten dividierten Differenzen oder l-ten Differenzenquotienten gemäß*

$$
\boxed{
\begin{array}{l}
\textit{für } i = 0, 1, \ldots, n: \\
\quad y[x_i] := y_i; \\
\textit{für } l = 1, \ldots, n: \\
\quad \textit{für } i = 0, \ldots, n-l: \\
\qquad y[x_i, \ldots, x_{i+l}] := \dfrac{y[x_{i+1}, \ldots, x_{i+l}] - y[x_i, \ldots, x_{i+l-1}]}{x_{i+l} - x_i}.
\end{array}
}
\tag{2.3.2}
$$

Bemerkungen: Die Berechnung nach (2.3.2) führt auf folgendes Dreieckschema, das spaltenweise $(l = 1, 2, \ldots, n)$ erzeugt wird.

$$
\begin{array}{c|ccccc}
x_i & y_i = y[x_i] & l=1 & l=2 & & l=n \\
\hline
x_0 & y_0 & & & & \\
x_1 & y_1 & y[x_0,x_1] & & & \\
x_2 & y_2 & y[x_1,x_2] & y[x_0,x_1,x_2] & & \\
x_3 & y_3 & y[x_2,x_3] & y[x_1,x_2,x_3] & & \\
\vdots & \vdots & \vdots & & \ddots & \\
x_n & y_n & y[x_{n-1},x_n] & y[x_{n-2},x_{n-1},x_n] & \cdots & y[x_0,\ldots,x_n]
\end{array}
\tag{2.3.3}
$$

Zur Durchführung genügt ein lineares Feldes $[0, 1, \ldots, n]$, in dem der Reihe nach die Werte $y_1, \ldots, y_n$ durch die ersten dividierten Differenzen und dann die ersten dividierten Differenzen $y[x_1, x_2], \ldots, y[x_{n-1}, x_{n-2}]$ durch die zweiten dividierten Differenzen überschrieben werden, usw. Am Ende bleiben die in (2.3.7) benötigten Werte übrig. Es ist auch eine zeilenweise Berechnung möglich, wenn man mit dem höchsten Index beginnt. Dabei kann dann bei Hinzunahme zusätzlicher Interpolationspunkte das Differenzenschema einfach ergänzt werden.

Rechenaufwand für (2.3.2): $3[n + (n-1) + \cdots + 1] = \frac{3}{2}n^2 + \mathcal{O}(n)$ Operationen.

Der Zusammenhang der in (2.3.2) eingeführten Differenzenquotienten mit den Koeffizienten a_i aus (2.3.1) ergibt sich im folgenden Neville-Algorithmus und dem anschließenden Satz 2.3.4.

Definition 2.3.3 (und Neville-Algorithmus) *Es seien $p_{i,l} \in \Pi_l$ die Polynome mit*

$$p_{i,l}(x_j) = y_j, \quad j = i, \dots, i+l. \tag{2.3.4}$$

Man erhält diese $p_{i,l}$ und den Wert des Polynoms p_n zu (2.2.2) an einer Stelle x durch

$$\boxed{\begin{array}{l} \textit{für } i = 0, 1, \dots, n: \\ \quad p_{i,0}(x) := y_i; \\ \textit{für } l = 1, \dots, n: \\ \quad \textit{für } i = 0, \dots, n-l: \\ \qquad p_{i,l}(x) := \dfrac{(x - x_i) p_{i+1,l-1}(x) + (x_{i+l} - x) p_{i,l-1}(x)}{x_{i+l} - x_i}; \\ p_n(x) = p_{0,n}(x). \end{array}} \tag{2.3.5}$$

Für $x \in (x_i, x_{i+l})$ ist $p_{i,l}(x)$ eine 'konvexe Linearkombination' der $p_{i,l-1}(x)$ und $p_{i+1,l-1}(x)$. Dadurch ergeben sich Stabilitätsvorteile bezüglich der Rundungsfehler.

<u>**Rechenaufwand**</u> für den Neville-Algorithmus (2.3.5) bei vorberechneten Differenzen $x - x_i$:

$$\frac{5}{2} n^2 + \mathcal{O}(n) \text{ Operationen.}$$

Den Zusammenhang zwischen den dividierten Differenzen aus (2.3.2), den Teilpolynomen $p_{i,l}$ aus (2.3.4) und den Koeffizienten a_i aus der Newton-Interpolationsform (2.3.1) gibt der folgende

Satz 2.3.4 *a) Die in (2.3.5) berechneten Polynome stimmen mit den in (2.3.4) definierten überein.*

b) Für die dividierten Differenzen in (2.3.2) und die Teilpolynome in (2.3.4) gilt:

$$y[x_i, \dots, x_{i+l}] = \frac{1}{l!}\, p_{i,l}^{(l)} \equiv \frac{1}{l!}\, p_{i,l}^{(l)}(x), \quad 0 \le i \le i + l \le n. \tag{2.3.6}$$

c) Das Lagrange-Interpolationspolynom ist in der Newtonform (2.3.1) darstellbar als

$$p_n(x) = y[x_0] + (x - x_0)\, y[x_0, x_1] + \dots + (x - x_0) \cdots (x - x_{n-1})\, y[x_0, \dots, x_n]. \tag{2.3.7}$$

Beweis Durch Induktion, die Aussagen bezüglich (2.3.4) bis (2.3.7) sind offensichtlich richtig für $n = l = 0$.

a) Unter der Induktionsvoraussetzung, daß $p_{i,l-1}(x)$ und $p_{i+1,l-1}(x)$ ihre Interpolationsbedingungen in (2.3.4) erfüllen, gilt für das in (2.3.5) definierte $p_{i,l}(x)$ die Aussage

$$p_{i,l}(x_j) = y_j, \quad j = i, \dots, i+l.$$

Für $j = i$ und $i + l$ treten nämlich in (2.3.5) nur der zweite bzw. erste Summand im Zähler auf, für die Fälle $j = i + 1, \ldots, i + l - 1$ kann man aufgrund der Induktionsvoraussetzung den Faktor y_j ausklammern, die restlichen Faktoren ergeben dann zusammen eins.

b) Nun wird angenommen, (2.3.6) sei richtig für $l - 1$. Dann gilt nach (2.3.1) und (2.3.4)

$$p_{i,l}(x) = p_{i,l-1}(x) + (x - x_i) \cdots (x - x_{i+l-1}) \cdot a. \tag{2.3.8}$$

Zunächst wird gezeigt

$$a = y[x_i, \ldots, x_{i+l}]. \tag{2.3.9}$$

Durch Vergleich der Koeffizienten von x^l in (2.3.8) und (2.3.5) ergibt sich mit der Induktionsvoraussetzung die Gleichung

$$a = \frac{y[x_{i+1}, \ldots, x_{i+l}] - y[x_i, \ldots, x_{i+l-1}]}{x_{i+l} - x_i}.$$

Wegen (2.3.2) ist (2.3.9) bewiesen. Durch l-malige Differentiation von (2.3.8) folgt (2.3.6).

c) Ergibt sich durch Anwendung von (2.3.8) und (2.3.9) für $i = 0$ und $l = 0, \ldots, n$. ■

Beispiel 2.3.5 Newton-Polynom durch vier Interpolationspunkte (Werte in der Tabelle):

i	x_i	y_i			
0	-1	$\boxed{2}$			
1	0	4	$\boxed{2}$		
2	2	6	1	$\boxed{-\frac{1}{3}}$	
3	3	12	6	$\frac{5}{3}$	$\boxed{\frac{1}{2}}$

Die eingerahmten Werte sind die Koeffizienten im Interpolationspolynom

$$p_3(x) = 2 + 2(x + 1) - \frac{1}{3}(x + 1)x + \frac{1}{2}(x + 1)x(x - 2).$$

Bei den besprochenen Verfahren werden keinerlei spezielle Eigenschaften der verwendeten Datenpaare (x_j, y_j) berücksichtigt. Sind die y_j allerdings *Funktionswerte* einer bestimmten Funktion f, stellt sich die Frage, wie stark p und f voneinander abweichen. Diese und andere Eigenschaften sind im folgenden Satz zusammengefaßt.

Satz 2.3.6 *a) Das Interpolationspolynom p_n aus (2.2.5) hängt linear vom Datenvektor y ab.*

b) Die dividierten Differenzen sind unabhängig von der Reihenfolge der Wertepaare (x_i, y_i).

c) Sind die y_i Funktionswerte einer Funktion $f \in C^n[a, b]$, d.h. $x_i \in [a, b]$, $y_i = f(x_i)$, so gilt

$$f[x_i, \ldots, x_{i+k}] := y[x_i, \ldots, x_{i+k}] = \frac{1}{k!}\, f^{(k)}(\xi), \tag{2.3.10}$$

mit $\xi \in (\min_{j=i}^{i+k} x_j, \max_{j=i}^{i+k} x_j)$, $0 \le i \le i + k \le n$.

d) Für $f(x) = c_k x^k + \ldots \in \Pi_k$ gilt

$$f[x_0, \ldots, x_n] = \begin{cases} c_k & \textit{für} \quad k = n, \\ 0 & \textit{für} \quad k < n. \end{cases}$$

e) Für den Interpolationsfehler $r_n(x) := f(x) - p_n(x),\ x \in [a,b]$ ergibt sich bei einer Funktion $f \in C^{n+1}[a,b]$ mit $\omega_{n+1}(x) := (x - x_0)\ldots(x - x_n)$ die Darstellung

$$r_n(x) = \omega_{n+1}(x) f[x_0, \ldots, x_n, x] = \omega_{n+1}(x) \frac{f^{(n+1)}(\xi)}{(n+1)!}, \quad \xi \in (a,b). \tag{2.3.11}$$

Beweis Teil a) ist nach Definition offensichtlich, b) folgt aus Satz 2.2.2 und (2.3.6).

c) Nach (2.3.6) ist $y[x_i, \cdots, x_{i+k}] = p_{i,k}^{(k)}(x)/k!$. Nun verschwindet $r_k = f - p_{i,k}$ in den Punkten $x = x_i, \cdots, x_{i+k}$. Nach dem iterierten Satz von Rolle existiert $\xi \in \left(\min_{j=i}^{i+k} x_j, \max_{j=i}^{i+k} x_j\right)$ mit

$$0 = r_k^{(k)}(\xi) = f^{(k)}(\xi) - p_{i,k}^{(k)}(\xi).$$

d) Ergibt sich als Spezialfall von c).

e) Der Interpolationsfehler wird an einer festen Stelle $x = \bar{x}$ dargestellt und das Interpolationspolynom $p_{0,n+1}$ mit der zusätzlichen Stützstelle $x_{n+1} := \bar{x}, y_{n+1} := f(\bar{x})$, betrachtet. Nach Satz 2.3.4 ergibt sich

$$p_{0,n+1}(x) = p_{0,n}(x) + (x - x_0)\cdots(x - x_n) f[x_0, \cdots, x_n, x_{n+1} = \bar{x}].$$

Wegen $p_{0,n+1}(\bar{x}) = f(\bar{x})$ gilt für den Fehler

$$r_n(\bar{x}) = f(\bar{x}) - p_{0,n}(\bar{x}) = (\bar{x} - x_0)\cdots(\bar{x} - x_n)\, f[x_0, \cdots, x_n, \bar{x}].$$

Die Definition von ω_{n+1} und (2.3.10) liefern schließlich (2.3.11). ■

Wegen der Beziehung (2.3.10) liegt es nahe, das Hermite-Interpolationsproblem (2.2.3) durch Grenzübergang von mehreren Knoten in eine gemeinsame Stelle zu behandeln. Bekanntlich ist

$$\lim_{x_{i+1} \to x_i} f[x_i, x_{i+1}] = f'(x_i) \quad \text{für } f \in C^1[a,b].$$

Analoge Aussagen gelten in (2.3.10), wenn mehr als zwei Knoten zusammenrücken. Das Problem ist allerdings nur dann sinnvoll gestellt, wenn, wie in (2.2.3) gefordert, an einer Stelle Funktions- und Ableitungswerte lückenlos vorgeschrieben sind. Es seien also für die Hermite-Interpolationsaufgabe bei einer genügend oft differenzierbaren Funktion f die folgenden Werte bekannt

$$f(x_i), f'(x_i), \cdots, f^{(j)}(x_i), \quad \text{wenn } x_i = x_{i+1} = \cdots = x_{i+j}. \tag{2.3.12}$$

Mit diesen Ableitungen definieren wir gemäß (2.3.10) die Differenzenquotienten als

$$f[x_i, \cdots, x_{i+k}] := \frac{1}{k!} f^{(k)}(x_i), \ 0 \le k \le j.$$

Für all solche *Mehrfachenknoten* (2.3.12) werden diese Werte in das Differenzentableau (2.3.3) eingetragen. Die noch fehlenden Differenzenquotienten berechnet man dann nach (2.3.2).

Beispiel 2.3.7 Differenzentableau für $n = 4,\ x_1 = x_2 = x_3$:

$$\begin{array}{c|lllll}
x_0 & y_0 & & & & \\
x_1 & y_1 & y[x_0,x_1] & & & \\
x_1 & y_1 & y_1' & y[x_0,x_1,x_1] & & \\
x_1 & y_1 & y_1' & \frac{1}{2}y_1'' & y[x_0,x_1,x_1,x_1] & \\
x_4 & y_4 & y[x_1,x_4] & y[x_1,x_1,x_4] & y[x_1,x_1,x_1,x_4] & y[x_0,x_1,x_1,x_1,x_4]
\end{array}$$

Bemerkungen: Man kann das Hornerschema aus Algorithmus 1 verallgemeinern zur Berechnung von Ableitungswerten von Polynomen.

Satz 2.3.6 erlaubt eine Gegenüberstellung von Taylor- und Newton-Entwicklung für $f \in C^{n+1}[a,b]$. Mit den analog zu ω_{n+1} definierten $\omega_k = (x-x_0)\cdots(x-x_{k-1})$, vgl. (2.3.11), erhält man für die Entwicklung nach

$$\begin{array}{lrl}
\text{Taylor:} & f(x) = & \sum\limits_{k=0}^{n}(x-x_0)^k\,\dfrac{f^{(k)}(x_0)}{k!} + (x-x_0)^{n+1}\dfrac{f^{(n+1)}(\xi)}{(n+1)!}, \\
\text{Newton:} & f(x) = & \sum\limits_{k=0}^{n}\omega_k(x)f[x_0,\cdots,x_k] + \ \omega_{n+1}(x)\ \dfrac{f^{(n+1)}(\eta)}{(n+1)!},
\end{array}$$

$\xi,\eta \in (a,b)$. Insbesondere geht für $x_j \to x_0,\ j = 1,\cdots,n$, die Newton-Darstellung über in die Taylor-Darstellung.

2.4 Entwicklung nach Orthogonalpolynomen, dreistufige Rekursionen

In diesem Teilparagraph werden zunächst in $\Pi_n \subset C[a,b]$ Familien von Orthogonalpolynomen bezüglich geeigneter Skalarprodukte eingeführt. Es zeigt sich, daß sich solche Orthogonalpolynome *rekursiv* mittels *Drei-Term-Rekursionen* berechnen lassen. Darauf aufbauend kann man *Clenshaw-Algorithmen* zur Auswertung von Polynomen definieren, die hier nur für den wichtigsten Fall der Entwicklung nach *Tschebyscheff-Polynomen* angegeben werden.

Zu einer positiven Gewichtsfunktion $g : [a,b] \mapsto \mathbf{R}_+$ und $f,h \in C[a,b]$ sei

$$\begin{array}{rcccll}
(f,h) & := & (f,h)_g & := & \int\limits_a^b f(x)h(x)g(x)\ dx, & \text{bzw.} \\
(f,h) & := & (f,h)_g & := & \sum\limits_{k=0}^{n} f(x_k)h(x_k)g(x_k) &
\end{array} \tag{2.4.1}$$

mit $a \le x_0 < \ldots < x_n \le b$. Durch (2.4.1) sind Innenprodukte für Π_n definiert. Über den Zusammenhang zwischen den Innenprodukten aus (2.4.1) in Π_n wird im anschließenden Kapitel 3 nachgedacht. Nun wird eine Familie von Orthogonalpolynomen definiert durch

$$P_k \in \Pi_k \quad \text{und} \quad (P_j,P_k) = \delta_{jk}\cdot c_k,\ 0 \le j,k \le n. \tag{2.4.2}$$

Dabei bezeichnet δ_{jk} das übliche Kronecker-Symbol und die c_k sind entweder in geeigneter Weise vorgegeben (z.B. $c_k \equiv 1$) oder berechnen sich aus der Forderung

$$P_0(x) \equiv 1,\ P_k(x) = x^k + \dots \tag{2.4.3}$$

Bekanntlich ist durch (2.4.1) bis (2.4.3) eindeutig eine Familie von Orthogonalpolynomen definiert. Es gilt sogar

Satz 2.4.1 *Für die Orthogonalpolynome P_k in (2.4.2), (2.4.3) gilt mit $P_{-1} :\equiv 0,\ P_0 :\equiv 1$, eine Drei-Term-Rekursion der Form*

$$P_k(x) = (x + a_k)P_{k-1}(x) + b_k P_{k-2}(x), \quad k \geq 1, \tag{2.4.4}$$

wobei $b_1 = 0$ und

$$a_k = -\frac{(xP_{k-1}, P_{k-1})_g}{(P_{k-1}, P_{k-1})_g},\ k \geq 1, \quad b_k = -\frac{(P_{k-1}, P_{k-1})_g}{(P_{k-2}, P_{k-2})_g},\ k \geq 2. \tag{2.4.5}$$

Beweis Wegen (2.4.3) ergibt sich die Darstellung (2.4.4), wenn P_k durch Orthogonalisierung von xP_{k-1} berechnet werden soll. Die in (2.4.4) fehlenden Terme der Form $d_j P_j(x),\ 0 \leq j \leq k-3$, $d_j = -(xP_{k-1}, P_j)_g/(P_j, P_j)_g = -(P_{k-1}, xP_j)_g/(P_j, P_j)_g$, fallen wegen (2.4.2) weg und man erhält schließlich (2.4.4) mit den speziellen Koeffizienten aus (2.4.5). Dabei wurde $b_k = d_{k-2}$ mit Hilfe von $xP_{k-2} = P_{k-1} - a_{k-1}P_{k-2} - b_{k-1}P_{k-3}$ umgeformt, denn

$$(xP_{k-1}, P_{k-2})_g = (P_{k-1}, xP_{k-2})_g = (P_{k-1}, P_{k-1})_g. \qquad \blacksquare$$

Hier werden nur die Tschebyscheff-Polynome behandelt, im Kapitel 3 treten aber im Zusammenhang mit der numerischen Integration weitere Klassen von Orthogonalpolynomen im Sinne von (2.4.1) bis (2.4.3) auf. Auf viele Polynomfamilien sind die Ergebnisse von Satz 2.4.1 anwendbar. Ganz allgemein sind in vielen Gebieten der Analysis Entwicklungen nach Orthogonalsystemen zur Approximation von Funktionen sehr hilfreich: Kennt man für ein Innenprodukt aus (2.4.1) ein System orthogonaler Funktionen $\{T_k\}$, so kann man eine Funktion f approximieren in der Form (Lineare Algebra: Entwicklung nach Orthogonalbasis)

$$f \approx \sum_{j=0}^{n} \alpha_j P_j =: p_n, \quad \alpha_j := (f, P_j)/(P_j, P_j). \tag{2.4.6}$$

Definition 2.4.2 *Man bezeichnet die Funktionen*

$$T_n(x) := \cos(n \arccos x),\ |x| \leq 1, \quad n = 0, 1, .., \tag{2.4.7}$$

als Tschebyscheff-Polynome *und ihre Nullstellen*

$$x_k := x_{k,n} := \cos\Big(\frac{2k+1}{2n+2}\pi\Big), \quad k = 0, \dots, n \tag{2.4.8}$$

als Tschebyscheff-Punkte oder -Knoten.

Satz 2.4.3 *Die in (2.4.7) definierten Funktionen T_n sind Polynome vom Grad n. Sie können mit $T_0 = 1,\ T_1(x) = x$, rekursiv berechnet werden durch die Drei-Term-Rekursion*

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x),\ \ n = 1, 2, \ldots . \tag{2.4.9}$$

Die $x_k = x_{k,n}$ aus (2.4.8) sind die Nullstellen von T_{n+1}. Die Polynome T_n haben die Form

$$T_n(x) = 2^{n-1}x^n + \ldots \quad \in \Pi_n, \tag{2.4.10}$$

ihre Supremumsnorm ist eins,

$$\|T_n\|_{[-1,1],\infty} = 1, \quad n = 0, 1, \ldots \quad . \tag{2.4.11}$$

Beweis Mit $t := \arccos x$ folgen die Aussagen für T_0, T_1 unmittelbar aus (2.4.7). Die Rekursion in (2.4.9) ergibt sich aus dem Additionstheorem

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

Aus (2.4.9) folgt unmittelbar die Aussage, daß die $T_n \in \Pi_n$ die Form (2.4.10) besitzen. Aus (2.4.7) andererseits ergibt sich direkt, daß die x_k aus (2.4.8) die Nullstellen von T_{n+1} sind und (2.4.11) gilt. ∎

Satz 2.4.4 *Die Tschebyscheff-Polynome $T_k,\ k = 0, 1, \ldots, n$, bilden ein Orthogonalsystem bezüglich des Innenprodukts*

$$(f, g)_n := \sum_{k=0}^{n} f(x_k)g(x_k) \qquad \textit{mit } x_k = x_{k,n} \textit{ aus (2.4.8).} \tag{2.4.12}$$

Zur Vereinfachung der Formeln (wegen (2.4.15)) verwendet man bei Entwicklungen nach Tschebyscheff-Polynomen (wie bei Fourierentwicklungen) statt (2.4.6) die Darstellung

$$f \approx \frac{c_0}{2} + \sum_{j=1}^{n} c_j T_j =: p_n. \tag{2.4.13}$$

Die Koeffizienten ergeben sich bei den speziellen Knoten (2.4.8) in (2.4.12) durch

$$c_j = \frac{2}{n+1}(f, T_j)_n = \frac{2}{n+1}\sum_{k=0}^{n} f(x_k)\cos\left(j\frac{2k+1}{2n+2}\pi\right), \quad j = 0, 1, \ldots, n. \tag{2.4.14}$$

Das Polynom p_n aus (2.4.13) interpoliert f in $x_{k,n}, k = 0, \ldots, n$.

Beweis Die Aussagen über (2.4.12), (2.4.14) folgen aus der bekannten Beziehung

$$\begin{aligned}(T_i, T_j)_n &= \sum_{k=0}^{n} T_i(x_k)T_j(x_k) = \sum_{k=0}^{n} \cos\left(i\frac{2k+1}{2n+2}\pi\right)\cos\left(j\frac{2k+1}{2n+2}\pi\right) \\ &= \begin{cases} 0 & \text{für} \quad i \neq j, \\ \frac{n+1}{2} & \text{für} \quad 0 < i = k < n+1, \\ n+1 & \text{für} \quad i = j = 0. \end{cases}\end{aligned} \tag{2.4.15}$$

Die Relation $(T_0, T_0)_n = 2(T_j, T_j)_n$, $1 \le j \le n$, in Formel (2.4.15) ist der Grund für den Übergang von (2.4.7) zu (2.4.13). Da p_n nach einer Orthogonalbasis entwickelt wurde, ist

$$(f - p_n, f - p_n)_n = \min_{q \in \Pi_n} (f - q, f - q)_n = \min_{q \in \Pi_n} \sum_{k=0}^{n} [f(x_k) - q(x_k)]^2 = 0.$$

Das Minimum wird im Interpolationspolynom an die Funktion f zu den Punkten x_k aus (2.4.8) angenommen, mit dem p_n offensichtlich übereinstimmt. ■

<u>Rechenaufwand</u> zur Bestimmung der c_j aus (2.4.14):

$$2n^2 + \mathcal{O}(n) \text{ Operationen } + n^2 \cos\text{-Funktionsauswertungen.}$$

Zur Auswertung eines Polynoms (2.4.13) dient der folgende Algorithmus, der nur für den Fall von Tschebyscheff-Polynomentwicklungen besprochen wird. Analoge Algorithmen sind für alle Familien mit Drei-Term-Rekursionen möglich.

Algorithmus 2.4.5 Clenshaw-Algorithmus

$$\boxed{\begin{array}{l} d_{n+1} := 0;\ d_n := c_n; \\ \text{für } k = n-1, n-2, \ldots, 0: \\ \quad d_k := c_k + 2x d_{k+1} - d_{k+2}; \\ p_n(x) = (d_0 - d_2)/2; \end{array}} \tag{2.4.16}$$

zur Berechnung von

$$p_n(x) = \frac{c_0}{2} + \sum_{j=1}^{n} c_j T_j(x) \quad \text{für } x \in \mathbf{R}, (\text{ i.a. } |x| \le 1). \tag{2.4.17}$$

Im Clenshaw-Algorithmus ersetzt man rekursiv in dieser Entwicklung von p_n die T_j mit höherem Index mit Hilfe der Rekursionsformel (2.4.9) durch die einfacheren Tschebyscheff-Polynome mit niedrigerem Index:

$$\begin{aligned} p_n(x) &= d_n T_n + c_{n-1} T_{n-1} + c_{n-2} T_{n-2} + \ldots \\ &= \underbrace{(2x d_n + c_{n-1})}_{d_{n-1}} T_{n-1} + (c_{n-2} - d_n) T_{n-2} + \ldots = \underbrace{(2x d_{n-1} - d_n + c_{n-2})}_{d_{n-2}} T_{n-2} + \ldots \end{aligned}$$

Die folgende Tabelle veranschaulicht, von links nach rechts, die verschiedenen Ersetzungsschritte. Die Rekursionsformeln von (2.4.16) erscheinen in den Spalten. Das Argument (x) bei den T_j wurde weggelassen.

$$\begin{array}{c|ccccccccccccc} & T_n & & T_{n-1} & & T_{n-2} & & \ldots & T_2 & & T_1 & & T_0 \\ \hline & c_n & & c_{n-1} & & c_{n-2} & & \ldots & c_2 & & c_1 & & \frac{c_0}{2} \\ & 0 & & 0 & & -d_n & & \ldots & -d_4 & & -d_3 & & -d_2 \\ & & \nearrow & & \nearrow & & \nearrow & & & \nearrow & & \nearrow & \\ & 0 & & 2xd_n & & 2xd_{n-1} & & \ldots & 2xd_3 & & 2xd_2 & & * \\ \nearrow & & \nearrow & & \nearrow & & \nearrow & & & \nearrow & & & \\ \hline d_{n+1} = 0 & d_n & & d_{n-1} & & d_{n-2} & & \ldots & d_2 & & d_1 & & ** \end{array}$$

Der letzte Schritt von (2.4.16) ergibt sich aus

$$\begin{aligned} p_n(x) &= d_1T_1(x) + (\frac{1}{2}c_0 - d_2)T_0(x) = d_1x + \frac{1}{2}c_0 - d_2 \\ &= \frac{1}{2}(c_0 + 2d_1x - d_2) - \frac{1}{2}d_2 = \frac{1}{2}(d_0 - d_2). \end{aligned} \tag{2.4.18}$$

<u>**Rechenaufwand**</u> von (2.4.16): $3n + \mathcal{O}(1)$ Operationen.
Bemerkung: Wegen (2.4.7) und der daraus abgeleiteten Normabschätzung (2.4.11) ist der Fehlerverlauf bei einer (abgebrochenen) Tschebyscheff-Entwicklung wesentlich gleichförmiger als etwa beim Newton-Polynom mit Hornerschema und sonstigen Knoten.

Soll eine bestimmte Funktion f im Ablauf eines Programmes oder u.U. in Softwarepaketen für Rechner vielfach ausgewertet werden, ist eine Approximation durch Polynome vorteilhaft, die mit möglichst wenigen Termen auskommt und günstig ausgewertet werden kann. Mit dem folgenden Prozeß der *Ökonomisierung* kann dabei auf wohlbekannte Potenzreihenentwicklungen zurückgegriffen werden. Durch Taylorentwicklung bestimme man zur Funktion f ein Polynom

$$p_n(x) = a_0 + a_1x + \ldots + a_nx^n, \tag{2.4.19}$$

das folgende Bedingung erfüllt

$$\|f - p_n\|_{[-1,1],\infty} \leq q \cdot tol, \quad 0 < q < 1,$$

mit einer vorgegebenen Toleranz tol. Am Taylor-Restglied ist allerdings zu erkennen, daß der Fehler $f - p_n \cong a_{n+1}x^{n+1}$ in $[0,1]$ nicht gleichmäßig verteilt ist, sondern am Rand, bei ± 1 stark anwächst. Durch bessere Verteilung des Fehlers kann der Grad des Polynoms ohne wesentliche Verschlechterung der Approximation verringert werden. Dazu ersetzt man in einem zweiten Schritt in p_n mit (2.4.10) die höchste Potenz x^n durch T_n und erhält die Darstellung

$$p_n = q_{n-1} + d_nT_n, \quad q_{n-1} \in \Pi_{n-1}, \tag{2.4.20}$$

d.h., $d_n = a_n/2^{n-1}$. Dieser Prozeß kann mit x^{n-1} in q_{n-1} wiederholt werden und führt rekursiv auf Darstellungen

$$p_n = q_k + d_{k+1}T_{k+1} + \ldots + d_nT_n, \quad q_k \in \Pi_k,\ 0 \leq k \leq n.$$

Wählt man nun k so, daß

$$\|p_n - q_k\|_{[-1,1],\infty} \leq |d_{k+1}| + \ldots + |d_n| \overset{!}{\leq} (1-q)tol$$

gilt, erhält man wegen (2.4.11) insgesamt

$$\|f - q_k\|_{[-1,1],\infty} \leq tol$$

und hat mit diesem q_k eine Approximation an f von geringerem Grad als n gefunden.

2.5 Vergleich der Verfahren

In den Kapiteln 2.2 bis 2.4 wurden eine Reihe verschiedener Methoden zur Bestimmung von Interpolationspolynomen vorgestellt. In 2.2 und 2.3 waren die Knoten völlig willkürlich vorgebbar, in 2.4 wurde nur das Intervall $[-1,1]$ behandelt bei Verwendung der Knoten (2.4.8). In der folgenden Tabelle werden die verschiedenen Methoden gegenüberstellt. Dabei wird der Rechenaufwand getrennt nach einmaliger Vorberechnung von Koeffizienten und der Polynomauswertung bei bekannten Koeffizienten.

Verfahren	Rechenaufwand(Oper.): Vorbereitung,	1 Auswertung	Bemerkungen
Lagrange origin.	$L_i(x):\ 3n^2$	$2n$	L_i abhängig von x und Gitter, Stabilitätsprobleme
Lagr. baryzentr.	$w_i:\ \frac{3}{2}n^2$	$5n$	w_i nur gitterabhängig, flexibel, stabil
Newton	$y[.]:\ \frac{3}{2}n^2$	$3n$	$y[x_{0,\dots}]$ abhängig von Gitter und Daten, flexibel, Stabilitätsprobleme
Aitken-Neville	—	$\frac{5}{2}n^2$	für Spezialanwendungen
Clenshaw, n bel. Clenshaw, $n=2^m$	$c_i:\ 2n^2$ $m\cdot n$	$3n$ $3n$	Einfach, stabiler als Newton. F.F.T.(Numerik II), spezielle Gitter, Übergang von 2^m zu 2^{m+1}.

2.6 Interpolationsfehler, optimale Knoten

Bisher wurde davon ausgegangen, daß alle 'Eingangsdaten' festgelegt sind und nur die Berechnung und Auswertung des Interpolationspolynoms behandelt. Wird die Interpolation zur Approximation einer gegebenen Funktion verwendet, so läßt sich der Fehler, ohne Mehraufwand, durch eine geeignete Wahl der Interpolationsknoten sehr stark verkleinern. Dies ergibt sich aus einer genaueren Inspektion der Darstellung (2.3.11) des Interpolationsfehlers. Dabei wird die Supremumsnorm

$$\|g\|_\infty := \sup\{|g(x)|:\ x\in[\alpha,\beta]\},\quad g\in C[\alpha,\beta],$$

benutzt. Für eine 'glatte' Funktion $f\in C^{n+1}[\alpha,\beta]$ erhält man aus der Fehlerdarstellung (2.3.11) mit $\omega_{n+1}(x) := (x-x_0)\cdots(x-x_n)$ und $x\in[\alpha,\beta]$, $\xi\in(\alpha,\beta)$ die Schranke

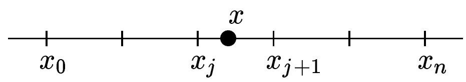
$$|f(x)-p_n(x)| = |\omega_{n+1}(x)|\frac{|f^{(n+1)}(\xi)|}{(n+1)!} \le |\omega_{n+1}(x)|\frac{\|f^{(n+1)}\|_\infty}{(n+1)!}. \tag{2.6.1}$$

Bei festem Polynomgrad n ist der Anteil $\|f^{(n+1)}\|_\infty$ durch die Funktion f vorgegeben. Dagegen kann aber der Beitrag $|\omega_{n+1}(x)|$ durch geeignete Wahl der Stützstellen beeinflußt werden. Zwei mögliche Situationen werden diskutiert:

Problem 1: Es sind sehr viele Funktionswerte von f gegeben an äquidistanten Stellen $x_i =$

$\alpha + ih,\ i = 0, 1, \ldots$ (z.B. Funktionstafel, Meßreihe). Welche Knoten eignen sich, bei festem n, am besten zur Auswertung von f an einer bestimmten Stelle x?

Es sei nun speziell $x \in [x_j, x_{j+1}],\ 0 \le j < n$.



Dann folgt

$$\begin{aligned}
|\omega_{n+1}(x)| &= |x - x_0| \cdots |x - x_j||x - x_{j+1}| \cdots |x - x_n| \\
&\le h(j+1-0)h(j+1-1)\cdots hh \cdots h(n-j) = h^{n+1}(j+1)!(n-j)! \\
&= h^{n+1}\frac{(n+1)!}{\binom{n+1}{j+1}}. \qquad (2.6.2)
\end{aligned}$$

Da die Binomialkoeffizienten bei $\frac{n+1}{2}$ den größten Wert haben, ist der Fehler minimal für

$$j + 1 = \lfloor \frac{n+1}{2} \rfloor,$$

d.h., es ist am günstigsten, links und rechts von x etwa gleich viele Stützstellen zu wählen.

Problem 2: Wenn f durch das Polynom p_n *gleichmäßig* möglichst gut approximiert werden soll, ist der maximale Fehler im Intervall $[\alpha, \beta]$ zu betrachten. Dieser erfüllt nach (2.6.1)

$$\|f - p_n\|_\infty \le \|\omega_{n+1}\|_\infty \frac{\|f^{(n+1)}\|_\infty}{(n+1)!}. \qquad (2.6.3)$$

In dieser Abschätzung soll der erste Faktor minimiert werden. Dazu sind diejenigen Knoten $x_0, \ldots, x_n$ zu suchen mit

$$\|\omega_{n+1}\|_\infty = \min_{\xi_j \in [\alpha,\beta]} \max_{x \in [\alpha,\beta]} |x - \xi_0||x - \xi_1| \cdots |x - \xi_n|. \qquad (2.6.4)$$

Die Lösung dieses Problems kann explizit angegeben werden, es gilt

Satz 2.6.1 *Die optimalen Knoten in (2.6.4) sind die* Tschebyscheff-*Punkte*

$$x_k := \frac{\alpha + \beta}{2} - \frac{\beta - \alpha}{2} \cos\left(\frac{2k+1}{2n+2}\pi\right), \quad k = 0, \ldots, n. \qquad (2.6.5)$$

Der Minimalwert der Norm ist

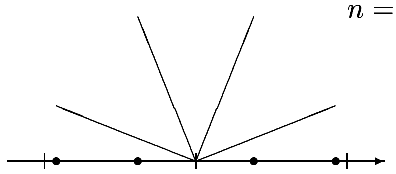
$$\|\omega_{n+1}\|_\infty = \max_{x \in [\alpha,\beta]} |\omega_{n+1}(x)| = 2\left(\frac{\beta - \alpha}{4}\right)^{n+1}. \qquad (2.6.6)$$

Bemerkung: Die Knoten ergeben sich aus den in (2.4.8) genannten durch affine Transformation des Intervalls

$$\left\{ \begin{array}{ccc} [-1, 1] & \mapsto & [\alpha, \beta] \\ \xi & \mapsto & \frac{\alpha+\beta}{2} - \frac{\beta-\alpha}{2}\xi = x \end{array} \right. . \qquad (2.6.7)$$

Die Punkte $\xi_k = \cos\frac{2k+1}{2n+2}\pi = Re\exp\left(i\frac{2k+1}{2n+2}\pi\right)$ sind die Realteile der $(4n+4)$-ten komplexen Einheitswurzeln. Sie liegen am Rand des Intervalls $[-1, 1]$ viel *dichter* als im Innern.

$n = 3$



Beweis des Satzes: Beim Standardintervall $[\alpha,\beta]=[-1,1]$ gilt $x_k=-\xi_k,\ k=0,\dots,n$, also

$$T_{n+1}(x_k)=\cos\left((n+1)\arccos x_k\right)=\cos\Big((n+1)\frac{2k+1}{2n+2}\pi\Big)=\cos\big(\frac{\pi}{2}+k\pi\big)=0.$$

Daher ist ω_{n+1}, bis auf eine Konstante, das Tschebyscheff-Polynom, der Vergleich des höchsten Koeffizienten mit (2.4.10) liefert genauer

$$\omega_{n+1}(x)=x^{n+1}+\ldots=2^{-n}T_{n+1}(x).$$

Die Funktion $\cos(n+1)\varphi$ nimmt ihre Extrema ± 1 in den Stellen $\varphi_j=\frac{j\pi}{n+1}$ an, das Polynom ω_{n+1} seine Extrema $\pm 2^{-n}$ daher in $t_j=\cos\frac{j\pi}{n+1},\ j=0,\dots,n+1$. Daher gilt

$$\|\omega_{n+1}\|_\infty=2^{-n}.$$

Dies ist tatsächlich der Minimalwert dieser Norm, gäbe es nämlich ein Polynom der Form $p_{n+1}=x^{n+1}+\ldots$ mit *kleinerer* Norm, dann müßte für dessen Differenz zu ω_{n+1} in den Stellen t_j gelten

$$\omega_{n+1}(t_j)-p_{n+1}(t_j)\ \begin{cases}<0 & \text{für}\quad n+1-j \text{ gerade}\\ >0 & \text{für}\quad n+1-j \text{ ungerade}\end{cases},\ j=0,\dots,n+1.$$

Das Polynom $\omega_{n+1}-p_{n+1}$ vom Grad n (!) hätte dann aber mindestens $n+1$ Nullstellen. Dies führt zu einem Widerspruch.

Durch die Transformation (2.6.7) wird

$$T_{n+1}(\xi)=T_{n+1}\Big(\frac{\alpha+\beta-2x}{\beta-\alpha}\Big)=2^n\Big(\frac{-2}{\beta-\alpha}\Big)^{n+1}x^{n+1}+\ldots,$$

die Normierung des höchsten Koeffizienten ergibt damit

$$\omega_{n+1}(x)=\Big(-\frac{\beta-\alpha}{2}\Big)^{n+1}2^{-n}T_{n+1}(\xi).$$

Daraus folgt die Schranke (2.6.6). ■

Nach dem Satz von Weierstraß kann jede stetige Funktion beliebig genau durch Polynome approximiert werden. Es ist jedoch ein Trugschluß, dieses Ergebnis unbesehen auf Interpolationspolynome zu übertragen, selbst wenn immer mehr Stützstellen verwendet werden. Dazu wird die Fehleraussage (2.6.3) für zwei Klassen von Knoten betrachtet. Für äquidistante Punkte $x_j=\alpha+j(\beta-\alpha)/n,\ j=0,\dots,n$, ergibt sich für ω_{n+1} aus (2.6.2) die einfache Schranke

$$\|\omega_{n+1}\|_\infty\le\Big(\frac{\beta-\alpha}{n}\Big)^{n+1}n!\le c\Big(\frac{\beta-\alpha}{e}\Big)^{n+1}\frac{1}{\sqrt{n}},\qquad \text{äquidistant},$$

deren asymptotisches Verhalten mit der Stirlingschen Formel approximiert wurde. Mit Hilfe dieser Aussage und der aus (2.6.6) ergeben sich explizite Schranken für die Interpolationsfehler bei äquidistanten und Tschebyscheff-Knoten:

$$\|p_n-f\|_\infty\ \le\ \frac{c}{\sqrt{n}}\Big(\frac{\beta-\alpha}{e}\Big)^{n+1}\frac{\|f^{(n+1)}\|}{(n+1)!},\quad \text{äquidistant},\tag{2.6.8}$$

$$\|p_n-f\|_\infty\ \le\ 2\Big(\frac{\beta-\alpha}{4}\Big)^{n+1}\frac{\|f^{(n+1)}\|}{(n+1)!},\quad \text{Tschebyscheff}.\tag{2.6.9}$$

Bei diesen Formeln ist zu beachten, daß sich bei einer Vergrößerung des Polynomgrads n auch die Ordnung der Ableitung $f^{(n+1)}$ erhöht. Wie das Beispiel

$$f(x) := \frac{1}{1+25x^2}, \quad x \in [-1,1],$$

bei äquidistanten Stützstellen zeigt, kann das Anwachsen der Ableitungen für $n \to \infty$ verhindern, daß der Fehler klein wird. Tatsächlich divergiert die Folge der Interpolationspolynome $\{p_n\}$ $(n \to \infty)$ bei diesem Beispiel in Randnähe bei ± 1. Im Vergleich von (2.6.8) und (2.6.9) verbessert der Übergang zu Tschebyscheffpunkten den wichtigsten Vorfaktor e^{-n} etwas auf 4^{-n}.

Diese Beobachtungen legen die Frage nahe, wie weit man mit dem *interpolierenden* Polynom noch vom *bestapproximierenden* entfernt ist, d.h. von demjenigen $\hat{p}_n \in \Pi_n$ mit

$$\|\hat{p}_n - f\|_\infty = \min_{p_n \in \Pi_n} \|p_n - f\|_\infty =: \operatorname{dist}(f, \Pi_n). \tag{2.6.10}$$

Dazu wird der Interpolationsprozeß als ein Operator

$$P_n : \begin{cases} C[\alpha,\beta] & \mapsto \quad \Pi_n, \\ \quad g & \mapsto \quad P_n g, \quad (P_n g)(x_i) = g(x_i), \; i = 0, \dots, n, \end{cases}$$

betrachtet. Dieser Operator ist linear und ein Projektor, $P_n^2 = P_n$, denn Polynome $p_n \in \Pi_n$ werden reproduziert, $P_n p = p$. Die Norm dieses Projektors kann aus den Lagrange-Polynomen berechnet werden. Dazu sei

$$\lambda_n(x) := \sum_{i=0}^{n} |L_i(x)| \qquad \text{die } \textit{Lebesgue-Funktion.} \tag{2.6.11}$$

Diese tritt im folgenden Satz auf, der ein sehr allgemeines Beweisprinzip benutzt.

Satz 2.6.2 *Für* $f \in C[\alpha,\beta]$ *gilt*

$$\begin{aligned} \|P_n f - f\|_\infty &\le (1 + \|P_n\|) \; \mathit{dist}(f, \Pi_n), & (2.6.12)\\ \|P_n\| &= \|\lambda_n\|_\infty \ge 1, & (2.6.13) \end{aligned}$$

wobei $\|P_n\| := \sup\Big\{ \|P_n g\|_\infty / \|g\|_\infty : \; g \in C[\alpha,\beta], \, g \not\equiv 0 \Big\}$.

Bemerkung: Dies zeigt, daß der Interpolationsfehler höchstens um den Faktor $1 + \|P_n\| \ge 2$ vom minimalen Fehler (2.6.10) abweicht. Daher interessieren Interpolationsknoten mit möglichst kleinem Wert $\|P_n\| = \|\lambda_n\|_\infty$.

Beweis Mit einem beliebigem $p \in \Pi_n$ gilt wegen $P_n p = p$ die Aussage

$$\begin{aligned} \|P_n f - f\|_\infty &= \|P_n f - P_n p + p - f\|_\infty = \|(I - P_n)(p - f)\|_\infty \\ &\le (1 + \|P_n\|) \|p - f\|_\infty. \end{aligned}$$

Auf der rechten Seite kann nun zum Infimum von $\|p - f\|_\infty$ übergegangen werden.

Für die Norm $\|P_n\|$ folgt aus der Lagrange-Darstellung

$$\|P_n g\|_\infty = \|\sum_{i=0}^{n} g(x_i)L_i\|_\infty \le \|g\|_\infty \|\sum_{i=0}^{n} |L_i|\|_\infty = \|g\|_\infty \|\lambda_n\|_\infty,$$

d.h., $\|P_n\| \le \|\lambda_n\|_\infty$. Zum Nachweis der Gleichheit sei $\hat{x} \in [\alpha, \beta]$ eine Maximalstelle von λ_n, d.h., $\lambda_n(\hat{x}) = \|\lambda_n\|_\infty$ und g eine spezielle Funktion mit $\|g\|_\infty = 1$ und $g(x_i) = \text{sign } L_i(\hat{x})$. Hierfür gilt

$$\|P_n g\|_\infty \ge |P_n g(\hat{x})| = |\sum_{i=0}^{n} L_i(\hat{x}) \text{ sign } L_i(\hat{x})| = \lambda_n(\hat{x}).$$

Dies zeigt die Gleichung in (2.6.13). Die untere Schranke für $\|P_n\|$ gilt wegen $\|p\| = \|P_n p\|_\infty \le \|P_n\|\|p\|_\infty \,\forall p \in \Pi_n$. ■

Die Bestimmung eines bestapproximierenden Polynoms (2.6.10) ist ein sehr aufwendiger Prozeß. Zur Klärung der Frage, ob im konkreten Fall nicht eine gute Interpolierende ausreicht, kann der 'Verlustfaktor' $1 + \|P_n\|$ herangezogen werden. Für Tschebyscheffknoten wächst er wie $1.52 + \frac{2}{\pi}\ln(n+1)$. Etwas günstiger verhält sich dieser Faktor für die *gestreckten Tschebyscheff-knoten*

$$x_k = \frac{\alpha+\beta}{2} - \frac{\beta-\alpha}{2} \frac{\cos\frac{2k+1}{2n+2}\pi}{\cos\frac{\pi}{2n+2}}, \; k = 0, \dots, n, . \tag{2.6.14}$$

bei denen $x_0 = \alpha, x_n = \beta$. Die folgende Tabelle enthält die entsprechenden Werte für einige Polynomgrade.

$n =$	5	8	10	15	20	
$1 + \|P_n\| \cong$	3.1	3.4	3.5	3.7	3.9	Tschebyscheffknoten (2.6.5)
	2.6	2.9	3.0	3.3	3.5	gestreckte Tscheb.Knoten (2.6.14)

3 Numerische Integration und Differentiation

In der Praxis stellt sich oft die Aufgabe, mit einer gegebenen Funktion f eine aufwendige, oder explizit nicht mögliche Operation durchzuführen. Als Ausweg kann man die Funktion durch eine 'einfachere' ersetzen, für die die Operation (einfach) ausführbar ist. Bei *Integration und Differentiation* bietet sich die Polynom-Interpolierende in der Lagrange-Form aus §2.2 an:

$$\begin{aligned}
f(x) &\cong & p_n(x) &= \sum_{j=0}^{n} L_j(x)f(x_j) & &\overset{?}{\Longrightarrow} \\
\int_a^b f(x)dx &\cong & \int_a^b p_n(x)dx &= \sum_{j=0}^{n} \Big(\int_a^b L_j(x)dx\Big) f(x_j) & &=: \sum_{j=0}^{n} \alpha_j f(x_j), \\
f'(0) &\cong & p_n'(0) &= \sum_{j=0}^{n} L_j'(0)\; f(x_j) & &=: \sum_{j=0}^{n} \delta_j f(x_j).
\end{aligned} \tag{3.0.1}$$

Die Näherungen für Integral- und Ableitungswerte ergeben sich also einfach als Linearkombinationen von Funktionswerten. Daher werden jetzt Approximationen dieses Typs behandelt, zunächst für die Integration.

3.1 Quadratur

Bei schwierigen Integranden (mit Singularitäten) ist es vorteilhaft, diesen als Produkt einer glatten Funktion und einer explizit aufgeführten, schwierigen *Gewichtsfunktion* $g(x) > 0$ (z.B. $\sqrt{x}, 1/\sqrt{x}, ..$) zu schreiben. Wir betrachten daher allgemeiner

Definition 3.1.1 *Falls $\int_a^b f(x)g(x)dx$ existiert, heißt*

$$\int_a^b f(x)g(x)dx = \sum_{j=0}^{n} \alpha_j f(x_j) + R_n(f) \tag{3.1.1}$$

eine Quadraturformel *mit den* Gewichten α_j *zu den* Knoten

$$a \le x_0 < x_1 < \ldots < x_n \le b.$$

Dabei ist $R_n(f)$ der Quadraturfehler *bzw. das Restglied . Die Formel (3.1.1) besitzt die* Ordnung $m \in \mathbf{N}$, *wenn*

$$R_n(p) = 0 \quad \forall p \in \Pi_{m-1}.$$

Für die speziellen Quadraturformeln, die sich durch Integration des Interpolationspolynoms ergeben, vgl. (3.0.1), gilt

$$\alpha_j = \int_a^b L_j(x)g(x)dx, \; j = 0, \ldots, n. \tag{3.1.2}$$

Aus der Darstellung (2.3.11) des Interpolationsfehlers $r_n = f - p_n$ folgt

$$\begin{aligned} R_n(f) &= \int_a^b f(x)g(x) - \sum_{j=0}^{n} \alpha_j f(x_j) = \int_a^b r_n(x)g(x)dx \\ &= \int_a^b \omega_{n+1}(x) f[x_0, \ldots, x_n, x] g(x)dx. \end{aligned} \tag{3.1.3}$$

Für $f \in C^{n+1}[a,b]$ ergibt sich daraus durch Betragsabschätzung die Schranke

$$|R_n(f)| \le \int_a^b |\omega_{n+1}(x)| g(x)dx \; \frac{1}{(n+1)!} \|f^{(n+1)}\|_\infty. \tag{3.1.4}$$

Damit folgt insbesondere der

Satz 3.1.2 *Die Ordnung einer* interpolatorischen *Quadraturformel (3.1.1), (3.1.2) ist mindestens $n+1$.*

Die Betragsschranke kann mit dem Mittelwertsatz der Integralrechnung noch verschärft werden, wenn das Nullstellenpolynom ω_{n+1} nur ein Vorzeichen besitzt, d.h., wenn

$$\omega_{n+1}(x) = (x - x_0) \cdots (x - x_n) \ge 0 \; \forall x \in [a,b] \quad (\text{bzw. } \le 0 \; \forall ..). \tag{3.1.5}$$

Dann gilt, mit $\xi_1, \xi_2 \in (a,b)$, sogar

$$\begin{aligned} R_n(f) &= \int_a^b \omega_{n+1}(x)g(x)dx \cdot f[x_0, \ldots, x_n, \xi_1] \\ &= \varrho_n \cdot f^{(n+1)}(\xi_2), \quad \varrho_n := \frac{1}{(n+1)!} \int_a^b \omega_{n+1}(x)g(x)dx. \end{aligned} \tag{3.1.6}$$

3.2 Newton-Cotes-Formeln

Speziell für $g \equiv 1$ und äquidistante Knoten $x_j = x_0 + jh, j = 0, \ldots, n$, die die Randpunkte enthalten (d.h. $x_0 = a,\ h = (b-a)/n$, 'abgeschlossene Formeln') oder ausschließen ($a < x_0,\ x_n < b$, 'offene Formeln') ergeben sich die Newton-Cotes-Formeln. Die einfachsten Spezialfälle sind:

Rechteckregel: $n = 0$, offene Formel, Ordnung 2:

$$\int_a^b f(x)dx = (b-a)\, f\Big(\frac{a+b}{2}\Big) + \frac{(b-a)^3}{24} f''(\xi). \tag{3.2.1}$$

Trapezregel: $n = 1$, abgeschlossene Formel, Ordnung 2:

$$\int_a^b f(x)dx = \frac{b-a}{2}\Big[f(a) + f(b)\Big] - \frac{(b-a)^3}{12} f''(\xi). \tag{3.2.2}$$

Simpsonregel: $n = 2$, abgeschlossene Formel, Ordnung 4:

$$\int_a^b f(x)dx = \frac{b-a}{6}\Big[f(a) + 4f(\frac{a+b}{2}) + f(b)\Big] - \frac{(b-a)^5}{2880} f^{(4)}(\xi). \tag{3.2.3}$$

In diesen Formeln steht ξ jeweils für eine unbekannte Zwischenstelle in (a, b). Eine Tabelle der Gewichte α_j und der Fehlerkoeffizienten ϱ_n für weitere Ordnungen folgt weiter unten. Zunächst wird nachgeprüft, daß bei der Rechteck- und Simpsonregel die Ordnung tatsächlich $n + 2$ ist, statt $n + 1$, wie nach Satz 3.1.2 zu erwarten.

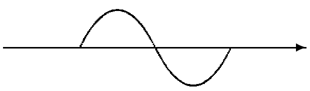
Beweis von (3.2.3): Für $a = 0$ (oBdA) sind die Lagrangepolynome zu $x_0 = 0,\ x_1 = h,\ x_2 = 2h$:

$$L_0(x) = \frac{(x-h)(x-2h)}{2h^2},\ L_1(x) = \frac{x(x-2h)}{-h^2},\ L_2(x) = \frac{x(x-h)}{2h^2} \quad \in \Pi_2.$$

Daraus ergeben sich die Gewichte

$$\alpha_0 = \int_0^{2h} L_0(x)dx = \frac{1}{2h^2}[\frac{1}{3}x^3 - \frac{3h}{2}x^2 + 2h^2x]_0^{2h} = \frac{h}{3} = \alpha_2, \quad \alpha_1 = \int_0^{2h} L_1(x)dx = \frac{4}{3}h.$$

Die Fehlerdarstellungen (3.1.4) bzw. (3.1.6) stimmen allerdings noch nicht mit der in (3.2.3) überein. Dazu ist die Zusatzüberlegung erforderlich, daß wegen



$$\int_0^{2h} \omega_3(x)dx = \int_0^{2h} x(x-h)(x-2h)dx = 0 \tag{3.2.4}$$

sogar kubische Polynome exakt integriert werden! Wir betrachten eine zusätzliche Interpolationsbedingung $p'(x_1) = f'(x_1)$. Nach (2.3.7) folgt dann mit $p_2 = f(x_0)L_0 + f(x_1)L_1 + f(x_2)L_2$ die Darstellung $p_3(x) = p_2(x) + \omega_3(x)f[x_0, x_1, x_1, x_2]$ und

$$\int_0^{2h} p_3(x)dx = \int_0^{2h} p_2(x)dx + \underbrace{\int_0^{2h} \omega_3(x)dx}_{=0}\, f[x_0, x_1, x_1, x_2] = \alpha_0 f(x_0) + \alpha_1 f(x_1) + \alpha_2 f(x_2).$$

Also gilt (3.1.3) sogar mit $n = 3$ und auch (3.1.6) mit $\varrho_3 = -\frac{1}{90}h^5$, da

$$\omega_4(x) = x(x-h)^2(x-2h) \leq 0\ \forall x \in [0, 2h]. \qquad \blacksquare$$

Dieser Beweis überträgt sich auf alle Newton-Cotes-Formeln mit geradem n, die Ordnung ist für diese $n+2$ statt $n+1$.

Die folgende Tabelle enthält die Gewichte und Fehlerkonstanten der Newton-Cotes-Formeln. Da die Gewichte rational sind, erfolgt die Angabe in der Form

$$\int_a^b f(x)dx = \frac{b-a}{\gamma}\sum_{j=0}^{n} \beta_j f(x_j) + c\Big(\frac{b-a}{n}\Big)^{m+1} f^{(m)}(\xi),\ \xi \in (a,b). \tag{3.2.5}$$

Abgeschlossene Newton-Cotes-Formeln, $x_j = a + jh,\ j = 0,\ldots,n,\ h = \frac{b-a}{n}$.

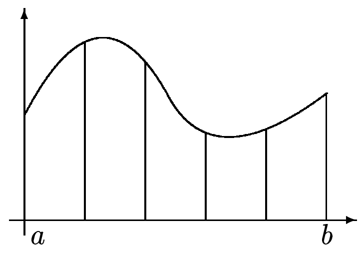
n	γ	β_0	β_1	β_2	β_3	β_4	c	m
1	2	1	1				-1/12	2
2	6	1	4	1			-1/90	4
3	8	1	3	3	1		-3/80	4
4	90	7	32	12	32	7	-8/945	6
5	288	19	75	50	50	75	-275/12096	6
6	840	41	216	27	272	27	-9/1400	8
7	17280	751	3577	1323	2989	2989	-8183/518400	8
8	28350	989	5888	-928	10496	-4540	-2368/467775	10

Die fehlenden Koeffizienten $\beta_5,\ldots,\beta_8$ sind symmetrisch zu ergänzen. Die Gewichte für $n \geq 8$ sind nicht mehr positiv. Dies erhöht etwas die Anfälligkeit der Formeln für Rundungsfehler.

Für die Konvergenz ($n \to \infty$) der Quadraturformel (3.2.5) gilt das gleiche wie bei der Polynom-Interpolation. Das Anwachsen der höheren Ableitungen $f^{(m)},\ m = 2(\lfloor \frac{n}{2} \rfloor + 1)$, kann die Konvergenz zerstören. Andererseits geht der Fehler

$$R_n(f) = c\left(\frac{b-a}{n}\right)^{m+1} f^{(m)}(\xi)$$

sehr schnell gegen Null für $(b-a) \to 0$. Dies kann durch Unterteilung des Gesamtintervalls ausgenutzt werden. Dabei wird eine Quadraturformel fester Ordnung *iteriert* angewendet. Bei der Trapezregel (3.2.2), z.B., ergibt sich mit Teilintervallen $[x_{i-1}, x_i],\ x_j = a + jh,\ j = 0,\ldots,n,\ h := (b-a)/n$, die Näherung



$$\begin{aligned}\int_a^b f(x)dx &= \sum_{i=1}^{n}\int_{x_{i-1}}^{x_i} f(x)dx \cong \sum_{i=1}^{n}\frac{h}{2}\Big[f(x_i) + f(x_{i-1})\Big]\\ &= \frac{h}{2}\Big[f(x_0) + 2f(x_1) + \ldots + 2f(x_{n-1}) + f(x_n)\Big].\end{aligned}$$

Für diese Näherung und die zur Simpsonregel gilt folgende Fehleraussage.

Satz 3.2.1 *Für $n \in \mathbf{N}$ sei $h := (b-a)/n,\ x_i = a + ih,\ f_i := f(x_i),\ i = 0,\ldots,n$. Dann gilt mit $\xi \in (a,b)$ für die* *a) iterierte Trapezregel bei* $f \in C^2[a,b]$*:*

$$\int_a^b f(x)dx = \frac{h}{2}\Big(f_0 + 2f_1 + \ldots + 2f_{n-1} + f_n\Big) - \frac{b-a}{12}h^2 f''(\xi), \tag{3.2.6}$$

b) iterierte Simpsonregel bei $f \in C^4[a,b]$, n *gerade:*

$$\int_a^b f(x)dx = \frac{h}{3}\Big(f_0 + 4f_1 + 2f_2 + 4f_3 + \ldots + 4f_{n-1} + f_n\Big) - \frac{b-a}{180}h^4 f^{(4)}(\xi). \tag{3.2.7}$$

Beweis Die Beweise von a) und b) verlaufen analog. Bei b) wird die Formel (3.2.3) über die $\frac{n}{2} =: k$ Intervalle $[x_{2i-2}, x_{2i}]$ summiert. Beim Restglied ergibt sich dabei mit $\xi_i \in (x_{2i-2}, x_{2i})$

$$-\frac{1}{2880}\sum_{i=1}^{k}(x_{2i} - x_{2i-2})^5 f^{(4)}(\xi_i) = -\frac{32}{2880}h^5\sum_{i=1}^{k} f^{(4)}(\xi_i) = -\frac{b-a}{180}h^4 f^{(4)}(\xi). \qquad \blacksquare$$

An diesen Darstellungen sieht man, daß es sinnvoll ist, in (3.2.5) als Ordnung der Formel die Zahl m und nicht $m+1$ anzugeben.

Der wesentliche Rechenaufwand bei beiden iterierten Formeln fällt bei den $n+1$ Funktionsauswertungen $f_0, \ldots, f_n$ an. Bei einem Vergleich der Fehler sieht man, daß

$$R_n(f) \sim \frac{1}{n^2} \text{ (Trapezregel)} \quad \text{bzw.} \quad R_n(f) \sim \frac{1}{n^4} \text{ (Simpsonregel)}$$

gilt bei einem genügend oft differenzierbaren Integranden f. Eine Verdopplung der Zahl n der Funktionsauswertungen verkleinert den Fehler $R_n(f)$ bei der Trapezregel um den Faktor $\frac{1}{4}$, bei der Simpsonregel dagegen um den Faktor $\frac{1}{16}$, bei vergleichbarem Aufwand. Daher sind bei *glatten* Integranden Formeln hoher Ordnung günstiger.

Beispiel 3.2.2 $\int_1^2 \frac{dx}{x} = \ln 2 = 0.6931472...$

h	Trapez	Fehler	Simpson	Fehler
1	0.75	0.056		
1/2	0.7083..	0.015..	0.694..	0.00129..
1/4	0.6970..	0.0038..	0.69325..	0.00010..
1/8	0.6941..	0.00097..	0.69315..	0.000007..
1/16	0.69339..	0.0002..	0.6931476..	0.0000004..

Diese Beobachtung führt natürlich sofort auf den Wunsch, mit möglichst wenig Funktionsauswertungen eine möglichst hohe Ordnung zu erzielen. Dieses Ziel wird bei der folgenden Klasse von Quadraturformeln erreicht.

3.3 Gauß-Quadratur

Bei den Newton-Cotes-Formeln ergab sich für geraden Polynomgrad n eine um eins erhöhte Konvergenzordnung aufgrund der Eigenschaft

$$\int_a^b \omega_{n+1}(x)dx = 0, \quad \omega_{n+1} = (x-x_0)\cdots(x-x_n),$$

die auf die Symmetrie der Knoten x_i zurückzuführen war. Durch geeignete Wahl der Knoten x_i kann es sogar erreicht werden, daß für den Quadraturfehler (3.1.1) gilt

$$R_n(f) = 0 \quad \forall f \in \Pi_{2n+1}. \tag{3.3.1}$$

Dazu sei $f \in \Pi_{2n+1}$ und $p_n(x) = \sum_{i=0}^n f(x_i)L_i(x)$ das zugehörige Interpolationspolynom vom Grad n. Dann ist $f = p_n + \omega_{n+1} \cdot q_n$ mit $q_n \in \Pi_n$. Damit entspricht die Forderung (3.3.1) der Beziehung

$$R_n(f) = \int_a^b f(x)g(x)dx - \int_a^b p_n(x)g(x)dx = \int_a^b \omega_{n+1}(x)q_n(x)g(x)dx \overset{!}{=} 0.$$

Daher ist (3.3.1) offensichtlich genau dann erfüllt, wenn

$$\int_a^b \omega_{n+1}(x)q(x)g(x)dx = 0 \quad \forall q \in \Pi_n. \tag{3.3.2}$$

Für $g > 0$ in (a, b) stellt

$$(u, v)_g := \int_a^b u(x)v(x)\, g(x)dx \tag{3.3.3}$$

ein Innenprodukt in $C[a, b]$ dar. Die Forderung (3.3.2) bedeutet daher

$$\omega_{n+1} \perp_g \Pi_n \quad \iff \quad (\omega_{n+1}, q)_g = 0\ \forall q \in \Pi_n, \tag{3.3.4}$$

es muß ω_{n+1} also g-orthogonal zu allen Polynomen vom Grad n gewählt werden!

Eine Familie von Orthogonalpolynomen $\omega_k,\ k \in \mathbf{N}$, zum Innenprodukt (3.3.3) kann mit Hilfe des E. Schmidtschen Orthogonalisierungsverfahrens aus der Basis $\{1, x, x^2, \ldots\}$ oder nach Satz 2.4.1 konstruiert werden. Diese Polynome besitzen glücklicherweise nur reelle Nullstellen in (a, b), die als Knoten $x_0, \ldots, x_n$ einer Quadraturformel verwendet werden können. Die wesentlichen Aussagen werden im nächsten Satz zusammengefaßt, später folgt ein Überblick über einige wichtige Klassen von Orthogonal-Polynomen.

Satz 3.3.1 *Die Stützstellen $x_i,\ i = 0, \ldots, n$, der Quadraturformel (3.1.1) seien als Nullstellen des Orthogonalpolynoms vom Grad $n+1$ zur Gewichtsfunktion $g \in C[a, b],\ g > 0$ in (a, b), gewählt, d.h. $\omega_{n+1} \in \Pi_{n+1},\ \omega_{n+1} \perp_g \Pi_n$. Dann besitzt diese* Gauß-sche Quadraturformel *die* Ordnung $2n + 2$, *d.h. $R_n(f) = 0\ \forall f \in \Pi_{2n+1}$. Für $f \in C^{2n+2}[a, b]$ gilt mit $\xi \in (a, b)$,*

$$R_n(f) = \int_a^b f(x)g(x)dx - \sum_{i=0}^n \alpha_i f(x_i) = \frac{1}{(2n+2)!}\int_a^b \omega_{n+1}^2(x)g(x)dx\, f^{(2n+2)}(\xi). \tag{3.3.5}$$

Bemerkung: Gegenüber den Newton-Cotes-Formeln erhält man also ungefähr die doppelte Konvergenzordnung. Bei der Verwendung in iterierten Formeln besitzten die Gaußformeln aber praktische Nachteile (vgl. §3.4).

Beweis Hier ist nur noch zu zeigen, daß das Restglied die von (3.1.6) abweichende Form hat. Analog zum Beweis bei der Simpsonregel (3.2.3) werden zusätzliche Interpolationsbedingungen benutzt, nämlich $f'(x_0), \ldots, f'(x_n)$. Die Newtonform dieses Interpolationspolynoms $p_{2n+1} \in \Pi_{2n+1}$ mit

$$p_{2n+1}(x_i) = f(x_i), \quad p'_{2n+1}(x_i) = f'(x_i),\ i = 0, \ldots, n,$$

hat im Vergleich zu $p_n \in \Pi_n,\ p_n(x_i) = f(x_i),\ i = 0, \ldots, n$, nach (2.3.7) die Gestalt

$$\begin{aligned} p_{2n+1}(x) &= p_n(x) + \omega_{n+1}(x) f[x_0, .., x_n, x_0] + \omega_{n+1}(x)(x - x_0) f[x_0, .., x_n, x_0, x_1] + \\ &\quad \ldots + \omega_{n+1}(x)(x - x_0) \cdots (x - x_{n-1}) f[x_0, \ldots, x_n, x_0, \ldots, x_n]. \end{aligned}$$

Außerdem ist nach (2.3.11)

$$f(x) = p_{2n+1}(x) + \omega_{n+1}^2(x) f[x_0, \ldots, x_n, x_0, \ldots, x_n, x]. \tag{3.3.6}$$

Aufgrund der Orthogonalität (3.3.4) gilt aber

$$\int_a^b \omega_{n+1}(x)(x - x_0) \cdots (x - x_k) g(x) dx = 0\ \forall k < n, \text{ d.h.,}$$

$$\int_a^b p_{2n+1}(x) g(x) dx = \int_a^b p_n(x) g(x) dx = \sum_{i=0}^n \alpha_i f(x_i). \tag{3.3.7}$$

Aus (3.3.6) folgt die Aussage (3.3.5), da die Fehlerformel (3.1.6) wegen $\omega_{n+1}^2 \geq 0$ jetzt auf p_{2n+1} angewendet werden kann. ■

Bemerkung: In (3.3.7) zeigt sich, daß die Gewichte β_i in der zu p_{2n+1} gehörigen Quadraturformel $\int_a^b p_{2n+1}(x) g(x) dx = \sum_{i=0}^n [\alpha_i f(x_i) + \beta_i f'(x_i)]$ alle verschwinden, $\beta_i \equiv 0$.

Zum Vergleich der verschiedenen Methoden betrachten wir folgendes

Beispiel 3.3.2 $\int_{-1}^1 e^x dx = e^1 - e^{-1} = 2 \sinh 1 = 2.3504024$. Näherungen

a) Trapezregel, Ordnung 2, 2 Schritte, 3 Punkte:

$$\frac{b-a}{4}[f(-1) + 2f(0) + f(1)] = 2 \cosh^2 \frac{1}{2} = 2.54308.., \quad \text{Fehler} = 0.193..$$

b) Simpsonregel, Ordnung 4, 3 Punkte:

$$\frac{b-a}{6}[f(-1) + 4f(0) + f(1)] = \frac{1}{3}[4 + 2\cosh 1] = 2.36205.., \quad \text{Fehler} = 0.012..$$

c) Gaußquadratur mit 2 Punkten, Ordnung 4. Bestimmung der Orthogonalpolynome: $\omega_0 = P_0 \equiv 1,\ \omega_1(x) = P_1(x) = x \perp \omega_0$, Ansatz $\omega_2 = x^2 + ax + b \perp \Pi_1$, d.h.

$$\alpha) \qquad 0 \stackrel{!}{=} \int_{-1}^1 [x^2 + ax + b] dx = \Big[\frac{x^3}{3} + a\frac{x^2}{2} + bx\Big]_{-1}^1 = \frac{2}{3} + 2b \ \Rightarrow b = -\frac{1}{3},$$

$$\beta) \qquad 0 \stackrel{!}{=} \int_{-1}^1 x[x^2 + ax + b] dx = \Big[\frac{x^4}{4} + a\frac{x^3}{3} + b\frac{x^2}{2}\Big]_{-1}^1 = \frac{2}{3}a \ \Rightarrow a = 0.$$

Dies ergibt $\omega_2(x) = x^2 - \frac{1}{3}$ mit den Nullstellen $x_0 = -\frac{1}{\sqrt{3}},\ x_1 = \frac{1}{\sqrt{3}} = 0.57735..$ Aus Symmetriegründen ist $\alpha_0 = \alpha_1 = 1$. Die Gaußnäherung hat daher den Wert

$$f(x_0) + f(x_1) = 2\cosh(x_1) = 2.342696.. \quad \text{Fehler} = -0.0077..$$

Je nach Art der im Integranden zu berücksichtigenden Singularität ($\rightarrow$ Gewichtsfunktion) ergeben sich verschiedene Polynomfamilien (vgl. folgende Tabelle). Die wichtigste Familie zur Gewichtsfunktion $g \equiv 1$ ist die der Legendrepolynome. Zur Vereinfachung wird meist das Standardintervall $[-1,1]$ zugrundegelegt, andere Intervalle können durch Variablensubstitution darauf zurückgeführt werden (wie (2.6.7)).

Polynom-Orthogonalfamilien nach Satz 2.4.1 und 3.3.1:

Intervall	Gewichtsfunkt.	Bezeichnung	Rekursionsformel
$[-1,1]$	$g \equiv 1$	Legendre-Pol.	$P_{n+1} = \frac{2n+1}{n+1} x P_n - \frac{n}{n+1} P_{n-1},\ P_0 = 1,\ P_1 = x$
$[-1,1]$	$g(x) = \sqrt{1-x^2}$	Tscheby. 2.Art	$U_{n+1} = 2xU_n - U_{n-1},\ U_0 = 1,\ U_1 = 2x$
$(-1,1)$	$g(x) = \frac{1}{\sqrt{1-x^2}}$	Tscheby. 1.Art	$T_{n+1} = 2xT_n - T_{n-1},\ T_0 = 1,\ T_1 = x$ Gewichte konstant, $\alpha_j = \frac{\pi}{n}$
$[0,\infty)$	$g(x) = e^{-x}$	Laguerre-Pol.	$L_{n+1} = \frac{2n+1-x}{n+1} L_n - \frac{n}{n+1} L_{n-1}$ $L_0 = 1,\ L_1 = 1-x$
$(-\infty,\infty)$	$g(x) = e^{-x^2}$	Hermite-Pol.	$H_{n+1} = 2xH_n - 2nH_{n-1},\ H_0 = 1,\ H_1 = 2x$

Über die angegebenen Polynomfamilien gibt es eine umfangreiche Literatur v.a. im Zusammenhang mit Reihenentwicklungen für Lösungen von Differentialgleichungen (z.B. Courant-Hilbert). Quadratur-Knoten und -Gewichte sind tabelliert, die Berechnung ist bei Stoer, §3.5, besprochen.

Eine explizite Darstellung von Orthogonalpolynomen ist mit Hilfe der Rodriguez-Formeln möglich. Die Legendre-Polynome, z.B., haben, bis auf Konstanten, die Gestalt

$$P_n(x) = \frac{n!}{(2n)!} \frac{d^n}{dx^n}(x^2-1)^n, \quad n = 0, 1, \dots\ . \tag{3.3.8}$$

Diese Formel ergibt sich aus den Orthogonalitätsbedingungen (3.3.2) durch partielle Integration. Aus dieser Darstellung (3.3.8) folgt übrigens direkt die Existenz von n reellen Nullstellen in $(-1,1)$ nach dem Satz von Rolle.

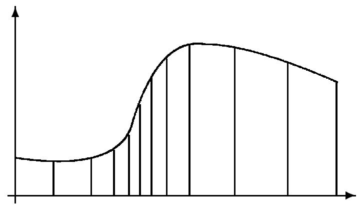
3.4 Adaptive Integration

Bisher ist die Frage völlig offen geblieben, wie in einem konkreten Fall, bei einem speziellen Integranden, das Quadraturverfahren und dessen Parameter (Schrittweite h, Ordnung) zu wählen sind. Im allgemeinen ist man daran interessiert, den Integralwert mit einer bestimmten Genauigkeit ε bei minimalem Aufwand zu bestimmen. Falls über den Integranden genügend analytische Information vorhanden ist, kann im Prinzip aus der Restglieddarstellung (3.1.4) die Ordnung der Formel und die Zahl der Teilintervalle so bestimmt werden, daß die Fehlerschranke den Wert ε nicht überschreitet.

Diese aufwendige Methode läßt sich umgehen, wenn die Quadraturformel mit einer Fehler(ab)schätzung verbunden wird. Dies soll bei der iterierten Trapezregel besprochen werden. Wenn das Intervall $[a,b]$ in n Teile $[x_{i-1}, x_i]$ zerlegt ist, gilt in jedem Teil

$$\int_{x_{i-1}}^{x_i} f(x)dx - \frac{1}{2}(x_i - x_{i-1})[f(x_{i-1}) + f(x_i)] = -\frac{1}{12}(x_i - x_{i-1})^3 f''(\xi_i) =: R[x_{i-1}, x_i]. \tag{3.4.1}$$

Der lokal auftretende Fehler setzt sich also aus der aktuellen Teilintervallänge und dem lokalen Wert der zweiten Ableitung zusammen. Eine effiziente Strategie bei der Quadratur sollte den vorgegebenen Fehler mit möglichst wenig Funktionsauswertungen erreichen. Im Bereich kleiner Ableitungen kann dazu mit großen Schrittweiten vorgegangen werden, während bei großen Ableitungswerten viel feinere Unterteilungen nötig sind.



Eine gute Strategie in diesem Sinn ist die der *Gleichverteilung* des Fehlers

$$\Big|R[x_{i-1},x_i]\Big| \overset{!}{\le} \frac{x_i-x_{i-1}}{b-a}\varepsilon,\ i=1,\dots,n. \tag{3.4.2}$$

Dann gilt nämlich

$$\left|\int_a^b f(x)dx - \sum_{i=1}^n \frac{x_i-x_{i-1}}{2}[f(x_{i-1})+f(x_i)]\right| \le \sum_{i=1}^n |R[x_{i-1},x_i]| \le \sum_{i=1}^n \frac{x_i-x_{i-1}}{b-a}\varepsilon = \varepsilon. \tag{3.4.3}$$

Zur Anwendung der Strategie (3.4.2) sind zwei Fragen zu klären:

a) Wie erhält man eine Abschätzung des lokalen Fehlers $R[x_{i-1},x_i]$?

b) Wie konstruiert man damit eine Unterteilung, die (3.4.2) erfüllt?

Zu a): Außer bei einfachen Integranden, bei denen $\|f''\|_{[x_{i-1},x_i]}$ abgeschätzt werden kann (evtl. mit Intervallrechnung), begnügt man sich mit einer *Schätzung* des lokalen Fehlers $R[x_{i-1},x_i]$. So gilt, z.B., mit $h_i := x_i - x_{i-1}$ für ein $\zeta_i \in (x_{i-1},x_i)$ die Aussage

$$f(x_{i-1}) - 2f\Big(\frac{x_{i-1}+x_i}{2}\Big) + f(x_i) = \frac{1}{4}h_i^2 f''(\zeta_i) \cong -\frac{3}{h_i}R[x_{i-1},x_i]. \tag{3.4.4}$$

Nachweis durch Taylorentwicklung um $\mu = \frac{1}{2}(x_{i-1}+x_i)$:

$$f(x_{i-1})-2f(\mu)+f(x_i) = f(\mu)-\frac{h_i}{2}f'(\mu)+\frac{h_i^2}{8}f''(\xi_-)-2f(\mu)+f(\mu)+\frac{h_i}{2}f'(\mu)+\frac{h_i^2}{8}f''(\xi_+).$$

Mit der Approximation (3.4.4) wird das Kriterium (3.4.2) *ersetzt* durch

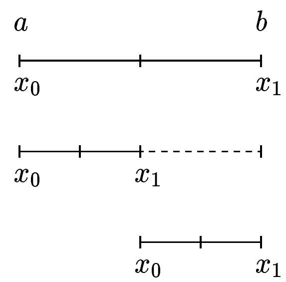
$$\Big|f(x_{i-1}) - 2f\Big(\frac{x_{i-1}+x_i}{2}\Big) + f(x_i)\Big| \overset{!}{\le} \frac{3\varepsilon}{b-a}. \tag{3.4.5}$$

Zu b): Die Gitterkonstruktion zur Einhaltung von (3.4.2) bzw. (3.4.5) kann ganz einfach über folgendes adaptive Verfahren geschehen:

— wähle $x_0 := a,\ x_1 := b$;

— Falls (3.4.5) verletzt ist, halbiere das Intervall: $x_1 := (x_0 + x_1)/2$;

— Falls (3.4.5) gilt, akzeptiere den Integralwert für $\int_{x_0}^{x_1}$ und fahre wie oben fort mit der Integration im Restintervall $\int_{x_1}^{b}$, d.h. mit $x_0 := x_1,\ x_1 := b$.



Der folgende Programmteil beschreibt diesen einfachen Algorithmus.

Algorithmus 3.4.1 Einfache adaptive Quadratur

$x_0 := a;\ f0 := f(x0);\ x1 := b;\ f1 := f(x1);$

$w := 0;\ e := 3\varepsilon/(b-a);$

repeat $xm := 0.5 * (x0 + x1);\ fm := f(xm);$

  if $abs(f0 - 2 * fm + f1) <= e$ then {akzeptieren}

  begin $w := w + 0.5 * (x1 - x0) * (f0 + f1);$

    $x0 := x1;\ f0 := f1;\ x1 := b;\ f1 := f(x1)$

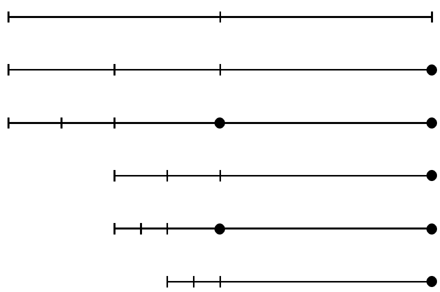
  end else {halbieren}

  begin $x1 := xm;\ f1 := fm$ end

until $x0 >= b;$

In der Praxis sind Funktionsauswertungen zu teuer, um sie im Falle eines nicht akzeptierten Teilintegrals zu verwerfen. Durch eine Einschränkung der Teilintervallängen auf die Werte $(b-a)2^{-j}, j \in \mathbf{N}$, und einfache Zusatzbedingungen kann erreicht werden, daß alle einmal *berechneten* Funktionswerte später auch *verwendet* werden. Die Verwaltung der gespeicherten Funktionswerte läßt sich elegant in einem *Keller*speicher (Stapel, stack, FILO) realisieren.

Beispiel:



halbieren
halbieren
akzeptieren
halbieren
akzeptieren
etc...

Schwarz markierte
Gitterwerte stehen im Keller

Bei professionellen Programmen wird außer der Schrittweite auch die Ordnung der Quadraturformel lokal gesteuert.

Demo-Beispiel: Adaptive Integration von

$$\int_0^1 e^x\,dx, \qquad \int_{-1}^1 \left(\sqrt{|x^2-0.25|} - \frac{1}{3}\right) dx, \qquad \int_0^1 \sin\frac{1}{x^2+0.05}\,dx.$$

3.5 Richardson-Extrapolation, Romberg-Verfahren

Im letzten Abschnitt wurde zum ersten Mal die theoretische Kenntnis des Fehlerverhaltens von Quadraturformeln praktisch ausgenutzt. Dies kann in noch viel stärkerem Maße zu einer erheblichen Verbesserung der Konvergenz verwendet werden. Dazu sei noch einmal an die Werte des Beispiels aus §3.2 erinnert:

Beispiel 3.5.1 : Iterierte Trapezregel zu $W := \int_1^2 \frac{1}{x}\,dx = \ln 2 = 0.6931472...$

h	1	1/2	1/4	1/8	1/16
T_h	0.75	0.70833	0.69702	0.694122	0.693391
Fehler	0.0568	0.01518	0.00387	0.000975	0.000244
Fehler-Quotient		3.74	3.92	3.99	3.996

Der Quotient aufeinanderfolgender Fehler konvergiert anscheinend gegen 4 für $h \to 0$,

$$\frac{W - T_h}{W - T_{h/2}} \to 4, \quad \text{d.h.,} \quad W - T_h - ah^2 = o(h^2).$$

Dabei ist a eine Konstante und $o(h^k)$, $(h \to 0)$, bezeichnet eine Funktion, die schneller gegen Null geht als h^k. Bei Vernachlässigung dieses Terms in der letzten Gleichung kann die unbekannte Konstante a bei Kenntnis von zwei Trapeznäherungen eliminiert werden in folgender Weise:

$$a \doteq \frac{1}{h^2}(W - T_h) \quad \Rightarrow \quad W \doteq T_{h/2} + a\frac{h^2}{4} \doteq T_{h/2} + \frac{h^2}{4}\frac{1}{h^2}(W - T_h), \text{ bzw.}$$

$$W \doteq \frac{4}{3}T_{h/2} - \frac{1}{3}T_h =: \widehat{T}_{h/2}. \tag{3.5.1}$$

Die auf diese Weise neu kombinierten Werte zum letzten Zahlenbeispiel lauten

$h =$ 1	1/2	1/4	1/8	1/16
$\widehat{T}_h=$	0.69444	0.6932	0.69316	0.693147

Diese besitzen ca. die doppelte Anzahl von gültigen Ziffern wie die ursprünglichen Trapezwerte. Der Erfolg des Vorgehens beruht auf folgender Eigenschaft.

Definition 3.5.2 *Die Größe T_h, $0 < h \in \mathbf{R}$, besitzt eine* asymptotische Entwicklung *(nach Potenzen von h^k), wenn $q \in \mathbf{N}$ und von h unabhängige Koeffizienten $a_j \in \mathbf{R}$ existieren so, daß*

$$T_h = W + a_1 h^k + a_2 h^{2k} + \ldots + a_q h^{qk} + \mathcal{O}(h^{qk+1}), \ h \to 0. \tag{3.5.2}$$

Bei der Trapezregel ist $k = 2$ (s.u.), die Linearkombination (3.5.1) liefert mit (3.5.2) die Entwicklung

$$\begin{aligned} \widehat{T}_{h/2} = \frac{4}{3}T_{h/2} - \frac{1}{3}T_h &= W + a_1 h^2(\frac{4}{3}\frac{1}{4} - \frac{1}{3}) + a_2(\frac{4}{3}\frac{1}{16} - \frac{1}{3}) + \ldots \\ &= W - \frac{1}{4}a_2 h^4 + \ldots \end{aligned}$$

In der Entwicklung wurde also der h^2-Fehleranteil eliminiert, $\widehat{T}_h$ ist jetzt eine Näherung der Ordnung 4. Genauer gilt $\widehat{T}_{h/2} = W + \hat{a}_2(\frac{h}{2})^4 + \ldots$. Daher kann das *Extrapolationsverfahren* von oben noch einmal auf $\widehat{T}$ angewendet werden. Bevor dieses Verfahren systematischer untersucht wird, sei zunächst die Aussage (3.5.2) für die Trapezregel formuliert.

Satz 3.5.3 (Euler-McLaurin-Summenformel) *Für $f \in C^{2q+2}[a,b]$ besitzt die iterierte Trapezregel (3.2.6) folgende h^2-Entwicklung, $h = (b-a)/n$,*

$$\begin{aligned} T_h &= \frac{h}{2}[f_0 + 2f_1 + \ldots + 2f_{n-1} + f_n] = \int_a^b f(x)dx \\ &\quad + \sum_{j=1}^{q} b_j h^{2j}\left[f^{(2j-1)}(b) - f^{(2j-1)}(a)\right] + c_{2q+2}h^{2q+2}f^{(2q+2)}(\xi), \ \xi \in (a,b). \end{aligned} \tag{3.5.3}$$

Beweis (-Idee, vgl. Stoer) Auf einem einzelnen Intervall, z.B. $[0,h]$, gilt

$$\frac{h}{2}[f(0)+f(h)] = \Big[(x-\frac{h}{2})f(x)\Big]_0^h = \int_0^h \Big[(x-\frac{h}{2})f(x)\Big]' \,dx = \int_0^h f(x)dx + \int_0^h (x-\frac{h}{2})f'(x)dx.$$

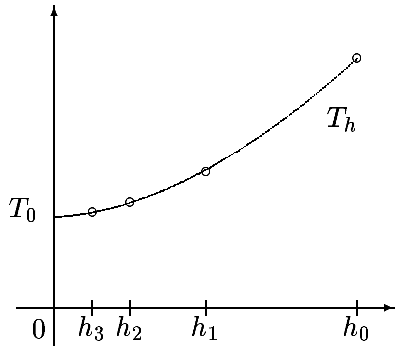
Hier kann nun wieder partiell integriert werden. Dabei lassen sich die Integrationskonstanten so wählen, daß Terme mit ungeraden h-Potenzen nicht auftreten. Im ersten Schritt etwa ist $p_2(x) := \frac{1}{2}(x^2 - hx + h^2/6)$ eine Stammfunktion von $(x - \frac{h}{2})$ mit $\int_0^h p_2(x)dx = 0$. Daher gilt

$$\frac{h}{2}[f(0)+f(h)] = \int_0^h f(x)dx + \frac{h^2}{12}[f'(h)-f'(0)] + \int_0^h p_3(x)f'''(x)dx,$$

also $b_1 = 1/12$, wobei $p_3' = p_2$, $p_3(0) = p_3(h) = 0$. Das Gesamtergebnis (3.5.3) folgt durch Summation über die Teilintervalle. ∎

Bemerkung: Der Summenformel entnimmt man, z.B., auch folgende Aussage: Wenn der Integrand $(b-a)$-periodisch ist, besitzt die Trapezregel sogar maximale Ordnung $2q+2$!

Folgendes Prinzip liegt den Extrapolationsverfahren zugrunde. Es soll eine Größe $W = \lim_{h\to 0} T_h =: T_0$ bestimmt werden, wobei T_h aber nur für (einzelne) Werte $h > 0$ tatsächlich berechnet werden kann. Anstatt nun die Elemente einer Folge von Werten $T_{h_0}, T_{h_1}, \ldots$ unabhängig voneinander zu betrachten, ist es günstiger, als Näherung für T_0 den Wert des Interpolationspolynoms zu $(h_0, T_{h_0}), \ldots, (h_m, T_{h_m})$ an der Stelle $h = 0$ heranzuziehen. Zur Auswertung des Polynoms in 0 eignet sich am besten der *Neville-Algorithmus* (2.3.5).



Dazu wird (3.5.2) als Polynom in h^k aufgefaßt. Für die Werte $p_{ij} = p_{ij}(0)$ ergibt sich daraus der folgende Algorithmus der *Richardson-Extrapolation:*

$$\begin{aligned} p_{i0} &:= T_{h_i},\ i = 0,\ldots,m \\ p_{ij} &:= p_{i+1,j-1} + \frac{p_{i+1,j-1} - p_{i,j-1}}{(h_i/h_{i+j})^k - 1}, \quad \left\{ \begin{array}{l} i = 0,\ldots,m-j \\ j = 1,\ldots,m \end{array} \right. . \end{aligned} \tag{3.5.4}$$

Die Berechnung erfolgt wieder in einem Dreieckschema spalten- oder zeilenweise. In der Praxis werden meist feste Schrittweitenfolgen, z.B. $h_i/h_{i+j} = 2^j$, verwendet. Dann benötigt (3.5.4) nur 3 Operationen pro Schritt. Die durch die Extrapolation (3.5.4) erreichte Erhöhung der Konvergenzordnung ergibt sich aus

Satz 3.5.4 *Wenn für die Größe T_h eine asymptotische Entwicklung (3.5.2) existiert, dann gilt für die nach (3.5.4) mit $h_0 > h_1 > \ldots$ extrapolierten Werte die Darstellung*

$$p_{ij} = T_0 + h_i^k \cdots h_{i+j}^k[(-1)^j a_{j+1} + \mathcal{O}(h_i^k)], \tag{3.5.5}$$

$h_0 \to 0$, $j < q$. *Bei einem konstanten Schrittweitenverhältnis $v = h_i/h_{i-1} < 1$ ergibt sich insbesondere*

$$p_{ij} = T_0 + h_{i+j}^{k(j+1)}[(-1)^j v^{-j(j-1)/2} a_{j+1} + \mathcal{O}(h_{i+j}^k)], \tag{3.5.6}$$

$h_0 \to 0,\ j < q$. Die Ordnung der Näherung hat sich also auf $h^{k(j+1)}$ erhöht.

Beweis Alle Größen werden als Funktionen von $s := h^k$ interpretiert. Da die Polynominterpolation eine lineare Operation darstellt, kann man sich beim Verfahren (3.5.4) auf die Betrachtung eines Summanden in (3.5.2) beschränken. Der Wert des Interpolationspolynoms $q_{l,j}(s)$ vom Grad j zu einem Monom

$$h^{lk} = s^l, \quad 1 = s^0,$$

im Punkt $s = 0$ ist $q_{0j}(0) = 1$ (für $l = 0$) bzw. $q_{lj}(0) = 0$ (für $0 < l \le j$). Dagegen entspricht der Wert für $l = j+1$ gerade dem Interpolationsfehler bei $g(s) := s^{j+1}$, nach (2.3.11) also

$$p_{j+1,j}(0) = g(0) - (0 - s_i)\cdots(0 - s_{i+j})g[s_i, \ldots, s_{i+j}, 0] = (-1)^j s_i \cdots s_{i+j}\frac{(j+1)!}{(j+1)!}.$$

Daher hat das Interpolationspolynom vom Grad j zu $T_0 + a_1 s + \ldots + a_{j+1}s^{j+1}$ in $s = 0$ den Wert $T_0 + (-1)^j s_i \cdots s_{i+j} a_{j+1}$. Dies entspricht der Behauptung. ■

Praktische Durchführung

1. In der Trapezregel brauchen nach einer Schrittweitenhalbierung bei der Berechnung von $T_{h/2}$ die in T_h enthaltenen Funktionswerte nicht mehr neu berechnet zu werden. Bei Extrapolation sind daher folgende Schrittweitenfolgen günstig.

   a) *Romberg-Folge*: $h_0 = \beta - \alpha,\ h_i = 2^{-i}h_0$. Diese Quadratur-Methode heißt *Romberg-Verfahren.* In Satz 3.5.4 gilt

   $$p_{ij} = T_0 + h_{i+j}^{2j+2}\bar{a}_{j+1} + \ldots\ .$$

   Wenn hiermit hohe Ordnungen erreicht werden sollen, werden allerdings sehr viele Funktionsauswertungen benötigt, nämlich $2^j + 1$ für p_{0j}. Eine Formel der Ordnung 16 etwa verwendet hier 129 Punkte, während die Gaußformel dafür mit nur 8 Punkten auskommt!

   b) *Bulirsch-Folge*: $h_0 = \beta - \alpha$ und für $i = 1, 2, \ldots$

   $$h_i/h_0 = \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \frac{1}{6}, \frac{1}{8}, \frac{1}{12}, \ldots = \begin{cases} 2^{-(i+1)/2}, & i \text{ ungerade}, \\ \frac{2}{3}2^{-i/2}, & i \text{ gerade}. \end{cases}$$

   Die Anzahl der für eine bestimmte Ordnung benötigten Stützstellen wächst hier langsamer, Ordnung 16, z.B., ist jetzt mit 25 Punkten erreichbar.

2. Wie bei den anderen Quadraturverfahren kennt man i. a. nicht die günstigste Ordnung (etwa die Differenzierbarkeitsordnung von f). Daher sollte die Durchführbarkeit der Extrapolation in der Praxis durch Überwachung der Konvergenzaussage (3.5.5), (3.5.6) sichergestellt werden. Für die Romberg-Folge gilt nach (3.5.6):

   $$p_{ij} = T_0 + h_{i+j}^{2j+2}\bar{a}_{j+1} + h_{i+j}^{2j+4}\bar{a}_{j+2} + \ldots\ .$$

Daher ist

$$\begin{aligned} p_{i,j} - p_{i-1,j} &= h_{i+j}^{2j+2}(1 - 2^{2j+2})\bar{a}_{j+1} + h_{i+j}^{2j+4}(1 - 2^{2j+4})\bar{a}_{j+2} + \dots \\ p_{i+1,j} - p_{i,j} &= h_{i+j}^{2j+2}(2^{-2j-2} - 1)\bar{a}_{j+1} + h_{i+j}^{2j+4}(2^{-2j-4} - 1)\bar{a}_{j+2} + \dots \end{aligned}$$

Für den Quotienten dieser Differenzen gilt daher

$$\frac{p_{i,j} - p_{i-1,j}}{p_{i+1,j} - p_{i,j}} = \frac{(1 - 2^{2j+2})\bar{a}_{j+1} + h_{i+j}^2(\dots)\bar{a}_{j+2} + \dots}{(2^{-2j-2} - 1)\bar{a}_{j+1} + h_{i+j}^2(\dots)\bar{a}_{j+2} + \dots} = 2^{2j+2} + \mathcal{O}(h_{i+j}^2).$$

Daher sollte bei einer zu großen Abweichung

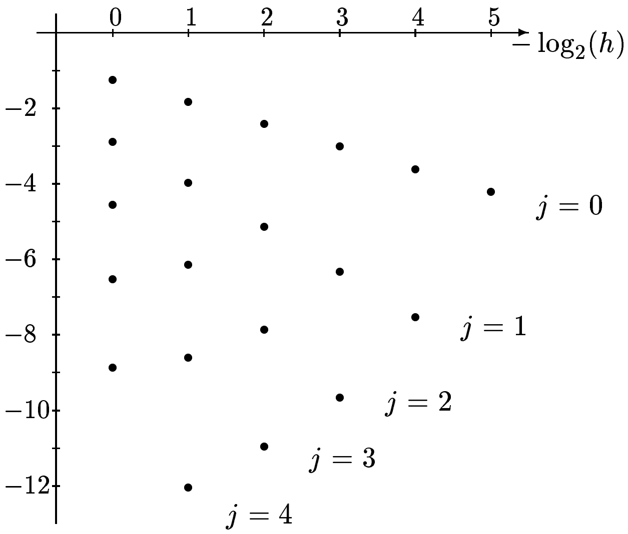
$$2^{-2j-2}\frac{p_{i,j} - p_{i-1,j}}{p_{i+1,j} - p_{i,j}} - 1$$

auf eine weitere Extrapolation $j \to j+1$ verzichtet werden.

Beispiel 3.5.5 $\int_1^2 \frac{1}{x} dx = \ln 2 = 0.69314718056$, Rombergfolge. Die Tabelle zeigt die Werte p_{ij}, $i = 1, .., 5$, $0 \le j \le i$.

h	j=0	1	2	3	4
1/2	0.70833333333	0.69325396825	0.69314790148	0.69314718307	0.69314718056
1/4	0.69702380952	0.69315453065	0.69314719430	0.69314718057	
1/8	0.69412185037	0.69314765282	0.69314718079		
1/16	0.69339120221	0.69314721029			
1/32	0.69320820827				

Der Wert p_{14} ist auf alle 11 Stellen genau. In der rechtsstehenden Graphik wird die Genauigkeit der Näherungen in den verschiedenen Spalten dieser Tabelle gezeigt. An der horizontalen Achse ist der negative Zweier-Logarithmus der Schrittweiten abgetragen, an der vertikalen der Zehner-Logarithmus der Fehler. Die Ordnung der Näherungen in einer Spalte der Tabelle läßt sich an der Steigung der Punktereihen ablesen.



Bemerkung: 1) Die Richardson-Extrapolation beruht allein auf der *Existenz* einer asymptotischen Entwicklung (3.5.2). Solche Entwicklungen existieren auch bei vielen anderen Verfahren (z.B. den Differenzenformeln in §3.6). Dort läßt sich die Extrapolation daher analog einsetzen.

2) Mit dem Romberg-Verfahren kann bei adaptiver Integration auch einfach eine Ordnungssteuerung auf den einzelnen Teilintervallen implementiert werden.

3.6 Numerische Differentiation

Zur Approximation von Ableitungen einer Funktion kann nach dem gleichen Prinzip wie bei den einfachen Quadraturformeln vorgegangen werden durch Verwendung von Ableitungen des Interpolationspolynoms:

$$f^{(k)}(\hat{x}) \cong p_n^{(k)}(\hat{x}) = \sum_{j=0}^{n} f(x_j) L^{(k)}(\hat{x}).$$

Da hier weniger Gestaltungsmöglichkeiten bestehen als in §3.1, werden nur kurz einige Beispiele und Besonderheiten beim Fehlerverhalten besprochen. Im folgenden sei $[\alpha, \beta]$ ein (beliebig kleines) Intervall, das die Stelle $\hat{x}$ enthält.

Satz 3.6.1 *Es sei $p_n \in \Pi_n$ das Interpolationspolynom zu $f \in C^{n+1}[\alpha, \beta]$ und einfachen Stützstellen $x_j \in [\alpha, \beta]$, $j = 0, \ldots, n$. Dann gilt für $k \leq n$ und $\hat{x} \in [\alpha, \beta]$ die Aussage*

$$\begin{aligned} p_n^{(k)}(\hat{x}) - f^{(k)}(\hat{x}) &= r_{kn}(\hat{x}) \qquad \textit{mit} \\ r_{kn}(x) &= (x - x_0^{(k)}) \cdots (x - x_{n-k}^{(k)}) \frac{f^{(n+1)}(\xi_k)}{(n+1-k)!}, \end{aligned} \tag{3.6.1}$$

$\xi_k \in (\alpha, \beta)$, wobei $\alpha < x_0^{(k)} < \ldots < x_{n-k}^{(k)} < \beta$. Mit $\varrho := \max_{j=0}^n |\hat{x} - x_j| \leq \beta - \alpha$ gilt daher

$$|r_{kn}(\hat{x})| \leq \frac{\varrho^{n+1-k}}{(n+1-k)!} \|f^{(n+1)}\|_\infty.$$

Beweis Iterierter Satz von Rolle, vgl. [Stummel-Hainer]

Bemerkung: Nach (3.6.1) gibt es offensichtlich in $[\alpha, \beta]$ Stellen, in denen der Fehler verschwindet. Durch geeignete Wahl der Stützstellen kann $\hat{x}$ in die Nähe solcher Punkte gebracht werden um, wie bei der Quadratur, eine bessere Konvergenz zu erhalten. Zur Untersuchung wird wieder eine zusätzliche Interpolationsstelle x_{n+1} herangezogen. In der Newtondarstellung ist

$$f(x) = p_n(x) + \omega_{n+1}(x) f[x_0, \ldots, x_{n+1}] + r_{n+1}(x),$$

$\omega_{n+1}(x) = (x - x_0) \cdots (x - x_n)$. Aus dem letzten Satz folgt dann

$$f^{(k)}(\hat{x}) - p_n^{(k)}(\hat{x}) = \omega_{n+1}^{(k)}(\hat{x}) f[x_0, \ldots, x_{n+1}] + r_{k,n+1}(\hat{x}), \tag{3.6.2}$$

mit $|r_{k,n+1}(\hat{x})| \leq \varrho^{n+2-k} \|f^{(n+2)}\|_\infty / (n+2-k)!$. Daher wird also bei Auswertung in den Nullstellen von $\omega_{n+1}^{(k)}$ eine um eins erhöhte Konvergenzordnung erreicht, wenn $\varrho \to 0$. Für Approximationen mit minimaler Knotenzahl, d.h. $n = k$, ergibt sich dazu

$$\omega_{n+1}(x) = x^{n+1} - \Big(\sum_{j=0}^{n} x_j\Big) x^n + \ldots \;\Rightarrow\; \omega_{n+1}^{(n)}(x) = (n+1)!\, x - n! \sum_{j=0}^{n} x_j.$$

Die Ableitung $p_n^{(n)} \equiv n! f[x_0, \ldots, x_n]$ ist konstant (vgl. (2.3.10)). Wegen (3.6.2) ist dieser Wert eine besonders gute Approximation für die Ableitung $f^{(n)}(x)$ im Mittelpunkt der Knoten

$$\hat{x} = \frac{1}{n+1} \sum_{j=0}^{n} x_j, \tag{3.6.3}$$

z.B., wenn die Knoten um $\hat{x}$ symmetrisch verteilt sind. In diesem Fall gilt also

$$f^{(n)}(\hat{x}) - n!f[x_0, \ldots, x_n] \leq \frac{\varrho^2}{2}\|f^{(n+2)}\|_\infty.$$

Bei äquidistanten Knoten $x_j = x_0 + jh$ ist $\varrho \leq nh$, in den einfachsten Fällen ergeben sich folgende Approximationen (sog. *Differenzenformeln*) für $\hat{x} = 0$:

$$\begin{aligned}
f'(0) &= \frac{1}{h}[f(h) - f(0)] + chf''(\xi) \\
f'(0) &= \frac{1}{2h}[f(h) - f(-h)] + ch^2 f^{(3)}(\xi), \quad \text{vgl. (3.6.3)} \\
f''(0) &= \frac{1}{h^2}[f(-h) - 2f(0) + f(h)] + ch^2 f^{(4)}(\xi), \quad \text{vgl. (3.6.3)}
\end{aligned}$$

Eine Formel höherer Ordnung ist

$$f'(0) = \frac{1}{12h}[f(-2h) - 8f(-h) + 8f(h) - f(2h)] + ch^4 f^{(5)}(\xi).$$

Die Differenzenformeln sind umso genauer, je kleiner die Schrittweite h ist. Allerdings wird dann in der Formel durch die kleinen Zahlen h oder h^2 dividiert. Daher wird der bei der Berechnung der Werte $f(x_j)$ im Computer gemachte Fehler vergrößert und macht die Verwendung sehr kleiner Schrittweiten unsinnig. Dies soll an der symmetrischen ersten Differenz erläutert werden, detaillierter wird das Problem in §6 untersucht. Jetzt seien $\tilde{f}(x_j) = f(x_j) + \epsilon_j$ die tatsächliche berechneten Werte, deren Fehler durch eine kleine Konstante ϵ (ca. Maschinengenauigkeit, z.B. 10^{-11}) beschränkt sind, $|\epsilon_j| \leq \epsilon$. Dann gilt für den Gesamtfehler der *berechneten* Approximation

$$\left|\frac{1}{2h}[\tilde{f}(h) - \tilde{f}(-h)] - f'(0)\right| = \left|ch^2 f''(\xi) + \frac{\epsilon_1 - \epsilon_0}{2h}\right| \leq \underbrace{ch^2\|f''\|_\infty}_{\searrow 0} + \underbrace{\frac{\epsilon}{h}}_{\nearrow\infty} \quad (h \to 0). \tag{3.6.4}$$

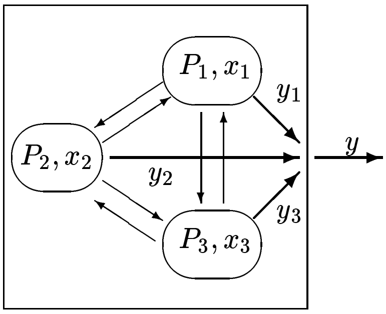
Die Schranke wird minimal bei $\hat{h} \cong \epsilon^{1/3}$ und einem Minimalwert $\cong \epsilon^{2/3}$. Man kann also bei diesem Beispiel nicht erwarten, eine Genauigkeit von mehr als $\frac{2}{3}$ der im Computer verfügbaren Stellenzahl zu erreichen. Allgemein ist die erreichbare Genauigkeit bei Formeln höherer Konvergenzordnung besser.

4 Lineare Gleichungssysteme

Viele Näherungsverfahren der Numerik im Bereich der Differentialgleichungen, Integralgleichungen, etc. führen auf *Lineare Gleichungssysteme* (LGSe). In anderen Anwendungen können Probleme direkt in dieser Form mathematisiert werden.

4.1 Beispiele

Beispiel 4.1.1 *Input-Output-Analyse*: Zur Analyse der Wechselwirkungen in einer Volkswirtschaft und der Auswirkungen geänderter Nachfrage wird die Wirtschaft in Produktionsbereiche $P_j,\ j = 1, \ldots, n$, unterteilt. Mit y_j wird die Endnachfrage nach Produkten aus dem Bereich P_j bezeichnet, zusammengefaßt zu $y \in \mathbf{R}^n$. Bei der Bestimmung der dazu erforderlichen Produktionsmengen x_i (in P_i) muß berücksichtigt werden, daß zur Herstellung eines Produktes in P_i Produkte aus anderen Bereichen benötigt werden (Auto: Maschinen, Stahl, Kunststoff, Elektroteile).



Diese Beziehungen faßt man in einer Bedarfsmatrix $B = (b_{ij})$ zusammen:

$$b_{ij} : \begin{cases} \text{Bedarf an Produkten aus } P_i \text{ zur Produktion einer} \\ \text{Einheit (Geldwert) in } P_j,\ b_{ij} \geq 0. \end{cases} \tag{4.1.1}$$

Die benötigte Gesamtproduktion $x = (x_i) \in \mathbf{R}^n$ ergibt sich also aus der Bilanzgleichung

$$x = \underbrace{y}_{\text{End-Nachfrage}} + \underbrace{Bx}_{\text{interne Nachfrage}}$$

$$\Leftrightarrow \qquad x - Bx = y \ \Leftrightarrow\ (I - B)x = y. \tag{4.1.2}$$

Somit beschreibt die Matrix $(I - B)^{-1}$ die Auswirkungen einer sich ändernden Endnachfrage y.

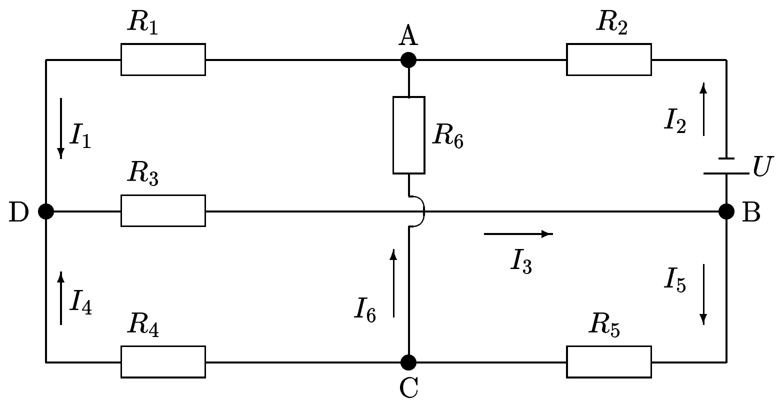
Beispiel 4.1.2 *Elektrisches Netzwerk*: Sind in einem Bauwerk die Leitungen stark genug, um eine Überhitzung zu vermeiden? Zur Untersuchung dieser Frage sind die angelegten Einzelspannungen U_k mit den Teilwiderständen R_k und den Teilströmen I_k zu verknüpfen. Das geschieht in den Kirchhoff'schen Regeln:

a) *Knotenregel:* Unter Beachtung der Richtung aller auf einen Knoten zu- bzw. abfließenden Ströme I_k gilt für jeden Knoten

$$\sum_k I_k = 0$$

b) *Maschenregel:* Für jeden geschlossenen Teilstromkreis gilt (wieder unter Berücksichtigung der Stromrichtungen)

$$\sum_k I_k R_k = \sum_k U_k$$



Für das abgebildete Netzwerk gilt somit das folgende LGS:

$$\text{Knoten}\qquad \begin{array}{lcccccccl} A: & -I_1 & +I_2 & & & & +I_6 & = & 0 \\ B: & & -I_2 & +I_3 & & -I_5 & & = & 0 \\ C: & & & & -I_4 & +I_5 & -I_6 & = & 0 \\ D: & I_1 & & -I_3 & +I_4 & & & = & 0 \end{array} \tag{4.1.3}$$

$$\text{Maschen}\qquad \begin{array}{lcccccccl} ABC: & & I_2R_2 & & & -I_5R_5 & -I_6R_6 & = & U \\ ADB: & I_1R_1 & +I_2R_2 & +I_3R_3 & & & & = & U \\ ADC: & I_1R_1 & & & -I_4R_4 & & +I_6R_6 & = & 0 \\ BDC: & & & -I_3R_3 & -I_4R_4 & -I_5R_5 & & = & 0 \end{array} \tag{4.1.4}$$

Die Gleichungen sind sicher *nicht linear unabhängig.* Man wird daher versuchen die linear abhängigen Gleichungen zu eliminieren. In Matrixform geht (4.1.3,4.1.4) über in

$$\begin{array}{c} A \\ B \\ C \\ D \\ ACB \\ ADB \\ ADC \\ BDC \end{array} \begin{pmatrix} -1 & 1 & 0 & 0 & 0 & 1 \\ 0 & -1 & 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 & 1 & -1 \\ 1 & 0 & -1 & 1 & 0 & 0 \\ 0 & R_2 & 0 & 0 & -R_5 & -R_6 \\ R_1 & R_2 & R_3 & 0 & 0 & 0 \\ R_1 & 0 & 0 & -R_4 & 0 & R_6 \\ 0 & 0 & -R_3 & -R_4 & -R_5 & 0 \end{pmatrix} \begin{pmatrix} I_1 \\ I_2 \\ I_3 \\ I_4 \\ I_5 \\ I_6 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ U \\ U \\ 0 \\ 0 \end{pmatrix} \tag{4.1.5}$$

Eine Addition der Gleichungen eins, zwei, drei, vier bzw. fünf, sieben und Subtraktion von sechs und acht ergibt jeweils null. Damit bleiben sechs Gleichungen mit sechs Unbekannten.

Beispiel 4.1.3 *Differenzenverfahren für ein Randwertproblem:* Im Anschluß an die Einleitung wird das Randwertproblem

$$\begin{gathered} z''(x) + f(x)z(x) = g(x) \\ z(0) = \alpha, z(1) = \beta \end{gathered} \tag{4.1.6}$$

betrachtet. Mit den Bezeichnungen

$$z_j \approx z(x_j), \quad x_j = j \cdot h, \quad h = \frac{1}{m}, \quad f_j := f(x_j), \quad g_j := g(x_j),$$

wird dazu das lineare Gleichungssystem

$$\frac{z_{j+1} - 2z_j + z_{j-1}}{h^2} + f_j z_j = g_j \qquad j = 1, \cdots, m-1$$
$$z_0 = \alpha, \quad z_n = \beta. \tag{4.1.7}$$

gelöst. Dieses System führt auf eine sogenannte *Tridiagonalmatrix*

$$\begin{pmatrix} -2+h^2 f_1 & 1 & 0 & \cdots & 0 \\ 1 & -2+h^2 f_2 & 1 & & \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ & & 1 & -2+h^2 f_{m-2} & 1 \\ 0 & \cdots & 0 & 1 & -2+h^2 f_{m-1} \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_{m-2} \\ z_{m-1} \end{pmatrix} = h^2 \begin{pmatrix} g_1 - \alpha/h^2 \\ g_2 \\ \vdots \\ g_{m-2} \\ g_{m-1} - \beta/h^2 \end{pmatrix}.$$

Matrizen dieser Form spielen daher im weiteren eine wesentliche Rolle.

Bemerkung: Auch bei anderen Probleme kommt man auf Bilanzgleichungen wie in Beispiel 1. In der Computergrafik, z.B., ergibt sich die absolute Helligkeit und Farbe eines Flächenelements aus der direkt und indirekt empfangenen Lichtmenge.

Diese Beispiele führen sofort auf weitere Fragen:
— wann besitzen die LGSe eine Lösung?
— Wie löst man solche LGSe numerisch zweckmäßig auf?
— Spiegeln sich spezielle Eigenschaften der rechten Seite, z.B. positive Nachfrage y bzw. Spannung U in Beispiel 1 bzw. 2, auch in der Lösung x bzw. $(I_1, \ldots, I_6)$ wieder?

4.2 Hilfsmittel

Im $\mathbf{R}^n$ bzw. $\mathbf{C}^n$ (Sammelbegriff $\mathbf{K}^n$) werden Spaltenvektoren betrachtet

$$x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}.$$

Eine wichtige Klasse von Normen sind, mit $p \in \mathbf{R}$, $p \geq 1$, die Hölder-Normen

$$\|x\|_p := \left(\sum_{j=1}^{n} |x_j|^p \right)^{1/p}, \tag{4.2.1}$$

mit den wichtigsten Vertretern

$$\|x\|_1 := \sum_{j=1}^{n} |x_j| \qquad \text{Summennorm} \tag{4.2.2}$$

$$\|x\|_2 := \sqrt{\sum_{j=1}^{n} |x_j|^2} \qquad \text{Euklidnorm} \tag{4.2.3}$$

$$\|x\|_\infty := \max_{j=1}^{n} |x_j|, \qquad \text{Maximumnorm} \tag{4.2.4}$$

Mit diesen Normen $\|\cdot\|_p$ ist $\mathbf{K}^n$ ein Banachraum. Da alle Normen im $\mathbf{K}^n$ äquivalent sind, existieren Konstanten mit

$$\|x\|_p \le c_{pq}\|x\|_q \quad \forall x \in \mathbf{K}^n,\ p,q \ge 1.$$

Diese c_{pq} hängen im allgemeinen aber von n ab. Lineare Abbildungen von $\mathbf{K}^n \to \mathbf{K}^m$ werden (in der Einheits-Basis) durch Matrizen

$$A = (a_{ij})_{i=1,j=1}^{m,n} \in \mathbf{K}^{m\times n}$$

beschrieben. Jedes Paar von Vektornormen in $\mathbf{K}^m, \mathbf{K}^n$ (mit gleichem Index p) *induziert* eine zugehörige Matrixnorm, die mit dem gleichen Symbol bezeichnet wird

$$\|A\|_p := \sup_{x\ne 0} \frac{\|Ax\|_p}{\|x\|_p} = \sup_{\|x\|_p=1} \|Ax\|_p. \tag{4.2.5}$$

In (4.2.5) wird das Supremum als Maximum angenommen. Die durch (4.2.2ff) *induzierten Normen* sind:

$$\|A\|_\infty = \max_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}| \qquad \text{Zeilensummennorm}, \tag{4.2.6}$$

$$\|A\|_1 = \max_{j=1}^{n} \sum_{i=1}^{m} |a_{ij}| \qquad \text{Spaltensummennorm}, \tag{4.2.7}$$

$$\|A\|_2 = \rho(A^*A)^{1/2} = \sqrt{\max_{j=1}^{n} \lambda_j(A^*A)} \qquad \text{Spektralnorm}, \tag{4.2.8}$$

wobei $\lambda_j(A^*A)$ die Eigenwerte von A^*A sind und $A^* = \bar{A}^T$. Eine Matrixnorm heißt *verträglich* mit den Vektornormen, wenn

$$\|Ax\| \le \|A\|\|x\| \quad \forall x \in \mathbf{K}^n,\ A \in \mathbf{K}^{m\times n}. \tag{4.2.9}$$

Als obere Schranke für $\|A\|_2$ betrachtet man oft die einfachere

$$\|A\|_F := \left(\sum_{i=1}^{m}\sum_{j=1}^{n} |a_{ij}|^2\right)^{1/2} \qquad \text{Frobeniusnorm}. \tag{4.2.10}$$

Sie ist verträglich mit der Euklidnorm, denn

$$\|Ax\|_2 \le \|A\|_F\|x\|_2.$$

Allerdings ist sie nicht die induzierte Norm, da z.B. $\|I\|_F = \sqrt{n} > \|I\|_2 = 1$. Mit I wird die Einheitsmatrix, $I = (\delta_{ij})$ bezeichnet. Für induzierte Nomen und für $\|\cdot\|_F$ gilt die Produktformel

$$\|AB\| \le \|A\|\,\|B\| \tag{4.2.11}$$

In dieser Vorlesung werden nur Normen benutzt, *die (4.2.11) erfüllen.* Für viele Überlegungen zu quadratischen Matrizen $A \in \mathbf{K}^{n\times n}$ ist der *Spektralradius*

$$\rho(A) := \max\{|\lambda_j(A)| :\ \lambda_j(A) \text{ EW zu } A\} \tag{4.2.12}$$

eine wichtige Größe. Wegen $\rho\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} = 0$ ist $\rho(A)$ sicherlich keine Norm. Der folgende Satz zeigt aber, daß der Spektralradius eine untere Schranke für jede beliebige Matrixnorm ist, sich von geeigneten Normen beliebig wenig unterscheidet und für diagonalisierbare Matrizen sogar gleich sein kann.

Satz 4.2.1 *a) Für jede Matrixnorm $\|\cdot\|$ gilt*

$$\rho(A) \leq \|A\|.$$

b) Für jede Matrix A und jedes $\varepsilon > 0$ existiert eine (spezielle) Norm mit

$$\|A\|_M \leq \rho(A) + \varepsilon. \tag{4.2.13}$$

c) Für diagonalisierbare Matrizen A kann in (4.2.13) $\varepsilon = 0$ gewählt werden, für hermitesche, symmetrische und normale Matrizen gilt $\|A\|_2 = \rho(A)$.

Beweis a) x sei Eigenvektor von $A \Rightarrow \ |\lambda|\|x\| = \|\lambda x\| = \|Ax\| \leq \|A\|\|x\|$.
b) Die Jordan-Normalform von A sei

$$\Lambda = SAS^{-1} = \begin{pmatrix} \lambda_1 & \delta_1 & & \\ & \lambda_2 & \delta_2 & \\ & & \ddots & \\ & & & \lambda_n \end{pmatrix}$$

mit $\delta_i \in \{0,1\}, i = 1,\ldots,n-1$. Eine weitere Ähnlichkeitstransformation mit der Diagonalmatrix

$$P = \begin{pmatrix} \varepsilon & & & \\ & \varepsilon^2 & & \\ & & \ddots & \\ & & & \varepsilon^n \end{pmatrix}$$

führt Λ über in

$$\tilde{\Lambda} = P^{-1}\Lambda P = (P^{-1}S)A(S^{-1}P).$$

Da $\tilde{\Lambda}$ die gleiche Hauptdiagonale wie Λ, in der Nebendiagonale aber Elemente $\varepsilon\delta_j$ hat, gilt

$$\|\tilde{\Lambda}\|_\infty = \begin{cases} \max_{j=1}^n\{|\lambda_j| + \varepsilon|\delta_j|\} \leq \rho(\tilde{\Lambda}) + \varepsilon = \rho(A) + \epsilon, & \text{wenn ein } \delta_j \neq 0, \\ \max_{j=1}^n\{|\lambda_j|\} = \rho(\tilde{\Lambda}) = \rho(A), & \text{wenn alle } \delta_j = 0. \end{cases}$$

Für die Matrixnorm

$$\|B\|_M := \|MBM^{-1}\|_\infty, \quad M := P^{-1}S,$$

gilt also b), bzw. c) im Fall $\delta_j \equiv 0$. ■

Mit Satz 4.2.1 kann eine Aussage über die Lösbarkeit von Gleichungssystemen mit Matrix $A = I - B$ gemacht werden.

Satz 4.2.2 *(Neumannreihe) Unter der Voraussetzung*

$$\rho(B) < 1 \tag{4.2.14}$$

- *ist das Gleichungssystem* $(I - B)x = b$ *eindeutig lösbar,*

- *ist* $(I - B)^{-1} = \sum\limits_{j=0}^{\infty} B^j$,

- *gibt es eine Matrixnorm so, daß* $\|B\| < 1$ *ist. Für diese gilt*

$$\|(I - B)^{-1}\| \le \frac{1}{1 - \|B\|}. \tag{4.2.15}$$

Bemerkung: Dieser Satz wird für Störungsaussagen sehr häufig benutzt.

Beweis Da $I - B$ regulär ist genau dann, wenn die Folgerung $0 = (I - B)x = x - Bx \Rightarrow x = 0$ gilt, d.h. $\lambda = 1$ kein EW von B ist, folgt aus $\rho(B) < 1$, daß $(I - B)^{-1}$ existiert.

Mit den Teilsummen $\sum_{k=0}^{m} B^k$ gilt

$$S_m := (I - B)\sum_{k=0}^{m} B^k = I - B + B - B^2 + \ldots - B^{m+1} = I - B^{m+1}.$$

Nach Satz 4.2.1 existiert eine Norm mit $\|B\| < 1$, da $\rho(B) < 1$. Daher gilt $\|S_m - I\| = \|B^{m+1}\| \le \|B\|^{m+1} \to 0$ für $m \to \infty$ und somit $S_m \to I$ und

$$S_m \to (I - B)\sum_{k=0}^{\infty} B^k = I.$$

Damit folgt die Behauptung. Außerdem gilt

$$\|(I - B)^{-1}\| = \|\sum_{k=0}^{\infty} B^k\| \le \sum_{k=0}^{\infty} \|B\|^k = \frac{1}{1 - \|B\|}. \qquad \blacksquare$$

4.3 Der Gauß-Algorithmus

Dieser dient zur Lösung von linearen Gleichungssystemen

$$\sum_{j=1}^{n} a_{ij}x_j = b_i,\ i = 1, \ldots, n \quad \Leftrightarrow \quad Ax = b,\ A \in \mathbf{K}^{n\times n} \tag{4.3.1}$$

in endlich vielen Schritten und beruht auf der Beobachtung, daß die Lösung x von (4.3.1) unverändert bleibt, wenn Gleichungen in (4.3.1) addiert oder subtrahiert werden (allg: Linearkombination). Ziel ist die Konstruktion eines einfach lösbaren Systems in Dreieckform:

$$\left.\begin{array}{cccccc} r_{11}x_1 & +r_{12}x_2 & +\ldots & +r_{1n}x_n & = & b'_1 \\ & r_{22}x_2 & +\ldots & +r_{2n}x_n & = & b'_2 \\ & & \ddots & \vdots & & \vdots \\ & & & r_{nn}x_n & = & b'_n \end{array}\right\} \quad \Leftrightarrow \quad Rx = b' \tag{4.3.2}$$

Dieses kann, beginnend mit der letzten Zeile, schrittweise aufgelöst werden, falls alle $r_{ii} \neq 0$ sind. Wir betrachten nur reguläre Matrizen A. Dann erhält man ein Dreiecksystem (4.3.2) in $n-1$ Schritten auf folgende Weise.

1. für $a_{11} = 0$ wird eine Zeile mit $a_{k1} \neq 0$ gesucht und diese dann mit der ersten vertauscht (*Pivot-Schritt*, k-te Zeile = Pivot-Zeile);

2. es wird dasjenige Vielfache der ersten Zeile zur i-ten Zeile addiert, das dort den Koeffizienten bei x_1 zu Null macht (*Eliminationsschritt*, Faktor $-a_{i1}/a_{11}$). Nach Behandlung aller Zeilen $i = 2, \ldots, n$ geht das LGS über in die Form
$$A^{(2)}x = b^{(2)}, \text{ mit } \quad A^{(2)} = \begin{pmatrix} a_{11}^{(2)} & a_{12}^{(2)} & \ldots & a_{1n}^{(2)} \\ 0 & a_{22}^{(2)} & \ldots & a_{2n}^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & a_{n2}^{(2)} & \ldots & a_{nn}^{(2)} \end{pmatrix}$$

3. die Schritte 1 und 2 werden auf das Restsystem mit
$$B^{(2)} := \begin{pmatrix} a_{22}^{(2)} & \ldots & a_{2n}^{(2)} \\ \vdots & \ddots & \vdots \\ a_{n2}^{(2)} & \ldots & a_{nn}^{(2)} \end{pmatrix} \tag{4.3.3}$$
angewendet.

4. Nach $n-1$ Wiederholungen mit Matrizen $A^{(1)} := A, A^{(2)}, \ldots, A^{(n)}$ ergibt sich ein Dreicksystem (4.3.2) mit $R = A^{(n)}$, das einfach aufzulösen ist.

Die Eliminationskoeffizienten
$$l_{ij} := \frac{a_{ij}^{(j)}}{a_{jj}^{(j)}}$$
haben eine besondere Bedeutung, die später erläutert wird. Man speichert sie bei Durchführung des Algorithmus an die freiwerdenden Plätze der $a_{ij}^{(j+1)} = 0 \ (i > j)$.

Algorithmus 4.3.1 Gauß-Algorithmus ohne Pivotisierung, durchgeführt im Speicherplatz der Originalmatrix A (d.h. $l_{ij} \mapsto a_{ij}$),

für $j := 1, \ldots, n-1$:
  für $i := j+1, \ldots, n$:
    $l_{ij} := a_{ij}/a_{jj}$; $b_i := b_i - l_{ij} b_j$;
    für $k := j+1, \ldots, n$: $a_{ik} := a_{ik} - l_{ij} a_{jk}$;   (4.3.4)
$x_n := b_n / a_{nn}$;
für $i := n-1, n-2, \ldots, 1$:
  $x_i := \left(b_i - \sum_{j=i+1}^{n} a_{ij} x_j\right) / a_{ii}$;

Der Rechenaufwand beträgt also i.w. $\frac{2}{3}n^3$ Operationen. Soll jedoch das gleiche LGS mit einer anderen rechten Seite c gelöst werden, also $Ay = c$, müssen im obigen Algorithmus nur die b betreffenden Schritte wiederholt werden. Dies ist möglich, da die l_{ij} gespeichert wurden. Der Aufwand ist dann

$$\sum_{j=1}^{n-1} 2(n-j) = n(n-1) \text{ Operationen für } b^{(n)}.$$

Die abschließende Auflösung des Dreiecksystems $Rx = b^{(n)}$ benötigt ebenfalls

$$1 + \sum_{i=1}^{n-1} [1 + 2(n-i)] = n^2 \text{ Operationen.}$$

<u>Rechenaufwand</u> Der Gauß-Algorithmus benötigt i.w. $\frac{2}{3}n^3$ arithmetische Operationen. Zur wiederholten Auflösung mit neuer rechter Seite b sind nur $2n^2$ Operationen erforderlich.

LR-Zerlegung

Die Durchführung eines Eliminationsschritts ohne Pivotisierung beim System $A^{(j)}x = b^{(j)}$,

$$\boxed{\begin{array}{l} \text{für } i = j+1, \ldots, n: \\ \quad b_i^{(j+1)} := b_i^{(j)} - l_{ij} b_j^{(j)}; \\ \quad \text{für } k := j+1, \ldots, n: \\ \qquad a_{ik}^{(j+1)} := a_{ik}^{(j)} - l_{ij} a_{jk}^{(j)}; \end{array}}$$

entspricht der Multiplikation dieses Systems mit einer speziellen Matrix:

$$A^{(j+1)}x = L_j^{-1} A^{(j)} x = L_j^{-1} b^{(j)} = b^{(j+1)},$$

wobei

$$L_j^{-1} = \begin{pmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & -l_{j+1,j} & 1 & & \\ & & \vdots & & \ddots & \\ & & -l_{nj} & & & 1 \end{pmatrix}, \quad l_j := \begin{pmatrix} 0 \\ \vdots \\ 0 \\ l_{j+1,j} \\ \vdots \\ l_{nj} \end{pmatrix}. \tag{4.3.5}$$

Wegen $e_j^T l_j = 0$ ($e_j = j$-ter Einheitsvektor) gilt $(I - l_j e_j^T)(I + l_j e_j^T) = I$, also $L_j = I + l_j e_j^T$ und $L_j^{-1} = I - l_j e_j^T$. Somit wird die obere Dreieckmatrix $R = A^{(n)}$ erzeugt in der Form

$$R = A^{(n)} = L_{n-1}^{-1} A^{(n-1)} = \ldots = L_{n-1}^{-1} \cdots L_1^{-1} A \quad \Leftrightarrow$$

$$A = L_1 L_2 \cdots L_{n-1} R =: LR. \tag{4.3.6}$$

Da die Menge der unteren bzw. oberen Dreieckmatrizen abgeschlossen ist unter Matrixmultiplkation, ist L eine *untere* und R eine *obere* Dreieckmatrix:

$$A = LR = \begin{pmatrix} 1 & 0 & \dots & 0 \\ * & 1 & & \\ \vdots & & \ddots & \\ * & * & \dots & 1 \end{pmatrix} \cdot \begin{pmatrix} * & * & \dots & * \\ 0 & * & & * \\ & & \ddots & \vdots \\ 0 & & & * \end{pmatrix} \tag{4.3.7}$$

Satz 4.3.2 *Der Gaußalgorithmus (4.3.4) (ohne Pivotisierung) erzeugt eine* LR-Zerlegung *der Matrix A. Damit reduziert sich die Lösung von $Ax = b$ auf*

$$b = Ax = L\underbrace{(Rx)}_{b^{(n)}} \quad \Leftrightarrow \quad Lb^{(n)} = b,\ Rx = b^{(n)}, \tag{4.3.8}$$

also auf die Auflösung von zwei LGSen mit Dreieckmatrizen. Die beiden Faktoren L und R werden bei (4.3.4) wieder in A abgespeichert mit Ausnahme der Hauptdiagonalen von L.

Bemerkung: Die LR-Zerlegung kann auch direkt erzeugt werden ohne die Zwischenmatrizen $A^{(j)}$. Die Identität $A = L \cdot R$ wird dabei als Gleichungssystem für L und R aufgefaßt und liefert

$$a_{ij} = \sum_{k=1}^{\min\{i,j\}} l_{ik} r_{kj} \iff \begin{cases} a_{ij} = \sum_{k=1}^{j-1} l_{ik} r_{kj} + l_{ij} r_{jj}, & i > j, \\ a_{ij} = \sum_{k=1}^{i-1} l_{ik} r_{kj} + r_{ij}, & i \le j. \end{cases} \tag{4.3.9}$$

Es können verschiedene Indexfolgen (i,j) angegeben werden so, daß man in diesen Gleichungen direkt nach den l_{ij} (obere Gleichungen) bzw. r_{ij} (untere Gleichungen) auflösen kann. Die Effizienz ist dabei sprach- und maschinenabhängig (Matrizen zeilen-/spaltenweise gespeichert?) und führt auf die Algorithmen von Banachiewicz, Crout, etc.

Pivotisierung

Im allgemeinen ist der Gauß-Algorithmus nicht ohne Zeilenvertauschungen durchführbar. Dies führt auf zwei Fragen

— Für welche Problemmklassen ist keine Pivotisierung nötig?
— welche Pivotstrategien sind für die übrigen sinnvoll?

Bei der Diskussion ist die Tatsache hilfreich, daß der Gauß-Algorithmus bestimmte Strukturen bzw. Eigenschaften der Matrix A erhält.

Definition 4.3.3 *A sei eine $n \times n$-Matrix.*

— *A heißt* hermitesch positiv definit *(HPD), wenn $A = A^*$ und $x^*Ax > 0$ für $x \neq 0$ gilt (alle Eigenwerte von A sind dann reell und positiv).*

— *A heißt* streng diagonaldominant, *wenn*

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|, \quad i = 1, \ldots, n. \tag{4.3.10}$$

— *A heißt* $(2m+1)$-Bandmatrix *(Tridiagonalmatrix für* $m = 1$*), wenn*

$$a_{ij} = 0 \quad \textit{für } |i - j| > m.$$

Bei definiten und streng diagonaldominanten Matrizen haben insbesondere die Hauptdiagonalelemente einen Wert ungleich Null. Die Erhaltung bestimmter Strukturen ermöglicht oft eine Reduktion des Aufwands beim Gauß-Algorithmus.

Satz 4.3.4 *a) Der Gaußalgorithmus sei bei einer Matrix A ohne Pivotisierung durchführbar. Ist die Matrix A hermitesch, positiv (negativ) definit, streng diagonaldominant oder* $2m+1$*-Bandmatrix, dann hat jede Restmatrix (4.3.3) wieder die gleiche Eigenschaft. Bei definiten und streng diagonaldominanten Matrizen ist daher keine Pivotisierung nötig.*

b) Für eine positiv (negativ) definite Matrix $A \in \mathbf{K}^{n \times n}$ *existiert genau eine Dreieckzerlegung* $A = \tilde{L}\tilde{R}$ $(A = -\tilde{L}\tilde{R})$ *mit* $\tilde{l}_{ii} = \tilde{r}_{ii} > 0$, $i = 1, \ldots, n$, *und* $\tilde{L}^* = \tilde{R}$. *Diese* Cholesky-Zerlegung $A = \tilde{L}\tilde{L}^*$ *erfordert etwa den halben Rechenaufwand der gewöhnlichen LR-Zerlegung.*

Beweis a) Der erste Eliminationsschritt erzeugt die Umformung

$$A = A^{(1)} = \begin{pmatrix} a_{11} & z^* \\ s & B^{(1)} \end{pmatrix} \text{ mit } s = \begin{pmatrix} a_{21} \\ \vdots \\ a_{n1} \end{pmatrix}, \; z^* = (a_{12}, \ldots, a_{1n}), \quad \longrightarrow$$

$$A^{(2)} = \begin{pmatrix} a_{11} & z^* \\ 0 & B^{(2)} \end{pmatrix}, \quad B^{(2)} := \Big(a_{ik} - \frac{1}{a_{11}} a_{i1} a_{1k}\Big)_{i,k=2}^{n} = B^{(1)} - \frac{1}{a_{11}} s z^*.$$

Daraus ergeben sich die Aussagen, von denen hier nur $A = A^*$ und Definitheit gezeigt werden.

Wenn A hermitesch ist, gilt $s = z$ und $B^{(1)} = B^{(1)^*}$. Daher ist auch $B^{(2)}$ hermitesch. Wenn A noch positiv definit ist, gilt $x^* A x > 0$ für alle $x \in \mathbf{K}^n$. Für $x = (1, 0, \ldots, 0)^T$ folgt $a_{11} > 0$, im ersten Schritt ist also keine Zeilenvertauschung notwendig. Nun sei mit $y \in \mathbf{K}^{n-1}$ speziell

$$\hat{x} = \begin{pmatrix} \frac{-1}{a_{11}} s^* y \\ y \end{pmatrix} \Rightarrow A\hat{x} = \begin{pmatrix} a_{11} & s^* \\ s & B^{(1)} \end{pmatrix} \begin{pmatrix} \frac{-1}{a_{11}} s^* y \\ y \end{pmatrix} = \begin{pmatrix} 0 \\ B^{(1)} y - \frac{1}{a_{11}} s s^* y \end{pmatrix} = \begin{pmatrix} 0 \\ B^{(2)} y \end{pmatrix}.$$

Also ist $y^* B^{(2)} y = \hat{x}^* A \hat{x} > 0$, d.h., $B^{(2)}$ wieder positiv definit.

b) Die Existenz einer Dreieckzerlegung $A = \tilde{L}\tilde{R}$ mit $\tilde{l}_{ii} = \tilde{r}_{ii} > 0$ folgt sofort aus dem ersten Beweisteil. Es kann nämlich mit $D := \operatorname{diag}(\sqrt{r_{11}}, \ldots, \sqrt{r_{nn}})$ gesetzt werden $\tilde{L} := LD$, $\tilde{R} := D^{-1}R$. Durch Induktion läßt sich auch zeigen, daß diese Zerlegung eindeutig ist. Wegen $\tilde{L}\tilde{R} = A = A^* = \tilde{R}^*\tilde{L}^*$ erhält man dann sofort $\tilde{L} = \tilde{R}^*$ und $\tilde{R} = \tilde{L}^*$. ■

Kriterium 4.3.5 *Falls A symmetrisch und streng diagonaldominant ist mit $a_{ii} > 0$, dann ist A positiv definit.*

Diese Aussage und weitere Kriterien findet man bei Rutishauser: *Vorlesungen über numerische Mathematik, 1976*. Damit ist in vielen wichtigen Fällen gesichert, daß der Gauß-Algorithmus ohne Pivotisierung durchführbar ist.

Bemerkung: Bei großen praktischen Problemen können LGSe mit $n = 10^5 \dots 10^6$ Unbekannten auftreten. Die Matrizen haben dabei aber nur wenige Elemente $a_{ij} \neq 0$ (Anteil $< 10\%$, 'dünnbesetzt'), wie, z.B., bei Bandstruktur. Ohne die Ausnutzung solcher Struktureigenschaften wären so große Probleme auch auf modernen Supercomputern nicht lösbar.

In der Theorie ist eine Zeilenvertauschung beim Gauß-Algorithmus nur dann nötig, wenn das Pivotelement verschwindet, $a_{jj}^{(j)} = 0$. Bei der praktischen Rechnung im Computer mit endlichen Zahldarstellungen sollte aber auch die Verwendung zu kleiner Pivotelemente vermieden werden (genauere Analyse in §6.)

Beispiel 4.3.6 Alle (Zwischen-)Ergebnisse werden auf zwei Stellen gerundet. Aus

$$\begin{array}{rcrcl} 0.01\,x_1 & + & 2\,x_2 & = & 1 \\ x_1 & + & x_2 & = & 3 \end{array} \quad \text{wird} \quad \begin{array}{rcrcl} 0.01\,x_1 & + & 2x_2 & = & 1 \\ 0\cdot x_1 & - & 200\,x_2 & = & -97 \end{array}, \tag{4.3.11}$$

denn $l_{21} = a_{21}/a_{11} = 1/0.01 = 1.0 \cdot 10^2$ (Gleitpunktdarstellung) $\Rightarrow$

$$\begin{aligned} a_{22}^{(2)} &= a_{22} - l_{21}a_{12} = 1 - 100\cdot 2 = -199 \quad \text{Rundung: } \widetilde{a_{22}^{(2)}} = -2\cdot 10^2 \\ b_2^{(2)} &= b_2 - l_{21}b_1 = 3 - 100 = -97 \quad \text{(ist exakt).} \end{aligned}$$

Damit erhält man

$$\widetilde{x_2} = -97/(-200) = 0.49, \quad \widetilde{x_1} = (1 - 2\cdot 0.49)/0.01 = 2.0.$$

Die exakte Lösung ist aber $(2.512\dots, 0.487\dots)$. Vertauscht man aber die beiden Gleichungen in (4.3.11), so ergibt die Elimination

$$\begin{array}{rrl} x_1 & +x_2 & = 3 \\ 0.01x_1 & +2x_2 & = 1 \end{array} \quad \to \quad \begin{array}{rrl} x_1 & +x_2 & = 3 \\ & 2x_2 & = 0.97 \end{array} \quad \to \quad \begin{array}{rl} x_1 & = 2.5 \\ x_2 & = 0.49 \end{array}$$

Um diese Ungenauigkeiten zu verhindern, verwendet man verschiedene *Pivot-Strategien*: Im Schritt j wird zur Vertauschung mit Zeile j die Zeile Nr. $p \in \{j, \dots, n\}$ nach folgender Regel gewählt:

1. Spaltenpivotisierung, absolutes Maximum:

$$|a_{pj}^{(j)}| = \max_{k=j}^{n} |a_{kj}^{(j)}|.$$

2. Spaltenpivotisierung, relatives Maximum:
$$\frac{|a_{pj}^{(j)}|}{\sum_{\nu=j}^{n}|a_{p\nu}^{(j)}|}=\max_{k=j}^{n}\frac{|a_{kj}^{(j)}|}{\sum_{\nu=j}^{n}|a_{k\nu}^{(j)}|}.$$
3. Vollständige Pivotisierung, zusätzlich werden auch Spaltenvertauschungen (Umnumerierung der Variablen) vorgenommen
$$|a_{pq}^{(j)}|=\max_{i,k=j}^{n}|a_{ik}^{(j)}|.$$

Wegen des hohen Aufwands werden die Strategien 2) und 3) nur in Ausnahmefällen benutzt. Strategie 2) unterdrückt den Einfluß der Multiplikation von Zeilen mit einer Konstanten (Skalierung). Ein ähnlicher Effekt ist bei 1) erzielbar durch *Äquilibrierung* der Matrix, bei der zu Beginn alle Zeilen auf Summennorm 1 skaliert werden. Auch die einfache Strategie 1) erhöht den Aufwand um 50% :

Rechenaufwand für den Gauß-Algorithmus mit Pivotisierung 1): n^3 Operationen

Inverse Matrix A^{-1}:

Die Lösung von $Ax=b$ läßt sich theoretisch sehr elegant in der Form $x=A^{-1}b$ angeben. Selbst zur Lösung vieler LGSe $Ax^{(j)}=b^{(j)},\ j=1,2,\ldots,$ lohnt es sich aber **nicht**, die Inverse A^{-1} explizit zu berechnen. Der Rechenaufwand für das Produkt $x=A^{-1}b$ ist der *gleiche*, wie der zur Lösung der beiden Dreiecksysteme bei $LRx=b$, nämlich $2n^2$ (vgl. Berechnung nach (4.3.4)).

Manchmal ist es aber für theoretische Aussagen günstig, die Elemente von A^{-1} zu kennen (z.B. Input-Output-Analyse: Wechselwirkungen zwischen Produktionszweigen). Bei gegebener LR-Zerlegung von A ergeben sich die Spalten von
$$A^{-1}=C=\left(c^{(1)},\ldots,c^{(n)}\right)\quad\text{aus}\quad LRc^{(j)}=e^{(j)},\ j=1,\ldots,n.$$
Unter Ausnutzung der speziellen Gestalt der Einheitsvektoren $e^{(j)}$ benötigt man zur Berechnung $\frac{4}{3}n^3$ Operationen zusätzlich zur LR-Zerlegung, also $2n^3$ Operationen insgesamt.

Rechteckige Matrizen $A\in\mathbf{R}^{m\times n},\ m<n$:

Die Gauß-Elimination führt hier auf eine LR-Zerlegung
$$A=LR=\begin{pmatrix}1&&\\ \vdots&\ddots&\\ *&\cdots&1\end{pmatrix}\begin{pmatrix}*&\cdots&*&\cdots&*\\ &\ddots&\vdots&&\vdots\\ &&*&\cdots&*\end{pmatrix}$$
mit $L\in\mathbf{R}^{m\times m},\ R\in\mathbf{R}^{m\times n}$. Wenn die Diagonalelemenete $r_{11},\ldots,r_{mm}$ alle nicht verschwinden, kann die allgemeine Lösung des unterbestimmten Gleichungssystems $Ax=b$ mit den freien Parametern $x_{m+1},\ldots,x_n$ aus dem Dreiecksystem $Rx=L^{-1}b$ bestimmt werden.

4.4 Fehlerausbreitung, Kondition

Die Auswirkungen von Rundungsfehlern bei der Durchführung des Gauß-Algorithmus werden später diskutiert. Aber auch die Verfälschung von Eingangsdaten (z.B. durch Rundung auf endliche Genauigkeit) wirkt sich auf die Lösung des LGS aus. Zur Untersuchung wird angenommen, daß statt des ursprünglichen Problems $Ax = b$ mit regulärer Matrix A das gestörte Problem

$$(A + \delta A)y = b + \delta b, \quad y = x + \delta x \tag{4.4.1}$$

exakt gelöst wird. Wie groß ist dann die Störung δx von x?

Multiplikation der Gleichung mit A^{-1} ergibt

$$(I + A^{-1}\delta A)y = A^{-1}(b + \delta b) = x + A^{-1}\delta b. \tag{4.4.2}$$

Nach Satz 4.2.2 (v. Neumann) ist dieses LGS lösbar, wenn in einer Norm gilt

$$\|A^{-1}\delta A\| \leq \|A^{-1}\|\|\delta A\| =: q < 1. \tag{4.4.3}$$

Dann folgt die Schranke

$$\|y\| \leq \frac{\|x + A^{-1}\delta b\|}{1-q}. \tag{4.4.4}$$

Nach (4.4.2) gilt auch $\delta x = y - x = A^{-1}\delta b - A^{-1}\delta A y$. Daraus folgt mit (4.4.3) und (4.4.4)

$$\begin{aligned}
\|\delta x\| = \|y - x\| &\leq \|A^{-1}\delta b\| + q\|y\| \\
&\leq \|A^{-1}\delta b\| + \frac{q}{1-q}(\|x\| + \|A^{-1}\delta b\|) \\
&\leq \frac{q}{1-q}\|x\| + \frac{1}{1-q}\|A^{-1}\|\|\delta b\|.
\end{aligned} \tag{4.4.5}$$

Besonders aussagekräftig sind Abschätzungen, die *skalierungsinvariant* sind, bei einer Multiplikation des LGS mit Konstanten also unverändert bleiben. Dazu betrachtet man relative Fehler. Mit $\|b\| = \|Ax\| \leq \|A\|\|x\|$ folgt aus (4.4.5):

$$\frac{\|\delta x\|}{\|x\|} \leq \frac{q}{1-q} + \frac{\|A^{-1}\| \cdot \|A\|}{1-q} \, \frac{\|\delta b\|}{\|b\|}.$$

Bezieht man sich auch bei δA auf den relativen Fehler, etwa bei

$$q = \|A^{-1}\|\|\delta A\| = \|A^{-1}\| \cdot \|A\| \, \frac{\|\delta A\|}{\|A\|},$$

dann taucht in der Abschätzung außer den Fehlern nur **eine** zusätzliche Konstante auf:

Definition 4.4.1 *Zu einer gegebenen Matrixnorm $\|\cdot\|$ heißt*

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

die Konditionszahl *der Matrix A. Bei speziellen Normen übernimmt man den Index, z.B.,* $\kappa_\infty(A) = \|A\|_\infty \|A^{-1}\|_\infty$.

Die Konditionszahl beschreibt die Empfindlichkeit des LGS gegenüber Störungen. Für verträgliche Matrixnormen gilt $\kappa(A) \geq 1$ wegen $1 \leq \|I\| = \|AA^{-1}\| \leq \|A\|\|A^{-1}\|$. Die gerade durchgeführte Diskussion hat folgenden Satz bewiesen:

Satz 4.4.2 *Mit φ seien die relativen Fehler im gestörten System (4.4.1) bezeichnet,*

$$\varphi_x := \frac{\|\delta x\|}{\|x\|}, \quad \varphi_b := \frac{\|\delta b\|}{\|b\|}, \quad \varphi_A := \frac{\|\delta A\|}{\|A\|}.$$

Damit gilt: Das gestörte LGS (4.4.1) ist eindeutig lösbar für

$$\kappa(A) \cdot \varphi_A < 1. \tag{4.4.6}$$

Der relative Fehler in der Lösung ist beschränkt durch

$$\varphi_x \leq \frac{\kappa(A)}{1 - \kappa(A)\varphi_A} \Big(\varphi_A + \varphi_b\Big). \tag{4.4.7}$$

Für große $\kappa(A) \gg 1$ ist das Problem *schlecht konditioniert.* Dann können schon kleine Störungen der Daten zu erheblichen Änderungen der Lösung des LGS führen.

Bei der Berechnung einer Lösung von (4.4.1) können weitere (Rundungs-) Fehler auftreten. Die Genauigkeit einer Näherungslösung $\tilde{x}$ kann mit Hilfe des *Defekts* oder *Residuums*

$$r := \tilde{A}\tilde{x} - \tilde{b}, \qquad \tilde{A} = A + \delta A,\ \tilde{b} = b + \delta b, \tag{4.4.8}$$

überprüft werden. Dazu gilt

Satz 4.4.3 *Mit den Bezeichnungen von Satz 4.4.2 und r aus (4.4.8) gilt für den relativen Fehler der Näherungslösung $\tilde{x}$ zu $x = A^{-1}b$ die Abschätzung*

$$\frac{\|\tilde{x} - x\|}{\|x\|} \leq \frac{\kappa(A)}{1 - \kappa(A)\varphi_A} \left(\varphi_a + \varphi_b + \frac{\|r\|}{\|b\|}\right). \tag{4.4.9}$$

Beweis $\tilde{x}$ unterscheidet sich von y aus Satz 4.4.2 folgendermaßen:

$$\tilde{A}y = \tilde{b}, \quad \tilde{A}\tilde{x} = \tilde{b} + r \quad \Rightarrow \quad \tilde{x} - y = \tilde{A}^{-1}r.$$

Mit den Schranken aus Satz 4.4.2 folgt

$$\frac{\|\tilde{x} - x\|}{\|x\|} \leq \frac{\|y - x\|}{\|x\|} + \frac{\|\tilde{A}^{-1}r\|}{\|x\|} \leq \varphi_x + \frac{\|A^{-1}\|}{1 - \kappa(A)\varphi_A} \cdot \frac{\|A\|\|r\|}{\|b\|}$$

also die Behauptung, denn

$$\|\tilde{A}^{-1}\| = \|(A + \delta A)^{-1}\| \leq \frac{\|A^{-1}\|}{1 - \|A^{-1}\delta A\|},$$

nach (4.4.2), (4.4.4). ∎

Auch hier geht die Konditionszahl entscheidend ein. Wegen dieser Bedeutung sollte bei der praktischen Rechnung immer auch die Kondition der Matrix (ab-)geschätzt werden, um die Genauigkeit der tatsächlich berechneten Lösung einstufen zu können.

Schranken für $\kappa(A)$

Da die Zeilensummennorm $\|\cdot\|_\infty$ einer Matrix einfacher bestimmt werden kann als, z.B., $\|\cdot\|_2$, werden hier Ansätze diskutiert, diese Norm der Inversen $\|A^{-1}\|_\infty$ abzuschätzen:

1. Für streng diagonaldominante Matrizen (4.3.10) ist $A = D(I - D^{-1}B)$ mit $D = \operatorname{diag}(a_{ii})$ und $\|D^{-1}B\|_\infty < 1$. Aus Satz 4.2.2 folgt daher

$$\kappa_\infty(A) \le \|A\|_\infty \frac{\|D^{-1}\|_\infty}{1 - \|D^{-1}B\|_\infty}.$$

2. Wenn die LR-Zerlegung einer Matrix $A = LR$ vorliegt, kann mit akzeptablem Rechenaufwand ($\approx n^2$) auch eine Schranke für $\|A^{-1}\|_\infty \le \|L^{-1}\|_\infty \|R^{-1}\|_\infty$ berechnet werden. Die Methode beruht auf folgenden Eigenschaften nichtnegativer Matrizen, die hier nur für R formuliert werden. Ungleichungen und Beträge bei Matrizen sind dabei komponentenweise zu verstehen.

   (a) Für $B = (b_{ij})$ mit $b_{ij} \ge 0$ (kurz: $B \ge 0$) gilt

   $$\|B\|_\infty = \|Be\|_\infty \,, \; e = (1, 1, \dots, 1)^\mathsf{T}.$$

   (b) Für $C^{-1} \ge 0$ ist $\|C^{-1}\|_\infty = \|z\|_\infty$, wobei z das LGS $Cz = e$ löst. Daher kann die Norm $\|C^{-1}\|_\infty$ berechnet werden, ohne C^{-1} explizit zu kennen.

   (c) Die Norm $\|\cdot\|_\infty$ ist monoton, d.h. aus

   $$|b_{ij}| \le |c_{ij}| \;\forall i, j \text{ (kurz: } |B| \le |C|), \quad \text{folgt} \quad \|B\|_\infty \le \|C\|_\infty.$$

   (d) Zerlegt man $R = D(I - B)$ mit $D = \operatorname{diag}(r_{ii})$, dann ist B strikt obere Dreieckmatrix mit $\rho(B) = 0 \Rightarrow B^n = 0$. Aus Satz 4.2.2 folgt

   $$|R^{-1}| = |(I - B)^{-1} D^{-1}| = |\sum_{j=0}^{n-1} B^j| |D^{-1}| \le \sum_{j=0}^{n-1} |B|^j |D|^{-1} = \Big(|D|(I - |B|)\Big)^{-1}.$$

   Die Eigenschaften (a) - (d) führen auf

Satz 4.4.4 *$R \in \mathbf{R}^{n\times n}$ sei eine reguläre obere Dreieckmatrix. Der (positive) Vektor $z \in \mathbf{R}^n$ sei definiert durch das LGS*

$$|r_{ii}| z_i - \sum_{j=i+1}^{n} |r_{ij}| z_j = 1, \quad i = n, n-1, \dots, 1. \tag{4.4.10}$$

Dann gilt

$$\|R^{-1}\|_\infty \le \|z\|_\infty.$$

Beispiel 4.4.5

$$A = \begin{pmatrix} 2 & 3 & -5 \\ 4 & 8 & -3 \\ -6 & 1 & 4 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ -3 & 5 & 1 \end{pmatrix} \begin{pmatrix} 2 & 3 & -5 \\ 0 & 2 & 7 \\ 0 & 0 & -46 \end{pmatrix} = L \cdot R.$$

Zur Abschätzung von $\|R^{-1}\|_\infty$ ist zu lösen

$$\begin{pmatrix} 2 & -3 & -5 \\ 0 & 2 & -7 \\ 0 & 0 & 46 \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ z_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \Longrightarrow \begin{pmatrix} z_1 \\ z_2 \\ z_3 \end{pmatrix} = \begin{pmatrix} 1.418 \\ 0.576 \\ 0.0217 \end{pmatrix}.$$

Dies führt auf die Schranke $\|R^{-1}\|_\infty \leq \|z\|_\infty \doteq 1.418$. Für $\|L^{-1}\|_\infty$ ergibt sich analog

$$\begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -3 & -5 & 1 \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \Longrightarrow \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 3 \\ 19 \end{pmatrix}.$$

Hier erhält man die Schranke $\|L^{-1}\|_\infty \leq \|u\|_\infty = 19$, zusammen mit der ersten führt sie auf die Abschätzung $\|A^{-1}\|_\infty \leq 26.95$ (zum Vergleich: $\|A^{-1}\|_\infty \doteq 0.451$). Mit $\|A\|_\infty = 15$ folgt die Schranke $\kappa_\infty(A) \leq 404.25$ für die Konditionszahl.

Spalten-Pivotisierung (hier: Vertauschung der Zeilen 1 und 3) wirkt sich auch auf die berechneten Schranken günstig aus, es ist

$$\begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} A = \begin{pmatrix} -6 & 1 & 4 \\ 4 & 8 & -3 \\ 2 & 3 & -5 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ -\frac{2}{3} & 1 & 0 \\ -\frac{1}{3} & \frac{5}{13} & 1 \end{pmatrix} \begin{pmatrix} -6 & 1 & 4 \\ 0 & \frac{26}{3} & -\frac{1}{3} \\ 0 & 0 & -\frac{46}{13} \end{pmatrix} =: \hat{L}\hat{R}.$$

Als Abschätzung erhält man jetzt $\|\hat{R}^{-1}\|_\infty \leq 0.376$, $\|\hat{L}^{-1}\|_\infty \leq 2.616$ und daher die viel schärfere Schranke $\|A^{-1}\|_\infty \leq 0.984$, da $\|A^{-1}\|_\infty = \|(PA)^{-1}\|_\infty$ mit $P = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} = P^{-1}$.

4.5 Iterationsverfahren für lineare Gleichungssysteme

Für lineare Gleichungssysteme der Form $(I - B)x = r$ (vgl. Input-Output-Analyse) mit ‘kleiner’ Matrix B, genauer, wenn $\rho(B) < 1$, ist eine explizite Darstellung der Lösung aus Satz 4.2.2 (Neumannreihe) bekannt,

$$x = (I - B)^{-1} r = \sum_{j=0}^{\infty} B^j r = r + B \sum_{j=0}^{\infty} B^j r. \tag{4.5.1}$$

Die Partialsummen dieses Ausdrucks ergeben sich aus der Rekursion bzw. *Iteration* $x^{(0)} := 0$,

$$x^{(k+1)} := Bx^{(k)} + r, \quad k = 0, 1, \ldots, \tag{4.5.2}$$

denn: $x^{(1)} = r,\ x^{(2)} = r + Br,\ x^{(3)} = r + Br + B^2r, \ldots$. Diese Konvergenzaussage läßt sich erheblich verallgemeinern. Die Iteration (4.5.2) dient nämlich zur Bestimmung eines *Fixpunkts* der Abbildung $g(x) := Bx + r$, d.h. eines Vektors mit

$$z = g(z), \qquad g : \mathbf{R}^n \to \mathbf{R}^n. \tag{4.5.3}$$

Allgemein gilt für *kontrahierende* Abbildungen g der

Satz 4.5.1 (Banachscher Fixpunksatz) *Gegeben sei ein Banachraum H und eine Abbildung $g:\ H \to H$. Gelten auf einer abgeschlossenen Teilmenge $\Omega \subset H$ und für $q \in \mathbf{R}, 0 < q < 1$, die Aussagen*

$$g(\Omega) \subset \Omega \qquad \text{('g bildet } \Omega \text{ auf sich ab')} \tag{4.5.4}$$

$$\|g(x) - g(y)\| \ \leq\ q\|x - y\|\ \forall x, y \in \Omega, \quad \text{('g ist kontrahierend')} \tag{4.5.5}$$

dann besitzt g genau *einen Fixpunkt $z = g(z) \in \Omega$. Dieser kann durch die* Iteration

$$x^{(k+1)} := g(x^{(k)}),\ k = 0, 1, \ldots, \quad x^{(0)} \in \Omega \text{ beliebig}, \tag{4.5.6}$$

berechnet werden, es ist $\lim_{k\to\infty} x^{(k)} = z$. Quantitativ gelten die Fehlerabschätzungen

$$\|x^{(k)} - z\| \ \leq \frac{q}{1-q}\|x^{(k)} - x^{(k-1)}\| \qquad \text{(a-posteriori)} \tag{4.5.7}$$

$$\leq \frac{q^k}{1-q}\|x^{(1)} - x^{(0)}\| \qquad \text{(a-priori)}. \tag{4.5.8}$$

Eine Bedingung der Form (4.5.5) (mit i.a. beliebigem $q > 0$) heißt *Lipschitz-Bedingung.*

Beweis Wegen (4.5.5) gilt

$$\begin{aligned} \|x^{(k+1)} - x^{(k)}\| &= \|g(x^{(k)}) - g(x^{(k-1)})\| \leq q\|x^{(k)} - x^{(k-1)}\| \leq \ldots \\ &\leq q^k\|x^{(1)} - x^{(0)}\|. \end{aligned}$$

Analog ergibt sich, für beliebiges $m > 0$, aus der Dreiecksungleichung die Schranke

$$\begin{aligned} \|x^{(k+m)} - x^{(k)}\| &\leq \sum_{j=0}^{m-1} \|x^{(k+j+1)} - x^{(k+j)}\| \leq \sum_{j=0}^{m-1} q^{k+j}\|x^{(1)} - x^{(0)}\| \\ &\leq \frac{q^k}{1-q}\|x^{(1)} - x^{(0)}\| \to 0, \quad k \to \infty. \end{aligned}$$

Daher ist $\{x^{(k)}\}$ Cauchyfolge, wegen der Vollständigkeit von Ω also auch konvergent, $\Rightarrow \hat{x} = \lim_{k\to\infty} x^{(k)}$ existiert. Aus der Stetigkeit von g folgt

$$\hat{x} = \lim_{k\to\infty} x^{(k)} = \lim_{k\to\infty} x^{(k+1)} = \lim_{k\to\infty} g(x^{(k)}) = g(\lim_{k\to\infty} x^{(k)}) = g(\hat{x}),$$

d.h. $\hat{x}$ ist Fixpunkt von g. Dieser ist eindeutig, denn wegen $q < 1$:

$$\|\hat{x} - z\| = \|g(\hat{x}) - g(z)\| \leq q\|\hat{x} - z\| \Rightarrow \hat{x} = z.$$

Wie oben folgt aus $x^{(k)} - z = \lim_{m\to\infty}(x^{(k)} - x^{(k+m)})$ die Schranke

$$\|x^{(k)} - z\| \le \sum_{j=0}^{\infty} q^{j+1}\|x^{(k)} - x^{(k-1)}\| = \frac{q}{1-q}\|x^{(k)} - x^{(k-1)}\|. \qquad \blacksquare$$

Dieser Satz ist die Grundlage der meisten Konvergenzaussagen bei Iterationsverfahren, auch bei nichtlinearen Problemen. Wegen Satz 4.2.1 kann die Bedingung (4.5.5), $\|B\| < 1$, bei der affin-linearen Funktion (4.5.2) natürlich durch die Bedingung $\rho(B) < 1$ an den Spektralradius ersetzt werden. Außerdem gilt die Umkehrung

Satz 4.5.2 *Für $\rho(B) > 1$ gibt es Startvektoren $x^{(0)} \in \mathbf{R}^n$ so, daß (4.5.2) divergiert.*

Beweis (-idee): Wähle $x^{(0)} = z + y$ so, daß $By = \lambda y$ mit $|\lambda| > 1$. $\blacksquare$

Außer in einigen Spezialfällen liegt die Matrix A unseres LGSs natürlich nicht in der Form $A = I - B$ vor. Oft kann man sie aber als *Störung* eines Hauptteils M schreiben. Von praktischem Interesse sind dabei allerdings nur Hauptteile M, die auf einfach auflösbare Gelichungssysteme führen (z.B. Diagonal-, Dreieck-Matrix).

Definition 4.5.3 *Die Zerlegung einer Matrix*

$$A = M - N \tag{4.5.9}$$

heißt regulär, *wenn M regulär ist und*

$$\rho(M^{-1}N) \; < \; 1. \tag{4.5.10}$$

Nach Satz 4.5.1 definiert jede reguläre Zerlegung von A ein konvergentes Iterationsverfahren:

$$b = Ax = Mx - Nx \iff Mx = Nx + b \iff x = M^{-1}Nx + M^{-1}b,$$

d.h. (4.5.3) mit $B := M^{-1}N = I - M^{-1}A,\ r := M^{-1}b$. Dazu gehört die Iteration: $x^{(0)} := 0$,

$$Mx^{(k+1)} := Nx^{(k)} + b \iff M(x^{(k+1)} - x^{(k)}) := b - Ax^{(k)} \tag{4.5.11}$$

$$x^{(k+1)} := M^{-1}Nx^{(k)} + M^{-1}b \iff x^{(k+1)} := M^{-1}b + (I - M^{-1}A)x^{(k)}. \tag{4.5.12}$$

Dabei wird die Form (4.5.11) bei der praktischen Durchführung benutzt, während (4.5.12) daraus durch Multiplikation mit M^{-1} hervorgeht und sich für theoretische Untersuchungen anbietet. Eine Zerlegung ist umso günstiger, je kleiner der Konvergenzfaktor $\rho(M^{-1}N)$ ist und je weniger Aufwand ein Schritt (4.5.11) kostet. Beide Kriterien lassen sich auf folgende Weise miteinander in Beziehung setzen.

Bemerkung: Benutzt man in Satz 4.5.1 die Konstante $q = \rho(M^{-1}N)$, so folgt aus (4.5.8), daß zur Verringerung des Startfehlers $\|x^{(1)} - x^{(0)}\|$ auf die Endgenauigkeit $\|x^{(k)} - z\| \le 10^{-l}\|x^{(1)} - x^{(0)}\|$ ungefähr

$$k = \frac{l}{-\log_{10}\rho(M^{-1}N)} \text{ Schritte}$$

nötig sind. Verschiedene Zerlegungen von A lassen sich daher anhand der Größe

$$\frac{1}{-\log_{10}\rho(M^{-1}N)}\cdot(\text{Aufwand für einen Schritt (4.5.11)}) \tag{4.5.13}$$

vergleichen. (4.5.13) gibt den zur Verkleinerung des Fehlers um $\frac{1}{10}$ benötigten Aufwand an.

Bevor die Standard-Iterationsverfahren behandelt werden, sei hier eine Anwendung beim Gauß-Algorithmus erwähnt, die *Nachiteration:*

Bei sehr schlecht konditionierten linearen Gleichungssystemen, $\kappa(A) \gg 1$, kann die mit Hilfe des Gaußalgorithmus berechnete Lösung durch Rundungsfehler stark verfälscht sein. Dann gilt für die berechnete LR-Zerlegung nur noch $\tilde{L}\tilde{R} \cong A$. In diesem Fall kann oft noch eine vernünftige Lösung berechnet werden mit $M := \tilde{L}\tilde{R}$ in (4.5.11), (vgl. auch Satz 4.4.3) d.h. durch

$$\tilde{L}\tilde{R}(x^{(k+1)} - x^{(k)}) = b - Ax^{(k)}, \quad \text{wenn } \rho(I - \tilde{R}^{-1}\tilde{L}^{-1}A) < 1.$$

Der Speicherbedarf verdoppelt sich allerdings, da die Originalmatrix A zur Berechnung des Defekts $b - Ax^{(k)}$ benötigt wird. Die Defektberechnung sollte mit doppelter Zahlengenauigkeit erfolgen.

Gesamt- und Einzelschrittverfahren

Beispiel 4.5.4 Betrachtet werde das folgende 2×2-Gleichungssystem mit Lösung $(2,1)^T$.

$$\begin{array}{rcrcl c rcll}
x_1 & + & 0.01x_2 & = & 2.01 & \Longleftrightarrow & x_1 & = & 2.01 & -0.01x_2 \\
-0.02x_1 & + & 4x_2 & = & 3.96 & \Longleftrightarrow & 4x_2 & = & 3.96 & \underbrace{+0.02x_1}_{\text{'Störung'}}
\end{array}$$

'Sukzessives Einsetzen', beginnend mit $x^{(0)} = \binom{0}{0}$ liefert der Reihe nach Vektoren

$x^{(0)}$	$x^{(1)}$	$x^{(2)}$	$x^{(3)}$
0	2.01	2.0001	1.9999995
0	0.99	1.00005	1.0000005

Dabei wurde sozusagen die Diagonale der Matrix $\begin{pmatrix} 1 & 0.01 \\ -0.02 & 4 \end{pmatrix}$ als *Hauptteil* M benutzt, der Rest als 'Störung'. Eine weitere Variante ergibt sich, wenn man zur Berechnung von $x_2^{(k)}$ den schon bekannten Wert $x_1^{(k)}$ statt $x_1^{(k-1)}$ benutzt. Zugrunde liegt beiden Verfahren eine Zerlegung der Matrix $A = L + D + R$ in Hauptdiagonale, unteres und oberes Dreieck,

$$A = \begin{pmatrix} \ddots & & R \\ & D & \\ L & & \ddots \end{pmatrix}, \quad \text{d.h., } L = \begin{pmatrix} 0 & \cdots & 0 \\ * & \ddots & \vdots \\ * & * & 0 \end{pmatrix}, \quad R = \begin{pmatrix} 0 & * & * \\ \vdots & \ddots & * \\ 0 & \cdots & 0 \end{pmatrix} \tag{4.5.14}$$

und $D = diag(a_{11}, a_{22}, \ldots, a_{nn}),\ a_{ii} \neq 0$.

Gesamtschrittverfahren, Jacobi-Iteration: Zerlegung $M := D,\ N := -L - R$, Iterationsmatrix $B_G = -D^{-1}(L + R)$.

$$\left\{\begin{array}{rcll} Dx^{(k+1)} & = & b - (L+R)x^{(k)}, & k = 0, 1, \ldots \\ x_i^{(k+1)} & = & \frac{1}{a_{ii}}[b_i - \sum\limits_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum\limits_{j=i+1}^{n} a_{ij}x_j^{(k)}], & i = 1, \ldots, n. \end{array}\right. \tag{4.5.15}$$

Einzelschrittverfahren, Gauß-Seidel-Iteration: Zerlegung $M := D + L,\ N := -R$, Iterationsmatrix $B_E = -(D + L)^{-1}R$.

$$\left\{\begin{array}{rcll} (D+L)x^{(k+1)} & = & b - Rx^{(k)}, & k = 0, 1, \ldots \\ x_i^{(k+1)} & = & \frac{1}{a_{ii}}[b_i - \sum\limits_{j=1}^{i-1} a_{ij}x_j^{(k+1)} - \sum\limits_{j=i+1}^{n} a_{ij}x_j^{(k)}], & i = 1, \ldots, n. \end{array}\right. \tag{4.5.16}$$

Die Formeln (4.5.15) und (4.5.16) lassen direkt den Rechenaufwand für einen Iterationsschritt erkennen, er beträgt bei beiden Verfahren maximal $2n^2$ Operationen. Für Matrizen, bei denen nur wenige Elemente von Null verschieden sind, gilt genauer

Rechenaufwand pro Schritt (4.5.15), (4.5.16) $= 2\cdot$(Anzahl der nichttrivialen Matrixelemente)

Diese einfache Abhängigkeit ist ein entscheidender Vorteil von Iterationsverfahren da eine dünne Besetzung von A ohne Schwierigkeit ausgenutzt werden kann, im Gegensatz zu den Eliminationsverfahren.

Ein einfaches, hinreichendes Kriterium für die Konvergenz der Iteration ist wieder die *Diagonaldominanz* von A.

Satz 4.5.5 *Die Matrix A sei streng diagonaldominant, (4.3.10). Dann konvergieren Gesamt- und Einzelschrittverfahren, genauer gilt*

$$\|B_E\|_\infty = \|(D+L)^{-1}R\|_\infty \le \|D^{-1}(L+R)\|_\infty = \|B_G\|_\infty < 1. \tag{4.5.17}$$

Somit gelten die Abschätzungen (4.5.8) in der Maximumnorm mit $q = \|B_G\|_\infty$ bzw. $q = \|B_E\|_\infty$.

Beweis Die betrachtete Matrixnorm ist $\|.\|_\infty$.

a) Strenge Diagonaldominanz ist äquivalent mit $\|B_G\| = \max_i \sum_{j \ne i} |a_{ij}/a_{ii}| < 1$.

b) Es wird die stärkere Aussage

$$\|C\| \le \|B_G\|, \quad C := (|D| - |L|)^{-1}|R| \tag{4.5.18}$$

bewiesen. Wie im (vorgezogenen) Beweis von Satz 4.4.4 gilt

$$|B_E| = |(D+L)^{-1}R| \le (|D| - |L|)^{-1}|R| = C.$$

Mit (4.5.18) folgt daraus die Behauptung (4.5.17), denn $\|B_E\| \le \|C\| \le \|B_G\|$. Zur Ungleichung (4.5.18): Da $C \ge 0$ ist $\|C\| = \|z\|_\infty =: q$ wobei $Ce =: z,\ e = (1, \ldots, 1)^T$. Mit

$$(I - |D^{-1}L|)^{-1}|D^{-1}R|e = z \iff z = |D^{-1}L|z + |D^{-1}R|e$$

folgt

$$\begin{aligned} q &= \|z\|_\infty = \max_i \{ \sum_{j<i} |\frac{a_{ij}}{a_{ii}}| z_j + \sum_{j>i} |\frac{a_{ij}}{a_{ii}}| \} \\ &\le \max_i \{ q \sum_{j<i} |\frac{a_{ij}}{a_{ii}}| + \sum_{j>i} |\frac{a_{ij}}{a_{ii}}| \} = \|D^{-1}(qL+R)\|. \end{aligned} \tag{4.5.19}$$

Nach Division durch q wird daraus die Ungleichung

$$1 \le \max_i \{ \sum_{j<i} |\frac{a_{ij}}{a_{ii}}| + \frac{1}{q} \sum_{j>i} |\frac{a_{ij}}{a_{ii}}| \} = \|D^{-1}(L + \frac{1}{q}R)\|,$$

die wegen der Diagonaldominanz aber nur für $q < 1$ erfüllt sein kann. Also gilt $\|C\| = q \le \|D^{-1}(qL+R)\| \le \|D^{-1}(L+R)\| = \|B_G\|$, d.h. (4.5.19)$\Rightarrow$(4.5.18)$\Rightarrow$(4.5.17). ■

Der Satz zeigt, daß bezüglich der Zeilensummennorm das Einzelschrittverfahren nicht schlechter konvergiert als das Gesamtschrittverfahren. Auf die asymptotisch ($k \to \infty$) entscheidende Größe, den Spektralradius, laßt sich diese Aussage in einem wichtigen Spezialfall übertragen.

Satz 4.5.6 (Stein-Rosenberg) *Die Matrix $B_G = -D^{-1}(L+R)$ sei nicht-negativ, $B_G \ge 0$. Dann gilt einer der Fälle*

$$\left.\begin{array}{rlcccc} 0 = & \rho(B_E) & = & \rho(B_G) & & \\ 0 < & \rho(B_E) & < & \rho(B_G) & < & 1 \\ & \rho(B_E) & = & \rho(B_G) & = & 1 \\ & \rho(B_E) & > & \rho(B_G) & > & 1 \end{array}\right\} \iff \quad |\rho(B_G) - 1| \le |\rho(B_E) - 1|,$$

mit Gleichheit genau für $\rho = 0, 1$.

Im nichttrivialen Konvergenzfall konvergiert also das Einzelschrittverfahren für solche Matrizen besser. Es gibt folgende *hinreichenden* Kriterien für die Konvergenz von Gesamt- und Einzelschritt-Verfahren:

1. (4.3.10), strenge Zeilen-Diagonaldominanz: $|a_{ii}| - \sum_{j\ne i} |a_{ij}| > 0 \ \forall i$.
2. strenge Spalten-Diagonaldominanz $|a_{jj}| - \sum_{i\ne j} |a_{ij}| > 0 \ \forall j$.
3. Symmetrie und positive Definitheit.
4. schwache Zeilen- bzw. Spalten-Diagonaldominanz: A unzerlegbar und $|a_{ii}| - \sum_{j\ne i} |a_{ij}| \ge 0$ bzw. $|a_{jj}| - \sum_{i\ne j} |a_{ij}| \ge 0$ mit echter Ungleichung ‘ $>$’ an mindestens einer Stelle.

Im letzten Fall mußte ausgeschlossen werden, daß das LGS in kleinere Teilprobleme zerfällt:

Definition 4.5.7 *Eine Matrix $A = (a_{ij})$ heißt* unzerlegbar, *wenn keine Umordnung $(i_1, i_2, \ldots, i_n)$ der Indizes $(1, 2, \ldots, n)$ existiert so, daß die Matrix dann folgende Struktur hat*

$$\left(a_{i_k i_j}\right)_{k,j=1}^n = \begin{pmatrix} A_{11} & A_{12} \\ 0 & A_{22} \end{pmatrix}.$$

Relaxationsverfahren

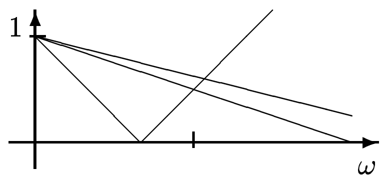
Die gerade beschriebenen Verfahren können oft durch Einführung eines Beschleunigungsparameters ohne praktischen Mehraufwand verbessert werden. Zur Vereinfachung der Darstellung wird hier davon ausgegangen, daß die Matrix des Gleichungssystems schon in der Form $A = I - B$ vorliegt mit einer geeigneten Iterationsmatrix B. Dies ist o.B.d.A. möglich, da für jede reguläre Zerlegung $A = M - N$ das LGS $Ax = b$ äquivalent zu $(I - B)x = M^{-1}b$, $B = M^{-1}N$, ist. Mit $0 < \omega \in \mathbf{R}$ wird das Iterationsverfahrens (4.5.2) modifiziert:

$$x^{(k+1)} := (1-\omega)x^{(k)} + \omega(Bx^{(k)} + r) = x^{(k)} + \omega(r - Ax^{(k)}), \tag{4.5.20}$$

$k = 0, 1, \ldots$. Die Iterationsmatrix ist hier also

$$B(\omega) := (1-\omega)I + \omega B = I - \omega A, \tag{4.5.21}$$

wobei $B(1) = B$. Das Ziel beim Ansatz (4.5.20),(4.5.21) ist es natürlich, ω so zu bestimmen, daß der Konvergenzfaktor $\rho(B(\omega))$ möglichst klein wird. Allgemein ist $\rho(B(0)) = 1$.



Eine Optimierung des Parameters ω ist einfach, wenn A und B nur reelle Eigenwerte besitzen. Die Eigenwerte von A seien daher $\lambda_n \le \ldots \le \lambda_1$, die von B sind dann $\mu_i = 1 - \lambda_i$. Wenn $\lambda_n > 0$ ist, kann ein $\omega > 0$ so gewählt werden, daß $\rho(B(\omega)) = \max_i |1 - \omega\lambda_i| < 1$ wird. Dazu muß man nur die flachste und steilste Gerade betrachten,

$$\rho(B(\omega)) = \max\{|1-\omega\lambda_1|, |1-\omega\lambda_n|\}. \tag{4.5.22}$$

Der optimale Parameter $\hat\omega$ ergibt sich im Schnittpunkt dieser beiden (geknickten) Geraden:

$$-(1-\omega\lambda_1) = 1 - \omega\lambda_n \iff \frac{1}{\hat\omega} = \frac{\lambda_1 + \lambda_n}{2}.$$

Satz 4.5.8 *Die Eigenwerte λ_i von A seien reell und positiv, $0 < \lambda_n \le \ldots \le \lambda_2 \le \lambda_1$. Dann ist der optimale Relaxationsparameter $\hat\omega$ bei (4.5.20) gegeben durch*

$$\hat\omega = \frac{2}{\lambda_1 + \lambda_n} \quad \textit{mit} \quad \rho\Big(B(\hat\omega)\Big) = \frac{\lambda_1 - \lambda_n}{\lambda_1 + \lambda_n} < 1.$$

Bemerkung: 1) Die Konvergenz des einfachen Verfahrens ($\omega = 1$) war nicht vorausgesetzt, d.h. $\mu_1 < -1$, $\lambda_1 > 2$, ist zugelassen.

2) Falls der Mittelpunkt des Spektrums von A kleiner Eins ist, $\frac{1}{2}(\lambda_1 + \lambda_n) < 1$, gilt $\hat\omega > 1$. Man spricht dann von *Überrelaxation.*

Das Verfahren (4.5.20) läßt sich sehr einfach analysieren, da die Iterationsmatrix $B(\omega)$ linear von ω abhängt. Es hat aber, etwa in Verbindung mit dem Einzelschrittverfahren mit $B = -(D+L)^{-1}R$ den Nachteil, daß für den Vektor $x^{(k)}$ und $(D+L)^{-1}Rx^{(k)}$ getrennte Speicherplätze erforderlich sind. Geschickter ist es, die Relaxation direkt in jeder Komponente durchzuführen. Für das LGS $Ax = b$ hat ein einzelner Schritt die Form

Algorithmus 4.5.9 *SOR-Verfahren (successive overrelaxation)*

```
für i := 1 bis n:
    s := b_i;
    für j := 1 bis n: s := s - a_ij * x_j;
    x_i := x_i + ω * s/a_ii;
```
(4.5.23)

Mit Iterationsindizes $k = 0, 1, \ldots,$ versehen lautet ein Verfahrensschritt einfach

$$a_{ii}x_i^{(k+1)} := a_{ii}x_i^{(k)} + \omega[b_i - \sum_{j<i} a_{ij}x_j^{(k+1)} - \sum_{j\geq i} a_{ij}x_j^{(k)}], \quad i = 1, \ldots, n \ .$$

In der Matrix-Schreibweise

$$(D + \omega L)x^{(k+1)} := [(1 - \omega)D - \omega R]x^{(k)} + \omega b \tag{4.5.24}$$

wird allerdings erkennbar, daß die Iterationsmatrix

$$B_S(\omega) := (D + \omega L)^{-1}[(1 - \omega)D - \omega R]$$

nichtlinear vom Parameter ω abhängt. Eine Analyse des Verfahrens ist für diese Vorlesung zu aufwendig. Als Beispiel einer Konvergenzaussage sei zitiert

Satz 4.5.10 *Für eine symmetrische und positiv definite Matrix A gilt*

$$|\omega - 1| \leq \rho(B_S(\omega)) < 1 \quad \forall 0 < \omega < 2.$$

Meist wird eine Beschleunigung nur für $\omega > 1$ erreicht, d.h. bei *Überrelaxation.* Eine sehr genaue Analyse ist bei bestimmten, praktisch wichtigen Matrixstrukturen möglich.

Demo-Beispiel: Poisson-Gleichung auf Einheitsquadrat mit $n = 10$, das LGS besteht i.w. aus Gleichungen der Form $4x_i - x_{i-1} - x_{i+1} - x_{i-10} - x_{i+10} = 0$, wobei für Komponenten mit Indizes außerhalb $\{1, \ldots, 100\}$ feste Werte eingesetzt werden. Theoretisch berechenbar ist $\rho(B_G) = 0.959$, $\rho(B_E) = \rho(B_G)^2 = 0.9206$, $\rho(B_S(\hat{\omega})) = 0.560 \cong \rho(B_E)^7$. Also hat eine SOR-Iteration den gleichen Effekt wie ca. 7 Einzelschritt- oder 14 Gesamtschritt-Iterationen (fast) ohne Mehraufwand.

5 Nichtlineare Gleichungssysteme

Probleme, bei denen Unbekannte $x \in \mathbf{R}^n$ zu bestimmen sind, die Bedingungen in einer der beiden Formen

$$f(x) = 0, \qquad f : \mathbf{R}^n \mapsto \mathbf{R}^n, \tag{5.0.1}$$

$$x = g(x), \qquad g : \mathbf{R}^n \mapsto \mathbf{R}^n, \tag{5.0.2}$$

erfüllen, heißen *nichtlineare Gleichungssysteme*, in ausführlicher Form geschrieben

$$\left\{ \begin{array}{ll} f_1(x_1, \ldots, x_n) & = 0 \\ \vdots & \\ f_n(x_1, \ldots, x_n) & = 0 \end{array} \right. \qquad \left\{ \begin{array}{ll} x_1 = & g_1(x_1, \ldots, x_n) \\ \vdots & \\ x_n = & g_n(x_1, \ldots, x_n) \end{array} \right. .$$

Dabei wird im folgenden mindestens einmalige stetige Differenzierbarkeit der Funktionen f bzw. g angenommen. Die Gleichungen (5.0.1) und (5.0.2) sind Standardformen solcher Probleme, (5.0.1) heißt *Nullstellen-*, (5.0.2) *Fixpunktproblem.* Für die Lösung von (5.0.2) ist im Banachschen Fixpunktsatz ein hinreichendes Existenzkriterium *und* ein Lösungsverfahren bekannt (Iteration). Beim Nullstellenproblem (5.0.1) werden meist Umformungen auf die Gestalt (5.0.2) untersucht, für die der Fixpunktsatz anwendbar ist. Zunächst wird der einfachste Spezialfall $n = 1$ betrachtet.

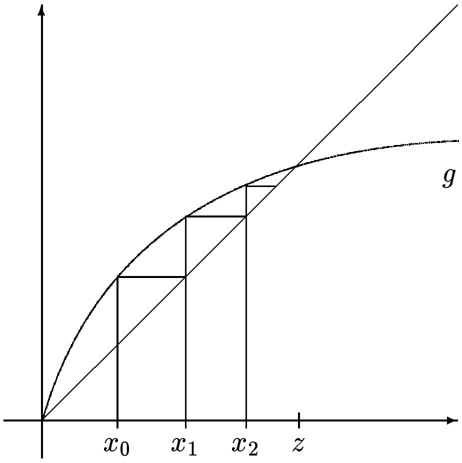
5.1 Nullstellen einer skalaren Funktion

Für reelle Funktionen g können die Aussagen des Fixpunktsatzes durch Monotoniebetrachtungen verfeinert werden. Es sei $g \in C^1[a, b]$.

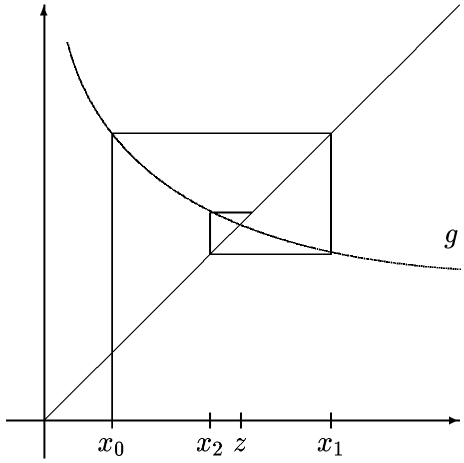
- Die Existenz eines Fixpunkts (oder einer Nullstelle) kann allein aus dem Zwischenwertsatz gefolgert werden:
$$a \leq g(a),\ g(b) \leq b \ \Rightarrow\ \exists\ z = g(z) \in [a, b].$$
- Die Überprüfung der Lipschitzbedingung kann zur Vereinfachung auf den Fixpunkt z selbst beschränkt werden. Wenn $|g'(z)| < 1$ gilt, folgt aus der Stetigkeit von g' die Existenz einer ganzen Umgebung von z, in der Konvergenz eintritt. Zusätzlich wirkt sich das Vorzeichen von $g'(z)$ auf die Art der Konvergenz aus, bei positiver Steigung von g ist die Konvergenz monoton, bei negativer alternierend. Die Umgebung von z, in der Konvergenz gegen z eintritt, der sogenannte *Einzugsbereich* von z, kann sogar wesentlich größer sein als der Bereich $\{x : |g'(x)| < 1\}$ aus dem Fixpunktsatz. Im folgenden linken Beispiel hat die Funktion g den weiteren Fixpunkt 0 mit $g'(0) > 1$ ('abstoßender Fixpunkt'). Offensichtlich konvergiert die Iteration hier für alle Startwerte $x_0 \in (0, z]$ gegen z. Dies läßt sich induktiv ganz einfach aus den Eigenschaften
$$x < g(x) < g(z)\ \forall x \in (0, z)$$

herleiten, denn dann gilt $x_k < g(x_k) = x_{k+1} < z$.

$0 < g'(z) < 1$: monotone Konvergenz



$-1 < g'(z) < 0$: alternierende Konvergenz



- Im Fall $g'(z) = 0$ tritt ein weiterer Effekt auf:

Beispiel 5.1.1 $f(x) = x^2 - a = 0\,(a > 0) \iff x = \frac{a}{x} \iff 2x = x + \frac{a}{x}$.

Es sei $g(x) := \frac{1}{2}(a + \frac{a}{x}) \;\Rightarrow\; z = g(z) = \sqrt{a}$, die Iteration $x_{k+1} := g(x_k)$ heißt *Heron-Verfahren.* Die nebenstehende Tabelle zeigt die Iterierten für $a = 2$. Offensichtlich verdoppelt sich die Anzahl der richtigen Ziffern bei jedem Schritt. Dieser Effekt ist mit dem Banachschen Fixpunktsatz alleine nicht erklärbar.

k	x_k
0	1
1	$\underline{1.}5$
2	$\underline{1.41}66$
3	$\underline{1.41421}56$
4	$\underline{1.414213562}$

Eine Analyse ergibt im speziellen Beispiel die Identität

$$x_{k+1} - z = \frac{x_k^2 + a}{2x_k} - \sqrt{a} = \frac{1}{2x_k}(x_k^2 - 2x_k\sqrt{a} + a) = \frac{1}{2x_k}(x_k - z)^2,$$

bzw. allgemein mit der Eigenschaft $g'(z) = \frac{1}{2}(1 - a/z^2) = 0$ und dem Satz von Taylor

$$x_{k+1} - z = g(x_k) - g(z) = \underbrace{g'(z)}_{=0}(x_k - z) + \frac{1}{2}g''(\xi_k)(x_k - z)^2. \tag{5.1.1}$$

Aus beiden Formeln folgt, daß der Fehler bei jedem Iterationsschritt ungefähr quadriert wird mit der Konsequenz

$$|x_k - z| \le 10^{-m} \;\Rightarrow\; |x_{k+1} - z| \le \frac{1}{2}|g''(z)|\,10^{-2m}.$$

Dies entspricht ungefähr einer Verdopplung der Zahl der gültigen Ziffern pro Schritt.

Zur Beschreibung solcher Effekte führt man die *Konvergenzordnung* von Iterationsverfahren ein.

Definition 5.1.2 *Eine Folge $\{x_k\}$ mit $\lim_{k\to\infty} x_k = z$, bzw. ein sie erzeugendes Iterationsverfahren, heißen konvergent von der* Ordnung $p \ge 1$, *wenn $k_0 \in \mathbf{N}$, $q \in \mathbf{R}_+$ (und $q < 1$ bei Ordnung $p = 1$) existieren mit*

$$|x_{k+1} - z| \le q\,|x_k - z|^p \quad \forall k \ge k_0. \tag{5.1.2}$$

Im allgemeinen wird als Ordnung das maximal mögliche p angegeben. Für $p = 1, 2, 3$ spricht man speziell von linearer, quadratischer und kubischer Konvergenz. Die Ordnung eines Iterationsverfahrens läßt sich an der Taylor-Entwicklung der Iterationsfunktion g im Fixpunkt z ablesen (wie im Beispiel).

Satz 5.1.3 *Es sei $z = g(z) \in U$ ein Fixpunkt von $g \in C^p(U)$, $p > 1$, U offen. Es gelte*

$$g'(z) = \ldots = g^{(p-1)}(z) = 0, \quad g^{(p)}(z) \neq 0.$$

Dann konvergiert die Iterationsfolge $\{x_k\}$, $x_k := g(x_{k-1})$, für Startwerte x_0 genügend nahe an z mit der Ordnung p gegen z.

Beweis Wenn $0 < |x_0 - z| \leq \varepsilon$ mit genügend kleinem ε gilt, ist

$$x_{k+1} - z = g(x_k) - g(z) = g'(z)(x_k - z) + \ldots + \frac{g^{(p-1)}(z)}{(p-1)!}(x_k - z)^{p-1} + \frac{g^{(p)}(\xi_k)}{p!}(x_k - z)^p$$

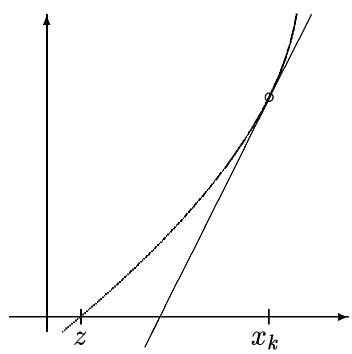
wobei rechts alle Summanden, bis auf den letzten, verschwinden ($g^{(p)}(\xi) \neq 0$ wegen der Stetigkeit). Daher gilt (5.1.2). Die Konvergenz folgt aus

$$|x_{k+1} - z| \leq q|x_k - z|^p \leq (q\varepsilon^{p-1})|x_k - z|,$$

wenn $q\varepsilon^{p-1} < 1$. ∎

Bei der Konstruktion von Iterationsverfahren zur Nullstellenbestimmung geht man ähnlich vor wie früher: Die Funktion f wird durch ein einfacheres Modell approximiert (z.B. ein Polynom), für das sich eine Nullstelle einfach berechnen läßt. Dazu gibt es verschiedene Ansätze.

Newton-Verfahren



Approximation der Funktion f durch Abschnitte des Taylorpolynoms:

$$f(z) = \underbrace{\underbrace{0 = f(x_k) + f'(x_k)(z - x_k)}_{\to \text{ Newton-Verf.}} + \frac{1}{2}f''(x_k)(z - x_k)^2}_{\to \text{ Newton-Raphson-Verf.}} + \ldots$$

Das lineare Modell führt auf das *Newtonverfahren*

$$x_{k+1} := g_N(x_k), \quad g_N(x) := x - \frac{f(x)}{f'(x)}\ , \tag{5.1.3}$$

das quadratische Modell auf das *Newton-Raphson-Verfahren*

$$x_{k+1} := g_R(x_k), \quad g_R(x) := x - \frac{2f(x)}{f'(x) + \text{sign}f'\sqrt{f'(x)^2 - 2f(x)f''(x)}}\ . \tag{5.1.4}$$

Nur das Newtonverfahren wird im folgenden genauer analysiert. Offensichtlich gilt

$$z = g_N(z) = z - \frac{f(z)}{f'(z)} \;\Rightarrow\; f(z) = 0.$$

Daher ist jeder Fixpunkt von g_N auch Nullstelle von f. Als Anwendung von Satz 5.1.3 ergeben sich für $f'(z) \neq 0$ aus

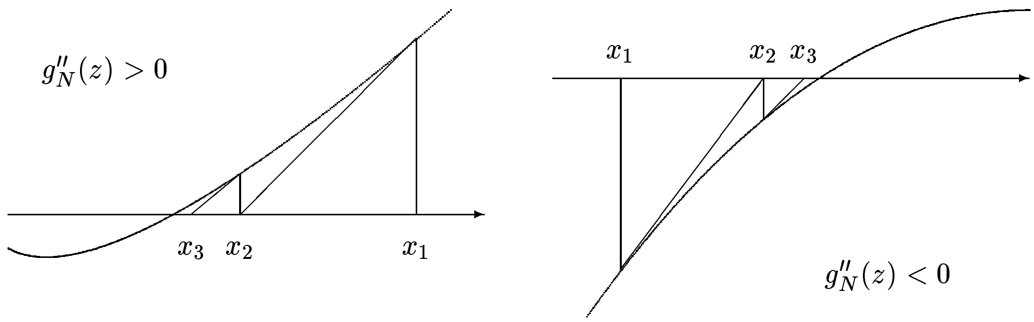
$$\begin{aligned} g_N'(x) &= 1 - \frac{f'(x)^2 - f(x)f''(x)}{f'(x)^2} = f(x)\frac{f''(x)}{f'(x)^2} = 0 \quad \text{in } x = z, \\ g_N''(x) &= \frac{f'(x)^2 f''(x) + f(x)[\cdots]}{f'(x)^3} = \frac{f''(z)}{f'(z)} \qquad\qquad \text{in } x = z, \end{aligned} \tag{5.1.5}$$

die Aussagen, daß (für $f'(z) \neq 0$) das Newtonverfahren *quadratisch* konvergiert, in einem Wendepunkt von f (mit $f''(z) = 0$) sogar kubisch. Das Newton-Raphson-Verfahren (5.1.4) konvergiert i.a. kubisch.

Beim Newton-Verfahren verläuft die Konvergenz meist *monoton*, im Beispiel 5.1.1, z.B., galt ab dem zweiten Schritt $x_k \searrow \sqrt{a}$. Dies folgt aus der Fehlerdarstellung (5.1.1),

$$x_{k+1} - z = \frac{1}{2} g_N''(\xi_k) \underbrace{(x_k - z)^2}_{\geq 0} \quad \Rightarrow \quad \text{sign}(x_{k+1} - z) = \text{sign } g_N''(z) = \text{sign } \frac{f''(z)}{f'(z)}$$

nahe bei z.



Bemerkung: 1) Wie die meisten Verfahren höherer Ordnung konvergiert das Newtonverfahren i.a. nur lokal, d.h. für genügend kleine Startfehler $|x_0 - z|$. Eine konkrete Aussage über dessen zulässige Größe folgt im nächsten Abschnitt für den $\mathbf{R}^n$. Der *Einzugsbereich* des Newtonverfahrens kann in ungünstigen Fällen sehr klein sein. Dann ist es sehr schwer, einen geeigneten Startwert x_0 zu finden.

2) Auch bei mehrfachen Nullstellen, d.h. bei $f'(z) = 0$, ist das Newtonverfahren noch verwendbar. Bei $f(x) = (x - z)^m r(x),\; r(z) \neq 0$, gilt nämlich

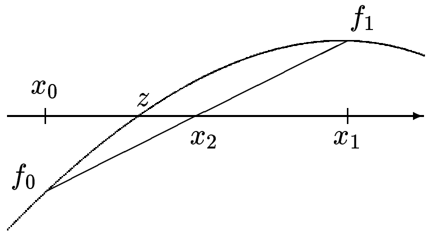
$$g_N(x) = x - \frac{(x - z)^m r(x)}{m(x - z)^{m-1} r(x) + (x - z)^m r(x)} = x - \frac{(x - z) r(x)}{m r(x) + (x - z) r'(x)} \quad \Rightarrow$$

$$x_{k+1} - z = g_N(x_k) - z = (x_k - z)\Big(1 - \frac{1}{m + (x_k - z) r'(x_k)/r(x_k)}\Big) \cong (1 - \frac{1}{m})(x_k - z).$$

Daher konvergiert das Newtonverfahren noch linear mit dem Kontraktionsfaktor $1 - \frac{1}{m} < 1$.

Sekantenverfahren, Regula falsi

Zur Approximation der Funktion f bei der Nullstellenbestimmung ist außer dem Taylorpolynom auch ein Interpolationspolynom einsetzbar. Ausgehend von zwei Funktions-Wertepaaren $(x_0, f_0), (x_1, f_1)$ läßt sich eine verbesserte Näherung über die Nullstelle der linearen Interpolierenden (*Sekante*) bestimmen:



$$s(x) = f[x_1] + (x - x_1) f[x_1, x_0] = f_1 + (x - x_1)\frac{f_1 - f_0}{x_1 - x_0} \stackrel{!}{=} 0.$$

Dies führt auf den neuen Wert

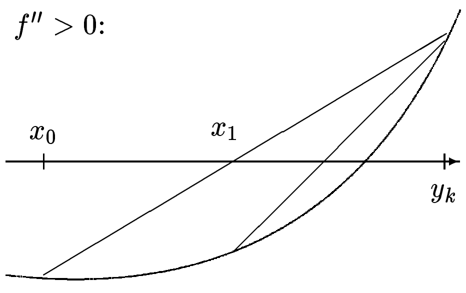
$$x_2 = x_1 - f_1 \frac{x_1 - x_0}{f_1 - f_0} = \frac{x_0 f_1 - x_1 f_0}{f_1 - f_0}. \tag{5.1.6}$$

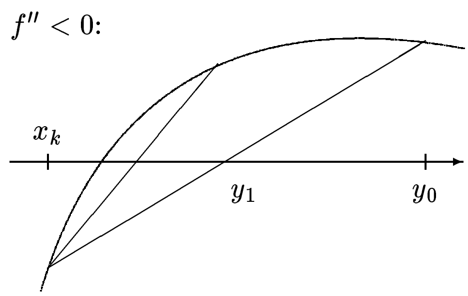
Die erste Form betont die Ähnlichkeit mit dem Newtonverfahren, da $(x_1 - x_0)/(f_1 - f_0) \cong 1/f'(x_1)$. Bei Wiederholung der Konstruktion (5.1.6) mit $\{x_1, x_2\}$ kann die nächste Näherung allerdings wieder schlechter als x_0 sein. Dies wird verhindert bei der

Regula falsi Mit (5.1.6) wird stets eine *Einschließung* der Nullstelle z konstruiert. Ausgehend von einem Intervall $[x_k, y_k]$ mit $f(x_k)f(y_k) < 0$ wird definiert

$$[x_{k+1}, y_{k+1}] := \begin{cases} [m_k, y_k] & \text{für} \quad f(m_k)f(y_k) < 0 \\ [x_k, m_k] & \text{für} \quad f(x_k)f(m_k) < 0 \end{cases} \quad \text{mit } m_k := \frac{x_k f(y_k) - y_k f(x_k)}{f(y_k) - f(x_k)}. \tag{5.1.7}$$

Nach Konstruktion ist dann $f(x_{k+1})f(y_{k+1}) < 0$. Allerdings hat das Verfahren den entscheidenden Nachteil, daß einer der Randpunkte nie verbessert wird, wenn f'' in der Nähe von z ein festes Vorzeichen besitzt:





Daher besitzt dieses Verfahren i.a. nur die Konvergenzordnung eins. Bei Verzicht auf die Einschließungseigenschaft erhält man das

Sekantenverfahren Ohne Beachtung des Vorzeichenwechsels der Funktionswerte wird (5.1.6) als Iterationsvorschrift angewendet:

$$x_{k+1} = x_k - f(x_k)\frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})}, \quad k = 1, 2, \ldots \tag{5.1.8}$$

Dieses Iterationsverfahren hat eine andere Struktur als alle bisherigen. Da zur Konstruktion von x_{k+1} zwei frühere Iterierte verwendet werden, handelt es sich nicht mehr um eine einfache Fixpunktiteration, sondern um eine *mehrstufige* Iteration.

Definition 5.1.4 *Ein Iterationsverfahren der Form*

$$x_{k+1} := G(x_k, x_{k-1}, \dots, x_{k-m+1}), \; k \geq m-1,$$

mit $G : \mathbf{R}^m \mapsto \mathbf{R}$ *heißt* m-stufiges Verfahren.

Zur Untersuchung der Konvergenz des Verfahrens betrachten wir die Fortpflanzung der Fehler. Aus (5.1.8) folgt

$$\begin{aligned} x_{k+1} - z &= x_k - z - \frac{f(x_k) - f(z)}{f[x_{k-1}, x_k]} = (x_k - z)\frac{f[x_{k-1}, x_k] - f[x_k, z]}{f[x_{k-1}, x_k]} \\ &= \frac{(x_k - z)(x_{k-1} - z)}{f[x_{k-1}, x_k]} \, \frac{f[x_{k-1}, x_k] - f[x_k, z]}{x_{k-1} - z} \\ &= (x_k - z)(x_{k-1} - z)\frac{f[x_{k-1}, x_k, z]}{f[x_{k-1}, x_k]}, \end{aligned} \tag{5.1.9}$$

oder kurz mit $e_k := x_k - z$:

$$e_{k+1} = e_k \, e_{k-1} \, \frac{f[x_{k-1}, x_k, z]}{f[x_{k-1}, x_k]}. \tag{5.1.10}$$

Aus (2.3.10) folgt, daß Stellen ξ_k, η_k aus der konvexen Hülle von x_k, x_{k-1}, z existieren mit

$$f[x_{k-1}, x_k] = f'(\xi_k), \;\; f[x_{k-1}, x_k, z] = \frac{1}{2} f''(\eta_k).$$

Für $f'(z) \neq 0$ gibt es daher eine Umgebung von z, in der (5.1.8) konvergiert.

Satz 5.1.5 *Es sei* $U = [z-r, z+r], r > 0$, *eine Umgebung der Nullstelle* z *von* $f \in C^2(U)$. *Es gelte*

$$|f'(x)| \geq m_1 > 0, \;\; \frac{1}{2}|f''(x)| \leq M_2 \;\; \forall x \in U.$$

Dann konvergiert das Sekantenverfahren (5.1.8) für zwei beliebige Startwerte $x_0, x_1 \in U$ *mit* $M|x_0 - z| < 1, \; M|x_1 - z| < 1$ *(*$M := M_2/m_1$*) gegen die Nullstelle* z*. Die Konvergenzordnung ist dabei mindestens* $p_S = \frac{1}{2}(1 + \sqrt{5}) = 1.618..$ *.*

Beweis a) Nach (5.1.10) gilt mit $q := M \max\{|e_0|, |e_1|\} < 1$ induktiv

$$|e_{k+1}| \leq M|e_k e_{k-1}| \leq q|e_k| < |e_k|, \;\; k \geq 1.$$

Somit übertragen sich die benötigten Eigenschaften auf e_{k+1}, es ist $|e_{k+1}| < r, M|e_{k+1}| < q$ und es folgt $|e_{k+1}| \leq q^k |e_1| \to 0$.

b) Wir betrachten nur den Fall $f''(z) \neq 0$ und zeigen, daß das Verfahren dann genau von der Ordnung $p = p_S$, mit $p^2 = p + 1$, konvergiert. Dazu sei $\hat{q} := \frac{1}{2} f''(z)/f'(z) \neq 0$ und

$e_0, e_1 \neq 0$. Nach Teil a) und (5.1.10) gilt dann $e_{k+1} = q_k e_k e_{k-1}$ mit $q_k \neq 0$ für $k \geq k_0$, $q_k \to \hat{q}$, und die neue Folge $c_k := e_{k+1}/e_k^p$ ist dann wohldefiniert. Für sie gilt

$$c_k = \frac{e_{k+1}}{e_k^p} = q_k e_k^{1-p} e_{k-1} = q_k c_{k-1}^{1-p} e_{k-1}^{1+p-p^2} = q_k c_{k-1}^{1-p}.$$

Dies ist eine einstufige Rekursionsformel, zu der wegen $0 < p - 1 < 1$ eine beschränkte Folge $\{c_k\}$ gehört (Übung). Daraus folgt die Behauptung. ∎

Bemerkungen: 1) Die Ordnung des Sekantenverfahrens ist etwas geringer als die des Newtonverfahrens. Allerdings benötigt ein Iterationsschritt (5.1.8) nur *eine* neue Funktionsauswertung, während beim Newtonverfahren $f(x_k)$ und $f'(x_k)$ zu berechnen sind. Dies ist bei einem Effizienzvergleich zu berücksichtigen: Mit Hilfe von $k = 2m$ Funktionsauswertungen wird der Anfangsfehler $|e_0|$ i.w. reduziert auf

$$|e_0|^{p_N^{k/2}} \text{ (Newton-Verf.)} \quad \text{bzw.} \quad |e_0|^{p_S^k} \text{ (Sekanten-Verf.)}.$$

Die *effektiven* Ordnungen (bezogen auf die Zahl der Funktionsauswertungen) sind also für Newton- und Sekanten-Verfahren

$$p_N^{1/2} = \sqrt{2} = 1.414\ldots < p_S = 1.618\ldots \; .$$

Da das Sekantenverfahren somit eine bessere effektive Ordnung besitzt und außerdem keine Ableitungen benötigt, ist es für skalare Probleme dem Newtonverfahren i.a. vorzuziehen (HP-Rechner: Solve-Taste).

Beispiel 5.1.6 $f(x) := \cos(x)\cosh(x) + 1$. Anwendung von Newton- und Sekantenverfahren:

Newton	Sekanten
x_0=1.8	x_0=1.8, x_1=1.9
	$x_2=\underline{1.87}36979$
$x_1=\underline{1.87}95674$	$x_3=\underline{1.875}0786$
	$x_4=\underline{1.8751040}95$
$x_2=\underline{1.8751}186$	$x_5=\underline{1.875104069}$

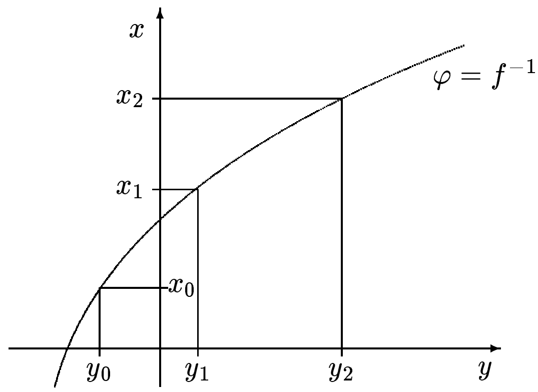
2) Die Vorteile von Regula falsi und Sekantenverfahren lassen sich durch einfache Überwachung vereinigen. Zusätzlich zur Folge $\{x_k\}$ aus (5.1.8) wird die bisher beste Einschließung

$$a_k := \max\{x_j : x_j \leq z, j \leq k\}, \quad b_k := \min\{x_j : x_j \geq z, j \leq k\},$$

gespeichert und ein $x_{k+1} \notin [a_k, b_k]$ durch den besseren Randwert ersetzt. Weitere Varianten sind in der Literatur beschrieben, ihre Konvergenzordnungordnungen reduzieren sich geringfügig auf ungefähr 1.6.

Interpolationsverfahren höherer Ordnung

Beim Übergang zu Interpolationspolynomen von f höherer Ordnung tritt das Problem auf, dessen Nullstellen bestimmen zu müssen. Dies läßt sich durch Betrachtung der (lokalen) Umkehrfunktion $\varphi := f^{-1}$ umgehen. Wenn nämlich z Nullstelle von f ist, $f(z) = 0$, dann gilt natürlich $z = \varphi(0)$. Dazu werden die Wertepaare $(x_j, y_j = f(x_j))$, $j = k, k-1, \ldots, k-m$, als Daten der Umkehrfunktion interpretiert, $(y_j, \varphi(y_j) = x_j)$, und so interpoliert:



$$q_m \in \Pi_m : \quad q_m(y_j) = x_j, \; j = k, k-1, \ldots, k-m.$$

Die nächste Iterierte ergibt sich als $x_{k+1} = q_m(0)$. Die Verbesserungen gegenüber dem Sekantenverfahren sind aber nicht mehr groß (Ordnung $< 2 \; \forall m$).

5.2 Newtonverfahren im $\mathbf{R}^n$

Im Gegensatz zum skalaren Fall $n = 1$ gibt es bei der Lösung nichtlinearer Gleichungs*systeme* (5.0.1) mit $n > 1$ kaum eine Alternative zum Newtonverfahren. Nach Definition der Ableitung in einer Stelle $\bar{x} \in \mathbf{R}^n$,

$$f'(\bar{x}) := \frac{\partial f}{\partial x}(\bar{x}) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1}(\bar{x}) & \ldots & \frac{\partial f_1}{\partial x_n}(\bar{x}) \\ \vdots & & \vdots \\ \frac{\partial f_n}{\partial x_1}(\bar{x}) & \ldots & \frac{\partial f_n}{\partial x_n}(\bar{x}) \end{pmatrix} = \begin{pmatrix} \nabla f_1 \\ \vdots \\ \nabla f_n \end{pmatrix} \quad \text{zu } f = \begin{pmatrix} f_1 \\ \vdots \\ f_n \end{pmatrix}, \tag{5.2.1}$$

gilt mit der Matrix $A := f'(\bar{x})$ die Aussage

$$f(x) = f(\bar{x}) + A(x - \bar{x}) + o(\|x - \bar{x}\|), \; x \to \bar{x}. \tag{5.2.2}$$

Falls der Abstand $\|z - \bar{x}\|$ zu einer Nullstelle z klein genug ist, kann zur Approximation von z das *lineare Modell* benutzt werden:

$$0 = f(z) = f(\bar{x}) + A(z - \bar{x}) + \ldots \quad \Rightarrow \quad z \cong \bar{x} - A^{-1} f(\bar{x}).$$

Dies führt auf das folgende *Newtonverfahren* im $\mathbf{R}^n$, bei dem mit einem Startvektor $x^{(0)} \in \mathbf{R}^n$ die Iteration

$$f'(x^{(k)})(x^{(k+1)} - x^{(k)}) = -f(x^{(k)}), \quad k = 0, 1, \ldots, \tag{5.2.3}$$

durchgeführt wird. In jedem Schritt (5.2.3) ist dabei ein lineares Gleichungssystem zu lösen, mit dessen Lösung die aktuelle Näherung korrigiert wird,

$$\begin{array}{rcl} A_k s^{(k)} & = & -f(x^{(k)}) \qquad \text{mit Matrix } A_k = f'(x^{(k)}), \\ x^{(k+1)} & := & x^{(k)} + s^{(k)}. \end{array} \tag{5.2.4}$$

Die Inverse A_k^{-1} wird dazu aber i.a. *nicht* berechnet (vgl. §4). Die Gestalt des Einzugsbereichs der Iteration (5.2.3) ist in der Regel sehr kompliziert. Unter geeigneten Voraussetzungen kann aber eine Kugel um z angegeben werden, in der Konvergenz $x^{(k)} \to z$ eintritt für jeden Startwert $x^{(0)}$ aus der Kugel. Dazu sei an den Mittelwertsatz im $\mathbf{R}^n$ erinnert:

Satz 5.2.1 *Es sei $D \subseteq \mathbf{R}^n$ offen, $f \in C^2(D)$ und $D_0 \subseteq D$ konvex und abgeschlossen. Dann gelten für $x, y \in D_0$ die Aussagen*

$$\begin{aligned}
f(x) - f(y) &= \int_0^1 f'\Big(y + t(x-y)\Big)\, dt\ (x-y), && (5.2.5)\\
\|f(x) - f(y)\| &\le L_1\|x-y\|, \quad L_1 := \max\{\|f'(x)\| : x \in D_0\}, && (5.2.6)\\
\Big(f'(x) - f'(y)\Big)z &= \int_0^1 f''(y + t(x-y))\, dt\ [z, x-y]\ \forall z \in \mathbf{R}^n, && (5.2.7)\\
\|f'(x) - f'(y)\| &\le L_2\|x-y\|, \quad L_2 := \max\{\|f''(x)\| : x \in D_0\}. && (5.2.8)
\end{aligned}$$

Bei (5.2.7) ist zu beachten, daß f'' eine bilineare Abbildung ist, $f''(x) : \mathbf{R}^n \times \mathbf{R}^n \to \mathbf{R}^n$. Die Komponenten f_i'' sind symmetrische Matrizen und $f''(x)[z,y] = \Big(z^T f_i''(x) y\Big)_{i=1}^n$.

Beweis Es werden die skalaren Funktionen $\varphi_i(t) := f_i(y + t(x-y))$, $t \in [0,1]$ betrachtet. Nach der Kettenregel gilt

$$\begin{aligned}
\frac{d\varphi_i}{dt}(t) &= \mathrm{grad} f_i(y + t(x-y)) \cdot (x-y) \quad \Rightarrow\\
\varphi_i(1) - \varphi_i(0) &= \int_0^1 \varphi_i'(t) dt = \int_0^1 \mathrm{grad} f_i(y + t(x-y)) \cdot (x-y),
\end{aligned}$$

d.h. (5.2.5) nach (5.2.1). Die Schranke (5.2.6) ergibt sich daraus sofort, denn

$$\|f(x) - f(y)\| \le \|\int_0^1 f'(y + t(x-y)) dt\|\, \|x-y\| \le \max_{t \in [0,1]} \|f'(y + t(x-y))\|\, \|x-y\|.$$

Der Beweis von (5.2.7),(5.2.8) geht analog. ■

Mit diesen Hilfsmitteln läßt sich nun die lokale Konvergenz des Newtonverfahrens mit Ordnung zwei zeigen.

Satz 5.2.2 *Es sei $D \subseteq \mathbf{R}^n$ offen, $f \in C^2(D)$ und $z \in D$ mit $f(z) = 0$. Die Ableitung $f'(z)$ sei regulär und es gelte*

$$\begin{aligned}
\|f'(z)^{-1}\| &\le \frac{1}{\lambda_1}, && (5.2.9)\\
\sup_{x \in D} \|f''(x)\| &\le L_2. && (5.2.10)
\end{aligned}$$

Die Kugel $K_\delta := \{x \in \mathbf{R}^n :\ \|x - z\| \le \delta\}$ sei in D enthalten,

$$K_\delta \subseteq D \quad \textit{für} \quad \delta < \frac{2\lambda_1}{3L_2}.$$

Dann konvergiert das Newtonverfahren (5.2.3) für jeden Startvektor $x^{(0)} \in K_\delta$ gegen z. Die Konvergenz ist quadratisch,

$$\|x^{(k+1)} - z\| \le \frac{1}{2}\frac{L_2}{\lambda_1 - \delta L_2}\|x^{(k)} - z\|^2. \tag{5.2.11}$$

Beweis O.B.d.A. wird der erste Schritt mit $x^{(0)} =: x$ betrachtet. Es gilt nach (5.2.3) und (5.2.5)

$$\begin{aligned} \|x^{(1)} - z\| &= \|\, x - z - f'(x)^{-1}(f(x) - f(z))\| \\ &= \|\Big(I - f'(x)^{-1}\int_0^1 f'(z + t(x-z))dt\Big)(x - z)\| \\ &\le \|f'(x)^{-1}\| \int_0^1 \|f'(x) - f'(z + t(x-z))\|dt\ \|x - z\|. \end{aligned} \tag{5.2.12}$$

Nach (5.2.8) mit $y = z + t(x - z)$, d.h., $x - y = (1-t)(x - z)$, läßt sich die Differenz der Ableitungen abschätzen durch

$$\int_0^1 \|f'(x) - f'(z + t(x-z))\|dt \le L_2 \int_0^1 (1-t)dt\,\|x - z\| = \frac{1}{2}L_2\|x - z\|. \tag{5.2.13}$$

Als nächstes wird die Regularität der Ableitung $f'(x)$ gezeigt. Es ist $f'(z)^{-1}f'(x) = I + f'(z)^{-1}(f'(x) - f'(z))$. Nach Voraussetzung (5.2.9) ist der zweite Summand klein,

$$\|f'(z)^{-1}(f'(x) - f'(z))\| \le \frac{L_2}{\lambda_1}\|x - z\| \le \frac{L_2\delta}{\lambda_1} < \frac{2}{3} < 1,$$

nach dem Satz von Neumann, Satz 4.2.2, existiert daher $f'(x)^{-1}$ und nach (4.2.15) ist

$$\|f'(x)^{-1}\| \le \|f'(z)^{-1}\|\frac{1}{1 - L_2\delta/\lambda_1} \le \frac{1}{\lambda_1 - L_2\delta}.$$

Diese Schranke und (5.2.13) führen mit (5.2.12) auf die letzte Behauptung (5.2.11),

$$\|x^{(1)} - z\| \le \frac{1}{\lambda_1 - \delta L_2}\frac{1}{2}L_2\|x - z\|^2.$$

Damit ist die quadratische Konvergenz gesichert, wenn noch $\|x^{(1)} - z\| \le q\|x - z\|,\ q < 1$, gilt. Dies kann für $\delta < 2\lambda_1/(3L_2)$ garantiert werden:

$$\|x^{(1)} - z\| \le \underbrace{\frac{1}{2}\frac{L_2\delta}{\lambda_1 - \delta L_2}}_{=:\, q < 1}\|x - z\| < \frac{1}{3}\frac{\lambda_1}{\lambda_1 - 2\lambda_1/3}\|x - z\| = \|x - z\|. \qquad \blacksquare$$

Durchführung des Newtonverfahrens

1. Satz 5.2.2 garantiert Konvergenz nur in einer, eventuell sehr kleinen, Kugel um die Nullstelle (*lokale Konvergenz*). Das globale Konvergenzverhalten des Verfahrens kann verbessert werden, indem der Fortschritt der Iteration anhand der Norm $\|f(x)\|_2$ überprüft wird. Man versucht dabei, die Funktion

$$\varphi(x) := \|f(x)\|_2^2 \qquad \text{zu } \textbf{minimieren}.$$

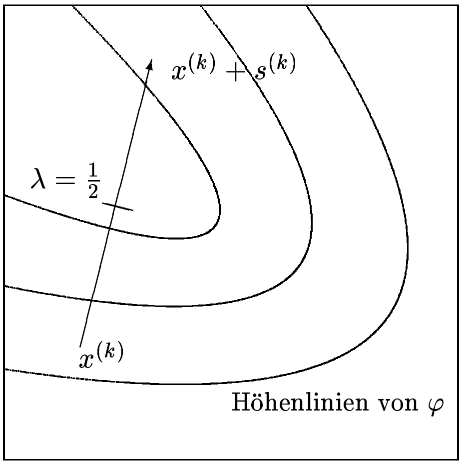
Dazu wird der Vektor des Newtonschritts $s^{(k)} := -f'(x^{(k)})^{-1}f(x^{(k)})$ nur als Such*richtung* benutzt, und der nächste Punkt $x^{(k+1)}$ auf dem Strahl

$$x^{(k+1)} := x^{(k)} + \lambda_k s^{(k)}, \quad 0 < \lambda_k \le 1,$$

so bestimmt, daß $\varphi(x^{(k+1)}) < \varphi(x^{(k)})$ gilt. Man führt eine sog. *Liniensuche* durch. Da

$$\frac{d}{d\lambda}\varphi(x^{(k)} + \lambda s^{(k)})\Big|_{\lambda=0} = -2\varphi(x^{(k)}) < 0$$

gilt, existiert ein solcher Punkt für genügend kleines λ. In der Praxis kann dazu die Strecke $x^{(k)} \dots x^{(k)} + s^{(k)}$ solange halbiert werden, bis φ einen kleineren Wert annimmt. Diese Wahl führt auf $\lambda_k \in \{1, \frac{1}{2}, \frac{1}{4}, \dots\}$. Da dann $\lambda_k \le 1$ gilt, spricht man vom *gedämpften Newtonverfahren.* Nahe bei der Nullstelle wählt diese Methode automatisch wieder $\lambda_k = 1$, sodaß die quadratische Konvergenz nicht zerstört wird.



2. Im $\mathbf{R}^n$ sind pro Schritt des Newtonverfahrens $n^2 + n$ Komponenten von f und f' zu berechnen und eine LR-Zerlegung durchzuführen. Wenn $x^{(k)}$ genügend nahe bei z liegt, kann evtl. die Ableitungsmatrix (und ihre LR-Zerlegung) von früher übernommen werden. Dies führt zum *vereinfachten Newtonverfahren*

$$f'(x^{(0)})(x^{(k+1)} - x^{(k)}) = -f(x^{(k)}), \quad k = 0, 1, \dots. \tag{5.2.14}$$

Bei den Gleichungssystemen ändert sich jetzt nur die rechte Seite (vgl. §4.3), $-f(x^{(k)})$. Die Konvergenz der vereinfachten Iteration (5.2.14) ist dafür nur noch linear mit dem asymptotischen Konvergenzfaktor $\|I - f'(x^{(0)})^{-1}f'(z)\|$.

Demo-Beispiel Im Quadrat $[-1, 1] \times [-1, 1] \subseteq \mathbf{R}^2$ besitzt die Funktion

$$f(x) = \begin{pmatrix} x_1^2 + 9x_2^2 + 2x_1 - 3 \\ 4x_1^2 - x_2^2 - x_2 - 1 \end{pmatrix}$$

4 Nullstellen bei $(-0.72, 0.66)$, $(0.62, 0.39)$, $(0.43, -0.46)$ und $(-0.44, -0.64)$. Die normale Newton-Iteration mit dem Startwert $(0, 0.4)$ erzeugt die Werte (12-stellige Rechnung)

k	$x_1^{(k)}$	$x_2^{(k)}$	$\varphi(x^{(k)})$
0	0.000000000	0.400000000	2.2E+00
1	3.900000000	-0.466666667	6.4E+01
2	1.974863569	-0.097182979	1.5E+01
3	1.024133838	-0.509788636	4.2E+00
4	0.603428523	-0.429913469	7.4E-01
5	0.457214218	-0.460213498	9.0E-02
6	0.433986060	-0.464807423	2.3E-03
7	0.433367993	-0.464932100	1.6E-06
8	0.433367555	-0.464932188	0.0E+00

Der erste Schritt führt nach weit außerhalb des Quadrats mit einer starken Vergrößerung des Wertes φ. Später konvergiert die Iteration offensichtlich quadratisch gegen eine Lösung in der unteren Halbebene. Bei einer Liniensuche im ersten Schritt wäre die nähergelegene Nullstelle bei $(0.62, 0.39)$ gefunden worden.

6 Analyse der Rundungsfehler

Bisher wurden alle Verfahren auf der Basis einer Rechnung mit reellen Zahlen diskutiert. In einem Computer stehen aber nur endlich viele Zahldarstellungen (Maschinenzahlen) zur Approximation reeller Zahlen zur Verfügung. Bei jeder Dateneingabe und -verknüpfung treten daher i.a. Fehler auf, die sich — eventuell katastrophal — auf das Endergebnis auswirken können.

6.1 Zahldarstellung und Rundungsfehler

Die gängige (Dezimal-) Darstellung reeller Zahlen, z.B. e=2.71828.., entspricht der Angabe der Koeffizienten in folgender Reihen-Entwicklung

$$e = 2 \cdot 10^0 + 7 \cdot 10^{-1} + 1 \cdot 10^{-2} + 8 \cdot 10^{-3} + \ldots$$

zur *Basis* 10. Auf Computern ist eine Zahldarstellung günstiger zu einer Basis B, die eine Zweierpotenz ist, z.B., $B = 2,\ B = 16$. Jede reelle Zahl x ist darstellbar in der Form

$$x = \pm(x_m B^m + x_{m-1} B^{m-1} + \ldots + x_1 B + x_0 + x_{-1} B^{-1} + \ldots), \tag{6.1.1}$$

mit $x_j \in \{0, 1, \ldots, B-1\}$, wenn $1 < B \in \mathbf{N}$. Diese Darstellung ist nur dann eindeutig, wenn von zwei möglichen Darstellungen die endliche gewählt wird (man beachte: $1.999\ldots = 2$). Eine Abbildung mit möglichst kleinem *relativem* Fehler von (6.1.1) auf eine endliche Zahlmenge läßt sich bei der Gleitpunktdarstellung erreichen. Bei dieser werden der höchste Exponent m und die ersten ℓ Ziffern $x_m, \ldots, x_{m-\ell+1}$ angegeben. Der zulässige Exponentenbereich muß dabei natürlich auch beschränkt werden. Außerdem ist eine Standardisierung in folgender Weise üblich:

Definition 6.1.1 *Die Menge M der* normalisierten Gleitpunktzahlen *enthält die Elemente $x = 0$ und die Zahlen*

$$\begin{aligned} x &= \pm 0.\xi_1\xi_2\ldots\xi_\ell \cdot B^d := \pm(\xi_1 B^{-1} + \xi_2 B^{-2} + \ldots + \xi_\ell B^{-\ell}) B^d, \\ & \quad \textit{mit } \xi_j \in \{0, 1, \ldots, B-1\},\ \xi_1 \neq 0,\ d \in \mathbf{Z},\ |d| \le e. \end{aligned} \tag{6.1.2}$$

Im folgenden wird allerdings ein unbeschränkter Exponentenbereich ($e = \infty$) angenommen.

Beispiel 6.1.2 Binärdarstellung, $B = 2$. Wegen der Normalisierung $\xi_1 \neq 0$ gilt immer $\xi_1 = 1$. Daher kann an Stelle von ξ_1 das Vorzeichen gespeichert werden. In Turbo-Pascal wird eine *real*-Zahl in 5+1=6 Byte gespeichert:

$\pm 0.\xi_1\xi_2\ldots\xi_\ell \cdot B^d \quad \mapsto$

d	$\pm\xi_2\xi_3..$	$\cdots$	$\cdots$	$\cdots$	$\cdots\xi_\ell$

Hier gilt also $\ell = 40, e = 128$. Die Zahlgenauigkeit beträgt ca. 12 Dezimalstellen, die Größe ist durch $2^d \cong 10^{38}$ beschränkt.

Ein zentraler Bestandteil jeder Computerarithmetik ist die Zuordnung von reellen Zahlen zu Gleitpunktzahlen, also die Abbildung

$$rd: \begin{cases} \mathbf{R} & \mapsto & M \\ x & \mapsto & rd(x) \end{cases}.$$

Eine triviale Variante der Rundung ist das Abschneiden der Darstellung (6.1.1) mit $\xi_1 \ldots \xi_\ell = x_m \ldots x_{m-\ell+1}$. Günstiger ist aber die gewöhnliche Rundung

$$rd(\pm 0.x_1x_2\ldots x_\ell x_{\ell+1}\ldots \cdot B^d) := \begin{cases} \pm 0.x_1x_2\ldots x_\ell \cdot B^d & \text{falls} \quad x_{\ell+1} < B/2 \\ \pm(0.x_1x_2\ldots x_\ell + B^{-\ell}) \cdot B^d & \text{falls} \quad x_{\ell+1} \geq B/2 \end{cases} \tag{6.1.3}$$

bei Dezimalzahlen etwa wird ab 5 aufwärts gerundet. Bei dieser Rundung kann für den *relativen Fehler* folgendes gezeigt werden.

Satz 6.1.3 *Für $x \neq 0$ gilt mit (6.1.3)*

$$\left|\frac{rd(x)-x}{x}\right| \leq \frac{1}{2}B^{1-\ell} =: eps \qquad \Longleftrightarrow \qquad rd(x) = x(1+\epsilon),\ |\epsilon| \leq eps.$$

Die Größe eps heißt *Maschinengenauigkeit*, bei $B = 2$ ist $eps = 2^{-\ell}$.

Beweis In (6.1.3) ist $|x| \geq B^{d-1}$. In beiden Fällen gilt dort

$$|rd(x)-x| \leq \frac{B}{2}B^{-\ell-1}B^d \leq \frac{1}{2}B^{1-\ell}B^{d-1} \leq |x| \cdot eps. \qquad \blacksquare$$

Dies bedeutet, daß die Funktion rd aus (6.1.3) jeweils auf die nächstgelegene Maschinenzahl abbildet. Da das Ergebnis der Verknüpfung von zwei Maschinenzahlen i.a. nicht wieder eine solche ist, werden bei jeder Verknüpfung neue Rundungsfehler erzeugt. Daher müssen Maschinenoperationen von reellen Verknüpfungen unterschieden werden:

Definition 6.1.4 *Für $a, b \in M$ und $* \in \{+, -, \times, /\}$ bezeichne (mit $b \neq 0$ im Fall der Division)*

$$a \,\tilde{*}\, b = gl(a * b)$$

das Ergebnis der entsprechenden (geräteabhängigen) Gleitpunktoperation.

Bei der Konstruktion eines Rechners (Hardware) ist anzustreben, daß der Fehler bei diesen Operationen den zu erwartenden Rundungsfehler nicht übersteigt. Dies kann durch Verwendung eines Hilfsregisters ('Akkumulator') mit doppelter Stellenzahl 2ℓ garantiert werden. Dort wird die Operation mit 2ℓ Stellen ausgeführt und dann auf ℓ Stellen gerundet.

Satz 6.1.5 *Für die Gleitpunktoperationen zu $* \in \{+, -, \times, /\}$ auf einem ℓ-ziffrigen Rechner mit doppeltlangem Akkumulator (2ℓ Stellen) gilt für $a, b \in M$ ($b \neq 0$ bei Division)*

$$a \,\tilde{*}\, b = (a * b)(1+\delta), \quad |\delta| \leq eps = \frac{1}{2}B^{1-\ell}. \tag{6.1.4}$$

Erläuterung: Bei Addition von $a = \pm 0.\alpha_1 \ldots \alpha_\ell B^c$, $b = \pm 0.\beta_1 \ldots \beta_\ell B^d$, (oBdA $c \geq d$) ist als erstes eine Exponentenanpassung durchzuführen:

$$\begin{array}{ll} \boxed{\pm 0.\alpha_1\alpha_2\ldots\alpha_\ell} & \\ \xleftrightarrow[c-d]{} & \boxed{\pm 0.\beta_1\beta_2\ldots\beta_\ell} \end{array}$$

Man sieht leicht, daß für $c-d \leq \ell$ das Ergebnis exakt mit 2ℓ Stellen darstellbar ist, für $c-d > \ell$ andererseits gilt $gl(a+b) = a$. Das Ergebnis der Multiplikation ist ebenfalls mit 2ℓ Stellen exakt darstellbar.

Die wichtigste Konsequenz der Rundung ist der Verlust der *Assoziativität* bei Gleitpunktoperationen. Eine wichtige Aufgabe der Fehleranalyse ist daher die Identifikation *günstiger* Anordnungen bei mehrfachen Verknüpfungen.

Beispiel 6.1.6 $\ell = 3,\ a = 0.721,\ b = 0.457_{10}2,\ c = -0.456_{10}2$. Es ist

$$\begin{aligned} (a\tilde{+}b)\tilde{+}c = \quad & (0.464_{10}2)\tilde{+}(-0.456_{10}2) && = 0.800_{10}0, \\ a\tilde{+}(b\tilde{+}c) = \quad & 0.721\tilde{+}(0.1) && = 0.821_{10}0. \end{aligned}$$

Beide Ergebnisse sind offensichtlich verschieden.

Der in diesem Beispiel bei der ersten Version beobachtete Effekt wird *Auslöschung* genannt: Besitzen zwei Zahlen $a, b \in M$ gleiche Exponenten und gleiche führende Ziffern, so ist bei Subtraktion das Ergebnis *exakt*! Sind in den Operanden aber *alte* Fehler vorhanden, werden diese sehr verstärkt, da dann $|a-b| \ll |a|+|b|$ ist in (6.1.4). Dies muß bei der Fehlerfortpflanzung besonders beachtet werden.

Zunächst wird die Bildung mehrfacher Produkte und Summen betrachtet, also

$$\widetilde{w_n} := gl(a_1 * a_2 * \ldots * a_n), \quad n \in \mathbf{N}, \tag{6.1.5}$$

mit $* \in \{+, \times\}$. Das Ergebnis $\widetilde{w_n}$ werde dabei rekursiv berechnet mit gerundeten Werten $\widetilde{a_j}$,

$$\widetilde{w_n} := gl(\ldots gl(gl(\widetilde{a_1} * \widetilde{a_2}) * \widetilde{a_3}) \ldots) = gl(\widetilde{w_{n-1}} * \widetilde{a_n}), \tag{6.1.6}$$

wobei $\widetilde{a_i} = a_i(1+\delta_i)$, $|\delta_i| \leq eps$. Für die weiteren relativen Fehler gelte ebenfalls $|\epsilon_i| \leq eps$.

Produkt: Hier gilt

$$\begin{aligned} \widetilde{w_2} &= gl(\widetilde{a_1} \cdot \widetilde{a_2}) = gl(a_1(1+\delta_1)a_2(1+\delta_2)) = a_1a_2(1+\delta_1)(1+\delta_2)(1+\epsilon_2) \\ \ldots & \\ \widetilde{w_n} &= gl(\widetilde{w_{n-1}} \cdot \widetilde{a_n}) = \widetilde{w_{n-1}} \cdot a_n(1+\delta_n)(1+\epsilon_n), \end{aligned}$$

Also

$$\begin{array}{cc} \widetilde{w_n} = w_n(1+\gamma_n) & \text{mit} \\ (1-eps)^{2n-1} \leq 1+\gamma_n \leq (1+eps)^{2n-1}. & \end{array} \tag{6.1.7}$$

Ausdrücke dieser Form treten jetzt öfter auf. Nach dem Mittelwertsatz gilt $(1+x)^m = 1 + mx(1+\xi)^{m-1}$ mit $|\xi| \le |x|$. Daraus folgt

$$|(1+x)^m - 1| \le m|x|(1+|x|)^{m-1} \le m|x|e^{(m-1)|x|} \le 1.06m|x| \quad \text{für } (m-1)|x| \le 0.1\ . \quad (6.1.8)$$

Daher gilt in (6.1.7) für γ_n die Aussage

$$|\gamma_n| \le 1.06(2n-1)eps, \text{ falls } 2(n-1)eps \le 0.1\ . \quad (6.1.9)$$

Summe: Bei der Addition ergibt sich

$$\begin{aligned}
\widetilde{w_2} &= gl(\widetilde{a_1} + \widetilde{a_2}) = (a_1(1+\delta_1) + a_2(1+\delta_2))(1+\epsilon_2) \\
&= a_1(1+\delta_1)(1+\epsilon_2) + a_2(1+\delta_2)(1+\epsilon_2) \\
\ldots \\
\widetilde{w_n} &= gl(\widetilde{w_{n-1}} + \widetilde{a_n}) = (\widetilde{w_{n-1}} + a_n(1+\delta_n)(1+\epsilon_n) \\
&= [\widetilde{w_{n-2}} + a_{n-1}(1+\delta_{n-1})](1+\epsilon_{n-1})(1+\epsilon_n) + a_n(1+\delta_n))(1+\epsilon_n) = \ldots \\
&= a_1(1+\sigma_1) + a_2(1+\sigma_2) + \ldots + a_n(1+\sigma_n),
\end{aligned}$$

mit $(1+\sigma_i) = (1+\delta_i)(1+\epsilon_i)\cdots(1+\epsilon_n)$. Aus (6.1.8) folgt daher

$$|\sigma_i| \le 1.06(n-i+2)eps, \quad \text{wenn } (n-i+1)eps \le 0.1\ . \quad (6.1.10)$$

Bemerkung: Bei der Produktbildung ist in der Fehlerschranke keine Bevorzugung einer bestimmten Anordnung der Operanden erkennbar. Bei der Summe dagegen liefern die ersten Summanden den größten relativen Fehlerbeitrag. Daher ist zu erwarten, daß der absolute Fehler in $\widetilde{w_n}$ kleiner wird, wenn zuerst die *betragskleinsten* Koeffizienten a_i summiert werden, da dann die Schranke

$$|\widetilde{w_n} - w_n| \le 1.06eps\ \sum_{i=1}^{n}(n-i+2)|a_i|$$

minimal wird. Daher sollte man, z.B., bei der Summation (monoton) konvergenter Reihen mit den letzten, d.h. kleinsten, Koeffizienten beginnen ('Rückwärts-Summation').

Ungünstige arithmetische Ausdrücke: Aufgrund der Rundungsfehler geht bei Gleitpunktrechnung nicht nur die Assoziativität von Summe und Produkt verloren, auch andere Gesetze gelten im allgemeinen nicht mehr (exakt). Daher können verschiedene, im Reellen äquivalente Ausdrücke, wie z.B. die der binomischen Formel $a^2 - b^2 = (a+b)(a-b)$, sehr unterschiedliche numerische Ergebnisse liefern. Dazu werde die Funktion

$$f(x) := \sqrt{1+x^2} - x \quad (6.1.11)$$

betrachtet. Für $x \ge 0$ ist $0 \le f(x) \le 1$. Da $|f'(x)| = |\frac{x}{\sqrt{1+x^2}} - 1| \le 1$ ist für $x \ge 0$, führt eine Störung im Argument x bei exakter Rechnung nur zu einer geringen Verfälschung im Funktionswert f, die nicht größer als die Abweichung in x selbst ist. Werden allerdings die Teilschritte in

(6.1.11) nur mit endlicher Genauigkeit ausgeführt, sieht das Ergebnis anders aus. Selbst wenn nur bei der Addition in (6.1.11) ein relativer Fehler ϵ, $|\epsilon| \le eps$, erzeugt wird, gilt

$$\tilde{f}(x) = \sqrt{1 \tilde{+} x^2} - x = \sqrt{(1+x^2)(1+\epsilon)} - x = f(x) + \frac{\epsilon}{2}\sqrt{1+x^2} + \mathcal{O}(\epsilon^2).$$

Für große $x \to \infty$ ist also ein großer absoluter Fehler $\cong \frac{1}{2}x\epsilon$ zu erwarte, wegen $|f| \le 1$ ist der relative Fehler noch viel größer. In der Gestalt (6.1.11) tritt nämlich bei der Differenzbildung der beiden großen Ausdrücke $\sqrt{1+x^2}$ und x eine *Auslöschung* führender Stellen auf. Diese kann durch Umformung in die äquivalente Form

$$f(x) = \sqrt{1+x^2} - x = \frac{1+x^2-x^2}{\sqrt{1+x^2}+x} = \frac{1}{x+\sqrt{1+x^2}} \tag{6.1.12}$$

vermieden werden.

Beispiel 6.1.7 Formel (6.1.11) wird mit 4-ziffriger Dezimalrechnung in $x = 9.999$ ausgewertet, bei exakter Rechnung ist $f(9.999) = 0.04988\ldots$ Dagegen ist $x \tilde{*} x = 99.98$, $1 \tilde{+} 99.98 = 101.0$, $\widetilde{\sqrt{101}} = 10.05$ und daher $\tilde{f}(x) = 0.05100$. Dies entspricht einem relativen Fehler von $0.022 \cong 450 \cdot eps$. Die letzte, äquivalente Form in (6.1.12) ergibt dagegen den korrekten 4-stelligen Wert 0.04988. Für $x \ge 100$ schließlich liefert 4-stellige Rechnung in (6.1.11) immer den Wert $\tilde{f}(x) = 0$.

6.2 Rundungsfehleranalyse einiger Algorithmen

Für einige der früher behandelten Verfahren werden nun Konsequenzen aus dem Auftreten von Rundungsfehlern behandelt. Bei numerischen Verfahren, die selbst nur Näherungen mathematischer Lösungen liefern, ist die wichtigste die, daß zum Approximationsfehler des (abstrakten) Verfahrens bei Durchführung auf einem Rechner noch Rundungsfehler hinzukommen, im *Gesamtfehler* des realen Verfahrens also beide zu berücksichtigen sind.

Approximationsverfahren

Bei Interpolation, Quadratur und Differentiation konnte die Güte der Approximation durch Vergrößerung eines Parameters n (Grad, Knotenzahl,..) verbessert werden. Für größere Werte von n treten dabei allerdings i.a. vermehrt Rundungsfehler auf.

Beispiel 6.2.1 Es soll $s := \sum_{k=1}^{\infty} a_k$ durch $s_n := \sum_{k=1}^{n} a_k$ angenähert werden, dabei sei $|a_k| \le ck^{-m-1}$ $\forall k$. Dann gilt für den Approximationsfehler

$$|s_n - s| \le \sum_{k=n+1}^{\infty} |a_k| \le c \sum_{k=n+1}^{\infty} k^{-m-1} \le c \int_n^{\infty} x^{-m-1} dx = \frac{c}{mn^m}.$$

Statt s_n wird aber berechnet $\widetilde{s_n}$ mit $|\widetilde{s_n} - s_n| \le 1.06 eps(n+1)c(1+\frac{1}{m})$, vgl. (6.1.10). Also ist

$$|\widetilde{s_n} - s| \le \frac{c_0}{n^m} + c_1 n \cdot eps.$$

Beispiel 6.2.2 Approximation von $f'(0)$ durch den symmetrischen Differenzenquotient $D_n = \frac{n}{2}[f(\frac{1}{n}) - f(-\frac{1}{n})]$ mit Schrittweite $h = \frac{1}{n}$. Nach (3.6.4) gilt

$$|\widetilde{D_n} - f'(0)| \le \frac{c_0}{n^2} + c_1 n \cdot eps.$$

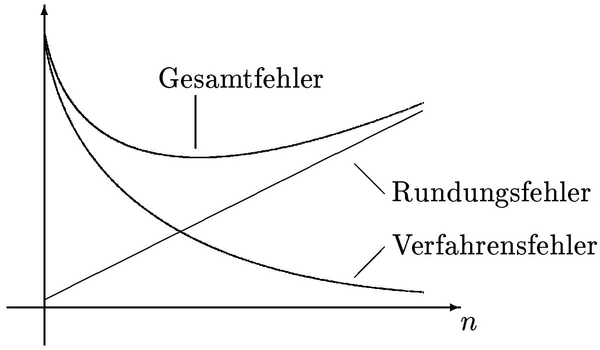
Beispiel 6.2.3 Zur Approximation von $I := \int_a^b f(x)dx$ kann die iterierte Trapezregel mit $h = (b-a)/n, x_j := a + jh$, dienen,

$$T_h = h\Big[\frac{1}{2}f(a) + \sum_{j=1}^{n-1} f(x_j) + \frac{1}{2}f(b)\Big].$$

Für die tatsächlich berechneten Funktionswerte $\tilde{f_j}$ gelte $\tilde{f_j} = f(x_j)(1+\delta_j)$ mit $|\delta_j| \le \delta \cdot eps$. Nach (6.1.10) gilt für den numerisch berechneten Wert $\widetilde{T_h}$ eine Abschätzung $|\widetilde{T_h} - T_h| = \frac{b-a}{n}|\frac{1}{2}f_0\sigma_0 + \sum_j f_j\sigma_j + \frac{1}{2}f_n\sigma_n| \le c(b-a)\|f\|_\infty n \cdot eps$. Mit (3.2.6) führt dies auf die Abschätzung für den Gesamtfehler

$$|\widetilde{T_h} - I| \le |T_h - I| + |\widetilde{T_h} - T_h| \le c_0 h^2 + c_1 n \cdot eps.$$

In all diesen Fällen verhält sich der Fehler so, wie in der Graphik gezeigt: Dem für $n \to \infty$ kleiner werdenden Verfahrensfehler wirkt ein wachsender Rundungsfehler entgegen, da sich beide i. a. addieren. Die mit diesen realen Verfahren überhaupt erreichbare Genauigkeit ist daher begrenzt.



Als Beispiel werde ein Verfahren der Ordnung m mit linear anwachsenden Rundungsfehlern betrachtet. Für den Gesamtfehler sei also eine Schranke F_n der Form

$$F_n = \frac{c_0}{n^m} + c_1 eps \cdot n$$

bekannt. Die Ableitung dieses Ausdrucks nach n ist $\partial F_n/\partial n = -mc_0/n^{m+1} + c_1 eps$ und führt auf die Minimalstelle

$$n_{\min} \cong \Big(\frac{c_0 m}{c_1 eps}\Big)^{1/(m+1)} \quad \Rightarrow \quad F_{\min} \ge c_2 eps^{m/(m+1)}.$$

Dies bedeutet, daß auf einem Rechner mit ℓ Stellen bei einem Verfahren der Ordnung m die Genauigkeit auf ca. $\frac{m}{m+1}\ell$ Stellen beschränkt ist. Daher können nur Verfahren hoher Ordnung Genauigkeiten im Bereich der vollen Maschinengenauigkeit erreichen.

Gauß-Algorithmus

Da der (reelle) Gauß-Algorithmus die exakte Lösung eines LGS liefert, treten kein Verfahrens- sondern nur Rundungsfehler auf. Allgemein kann ein ungenaues Ergebnis bei der numerischen Lösung eines bestimmten Problems (mindestens) zwei Ursachen haben.

1. Das *mathematische* Problem reagiert empfindlich auf Störungen der Eingangsdaten. Bei linearen Gleichungssystemen ist dies bei einer großen Konditionszahl $\kappa \gg 1$ der Fall, vgl. (4.4.7). In diesem Fall kann natürlich nicht erwartet werden, daß der *numerische* Algorithmus gute Ergebnisse liefert.

2. Das mathematische Problem ist unempfindlich gegen Störungen, der numerische Algorithmus erzeugt große Fehler.

Um hier die 'Verantwortlichkeiten' klar zu machen, beschränkt man sich meist auf eine sogenannte *Rückwärtsanalyse* der Algorithmen. Dem entspricht folgende Begriffsbildung.

> *Ein Algorithmus heißt gut konditioniert, wenn er das* exakte *Ergebnis eines wenig gestörten Ausgangsproblem liefert.*

Dieser Begriff schließt also nur den zweiten Fall aus, im ersten Fall sind große Fehler im Ergebnis auf die schlechten Eigenschaften des mathematischen Problems zurückzuführen.

Beim Gauß-Algorithmus ist in diesem Sinne der Nachweis zu führen, daß er bei Anwendung auf ein $n \times n$-Gleichungssystem $Ax = b$ die exakte Lösung eines gestörten Problems

$$(A + \delta A)\widetilde{x} = b + \delta b, \quad \text{mit } \|\delta A\| = c \cdot eps \tag{6.2.1}$$

berechnet, wobei die Konstante c nicht zu groß ist. Konkret läßt sich folgendes beweisen.

Satz 6.2.4 *Der Gauß-Algorithmus (ohne Pivotisierung) liefert beim System $Ax = b$ eine Lösung des gestörten Systems (6.2.1), wobei für δA elementweise gilt*

$$|\delta A| \le eps \cdot n(3|A| + 5|\widetilde{L}||\widetilde{R}|) + O(eps^2). \tag{6.2.2}$$

Dabei sind $\widetilde{L}, \widetilde{R}$ die tatsächlich berechneten LR-Faktoren von A.

Als wichtigster Anteil ist das Betragsprodukt $|\widetilde{L}||\widetilde{R}|$ zu erkennen, dessen Elemente die von $|A|$ möglicherweise stark überschreiten können. Dies ist besonders dann der Fall, wenn kleine Pivotelemente auftreten, da $l_{ij} = a_{ij}^{(j)}/a_{jj}^{(j)}$, vgl. (4.3.4). Der Satz ist aber auch bei Pivotisierung anwendbar, da der Gauß-Algorithmus bei Spaltenpivotisierung die LR-Zerlegung eines Systems $(PA)x = Pb$ bestimmt, wobei P eine Permutationsmatrix ist. Bei dieser Pivotisierungsstrategie gilt $|\widetilde{l_{ij}}| \le 1$, d.h. $\|\widetilde{L}\|_\infty \le n$ für den berechneten L-Faktor in $LR = PA$, vgl. Beispiel 4.4.5.

Trotz dieser günstigen Aussagen können große Fehler in einer Lösung auftreten, wenn die Konditionszahl der Matrix groß ist, $\kappa(A) \gg 1$. Dann ist aber, wie erläutert, schon das Ausgangsproblem fehleranfällig.

Fixpunkt-Iteration

Die zentrale Eigenschaft kontraktiver Iterationsverfahren ist die Tatsache, daß der Anfangsfehler in jedem Schritt um einen festen Faktor verkleinert wird. Dies trifft dann natürlich auch auf Fehler zu, die erst im Verlauf der Rechnung eingeschleppt wurden. Daher können solche Verfahren den Fixpunkt im wesentlichen mit derjenigen Genauigkeit bestimmen, die bei der Auswertung der Funktion g erreicht wird.

Satz 6.2.5 *Die Funktion g erfülle die Voraussetzungen (4.5.4),(4.5.5) des Banach'schen Fixpunktsatzes, z sei der Fixpunkt. Für die tatsächlich berechnete Funktion $\widetilde{g}$ sei*

$$\|\widetilde{g}(x) - g(x)\| \leq \epsilon \ \forall x \in \Omega. \tag{6.2.3}$$

Dann gilt bei der gestörten Iteration

$$\widetilde{x}^{(k+1)} := \widetilde{g}(\widetilde{x}^{(k)}), \tag{6.2.4}$$

die a-posteriori-Schranke

$$\|\widetilde{x}^{(k+1)} - z\| \leq \frac{1}{1-q}\Big(q\|\widetilde{x}^{(k+1)} - \widetilde{x}^{(k)}\| + \epsilon\Big). \tag{6.2.5}$$

Beweis Mit der Aussage (4.5.7)

$$\|g(x) - z\| \leq \frac{q}{1-q}\|g(x) - x\|$$

aus dem Fixpunktsatz 4.5.1 folgt

$$\begin{aligned}
\|\widetilde{x}^{(k+1)} - z\| &= \|\widetilde{g}(\widetilde{x}^{(k)}) - g(\widetilde{x}^{(k)}) + g(\widetilde{x}^{(k)}) - z\| \\
&\leq \epsilon + \|g(\widetilde{x}^{(k)}) - z\| \\
&\leq \epsilon + \frac{q}{1-q}\|g(\widetilde{x}^{(k)}) - \widetilde{x}^{(k)}\| \\
&\leq (1 + \frac{q}{1-q})\epsilon + \frac{q}{1-q}\|\widetilde{g}(\widetilde{x}^{(k)}) - \widetilde{x}^{(k)}\|. \qquad \blacksquare
\end{aligned}$$

Der Satz zeigt, daß bei der Iteration der Gesamtfehler $\epsilon/(1-q)$ i.a. nicht unterschritten werden kann. Daher lohnt es sich auch nicht, den Iterationsfehler wesentlich unter diesen Wert zu drücken. Ein Abbruchkriterium für die gestörte Iteration (6.2.4), das bei beiden Fehleranteilen in (6.2.5) für gleiche Größenordnung sorgt, ist

$$q\|\widetilde{x}^{(k+1)} - \widetilde{x}^{(k)}\| \overset{!}{\leq} \epsilon.$$

Mit diesem Kriterium wird dann die bestmögliche Genauigkeit im wesentlichen erreicht.

Index

a-posteriori, 56, 81
a-priori, 56
äquidistant, 9, 22
Äquilibrierung, 51
adaptive Integration, 32
Akkumulator, 75
Anfangswertproblem, 4
Assoziativitat, 76
asymptotische Entwicklung, 34
Auslöschung, 76, 78

Banachscher Fixpunktsatz, 56, 81
baryzentrische Form, 8, 20
Bulirsch-Folge, 36

Cholesky-Zerlegung, 49
Clenshaw-Algorithmus, 18, 20

Defekt, 53
Differentiation, 24, 38
Differenzen
  -Formel, 39
  -Quotient, 11, 15, 79
  -Verfahren, 41
  dividierte, 11, 12
Drei-Term-Rekursion, 15
Dreieck
  -Matrix, 47, 54
  -Schema, 9, 11, 35
  -System, 46

Eigenwert, 48, 61
Einzelschritt-Verfahren, 59
Einzugsbereich, 63, 66
Elektrisches Netzwerk, 40
Euler-McLaurin-Summenformel, 34
Extrapolations-Verfahren, 34

Fehler
  -Abschätzung, 56, 59
  -Gleichverteilung, 32
  -Schätzung, 31, 32
  relativer, 52, 75
Fixpunkt, 56, 63, 81
  abstoßender, 63

Gauß
  -Algorithmus, 46, 80
    Rundungsfehler, 79
  -Quadratur, 30
  -Seidel-Iteration, 59
Gesamt-Fehler, 39, 78
Gesamtschritt-Verfahren, 59
Gewichtsfunktion, 15, 25, 31
Gleichungssystem
  linear, 40
  nichtlinear, 63, 70
  Störung, 52
Gleitpunktzahlen, 74

Heron-Verfahren, 64
Hölder-Normen, 42
Horner-Schema, 11

Innenprodukt, 15, 29
Input-Output-Analyse, 40, 51, 55
Integration, 24
Interpolations-
  Fehler, 14, 20, 22
  Polynom, 7, 29, 38, 67, 70
    Hermite-, 7
    Lagrange-, 7, 20, 24
    Newton-, 10, 20, 30, 38
  Problem
    Hermite-, 14
  Punkte, neue, 8, 11
Iteration, 56
  lineare, 57
  mehrstufig, 68

Rundungsfehler, 80

Jacobi-Iteration, 59
Jordan-Normalform, 44

Keller-Speicher, 33
Kirchhoff'sche Regeln, 40
Konditionszahl, 52, 80
Schranken, 54
kontrahierende Abbildung, 56
Konvergenz, 27
-Faktor, 57
-Ordnung, 68
effektive, 69
alternierend, 63
linear, 65
lokal, 66
monoton, 63, 66
quadratisch, 65, 66

Lebesgue-Funktion, 23
Liniensuche, 73
Lipschitz-Bedingung, 56, 63
LR-Zerlegung, 48, 49, 54, 58, 80

Maschinengenauigkeit, 39, 75
Matrix, 43
-Norm, induzierte, 43
Band-, 49
diagonaldominant, 49, 54, 59
diagonalisierbar, 44
dünnbesetzt, 50, 59
inverse, 51
positiv definit, 48, 60
unzerlegbar, 60
Minimierungsverfahren, 72
Mittelwertsatz, 25, 71

Nach-Iteration, 58
Neumann-Reihe, 45, 55, 72
Neville-Algorithmus, 12, 20, 35
Newton
-Raphson-Verfahren, 65
-Verfahren, 65, 69
gedämpftes, 73
vereinfachtes, 73
Newton-Cotes
-Formeln, 26
Gewichte, 27
Nullstelle, 63
mehrfache, 66

Ordnung, 36, 38
eines Iterationsverfahrens, 64

Pendel, 3
Pivotisierung, 46, 48
Spalten-, 50, 80
Strategien, 50
Polynom
-Orthogonalfamilien, 31
bestapproximierendes, 23
Legendre-, 31
Orthogonal-, 15, 29, 30
Tschebyscheff-, 16

Quadratur, 25
-Fehler, 25, 33
-Formel, 25
Gauß-, 29
Gewichte, 25, 30
iteriert, 27, 29
Ordnung, 25, 29

Randwertproblem, 4
Rechenaufwand, 7, 9, 11, 12, 18, 19, 28, 47, 49, 51, 59
Rechteckregel, 26
regula falsi, 67
Reihenentwicklungen, 31
Relaxation, 61
Parameter, 61
Über-, 61, 62
Residuum, 53

Restglied, 25, 29, 31
Richardson-Extrapolation, 35
Rodriguez-Formel, 31
Romberg-Folge, 36
Rückwärts
  -Analyse, 80
  -Summation, 77
Rundung, 75
Rundungsfehler, 39, 50, 52, 58, 75

Satz von
  Neumann, 45
  Rolle, 14, 31, 38
  Stein-Rosenberg, 60
  Weierstraß, 22
Sekanten-Verfahren, 67
Simpsonregel, 26, 30
  iteriert, 28
SOR-Verfahren, 62
Spektral
  -Norm, 43
  -Radius, 43, 57, 60
Stützstellen, optimale, 20

Taylor-Entwicklung, 15, 19, 32, 65
Trapezregel, 26, 30, 79
  iteriert, 28, 31, 34
Tridiagonalmatrix, 42, 49
Tschebyscheff
  -Polynome, 16
  -Punkte, 16, 21, 22
    gestreckte, 24

Umkehrfunktion, 70

Volkswirtschaft, 40

Zeilen-Vertauschung, 46, 50
Zerlegung, reguläre, 57

Numerik IIA

Endlichdimensionale Probleme

B. Schmitt

Wintersemester 1994/95

Inhaltsverzeichnis

1 Interpolation und Approximation von Funktionen **3**

1.1 Spline-Funktionen 3

Existenz des Interpolations-Splines 5

Konvergenz des Interpolations-Splines 7

Spline-Darstellungen, Beziér-, B-Splines 10

Lokale Spline-Approximationen 14

1.2 Trigonometrische Interpolation 16

Die schnelle Fourier-Transformation (FFT) 18

2 Ausgleichsrechnung **23**

2.1 Kleinste-Quadrate-Lösung 23

2.2 Die Verallgemeinerte Inverse 24

2.3 QR-Zerlegung und Berechnung von A^+ 27

Praktische Durchführung, Rechenaufwand 29

3 Lineare Optimierung **32**

3.1 Problemstellung, Normalform 32

3.2 Eigenschaften und Grundlagen 33

3.3 Das Simplex-Verfahren 34

Praktische Durchführung 36

Durchführung des Basiswechsels 37

Bestimmung einer Startecke 38

4 Eigenwertprobleme von Matrizen 40

4.1 Grundlagen, Matrix-Normalformen 40

4.2 Reduktion auf einfachere Gestalt 44

4.3 Das unsymmetrische Eigenwertproblem 45

Die Vektor-Iteration (nach von Mises) 46

Inverse Iteration (nach Wielandt) 48

Orthogonale Iteration 50

Das QR-Verfahren 51

4.4 Das symmetrische Eigenwertproblem 56

Bisektionsverfahren 60

Jacobi-Verfahren 63

4.5 Überblick der Eigenwert-Verfahren 65

4.6 Eigenwert- und Fehler-Schranken 66

5 Iterationsverfahren für große Matrizen 72

5.1 Polynomielle Beschleunigungsverfahren 72

5.2 Das Verfahren der Konjugierten Gradienten 78

5.3 Vorkonditionierung 85

5.4 Lanczos-Verfahren zur Eigenwert-Berechnung 89

6 Fortsetzungsverfahren für nichtlineare Probleme 92

6.1 Die numerische Verfolgung von Lösungskurven 92

Index 99

1 Interpolation und Approximation von Funktionen

1.1 Spline-Funktionen

Der Fehler bei der Interpolation einer Funktion $f \in C^{k+1}[a,b]$ durch ein Polynom p_k vom Grad k mit Stützstellen $\xi_0, \ldots, \xi_k \in [a,b]$ hat die Gestalt ($x \in [a,b]$)

$$|f(x) - p_k(x)| \leq |(x-\xi_0)\cdots(x-\xi_k)| \frac{\|f^{(k+1)}\|_\infty}{(k+1)!} \leq \frac{(b-a)^{k+1}}{(k+1)!} \|f^{(k+1)}\|_\infty. \tag{1.1.1}$$

Beispiele zeigen, daß für $k \to \infty$ *i.a. keine Konvergenz* $\|f - p_k\|_\infty \to 0$ garantiert werden kann. Dagegen folgt aus (1.1.1) trivialerweise

$$|f - p_k| \to 0 \text{ für } (b-a) \to 0,\ k \text{ fest.}$$

Ähnlich wie bei den iterierten Quadraturformeln kann bei Unterteilung des Gesamtintervalls $[a,b]$ durch ein Gitter

$$\Delta: \quad a = x_0 < x_1 < \ldots < x_n = b, \quad h_i := x_{i+1} - x_i,\ i = 0, \ldots, n-1,\ H := \max_{i=0}^{n-1} h_i, \tag{1.1.2}$$

mit günstigen Eigenschaften gerechnet werden, wenn die Funktion f in jedem Teilintervall $[x_i, x_{i+1}]$ durch ein Polynom *festen Grades* k approximiert wird.

Definition 1.1.1 *Zu einem Gitter Δ nach (1.1.2) und $m, k \in \mathbf{N}_0,\ 0 \leq m < k$, wird der Raum $S^m_{\Delta,k}$ aller* Spline-*Funktionen s bezeichnet, die*

a) lokal, in jedem Teilintervall $(x_i, x_{i+1}]$, Polynome vom Grad k sind, $s|_{(x_i,x_{i+1}]} \in \Pi_k$,

b) global m-mal stetig differenzierbar sind, $s \in C^m[a,b]$.

Hier wird konkret nur der Fall der *kubischen Splines* ($k = 3$) mit $m \in \{1,2\}$ behandelt, einige Aussagen werden aber allgemeiner formuliert.

Anwendungsgebiete von Splines:

— Approximation (nicht-analytischer) Funktionen

— Approximation von Meßdaten

— Kurven- und Flächendarstellung in der Computer-Graphik

— Lösung von gewöhnlichen und partiellen Differentialgleichungen (FEM, Finite-Elemente-Methode)

Von besonderer Bedeutung sind die sogenannten *natürlichen Splines* aus $S^{2m-2}_{\Delta,2m-1}$ wegen der folgenden *Minimaleigenschaft.*

Satz 1.1.2 *Mit einem Gitter Δ nach (1.1.2) und $f \in C^2[a,b]$ erfülle die Splinefunktion $s \in S^2_{\Delta,3}$ die Interpolationsbedingungen*

$$s(x_i) = f(x_i),\ i = 0, 1, \ldots, n, \tag{1.1.3}$$

sowie eine der Bedingungen

$$\begin{array}{ll} a) & s''(a) = s''(b) = 0, \\ b) & s'(a) = f'(a),\ s'(b) = f'(b). \end{array}$$

Dann gilt

$$\|f'' - s''\|_2^2 = \|f''\|_2^2 - \|s''\|_2^2 \geq 0, \tag{1.1.4}$$

wobei $\|g\|_2^2 := \int_a^b g(x)^2 dx$*. Der so definierte Spline ist eindeutig bestimmt.*

Beweis Mit dem L_2-Innenprodukt $(f,g)_2 := \int_a^b f(x)g(x)\,dx$ gilt

$$\begin{aligned} \|f'' - s''\|_2^2 - \|f''\|_2^2 + \|s''\|_2^2 &= -2(f'', s'')_2 + 2(s'', s'')_2 = 2(s'' - f'', s'')_2 \\ &= 2\int_a^b [s''(x) - f''(x)]s''(x)\,dx. \end{aligned}$$

Durch partielle Integration auf den einzelnen Teilintervallen $[x_i, x_{i+1}]$ ergibt sich hier

$$\begin{aligned} \int_{x_i}^{x_{i+1}} [s'' - f'']s''\,dx &= (s' - f')s''\Big|_{x_i}^{x_{i+1}} - \int_{x_i}^{x_{i+1}} (s' - f')s'''\,dx \\ &= (s' - f')s''\Big|_{x_i}^{x_{i+1}} - \underbrace{(s - f)}_{=0,\,(1.1.3)}\ s'''\Big|_{x_i}^{x_{i+1}} + \int_{x_i}^{x_{i+1}} (s - f)\ \underbrace{s^{(4)}}_{=0,\,s\in\Pi_3}\ dx \end{aligned}$$

Der verbleibende Beitrag $(s' - f')s''$ ist stetig, bei Summation über die Teilintervalle zeigt sich

$$\int_a^b [s'' - f'']s''\,dx = \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} \ldots = \Big(s'(x) - f'(x)\Big)s''(x)\Big|_a^b = 0.$$

Dabei verschwindet die Klammer, wenn Bedingung b) erfüllt ist, bzw. die zweite Ableitung bei Bedingung a).

Zum Nachweis der Eindeutigkeit seien $s, \bar{s}$ zwei Splines, die die Voraussetzungen erfüllen. Für beide gilt dann (1.1.4), also

$$0 \leq \|s''\|_2^2 - \|\bar{s}''\|_2^2 = \|s'' - \bar{s}''\|_2^2 = \|\bar{s}''\|_2^2 - \|s''\|_2^2 \leq 0.$$

Daher ist $s'' - \bar{s}'' \equiv 0$, also $s(x) - \bar{s}(x) = cx + d,\ c, d \in \mathbf{R}$. Da die Differenz wegen (1.1.3) aber in allen x_i verschwindet, gilt $s = \bar{s}$. ■

Bemerkung: Unter allen Funktionen, die die Werte $(x_i, f(x_i))$ interpolieren, hat der natürliche kubische Spline nach (1.1.4) minimale zweite Ableitung im quadratischen Mittel, bei b) etwa ist

$$\|s''\|_2 = \min\{\|g''\|_2 :\ g \in C^2[a,b], g(x_i) = f(x_i)\,\forall i,\ g''(a) = g''(b) = 0\}.$$

In der Mechanik entspricht $\|g''\|_2$ der Biegenergie eines elastischen Lineals, wie es früher im Schiffsbau eingesetzt wurde (*Strak-Latte*, engl. *Spline*).

Existenz des Interpolations-Splines

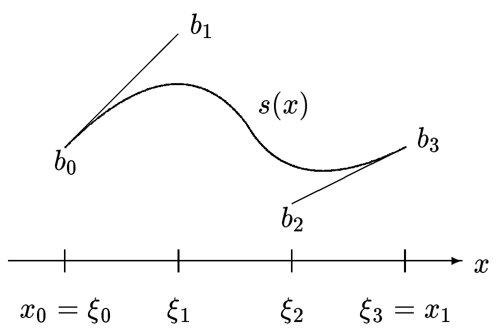
Für den Interpolationsspline $s \in S^2_{\Delta,3}$, der gegebene Werte y_i interpoliert, $s(x_i) = y_i$, $i = 0, \ldots, n$, wird in jedem Teilintervall eine Darstellung als Polynom benötigt. Dazu bietet sich die folgende *Beziér-Bernstein-Darstellung* an, die auf dem Standardintervall $[0, h]$, $h > 0$, lautet

$$s(x) = \frac{1}{h^k} \sum_{j=0}^{k} \binom{k}{j} x^j (h - x)^{k-j} b_j \tag{1.1.5}$$

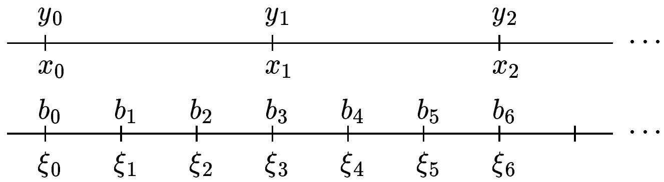
mit Koeffizienten $b_0, \ldots, b_k$. Diese Darstellung besitzt folgende Eigenschaften:

$$\left.\begin{array}{ll} (a) & b_j \equiv 1 \;\Rightarrow\; s(x) = \frac{1}{h^k} \sum_{j=0}^{k} \binom{k}{j} x^j (h-x)^{k-j} = \frac{1}{h^k}(x + h - x)^k \equiv 1 \\ (b) & s(0) = b_0, \quad s(h) = b_k \\ (c) & s'(x) = \frac{1}{h^k}[-k(h-x)^{k-1} b_0 + k(h - kx)(h-x)^{k-2} b_1 + x(\ldots)] \Rightarrow \\ & s'(0) = \dfrac{b_1 - b_0}{h/k}, \quad s'(h) = \dfrac{b_k - b_{k-1}}{h/k} \\ (d) & s''(0) = \frac{k(k-1)}{h^2}(b_0 - 2b_1 + b_2), \quad s''(h) = \frac{k(k-1)}{h^2}(b_{k-2} - 2b_{k-1} + b_k). \end{array}\right\} \tag{1.1.6}$$

Die Eigenschaft (c) führt bei $k = 3$ auf eine einfache *geometrische* Interpretation der Koeffizienten b_j. Werden die Paare $(\xi_j, b_j)^\top$ mit $\xi_j = jh/3$ betrachtet, dann ist die Gerade $\overline{(\xi_0, b_0)^\top\ (\xi_1, b_1)^\top}$ die Tangente an s in $x = \xi_0 = 0$ und $\overline{(\xi_2, b_2)^\top\ (\xi_3, b_3)^\top}$ die Tangente an s in $x = \xi_3 = h$. Durch b_0, b_3 werden also die Werte von s in den Randpunkten beeinflußt, durch b_1, b_2 die dortigen Ableitungen. Man bezeichnet die Koeffizienten oft als *Kontrollpunkte*.



Der Spline s wird auf dem Gesamtintervall $[a, b]$ durch lokale Darstellungen (1.1.5) zusammengesetzt. Dazu wird ein Zusatzgitter eingeführt,



$$\begin{array}{l} \xi_{ik+j} := x_i + \frac{j}{k} h_i, \\ j = 0, \ldots, k-1, \\ i = 0, \ldots, n-1, \end{array} \tag{1.1.7}$$

Die *Stetigkeitsbedingungen* an s im Punkt $x_j = \xi_{kj}$, $1 \le j \le n-1$, mit Polynomstücken (1.1.5) lauten für

C^0: erfüllt mit $b_{kj} = y_j$.

C^1: Nach (1.1.6,c) gilt

$$s'(x_j-) = \frac{k}{h_{j-1}}(b_{kj} - b_{kj-1}) \stackrel{!}{=} \frac{k}{h_j}(b_{kj+1} - b_{kj}) = s'(x_j+) \quad \Longleftrightarrow$$

$$\frac{1}{h_{j-1}}b_{kj-1}+\frac{1}{h_j}b_{kj+1}=(\frac{1}{h_{j-1}}+\frac{1}{h_j})b_{kj}=(\frac{1}{h_{j-1}}+\frac{1}{h_j})y_j. \tag{1.1.8}$$

2. Ableitung: Nach (1.1.6,d) ist $s''(x_j-)\stackrel{!}{=}s''(x_j+)$ äquivalent mit

$$\frac{1}{h_{j-1}^2}(b_{kj-2}-2b_{kj-1}+b_{kj})=\frac{1}{h_j^2}(b_{kj}-2b_{kj+1}+b_{kj+2})=:\mu_j \tag{1.1.9}$$

Im folgenden werden jetzt die Werte $\mu_j=\frac{1}{6}s''(x_j)$ aus (1.1.9) als eigentliche Parameter des Splines interpretiert, die Koeffizienten b lassen sich daraus bei $k=3$ berechnen. Die Beziehungen (1.1.9) an beiden Rändern von $[x_j,x_{j+1}]$,

$$\begin{aligned} 2b_{3j+1}-b_{3j+2} &= y_j-h_j^2\mu_j \\ -b_{3j+1}+2b_{3j+2} &= y_{j+1}-h_j^2\mu_{j+1} \end{aligned}$$

können nach b_{3j+1},b_{3j+2} aufgelöst werden und ergeben

$$\left.\begin{aligned} b_{3j+1} &= \tfrac{1}{3}(2y_j+y_{j+1})-\tfrac{1}{3}h_j^2(2\mu_j+\mu_{j+1}) \\ b_{3j+2} &= \tfrac{1}{3}(y_j+2y_{j+1})-\tfrac{1}{3}h_j^2(\mu_j+2\mu_{j+1}) \end{aligned}\right\}. \tag{1.1.10}$$

Diese Gleichungen führen auf folgende Formulierung der C^1-Bedingungen (1.1.8):

$$h_{j-1}\mu_{j-1}+2(h_{j-1}+h_j)\mu_j+h_j\mu_{j+1}=\frac{y_{j+1}-y_j}{h_j}-\frac{y_j-y_{j-1}}{h_{j-1}},\quad j=1,\ldots,n-1.$$

Bei Division durch $h_{j-1}+h_j=x_{j+1}-x_{j-1}$ vereinfacht sich insbesondere die rechte Seite (vgl. Numerik-I,(2.3.2)),

$$\underbrace{\frac{h_{j-1}}{h_{j-1}+h_j}}_{=:\,\alpha_j}\mu_{j-1}+2\mu_j+\underbrace{\frac{h_j}{h_{j-1}+h_j}}_{=:\,\beta_j}\mu_{j+1}=y[x_{j-1},x_j,x_{j+1}],\quad j=1,\ldots,n-1. \tag{1.1.11}$$

Zusätzlich sind noch die Randbedingungen aus Satz 1.1.2 zu berücksichtigen. Im Fall a), $s''(a)=s''(b)=0$ bedeuten diese einfach

$$\mu_0=\mu_n=0.$$

Nach Streichung dieser beiden Unbekannten ist (1.1.11) ein quadratisches Gleichungssystem. Im Fall b), $s'(a)=y_0'$, $s'(b)=y_n'$, ist nach (1.1.6,c) und (1.1.10) am linken Rand

$$y_0+\frac{h_0}{3}y_0'=b_1=\frac{1}{3}(2y_0+y_1)-\frac{h_0^2}{3}(2\mu_0+\mu_1).$$

Da rechts analoges gilt, führen diese Bedingungen b) also auf die zusätzlichen Gleichungen

$$\left.\begin{aligned} 2\mu_0+\mu_1 &= \tfrac{1}{h_0}\left(\tfrac{y_1-y_0}{h_0}-y_0'\right)=y[x_0,x_0,x_1] \\ \mu_{n-1}+2\mu_n &= y[x_{n-1},x_n,x_n] \end{aligned}\right\}. \tag{1.1.12}$$

In beiden Fällen ergeben sich die Momente $\mu=(\mu_j)_j$ als Lösung eines Gleichungssystems

$$A\mu=r$$

mit einer *Tridiagonalmatrix*

$$A = \underbrace{\begin{pmatrix} 2 & \beta_1 & & & \\ \alpha_2 & 2 & \beta_2 & & \\ & \alpha_3 & 2 & \beta_3 & \\ & & \ddots & \ddots & \ddots \end{pmatrix}}_{\in \mathbf{R}^{(n-1)\times(n-1)}} \quad \text{bzw.} \quad A = \underbrace{\begin{pmatrix} 2 & \beta_0 & & & \\ \alpha_1 & 2 & \beta_1 & & \\ & \alpha_2 & 2 & \beta_2 & \\ & & \ddots & \ddots & \ddots \end{pmatrix}}_{\in \mathbf{R}^{(n+1)\times(n+1)}}. \tag{1.1.13}$$

Lemma 1.1.3 *Für beide Matrizen A in (1.1.13) gilt*

$$\|A^{-1}\|_\infty \le 1.$$

Beweis Nach (1.1.11), (1.1.12) ist $\alpha_j, \beta_j \ge 0,\ \alpha_j + \beta_j = 1$, (mit $\alpha_0 := \beta_{n+1} := 0$). Diese Matrix ist diagonaldominant, daher gilt für jeden Vektor r und μ mit $A\mu = r$ und jeden Index j:

$$2|\mu_j| = |r_j - \alpha_j \mu_{j-1} - \beta_j \mu_{j+1}| \le |r_j| + \underbrace{(\alpha_j + \beta_j)}_{\le 1}\|\mu\|_\infty.$$

Betrachtet man auf der linken Seite den Index mit $|\mu_j| = \|\mu\|_\infty$, folgt

$$2\|\mu\|_\infty \le \|r\|_\infty + \|\mu\|_\infty \ \Rightarrow \|\mu\|_\infty = \|A^{-1}r\|_\infty \le \|r\|_\infty. \qquad \blacksquare$$

Da also insbesondere die Gleichungssysteme mit A immer lösbar sind, folgt

Satz 1.1.4 *Für beliebige Gitter Δ und beliebige Datenwerte y_j, y_0', y_n', existiert der eindeutig bestimmte Interpolationsspline $s \in S^2_{\Delta,3}$ (vgl. Satz 1.1.2) mit $s(x_j) = y_j,\ j = 0, \ldots, n$ sowie $s''(a) = s''(b) = 0$ oder $s'(a) = y_0', s'(b) = y_n'$.*

Mit Hilfe von Lemma 1.1.3 läßt sich auch einfach eine a-priori-Schranke für den Spline herleiten, bei einem äquidistantes Gitter ($h_j \equiv h$), etwa gilt

$$|s(x)| \le 3\,\|y\|_\infty \quad \Big(+\, h \max\{|y_0'|, |y_n'|\}\Big), \quad x \in [a,b].$$

Für das Interpolationspolynom existiert dagegen eine solche Schranke nicht mit einem für alle n beschränkten Vorfaktor.

Konvergenz des Interpolations-Splines

Wie nach der einleitenden Bemerkung zu erwarten, konvergiert der Interpolationsspline einer glatten Funktion gegen diese bei feiner werdendem Gitter ($H \to 0$). Überraschenderweise zeigt sich jedoch, daß dabei *keine weiteren* Voraussetzungen über das Gitter benötigt werden und daß sogar die ersten drei *Ableitungen* des Splines gegen die der Funktion konvergieren. Zum Nachweis der entsprechenden Aussage wird folgendes Hilfsmittel benötigt.

Lemma 1.1.5 *Die Funktion $g \in C^2[a,b]$ verschwinde in beiden Randpunkten, $g(a) = g(b) = 0$. Dann gilt*

$$\|g\|_\infty \le \frac{(b-a)^2}{8}\|g''\|_\infty, \quad \|g'\|_\infty \le (b-a)\|g''\|_\infty.$$

Beweis Taylorentwicklung um $x \in (a,b)$ ergibt

$$\begin{aligned} 0 = g(a) = \quad & g(x) + (a-x)g'(x) + \tfrac{1}{2}(a-x)^2 g''(\xi_1), \ \xi_1 \in (a,b), \\ 0 = g(b) = \quad & g(x) + (b-x)g'(x) + \tfrac{1}{2}(b-x)^2 g''(\xi_2), \ \xi_2 \in (a,b). \end{aligned}$$

Durch Multiplikation der Gleichungen mit $(b-x)$ bzw. $(x-a)$ und Addition wird $g'(x)$ eliminiert und man erhält

$$0 = (b-x+x-a)g(x) + \frac{1}{2}(b-x)(x-a)[(x-a)g''(\xi_1) + (b-x)g''(\xi_2)].$$

Daraus folgt dann die erste Ungleichung,

$$|g(x)| \le \frac{(b-x)(x-a)}{2(b-a)}(|x-a| + |b-x|)\|g''\|_\infty \le \frac{(b-a)^2}{8}\|g''\|_\infty.$$

Die zweite Ungleichung gilt, da nach dem Satz von Rolle ein $x_0 \in [a,b]$ existiert mit $g'(x_0) = 0$ und daher

$$|g'(x)| = |g'(x_0) + (x-x_0)g''(\xi)| = |x-x_0||g''(\xi)| \le (b-a)\|g''\|_\infty. \qquad \blacksquare$$

Die Konvergenz des Interpolationssplines nach Satz1.1.2,b wird im folgenden Satz beschrieben.

Satz 1.1.6 *Gegeben sei $f \in C^4[a,b]$ und ein Gitter Δ. Dann gelten für den Interpolationsspline $s \in S^2_{\Delta,3}$ mit*

$$s(x_j) = f(x_j), \ j = 0,\dots,n, \quad s'(a) = f'(a), \ s'(b) = f'(b),$$

und seine Ableitungen folgende Fehlerschranken. Für $x \in (x_j, x_{j+1}]$, $0 \le j < n$, ist

$$|s^{(i)}(x) - f^{(i)}(x)| \le c_i H^2 h_j^{2-i}\|f^{(4)}\|_\infty, \quad 0 \le i \le 3.$$

Beweis Es wird eine weitere Splinefunktion $\hat{s} \in S^0_{\Delta,3}$ eingeführt, die die Bedingungen

$$\hat{s}(x_j) = f(x_j), \ \hat{s}''(x_j) = f''(x_j), \quad j = 0,\dots,n,$$

erfüllt. Dann ist $\hat{b}_{3j} = b_{3j} = f(x_j) =: f_j$ und nach (1.1.10) gilt mit $\hat{\mu}_j := \frac{1}{6}\hat{s}''(x_j)$ die Beziehung

$$\left\{ \begin{array}{rcl} \hat{b}_{3j+1} & = & \frac{1}{3}(2f_j + f_{j+1}) - \frac{1}{3}h_j^2(2\hat{\mu}_j + \hat{\mu}_{j+1}) \\ \hat{b}_{3j+2} & = & \frac{1}{3}(f_j + 2f_{j+1}) - \frac{1}{3}h_j^2(\hat{\mu}_j + 2\hat{\mu}_{j+1}) \end{array} \right. .$$

Daraus folgt

$$|b_{3j+\nu} - \hat{b}_{3j+\nu}| \le h_j^2\|\mu - \hat{\mu}\|_\infty, \quad \nu = 1,2, \tag{1.1.14}$$

und aus der Beziérdarstellung (1.1.5) in $(x_j, x_{j+1}]$,

$$s(x) - \hat{s}(x) = \frac{3}{h_j^3} \sum_{\nu=1}^{2} (x - x_j)^\nu (x_{j+1} - x)^{3-\nu} (b_{3j+\nu} - \hat{b}_{3j+\nu}),$$

ergibt sich für die Ableitungen der Differenz die Schranke

$$\begin{aligned} |s^{(i)}(x) - \hat{s}^{(i)}(x)| &\leq \frac{3}{h_j^3} \sum_{\nu=1}^{2} \left| \frac{d^i}{dx^i} [(x - x_j)^\nu (x_{j+1} - x)^{3-\nu}] \right| \max_\nu |b_{3j+\nu} - \hat{b}_{3j+\nu}| \\ &\leq \frac{\tilde{c}_i}{h_j^i} \max_\nu |b_{3j+\nu} - \hat{b}_{3j+\nu}|, \end{aligned}$$

mit $\tilde{c}_0 = \frac{3}{4}$, $\tilde{c}_1 = 3$, $\tilde{c}_2 = 18$, $\tilde{c}_3 = 36$. Mit dieser Aussage und (1.1.14) wurde also bisher gezeigt

$$|s^{(i)}(x) - \hat{s}^{(i)}(x)| \leq \tilde{c}_i h_j^{2-i} \|\mu - \hat{\mu}\|_\infty, \; x \in (x_j, x_{j+1}]. \tag{1.1.15}$$

Die Momente von s werden aus dem Gleichungssystem $A\mu = r$, (1.1.11), (1.1.12), bestimmt. Daher gilt

$$\mu - \hat{\mu} = A^{-1}(r - A\hat{\mu}) \quad \Rightarrow \quad \|\mu - \hat{\mu}\|_\infty \leq \|r - A\hat{\mu}\|_\infty,$$

da $\|A^{-1}\|_\infty \leq 1$ nach Lemma 1.1.3. Die Komponenten r_j und $(A\hat{\mu})_j$ können durch Taylorentwicklung um x_j verglichen werden, für $1 \leq j < n$ ist

$$\begin{aligned} 6(A\hat{\mu})_j &= \frac{1}{h_{j-1} + h_j} \Big(h_{j-1} f''(x_{j-1}) + 2(h_{j-1} + h_j) f''(x_j) + h_j f''(x_{j+1}) \Big) \\ &= 3f''(x_j) + (h_j - h_{j-1}) f'''(x_j) + \frac{1}{2} \frac{1}{h_{j-1} + h_j} [h_{j-1}^3 f^{(4)}(\xi_1) + h_j^3 f^{(4)}(\xi_2)], \\ r_j &= f[x_{j-1}, x_j, x_{j+1}] = \frac{1}{h_{j-1} + h_j} \Big(\frac{1}{h_j}(f_{j+1} - f_j) - \frac{1}{h_{j-1}}(f_j - f_{j-1}) \Big) \\ &= \frac{1}{2} f''(x_j) + \frac{1}{6}(h_j - h_{j-1}) f'''(x_j) + \frac{1}{24} \frac{1}{h_{j-1} + h_j} [h_{j-1}^3 f^{(4)}(\xi_3) + h_j^3 f^{(4)}(\xi_4)], \end{aligned}$$

mit Zwischenstellen $\xi_\nu \in (x_{j-1}, x_{j+1})$. Für die gesuchte Differenz folgt daher die Schranke

$$|r_j - (A\hat{\mu})_j| \leq (\frac{1}{12} + \frac{1}{24}) \frac{h_{j-1}^3 + h_j^3}{h_{j-1} + h_j} \|f^{(4)}\|_\infty \leq \frac{H^2}{8} \|f^{(4)}\|_\infty,$$

die sich analog auch in den Fällen $j = 0, n$ nachweisen läßt. Nach Einsetzen dieser Ergebnisse in (1.1.15) zeigt sich für die Differenz der beiden Spline-Funktionen die Aussage

$$|s^{(i)}(x) - \hat{s}^{(i)}(x)| \leq \frac{1}{8} \tilde{c}_i H^2 h_j^{2-i} \|f^{(4)}\|_\infty, \; x \in (x_j, x_{j+1}].$$

Zum Nachweis der Behauptung ist nun nur noch zu zeigen, daß eine solche Abschätzung auch für die Differenz $\hat{s} - f$ gilt. Nach Konstruktion verschwinden $\hat{s} - f$ und $\hat{s}'' - f''$ in allen Gitterpunkten x_j, $j = 0, \ldots, n$. Daher besitzt in jedem Teilintervall $(x_j, x_{j+1}]$ die Funktion

$\hat{s} - f$	$\hat{s}' - f'$	$\hat{s}'' - f''$	$\hat{s}''' - f'''$
2 Nullstellen	1 Nullstelle	2 Nullstellen	1 Nullstelle

.

Bei der ersten und dritten Ableitung wurde dazu der Satz von Rolle angewendet. Daher folgen aus Lemma 1.1.5 rekursiv die Schranken

$$\begin{aligned}
&|\hat{s}''-f''|\le \tfrac{1}{8}h_j^2\|f^{(4)}\|_\infty, \quad |\hat{s}'''-f'''|\le h_j\|f^{(4)}\|_\infty \quad \Rightarrow\\
&|\hat{s}-f|\le \frac{1}{64}h_j^4\|f^{(4)}\|_\infty, \quad |\hat{s}'-f'|\le \tfrac{1}{8}h_j^3\|f^{(4)}\|_\infty.
\end{aligned}$$

Die beiden Schranken für die Differenzen in $s^{(i)}-f^{(i)}=s^{(i)}-\hat{s}^{(i)}+\hat{s}^{(i)}-f^{(i)}$ ergeben die Behauptung. ∎

Spline-Darstellungen, Beziér-, B-Splines

Zur Darstellung von Splines sind zwei Methoden verbreitet. Die erste, mit Beziér-Stücken (1.1.5) bietet sich an, wenn sowohl C^2- als auch C^1- oder C^0-Splines verwendet werden sollen. Allerdings benutzt diese Darstellung bei C^2-Splines unnötig viele Parameter. Dies wird bei Verwendung einer Basis von $S^2_{\Delta,3}$ aus B-Splines vermieden.

Die **Beziér-Darstellung** wird nur für äquidistante Gitter behandelt. Dies ist aber bei einer der wichtigsten Anwendungen, der Parameterdarstellung von Kurven $(s_1(x),s_2(x),\ldots)$ im $\mathbf{R}^2$ bzw. $\mathbf{R}^3$ keine starke Einschränkung. In diesem Fall, $h_j\equiv h$, gibt es für die verschiedenen Stetigkeitsbedingungen einfache geometrische Interpretationen.

- Die C^1-Bedingung (1.1.8) lautet $b_{3j}=\frac{1}{2}(b_{3j-1}+b_{3j+1})$. Da auch $\xi_{3j}=\frac{1}{2}(\xi_{3j-1}+\xi_{3j+1})$ gilt, ist also der $\mathbf{R}^2$-Punkt $(\xi_{3j},b_{3j})^\mathsf{T}$ Mittelpunkt der Verbindungsstrecke von $(\xi_{3j-1},b_{3j-1})^\mathsf{T}$ und $(\xi_{3j+1},b_{3j+1})^\mathsf{T}$.

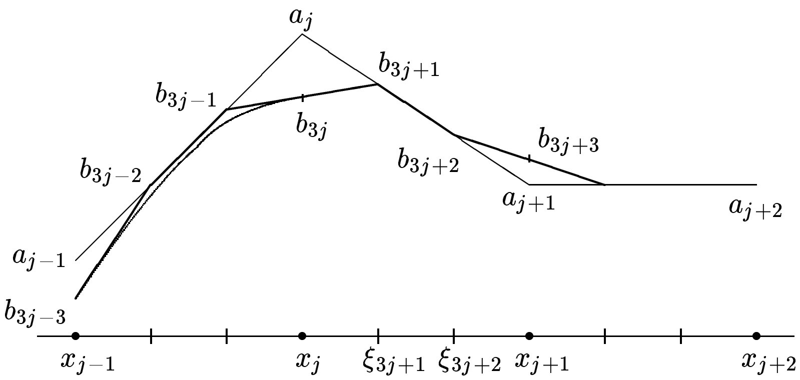
- Für die Ausdrücke in der C^2-Bedingung wird ein neuer Parameter eingeführt, damit wird aus (1.1.9):

$$\begin{aligned}
2b_{3j-1}-b_{3j-2} &= 2b_{3j+1}-b_{3j+2} =: a_j, &&\text{außerdem ist}\\
2\xi_{3j-1}-\xi_{3j-2} &= 2\xi_{3j+1}-\xi_{3j+2} = \xi_{3j}=x_j.
\end{aligned}$$

  Daher liegt der Punkt $(x_j,a_j)^\mathsf{T}$ im Schnittpunkt der Geraden durch $(\xi_\nu,b_\nu)^\mathsf{T}$ mit $\nu=3j-2,3j-1$ sowie $\nu=3j+1,3j+2$. Umgekehrt, dritteln die Punkte $(\xi_{3j+\nu},b_{3j+\nu})^\mathsf{T}$, $\nu=1,2$, die Strecke von $(x_j,a_j)^\mathsf{T}$ nach $(x_{j+1},a_{j+1})^\mathsf{T}$, denn

$$\left.\begin{aligned}2b_{3j+1}-b_{3j+2}&=a_j\\ -b_{3j+1}+2b_{3j+2}&=a_{j+1}\end{aligned}\right\} \quad\Rightarrow\quad \left\{\begin{aligned}b_{3j+1}&=\tfrac{1}{3}(2a_j+a_{j+1})\\ b_{3j+2}&=\tfrac{1}{3}(a_j+2a_{j+1})\end{aligned}\right. . \tag{1.1.16}$$

Diese Beziehungen der Beziér-Koeffizienten beim C^2-Spline erlauben folgende geometrische Interpretation (Hilfslinien), die analog auch für Kurven $(s_1(x),s_2(x))^\mathsf{T}$ gilt. Der Spline ist hier nur links von x_j skizziert:



Mit (1.1.16) können die C^1-Bedingungen umformuliert werden,

$$2b_{3j} = b_{3j-1} + b_{3j+1} = \frac{1}{3}(a_{j-1} + 4a_j + a_{j+1}).$$

Daher lassen sich die Koeffizienten a_j des C^2-Interpolationssplines aus dem Gleichungssystem

$$a_{j-1} + 4a_j + a_{j+1} = 6y_j, \quad j = 1, \ldots, n-1, \tag{1.1.17}$$

mit den Randbedingungen

$$\begin{array}{lll} a_0 = y_0, & a_n = y_n, & \text{bei } s''(a) = s''(b) = 0, \text{ bzw.} \\ 2a_0 + a_1 = 3y_0 + hy_0', & a_{n-1} + 2a_n = 3y_n - hy_n' & \text{bei } s'(a) = y_0', s'(b) = y_n', \end{array} \tag{1.1.18}$$

bestimmen. Die Beziér-Koeffizienten $\{b_\nu\}$ ergeben sich daraus nach (1.1.16).

Zur *Auswertung* eines Splines s in Beziér-Darstellung kann natürlich die Formel (1.1.5) direkt verwendet werden, nach (1.1.6a) ist sie eine konvexe Linearkombination und daher numerisch stabil. Wegen der auftretenden Binomialkoeffizienten läßt sich die Auswertung aber (analog zum Pascalschen Dreieck) auch durch mehrfache lineare Interpolation realisieren. Dazu sei für $x \in [x_j, x_{j+1}]$

$$s(x) = \sum_{i=0}^{k} \binom{k}{i} \xi^i (1-\xi)^{k-i} b_{kj+i}, \quad \xi := \frac{x - x_j}{h}.$$

Dann ergibt sich $s(x) = b_0^{(k)}$ aus dem Dreieckschema

$$\begin{array}{lcll} b_i^{(0)} & := & b_{kj+i}, & i = 0, \ldots, k; \\ b_i^{(\nu)} & := & (1-\xi) b_i^{(\nu-1)} + \xi b_{i+1}^{(\nu-1)}, & i = 0, \ldots, k-\nu;\ \nu = 1, \ldots, k. \end{array} \tag{1.1.19}$$

Auch (1.1.19) enthält nur konvexe Linearkombinationen.

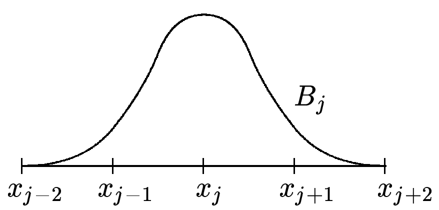
Beweis von (1.1.19): Die Identität $b_i^{(\nu)} = \sum_{\mu=0}^{\nu} \binom{\nu}{\mu} \xi^\mu (1-\xi)^{\nu-\mu} b_{i+\mu}^{(0)}$ läßt sich leicht durch Induktion über ν verifizieren. ■

Die wesentlichen Parameter eines Splines s sind die Größen a_j, aus denen sich die Splinewerte durch lineare Operationen (1.1.16), (1.1.19) berechnen. Daher existiert eine Basisdarstellung der Form

$$s(x) = \sum_{j=0}^{n} a_j B_j(x), \quad x \in [a, b], \tag{1.1.20}$$

wobei die Funktion $B_j,\ 0 \le j \le n$, dem Spline entspricht, den man für die speziellen Koeffizienten $(a_m)_m = (\delta_{jm})_m$ erhält. Dieser Spline hat im Innern des Intervalls die folgende Gestalt (vgl. die geometrische Interpretation)

- $a_j = 1$

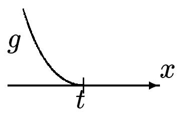


Der Träger von B_j für $2 \le j \le n-2$, ist $[x_{j-2}, x_{j+2}]$, umfaßt also $4 = k+1$ Teilintervalle. In Randnähe ergeben sich etwas unterschiedliche Formen.

Diese Basisfunktionen werden **B-Splines** genannt. Aus der Herleitung ist ersichtlich, daß diese eine *Zerlegung der Eins* bilden mit $B_j \ge 0,\ \sum_j B_j(x) \equiv 1$, wie dies bei den Bernsteinpolynomen, vgl. (1.1.6,a), schon der Fall war. Die B-Splines können aber auch vollkommen unabhängig davon und für beliebige Gitter Δ definiert werden. Die Bezeichnungen vereinfachen sich in Randnähe, wenn das Gitter um $2k = 6$ Punkte erweitert wird,

$$\overline{\Delta}: \quad x_{-3} < x_{-2} < x_{-1} < x_0 < \ldots < x_n < x_{n+1} < \ldots < x_{n+3}, \tag{1.1.21}$$

z.B. durch $x_{-1} := 2x_0 - x_{-1},\ x_{n+1} := 2x_n - x_{n-1}, \ldots$. Eine geschlossene Darstellung der B-Splines kann mit Hilfe der folgenden Funktion von zwei Argumenten angegeben werden.



$$g(t;x) := (t-x)_+^3 = \begin{cases} (t-x)^3 & \text{für} \quad t-x \ge 0, \\ 0 & \text{für} \quad t-x < 0. \end{cases} \tag{1.1.22}$$

Für festes t ist offensichtlich $g(t;\cdot) \in C^2(\mathbf{R})$.

Definition 1.1.7 *Zum Gitter* $\overline{\Delta}$ *werden die kubischen (normalisierten)* B-Splines *definiert durch die dividierte Differenz*

$$B_i(x) := (x_{i+2} - x_{i-2})\, g[x_{i-2}, x_{i-1}, \ldots, x_{i+2}; x], \quad i = -1, \ldots, n+1.$$

Die dividierten Differenzen (vgl. Numerik I,(2.3.2)) sind dabei bezüglich des ersten Arguments t von g zu bilden. Es gelten folgende Eigenschaften:

$$B_i(x) = 0 \quad \forall x \notin (x_{i-2}, x_{i+2}), \tag{1.1.23}$$

$$B_i(x) > 0 \quad \forall x \in (x_{i-2}, x_{i+2}), \tag{1.1.24}$$

$$\sum_{i=-1}^{n+1} B_i(x) = \sum_{i=j-1}^{j+2} B_i(x) \equiv 1, \text{ wenn } x \in (x_j, x_{j+1}],\ 0 \le j < n. \tag{1.1.25}$$

Beweis von (1.1.23): Für $x \ge x_{i+2}$ ist $g(x_j, x) = 0, j = i-2, .., i+2$. Für $x \le x_{i-2}$ ist $g[x_{i-2}, x_{i-1}, \ldots, x_{i+2}; x]$ vierte Differenz des kubischen Polynoms $(t-x)^3$ in t, also ebenfalls 0.

(1.1.24): Folgt aus dem nächsten Satz.

(1.1.25): Nach Definition ist $B_i(x) = g[x_{i-1},..,x_{i+2};x] - g[x_{i-2},..,x_{i+1};x]$. Nach (1.1.23) ist für $x \in (x_j, x_{j+1}]$:

$$\begin{aligned}\sum_i B_i(x) &= \sum_{i=j-1}^{j+2} B_i(x) \\ &= \underbrace{g[x_{j+1},..,x_{j+4};x]}_{1} - g[x_j,..,x_{j+3};x] + g[x_j,..,x_{j+3};x] + \ldots - \underbrace{g[x_{i-3},..,x_j;x]}_{0} = 1.\end{aligned}$$

Der Wert 1 ergibt sich dabei im ersten Fall aus Satz 2.3.6d (Numerik I), da $g(t;\cdot) = t^3 + \ldots$, und der Wert 0, da $g \equiv 0$ für die auftretenden Argumentwerte. Durch die Eigenschaften (1.1.23-1.1.25) ist umgekehrt der B-Spline B_i eindeutig festgelegt und stimmt daher mit dem aus (1.1.20) überein.

Zur numerischen Berechnung ist der Zugang über die Differenzen nicht sehr geeignet (Auslöschung). Ähnlich wie bei der Beziér-Darstellung kann aber auch für einen Spline in B-Spline-Darstellung ein stabiles Rekursionsschema angegeben werden.

Satz 1.1.8 *Die Splinefunktion $s \in S^2_{\Delta,3}$ sei dargestellt mit Hilfe der B-Splines zum erweiterten Gitter $\overline{\Delta}$,*

$$s(x) = \sum_{i=-1}^{n+1} a_i B_i(x).$$

Der Wert $s(x)$ mit $x \in (x_j, x_{j+1}]$, $0 \le j \le n$, ergibt sich aus der Rekursion

$$a_{i-2}^{(m)} = \frac{1}{x_{i-m} - x_{i-4}}\Big((x_{i-m} - x)a_{i-3}^{(m-1)} + (x - x_{i-4})a_{i-2}^{(m-1)}\Big), \quad \begin{cases} i = j+m+1, \ldots, j+4, \\ m = 1,2,3, \end{cases} \tag{1.1.26}$$

mit $a_i^{(0)} := a_i$, $i = 1, \ldots, n+1$. Es gilt

$$s(x) = a_{j+2}^{(3)} = \sum_{i=j-1}^{j+2} a_i B_i(x).$$

Mit Satz 1.1.8 kann ein C^2-Spline ohne Umweg über die Beziér-Form ausgewertet werden.

Bemerkung: a) Die Linearkombinationen in (1.1.26) sind konvex. Daher werden in den a_i evtl. vorhandene Fehler nicht verstärkt.

b) Am Funktionswert in $(x_j, x_{j+1}]$ sind nur beteiligt

$$\begin{array}{cccccc} & a_{j-1} & a_j & a_{j+1} & a_{j+2} & \\ x_{j-2} & x_{j-1} & x_j & x_{j+1} & x_{j+2} & x_{j+3}. \end{array}$$

Der Beweis des Satzes ist aufwendig und soll daher nicht vorgeführt werden. Ein wesentliches Hilfsmittel dabei ist die *Leibniz-Regel* für dividierte Differenzen,

$$(fg)[x_i, \ldots, x_{i+j}] = \sum_{\nu=0}^{j} f[x_i, \ldots, x_{i+\nu}]\, g[x_{i+\nu}, \ldots, x_{i+j}].$$

Sowohl B-Spline- als auch Beziér-Darstellung bieten eine Reihe von **praktischen Vorteilen**, die darauf beruhen, daß die verwendeten Basisfunktionen eine *lokale Zerlegung der Eins* bilden:

> Der Funktionswert $s(x)$ an einer beliebigen Stelle x im Intervall
> ist eine konvexe Linearkombination von $k+1=4$ Koeffizienten.

Konsequenzen:

— Bei Auswertung des Splines treten keine zusätzlichen Rundungsfehler auf.

— Die Koeffizienten schließen, als Punkte betrachtet, den Funktionsgraphen ein. Graphik-Algorithmen wie Skalierung, Clipping (Abschneiden bei Fenstern, Verdeckung) können eine Vorentscheidung anhand der Koeffizienten treffen.

— Bei Koordinatentransformationen (Zoom) sind alle Koeffizienten gleich zu behandeln.

— Änderungen einzelner Koeffizienten haben nur lokale Änderungen des Splines zur Folge ($\rightarrow$ interaktiver Entwurf).

Beispiel 1.1.9 Ausschnitt einer $\mathbf{R}^2$-Graphik aus kubischen Beziér-Splines mit Kontrollpunkten:

Hardcopy Micky-Maus, 64%

Lokale Spline-Approximationen

Um die Notwendigkeit zu umgehen, zur Bestimmung des Splines ein Gleichungssystem lösen zu müssen, und auch die Fernwirkung einzelner (großer) Funktionswerte zu unterbinden, kann man versuchen, Spline-Koeffizienten explizit anzugeben unter Verzicht auf die exakte Interpolationseigenschaft. Den einfachsten Fall behandelt

Satz 1.1.10 *Die Funktion f sei Lipschitz-stetig auf* $\mathbf{R}$ *mit der Lipschitz-Konstanten L. Mit den Eigenschaften (1.1.23)-(1.1.25) der Basis-Funktionen B_j gilt für*

$$s(x) := \sum_{j=-1}^{n+1} f(x_j)B_j(x)$$

die Konvergenzaussage

$$\|s-f\|_\infty \le 2HL.$$

Beweis Für $x \in (x_j, x_{j+1}]$ ist $1 = \sum_{i=j-1}^{j+2} B_i(x)$ und daher

$$|s(x)-f(x)| = |\sum_{i=j-1}^{j+2}(f(x_i)-f(x))B_i(x)| \le \max_{|x-y|\le 2H} |f(x)-f(y)| \le 2HL. \qquad \blacksquare$$

Allgemeiner werden Approximationen der Form

$$Qf := \sum_{i=-1}^{n+1} a_i(f)B_i \tag{1.1.27}$$

betrachtet für $f \in C[a-d, b+d]$ mit $d := \max\{|x_0 - x_{-3}|, |x_{n+3} - x_n|\}$. Diese heißen *lokale Spline-Approximation*, wenn $a_i(f)$ nur von Funktionswerten im Träger von B_i, also $[x_{i-2}, x_{i+2}]$, abhängt. Besonders interessant sind dabei lineare Funktionale a_i, z.B.,

$$a_i(f) = \sum_{j=1}^{\ell} \alpha_{ij} f(\xi_{ij}), \quad \xi_{ij} \in [x_{i-2}, x_{i+2}]. \tag{1.1.28}$$

Dann ist nämlich Q ein linearer Operator. Bei (1.1.28) gilt zunächst

$$|a_i(f)| \le \sum_{j=1}^{\ell} |\alpha_{ij}| \sup\{|f(x)| : \ x \in [x_{i-2}, x_{i+2}]\} =: A_i \|f\|_{[x_{i-2},x_{i+2}]}. \tag{1.1.29}$$

Daraus folgt eine *lokale* Abschätzung für Qf. Wenn $x \in [x_j, x_{j+1}]$ gilt, ist nämlich

$$|Qf(x)| \le \sum_{i=j-1}^{j+2} |a_i(f)| B_i(x) \le \max_{i=j-1}^{j+2} |a_i(f)| \le \max_{i=j-1}^{j+2} A_i \|f\|_{[x_{j-3},x_{j+4}]}. \tag{1.1.30}$$

Nach einem einfachen Prinzip kann mit dieser lokalen Schranke eine Konvergenzaussage abgeleitet werden.

Satz 1.1.11 *Zu einem erweiterten Gitter $\bar{\Delta}$ sei ein lokaler Approximationsoperator Q definiert, der (1.1.30) erfüllt. Wenn Q Polynome vom Grad m reproduziert, d.h., $Qp = p\,\forall p \in \Pi_m$, dann gilt für $f \in C^{m+1}[a-d, b+d]$ die Fehlerschranke*

$$\|f - Qf\|_{[a,b]} \le c\Big(1 + \max_{i=-1}^{n+1} A_i\Big) H^{m+1} \|f^{(m+1)}\|_{[a-d,b+d]}$$

Beweis Nach Voraussetzung gilt mit beliebigen Polynomen $p \in \Pi_m$

$$f - Qf = f - p - (Qf - p) = f - p - Q(f - p).$$

Mit (1.1.30) folgt daraus für $x \in (x_j, x_{j+1}]$, $0 \le j < n$, dann

$$|f(x) - Qf(x)| \le |f(x) - p(x)| + |Q(f - p)(x)| \le (1 + \max_{i=-1}^{n+1} A_i)\|f - p\|_{[x_{j-3},x_{j+4}]}.$$

Nun kann ein Polynom p so gewählt werden (z.B. durch Interpolation), daß

$$\|f - p\|_{[x_{j-3},x_{j+4}]} \le cH^{m+1}\|f^{(m+1)}\|_{[x_{j-3},x_{j+4}]}. \qquad \blacksquare$$

Die optimale Konvergenzaussage für kubische Splines, $\mathcal{O}(H^4)$, folgt aus diesem Satz also, wenn Q kubische Polynome reproduziert und die Konstante $\max_i A_i$ nicht vom Gitter $\bar{\Delta}$ abhängt.

Beispiel 1.1.12 (Schoenberg, H^2-Approximation):

$$a_i(f) := f(\bar{x}_i), \;\; \bar{x}_i := \frac{1}{3}(x_{i-1} + x_i + x_{i+1}),$$

bei beliebigem Gitter. Q reproduziert lineare Funktionen. Für äquidistantes Gitter stimmt Q mit der Approximation aus Satz 1.1.10 überein, da $\bar{x}_i = x_i$.

Beispiel 1.1.13 (H^4-Approximation) Die Gestalt der Funktionale (1.1.28) mit $\ell = 3$ bei allgemeinem Gitter ist sehr kompliziert, bei äquidistantem Gitter vereinfachen diese sich zu

$$a_i(f) := f(x_i) - \frac{h^2}{3} f[x_{i-1}, x_i, x_{i+1}],$$

(vgl. dazu (1.1.11)). Q reproduziert kubische Polynome.

Auch ohne Auflösung eines linearen Gleichungssystems (1.1.11) bzw. (1.1.17) kann also eine H^4-Approximation einer C^4-Funktion konstruiert werden, wenn auf die strenge Interpolationsforderung aus Satz 1.1.6 verzichtet wird.

1.2 Trigonometrische Interpolation

Zur Approximation periodischer Funktionen werden zweckmäßigerweise Polynome auf dem komplexen Einheitskreis bzw. direkt trigonometrische Polynome

$$p(x) := a_0 + a_1 e^{ix} + a_2 e^{2ix} + \ldots + a_{n-1} e^{(n-1)ix}, \;\; x \in [0, 2\pi), \tag{1.2.1}$$

betrachtet. Hier ist $i = \sqrt{-1}$ die imaginäre Einheit. Bei der Interpolation werden aus praktischen Gründen nur äquidistante Stützstellen $x_k = 2\pi\frac{k}{n}$, $k = 0, \ldots, n-1$, behandelt. Zu gegebenen Stützwerten $y_k \in \mathbf{C}$ ist das Interpolationspolynom p gesucht mit

$$p(x_k) = p\Big(2\pi\frac{k}{n}\Big) = y_k, \quad k = 0, 1, \ldots, n-1. \tag{1.2.2}$$

Wegen der 2π-Periodizität von p ist dann auch $p(2\pi k/n) = y_k \forall k \in \mathbf{Z}$, wenn definiert wird $y_{k+jn} := y_k \forall j \in \mathbf{Z}$. Nach Einführung der Größe $\omega_n := \exp(2\pi i/n)$ entspricht (1.2.2)

$$y_k = p(x_k) = \sum_{j=0}^{n-1} a_j e^{2\pi ijk/n} = \sum_{j=0}^{n-1} a_j\, \omega_n^{jk}, \quad k = 0, 1, \ldots, n-1. \tag{1.2.3}$$

Die Zahl ω_n ist eine *n-te primitive Einheitswurzel* mit $\omega_n^n = 1$. Aus dieser Eigenschaft folgen die Beziehungen

$$\begin{aligned} \omega_n^{k+jn} &= \omega_n^k \ \forall j \in \mathbf{Z} \\ \sum_{k=0}^{n-1} \omega_n^{jk}\omega_n^{-\ell k} &= n\delta_{j\ell} = \begin{cases} n & \text{für } j = \ell \\ 0 & \text{für } j \neq \ell \end{cases}. \end{aligned} \tag{1.2.4}$$

Beweis Die Nachweise für Periodizität und Summenwert bei $j = \ell$ sind trivial, für $j \neq \ell$ gilt

$$\sum_{k=0}^{n-1} \omega_n^{(j-\ell)k} = \frac{1 - \omega_n^{(j-l)n}}{1 - \omega_n^{j-\ell}} = 0, \quad \text{da } \omega_n^n = 1. \quad \blacksquare$$

Die Identität (1.2.4) besagt, daß die Vektoren $w^{(k)} := \left(\omega_n^{jk}\right)_{j=0}^{n-1} \in \mathbf{C}^n$ eine *Orthonormalbasis* bilden im Innenprodukt

$$(y, z) := \frac{1}{n}\sum_{j=0}^{n-1} y_j\, \bar{z}_j. \tag{1.2.5}$$

Daher läßt sich die Lösung des Interpolationsproblems (1.2.3) explizit angeben, denn dazu sind die Koeffizienten a_j in der Orthonormalentwicklung $y = \sum_{j=0}^{n-1} a_j w^{(j)}$ zu berechnen.

Satz 1.2.1 *Die Koeffizienten a_j des Interpolationspolynoms (1.2.1) mit (1.2.2) sind*

$$a_j = (y, w^{(j)}) = \frac{1}{n}\sum_{k=0}^{n-1} y_k\, \omega_n^{-jk}, \quad j = 0, 1, \ldots, n-1. \tag{1.2.6}$$

Beweis Da $\{w^{(j)}\}$ eine Orthonormalbasis ist, gilt $(y, w^{(j)}) = (\sum_{k=0}^{n-1} a_k w^{(k)}, w^{(j)}) = a_j$. $\blacksquare$

Bemerkung: Man beachte die Ähnlichkeit von (1.2.3) und (1.2.6).

Da bei der Interpolation nach einer Orthonormalbasis entwickelt wurde, sind Teilsummen $\sum_j a_j w^{(j)}$ auch jeweils Bestapproximierende an y:

Satz 1.2.2 *Für beliebiges $\ell \leq n-1$ minimiert das Abschnittspolynom*

$$p_\ell(x) := a_0 + a_1 e^{ix} + \ldots + a_\ell e^{i\ell x}$$

unter allen Polynomen q_ℓ vom Grad ℓ die Fehlerquadratsumme

$$\sum_{k=0}^{n-1} |y_k - q_\ell(x_k)|^2.$$

Beweis (Pythagoras) p_ℓ ist Orthogonalprojektion von p auf Π_ℓ bzgl. $(\cdot,\cdot)$. ■

Reelle Trigonometrische Interpolation:

Für reelle Daten $y_k \in \mathbf{R}$ kann das reelle trigonometrische Polynom

$$\varphi(x) := \frac{\alpha_0}{2} + \sum_{k=1}^{m-1} (\alpha_k \cos kx + \beta_k \sin kx) + \frac{\alpha_m}{2} \cos mx, \tag{1.2.7}$$

das die Werte y_k interpoliert, über (1.2.6) bestimmt werden.

Satz 1.2.3 *Die Koeffizienten a_j seien mit $n := 2m,\ y_k \in \mathbf{R}$, nach (1.2.6) definiert und*

$$\begin{aligned}
\alpha_j &:= \quad 2Re\ a_j = a_j + a_{n-j} &&= \frac{2}{n} \sum_{k=0}^{n-1} y_k \cos kx_j,\ j = 0, \ldots, m,\\
\beta_j &:= -2Im\ a_j = \tfrac{1}{i}(a_{n-j} - a_j) &&= \frac{2}{n} \sum_{k=0}^{n-1} y_k \sin kx_j,\ j = 1, \ldots, m-1.
\end{aligned}$$

Dann gilt für das trigonometrische Polynom (1.2.7) mit diesen Koeffizienten die Aussage

$$\varphi(x_k) = y_k \quad k = 0, \ldots, n-1.$$

Beweis Für reelle y_k ist

$$a_{n-j} = \frac{1}{n} \sum_{k=0}^{n-1} y_k \omega_n^{k(j-n)} = \frac{1}{n} \sum_{k=0}^{n-1} y_k \omega_n^{kj} = \bar{a}_j.$$

Damit folgt

$$\begin{aligned}
y_k &= p(x_k) = \sum_{j=0}^{n-1} a_j \omega^{jk} = \sum_{j=0}^{m-1} a_j \omega_n^{jk} + \sum_{\ell=1}^{m} a_{n-\ell} \omega_n^{kn-k\ell}\\
&= a_0 + \sum_{j=1}^{m-1} (a_j \omega_n^{jk} + a_{n-j} \omega_n^{-jk}) + a_m \omega^{mk}\\
&= a_0 + \sum_{j=1}^{m-1} 2\mathrm{Re}\,(a_j \omega_n^{jk}) + a_m (-1)^k = \varphi(x_k),
\end{aligned}$$

denn $2\mathrm{Re}\,(a_j \omega_n^{jk}) = \alpha_j \cos jx_k + \beta_j \sin jx_k$. ■

Die schnelle Fourier-Transformation (FFT)

Sowohl zur Bestimmung der Koeffizienten (1.2.6) als auch zur Auswertung des Polynoms (1.2.3) sind Summen der Form

$$z_k := \sum_{j=0}^{n-1} b_j \omega_n^{jk}, \quad k = 0, \ldots, n-1;\ \omega_n^n = 1,$$

zu berechnen (*diskrete Fourier-Transformation*). Im Normalfall werden dafür n^2 Operationen benötigt. Wenn n aber eine Zweierpotenz ist, kann der Aufwand auf $n \log_2 n$ Operationen gesenkt werden ($\rightarrow$ *schnelle* Fourier-Tr.) aufgrund der hohen Symmetrie der Koeffizienten ω_n^{jk}. Das Prinzip ist schon bei $n = 2m$ erkennbar, es gilt

$$\left.\begin{array}{rlll} z_{2k} = & \sum_{j=0}^{n-1} b_j \omega_{2m}^{2jk} = \sum_{j=0}^{m-1} (b_j + b_{j+m}) \omega_m^{jk} & =: \sum_{j=0}^{m-1} b_j^g \omega_m^{jk}, \\ & \text{da } \omega_{2m}^2 = \omega_m,\ \omega_m^{(j+m)k} = \omega_m^{jk}, & \\ z_{2k+1} = & \sum_{j=0}^{n-1} b_j \omega_{2m}^{2jk+j} = \sum_{j=0}^{m-1} [(b_j - b_{j+m}) \omega_n^j] \omega_m^{jk} & =: \sum_{j=0}^{m-1} b_j^u \omega_m^{jk}, \\ & \text{da } \omega_{2m}^2 = \omega_m,\ \omega_{2m}^m = -1. & \end{array}\right\} \qquad (1.2.8)$$

Durch den Übergang vom Gesamtvektor $(b_j)_{j=0}^{n-1}$ zu den beiden Teilvektoren $(b_j^g, b_j^u)_{j=0}^{m-1}$, der $2m$ Additionen und m Multiplikationen kostet, kann die Gesamt-Fourier-Transformation zu $n = 2m$ auf 2 Transformationen der halben Ordnung m zurückgeführt werden. Für den Aufwand A_n der Fourier-Transformation gilt also

$$A_{2m} = 2A_m + 3m, \quad A_1 = 0.$$

Für $n = 2^\ell$ ist die Methode (1.2.8) rekursiv anwendbar. Es läßt sich leicht nachrechnen, daß dann die Rekursionsformel durch $A_{2^\ell} = 3\ell 2^{\ell-1} = \frac{3}{2} n \log_2 n$ erfüllt wird, denn $A_{2^{\ell+1}} = 3(\ell+1)2^\ell = 2[3\ell 2^{\ell-1}] + 3\,2^\ell$. Also gilt

Satz 1.2.4 *Für $n = 2^\ell$ können die Summen (1.2.3) bzw. (1.2.6) mit $\frac{3}{2} n \log_2 n$ komplexen Operationen berechnet werden.*

Praktische Durchführung bei $n = 2^\ell$ [Cooley-Tuckey]:

Die Durchführung der ℓ Transformations-Stufen bei Anwendung von (1.2.8) kann ohne zusätzlichen Speicherbedarf, d.h., am Platz von (b_j), stattfinden. Dazu werden die Koeffizienten b_j^g, $j = 0, \ldots, m-1$, in die vordere Hälfte und die b_j^u, $j = 0, \ldots, m-1$, in die hintere Hälfte eingeordnet, die nächste Transformations-Stufe arbeitet getrennt auf beiden Hälften:

$$\left\{\begin{array}{rlll} b_j^{(1)} := & b_j^g = b_j + b_{j+m}, & j = 0, \ldots, m-1, & \text{Berechnung } \{z_{2j}\} \\ b_{j+m}^{(1)} := & b_j^u = (b_j - b_{j+m}) \omega_n^j, & j = 0, \ldots, m-1, & \text{Berechnung } \{z_{2j+1}\}. \end{array}\right. \qquad (1.2.9)$$

Zur korrekten Zuordnung werden die Indizes der Koeffizienten $b_j^{(1)}$ und die der Zielkoeffizienten z_k anhand ihrer Binärdarstellung (in den Kästchen) verglichen:

$$\begin{array}{ccccc} \text{Koeff. mit} & \left\{\begin{array}{c} k = 2j \\ k = 2j+1 \end{array}\right. & \text{kommt in Position} & \left\{\begin{array}{c} j \\ m+j \end{array}\right. \\ \text{Index binär:} & \boxed{s_{\ell-1}}\boxed{\cdots}\boxed{s_1}\boxed{s_0} & \longrightarrow & \boxed{s_0}\boxed{s_{\ell-1}}\boxed{\cdots}\boxed{s_1} \end{array}$$

Die weiteren Stufen lassen die höchste Binärstelle unverändert und ändern die restlichen analog. Daher hat sich nach $\ell = \log_2 n$ Schritten die Binärdarstellung des Index umgedreht:

$$z_k \text{ mit } k = \boxed{s_{\ell-1}}\boxed{\cdots}\boxed{s_1}\boxed{s_0} \text{ steht in } b_j^{(\ell)} \text{ mit } j = \boxed{s_0}\boxed{s_1}\boxed{\cdots}\boxed{s_{\ell-1}}.$$

Daher ist am Ende der FFT ein Permutationsschritt durchzuführen.

Beispiel 1.2.5 Schnelle Fourier-Transformation bei $n = 8 = 2^3$

	Index binär	1.Schritt	für Koeff	2.Schritt	für Koeff	3.Schritt	Index	Koeff
b_0	000	$b_0 + b_4$	xx0	$b_0 + b_2$	x00	$b_0 + b_1$	000	z_0
b_1	001	$b_1 + b_5$	xx0	$b_1 + b_3$	x00	$(b_0 - b_1)\omega^0$	100	z_4
b_2	010	$b_2 + b_6$	xx0	$(b_0 - b_2)\omega^0$	x10	$b_2 + b_3$	010	z_2
b_3	011	$b_3 + b_7$	xx0	$(b_1 - b_3)\omega^2$	x10	$(b_2 - b_3)\omega^0$	110	z_6
b_4	100	$(b_0 - b_4)\omega^0$	xx1	$b_4 + b_6$	x01	$b_4 + b_5$	001	z_1
b_5	101	$(b_1 - b_5)\omega^1$	xx1	$b_5 + b_7$	x01	$(b_4 - b_5)\omega^0$	101	z_5
b_6	110	$(b_2 - b_6)\omega^2$	xx1	$(b_4 - b_6)\omega^0$	x11	$b_6 + b_7$	011	z_3
b_7	111	$(b_3 - b_7)\omega^3$	xx1	$(b_5 - b_7)\omega^2$	x11	$(b_6 - b_7)\omega^0$	111	z_7

PASCAL-Programm zur schnellen Fourier-Transformation (die Unterprogramme cadd,csub,cmul realisieren die komplexen Operationen):

```
type complex = record  r,i: real  end;
      vektor = array[0..n_] of complex;
procedure wkoeff(n: integer; var om: vektor); {berechnet OM[J]=om^j}
var j,nh: integer; w: complex;
begin nh:=n div 2; w.r:=cos(pi/nh); w.i:=-sin(pi/nh);
  with om[0] do begin r:=1; i:=0 end;
  for j:=1 to nh-1 do cmul(om[j-1],w,om[j])
end; {wkoeff}
procedure FFT(n: integer; var z: vektor);
var i,j,jh,k,ia,id,ig,igr,ip,i1,i2,ng,ngr,L: integer;  zh: complex;
begin i:=n; L:=0;
  while i<>1 do begin L:=L+1; i:=i div 2 end; {N=2^L}
  id:=n div 2; ngr:=1;
  for ig:=1 to L do      { L=log2(N) Transformationen }
  begin ia:=0;
    for igr:=1 to ngr do { Transf. auf NGR=2^(IG-1) Teilfelder anwenden }
    begin ip:=0;
      for i:=0 to id-1 do { Laenge der Teilfelder: ID=2^(L-IG)}
      begin i1:=ia+i; i2:=i1+id;
        csub(z[i1],z[i2],zh);
        cadd(z[i1],z[i2],z[i1]);
        cmul(zh,om[ip],z[i2]);
        ip:=ip+ngr;
      end;{i}    ia:=ia+2*id;
    end;{igr}   id:=id div 2; ngr:=2*ngr;
```

```
  end;{ig}
  for j:=0 to n-1 do              {Index-Vertauschung:}
  begin jh:=j; k:=0;
    for i:=1 to L do { K:=bitinv(J) : }
    begin k:=k+k; if odd(jh) then k:=k+1; jh:=jh div 2 end;
    if k>j then   {vertausche einmal}
    begin zh:=z[j]; z[j]:=z[k]; z[k]:=zh end;
  end;{j}
end; {FFT}
```

Anwendungen der FFT:

a) Approximation von Koeffizienten der Fourier-Entwicklung

$$c_j = \frac{1}{2\pi}\int_0^{2\pi} f(x)e^{-ijx}dx,\ j \in \mathbf{Z}, \tag{1.2.10}$$

von 2π-periodische Funktionen f. Wenn f reell ist, gilt

$$\operatorname{Re} c_j = \frac{1}{2\pi}\int_0^{2\pi} f(x)\cos jx\,dx, \quad \operatorname{Im} c_j = -\frac{1}{2\pi}\int_0^{2\pi} f(x)\sin jx\,dx.$$

Mit den Werten $y_k = f(x_k) = f(2\pi k/n),\ k \in \mathbf{Z}$, in (1.2.6) gilt wegen der Periodizität

$$\begin{aligned} a_j &= \frac{1}{n}\sum_{k=0}^{n-1} f(x_k)\exp(-2\pi i\frac{jk}{n}) \\ &= \frac{h}{2\pi}\Big(\frac{1}{2}f(0) + \sum_{k=1}^{n-1} f(x_k)\exp(-2\pi i\frac{jk}{n}) + \frac{1}{2}f(2\pi)\Big), \end{aligned}$$

mit $h = 2\pi/n$. Dies ist aber gleichzeitig die Trapeznäherung an das Integral (1.2.10). Aus der Euler-McLaurin-Summenformel (vgl. Numerik I, Satz 3.5.3) folgt, daß die Trapezregel für periodische und genügend oft differenzierbare Integranden eine beliebig hohe Ordnung besitzt:

$$|a_j - c_j| \le k_m\Big(\frac{j+1}{n}\Big)^m \max_{\nu\le m}\|f^{(\nu)}\|_\infty,\ 0 \le j < n.$$

Daher bilden die ersten diskreten Fourierkoeffizienten a_j (z.B. mit $j \le \alpha n,\ \alpha < 1$) sehr gute Approximationen an die c_j. Dies gilt nicht mehr für $j \cong n$, da für eine reelle, stetige Funktion f zwar gilt $c_j \to 0,\ j \to \infty$, aber $a_{n-j} = \bar{a}_j$.

b) Schnelle Berechnung *diskreter Faltungen* von Vektoren. Zu Vektoren $u = (u_0, u_1, \dots, u_{n-1})^\mathsf{T}$, $y = (y_0, y_1, \dots, y_{n-1})^\mathsf{T} \in \mathbf{C}^n$, deren Komponenten bei Bedarf n-periodisch fortgesetzt werden, wird folgendes Produkt definiert,

$$z = u * y \quad :\Longleftrightarrow \quad z_k = \sum_{j=0}^{n-1} u_{k-j}\,y_j,\ k = 0, \dots, n-1, \tag{1.2.11}$$

bzw. $k \in \mathbf{Z}$. Dieses *Faltungsprodukt* $*$ ist kommutativ und assoziativ. Werden die diskreten Fouriertransformierten zu $u, y, z \in \mathbf{C}^n$ mit $\widehat{u}, \widehat{y}, \widehat{z}$ bezeichnet, etwa $\widehat{y} = (\widehat{y}_k) = (\sum_{j=0}^{n-1} y_j \omega_n^{jk})$, so gilt

$$\widehat{z} = (\widehat{u * y}) = \widehat{u} \circ \widehat{y} := (\widehat{u}_k \widehat{y}_k)_{k=0}^{n-1}. \tag{1.2.12}$$

Beweis Wegen der Periodizität ist

$$\widehat{z}_k = \sum_{j=0}^{n-1} z_j \omega_n^{jk} = \sum_{j=0}^{n-1} \sum_{m=0}^{n-1} u_{j-m} y_m \omega_n^{(j-m)k} \omega_n^{mk} = \sum_{m=0}^{n-1} y_m \omega_n^{mk} \sum_{j=-m}^{n-1-m} u_j \omega_n^{jk} = \widehat{y}_k \widehat{u}_k. \blacksquare$$

Wegen (1.2.12) läßt sich die Berechnung einer Faltung (Aufwand direkt: $2n^2$) auf drei Fouriertransformationen und eine komponentenweise Multiplikation zurückführen, dies erfordert insgesamt nur ca. $\frac{9}{4} n \log_2 n$ Operationen. Sind in (1.2.11) u und z bekannt, kann umgekehrt das Gleichungssystem für y mit der *zyklischen* Matrix $(u_{k-j})_{k,j=0}^{n-1}$ mit gleichem Aufwand (statt n^3 bei Gauß-Elimination) gelöst werden. Anwendungen von (1.2.11), (1.2.12):

— *digitale Filter*, z.B. gleitender Mittelwert $z_j := \frac{1}{4}(y_{j-1} + 2y_j + y_{j+1})$, *Tiefpaß-Filter*,

— Bildverarbeitung (Spezialfall), mehrdimensionale Fouriertransformation, Bildtelephon,

— Polynom- bzw. Langzahl-Multiplikation (statt über $\mathbf{C}$ in endlichen Zahlkörpern),

— Statistik

c) Polynom-Interpolation in Tschebyscheff-Knoten: vgl. Numerik I, §2.4.

Verallgemeinerungen, Weiterentwicklungen

- Permutationsfreie FFT: Durch geeignete Zusammenfassung von Schritten in den letzten $\lfloor \ell/2 \rfloor$ Stufen der FFT erscheinen die Koeffizienten am Ende in der richtigen Reihenfolge.
- Für allgemeinere n mit Primfaktorzerlegung $n = p_1 p_2 \cdots p_\ell$ ist analog eine Aufspaltung in ℓ Stufen möglich (z.B. $n = 1000 = 2^3 5^3$).
- Das Berechnungsprinzip ist auch auf einige andere Transformationen übertragbar, z.B., die Hartley-Transformation (*reelle* trigonometrische Entwicklung, reell durchführbar), Walsh-Transformation (reelle Entwicklung nach Treppenfunktionen).

2 Ausgleichsrechnung

Bei vielen wissenschaftlichen Problemen weiß oder vermutet man, wie eine meßbare Größe $y \in \mathbf{R}$ von Versuchsparametern $a = (a_1, \ldots, a_p)^\mathsf{T} \in \mathbf{R}^p$ abhängt,

$$y = f(a_1, \ldots, a_p; x_1, \ldots, x_n).$$

Dabei sind $(x_1, \ldots, x_n)^\mathsf{T} =: x$ unbekannte Konstanten einer bestimmten Funktion f, die aufgrund von Messungen zu bestimmen sind, im Idealfall in der Form

$$y_k = f(a_{k1}, \ldots, a_{kp}; x_1, \ldots, x_n),\ \ k = 1, \ldots, m. \tag{2.0.1}$$

Für $m > n$ existiert aber i.a. keine Lösung des Gleichungssystems (2.0.1), da die Meßwerte normalerweise Fehler enthalten. Daher beschränkt man sich darauf, (2.0.1) mit möglichst kleinem *Defekt* zu lösen:

$$\sum_{k=1}^{m} \Big| y_k - f(a_{k1}, \ldots, a_{kp}; x_1, \ldots, x_n) \Big|^2 \stackrel{!}{=} \min . \tag{2.0.2}$$

Diese, auf *Gauß* zurückgehende *Methode der kleinsten Quadrate* hat u. a. besonders gute statistische Eigenschaften. Im folgenden werden nur lineare Funktionen f und die Euklidnorm betrachtet.

2.1 Kleinste-Quadrate-Lösung

Das erste Ziel ist es hier, für beliebige lineare Gleichungssysteme einen verallgemeinerten Lösungsbegriff zu definieren.

Definition 2.1.1 *Zu $A \in \mathbf{C}^{m\times n}$, $m, n \in \mathbf{N}$, und $y \in \mathbf{C}^m$ heißt $\hat{x} \in \mathbf{C}^n$* Kleinste-Quadrate-Lösung, *wenn $\hat{x}$ das Funktional*

$$\|Ax - y\|_2^2 = (Ax - y)^*(Ax - y) \tag{2.1.1}$$

minimiert. Ist das Minimum von (2.1.1) nicht eindeutig, wird mit x^+ diejenige Kleinste-Quadrate-Lösung mit kleinster Norm bezeichnet,

$$\|x^+\|_2 = \min\{\|\hat{x}\|_2 :\ \hat{x} \in \mathbf{C}^n \textit{ minimiert (2.1.1)}\}. \tag{2.1.2}$$

Um die Lösungen aus Definition 2.1.1 genauer charakterisieren zu können, werden folgende Begriffe benötigt. Zu $A \in \mathbf{C}^{m\times n}$ heißen

$$\left.\begin{array}{rcll} N(A) & := & \{x \in \mathbf{C}^n :\ Ax = 0\} & \text{Nullraum bzw. Kern} \\ R(A) & := & \{y \in \mathbf{C}^m :\ \exists x \in \mathbf{C}^n \text{ mit } Ax = y\} = A\mathbf{C}^n & \text{Bildraum} \end{array}\right\} \text{von } A. \tag{2.1.3}$$

Folgende wichtige Beziehung aus der Linearen Algebra wird später verwendet:

$$R(A)^\perp = N(A^*). \tag{2.1.4}$$

Satz 2.1.2 *Es sei $P : \mathbf{C}^n \mapsto R(A)$ der Orthogonal-Projektor auf $R(A)$, d.h., $P^2 = P,\ P^* = P$. Dann sind die folgenden Aussagen äquivalent:*

$$\begin{aligned} \|A\hat{x} - y\|_2 &= \min\{\|Ax - y\|_2 :\ x \in \mathbf{C}^n\}, && (2.1.5)\\ A\hat{x} &= Py && (2.1.6)\\ A^*A\hat{x} &= A^*y,\ \textit{bzw.},\ A^*(A\hat{x} - y) = 0 \quad \textit{`Normalengleichung'}. && (2.1.7) \end{aligned}$$

Beweis a) (Äquivalenz der ersten beiden Aussagen) Wegen $Ax = PAx\,\forall x$ gilt

$$\begin{aligned} \|Ax - y\|_2^2 &= \|Ax - Py - (I - P)y\|^2 \\ &= \|Ax - Py\|_2^2 - 2\mathrm{Re}y^* \underbrace{(I - P^*)P}_{=0}(Ax - y) + \|(I - P)y\|_2^2. \end{aligned}$$

Da der letzte Summand nicht von x abhängt, wird das Minimum für $\|Ax - Py\|_2 = 0$ angenommen, da $Py \in R(A)$.

b) (Äquivalenz der beiden letzten Gleichungen)

$$\left.\begin{array}{lll} (2.1.6) \iff & \text{Defekt } d := A\hat{x} - y = (P - I)y & \in R(A)^\perp \\ (2.1.7) \iff & \text{Defekt } d := A\hat{x} - y & \in N(A^*) \end{array}\right\} \text{äquivalent wegen (2.1.4)} \quad \blacksquare$$

Aus diesem Satz läßt sich leicht ableiten

Satz 2.1.3 *Das Ausgleichsproblem (2.1.1) besitzt (mindestens) eine Kleinste-Quadrate-Lösung $\hat{x}$. Für $m \geq n$, Rang$(A) = n$, ist sie eindeutig. Die allgemeine Lösung von (2.1.1) hat die Form $\hat{x} + N(A)$. Die Lösung x^+ mit kleinster Norm (2.1.2) ist eindeutig bestimmt.*

Beweis a) Existenz, aus (2.1.6) und $Py \in R(A)$ oder aus (2.1.7): A^*A ist positiv semidefinit, denn $x^*A^*Ax = \|Ax\|_2^2 \geq 0$. Es ist $x^*A^*Ax = \|Ax\|_2^2 = 0 \iff Ax = 0$, also $N(A^*A) = N(A) \Rightarrow R(A^*A) = N(A^*A)^\perp = N(A)^\perp = R(A^*)$. Daher ist (2.1.7) lösbar, die allgemeine Lösung ist $\hat{x} + N(A)$.

b) Eindeutigkeit von x^+: Im affinen Unterraum $\hat{x} + N(A)$ existiert genau ein Element kleinster Norm, x^+, die Projektion des Nullpunkts auf diesen Unterraum. $\blacksquare$

2.2 Die Verallgemeinerte Inverse

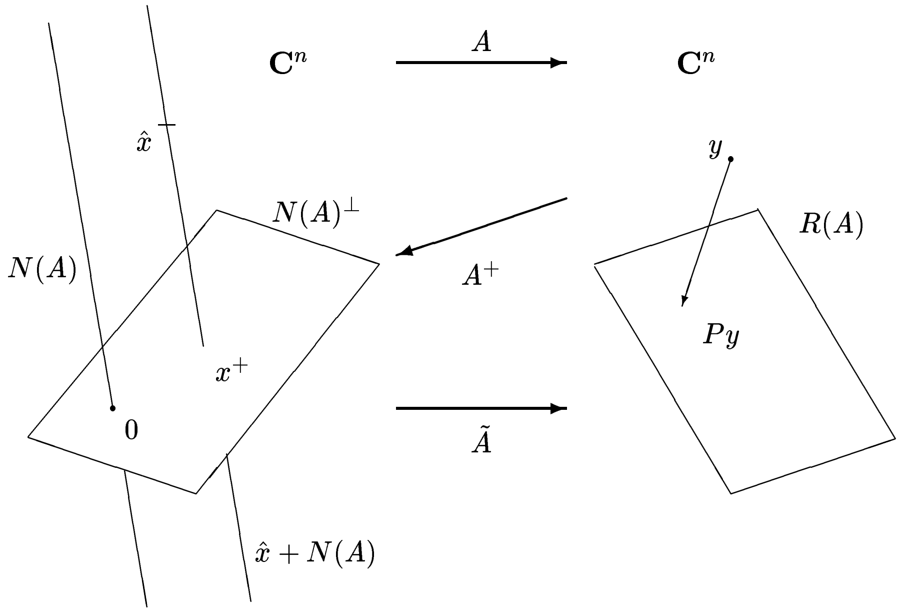
Analog zum regulären Fall mit quadratischer Matrix A, wo sich *die* Lösung in der Form $x = A^{-1}y$ angeben läßt, kann auch die Zuordnung $y \mapsto x^+$ durch eine Matrix realisiert werden.

Definition 2.2.1 *Mit $A \in \mathbf{C}^{m\times n}$, der Orthogonalprojektion $P :\ \mathbf{C}^m \mapsto R(A)$ und der restringierten Abbildung $\tilde{A} := A\Big|_{N(A)^\perp}$ heißt*

$$A^+ : \left\{ \begin{array}{ccl} \mathbf{C}^m & \mapsto & \mathbf{C}^n \\ y & \mapsto & A^+y := (\tilde{A})^{-1}Py \end{array}\right. \qquad (2.2.1)$$

Verallgemeinerte Inverse *von A (bzw. Moore-Penrose-Inverse, Pseudo-Inverse).*

Skizze:



Satz 2.2.2 *Die Minimallösung x^+ von (2.1.1), (2.1.2) stimmt mit*

$$x^+ = A^+ y \tag{2.2.2}$$

überein, es gilt $R(A^+) = N(A)^\perp$.

Beweis Jedes $x \in \mathbf{C}^n$ ist eindeutig zerlegbar, $x = x_0 + x_\perp$ in $x_0 \in N(A)$ und $x_\perp \in N(A)^\perp$. Damit ist $Ax = \tilde{A}x_\perp$. Mit einer solchen Zerlegung einer Minimallösung $\hat{x} = x_0 + x^+$ von (2.1.1) ergibt sich aus (2.1.6) und Satz 2.1.3 die Aussage:

$$\tilde{A}x^+ = A\hat{x} = Py. \qquad \blacksquare$$

Bemerkung: a) Jede Matrix besitzt eine Verallgemeinerte Inverse (auch die Nullmatrix).

b) Für Rang$(A) = n,\ m \geq n$, gilt also

$$A^+ = (A^*A)^{-1}A^*.$$

Ist außerdem noch $m = n$, stimmen A^+ und A^{-1} überein.

Die Verallgemeinerte Inverse kann unabhängig von Definition 2.2.1 als Lösung der im nächsten Satz angegebenen Gleichungen charakterisiert werden.

Satz 2.2.3 *Die Verallgemeinerte Inverse A^+ aus (2.2.1) ist die eindeutig bestimmte Lösung der Gleichungen*

$$\begin{array}{crcl} (a) & A^+A & = & (A^+A)^* \\ (b) & AA^+ & = & (AA^+)^* \\ (c) & AA^+A & = & A \\ (d) & A^+AA^+ & = & A^+ \end{array} \tag{2.2.3}$$

Es gilt $A^{++} = A$ und $(A^+)^* = (A^*)^+$.

Beweis a) Im folgenden spielen die Orthogonalprojektoren

$$\begin{aligned} P: &\quad \mathbf{C}^m \mapsto R(A) \\ U: &\quad \mathbf{C}^n \mapsto N(A)^\perp, \quad I-U: \ \mathbf{C}^n \mapsto N(A), \end{aligned}$$

eine wichtige Rolle. Als erstes wird nämlich gezeigt, daß $AA^+ = P$, $A^+A = U$ ist. Dazu wird definiert

$$R := A^+A = \tilde{A}^{-1}PA = \tilde{A}^{-1}A: \ \mathbf{C}^m \mapsto N(A)^\perp. \tag{2.2.4}$$

Mit $I = U + (I-U)$ gilt $A = AU + A(I-U) = AU = \tilde{A}U$. Also ist $R = \tilde{A}^{-1}A = \tilde{A}^{-1}AU = \tilde{A}^{-1}\tilde{A}U = U$, und da U hermitesch ist, gilt insgesamt

$$U = A^+A = (A^+A)^*, \qquad \text{d.h. (2.2.3,a).} \tag{2.2.5}$$

Analog folgt

$$P = AA^+ = (AA^+)^*, \qquad \text{d.h. (2.2.3,b).} \tag{2.2.6}$$

Daraus ergeben sich aber auch die restlichen Gleichungen, denn

$$\begin{aligned} (AA^+)A &= PA = A && \text{wegen (2.2.6),} && \text{d.h. (2.2.3,c),} \\ (A^+A)A^+ &= UA^+ = A^+ && \text{wegen (2.2.5),} && \text{d.h. (2.2.3,d).} \end{aligned}$$

Also löst A^+ aus (2.2.1) die Gleichungen (2.2.3).

b) Zum Nachweis der Eindeutigkeit sei $\hat{A}$ eine Lösung von (2.2.3). Dann gelten für $\hat{P} := A\hat{A}$ und $\hat{U} = \hat{A}A$ die Beziehungen $\hat{P}^* = \hat{P}$ und $\hat{U}^* = \hat{U}$, sowie

$$\hat{P}^2 = (A\hat{A}A)\hat{A} = A\hat{A} = \hat{P}, \quad \hat{U}^2 = (\hat{A}A\hat{A})A = \hat{A}A = \hat{U}. \tag{2.2.7}$$

Beide Abbildungen sind also Orthogonal-Projektoren, für die jetzt die Eigenschaften

$$N(\hat{U}) = N(A), \quad R(\hat{P}) = R(A)$$

nachgewiesen werden. Für $\hat{U}$ etwa folgt aus (2.2.3,c) die Identität $A = A\hat{A}A = A\hat{U}$, also $N(\hat{U}) \subseteq N(A)$. Andererseits ist für $x \in N(A)$ auch $\hat{U}x = \hat{A}Ax = 0$ und somit $N(\hat{U}) = N(A)$. Analog ergibt sich die Aussage für $R(\hat{P})$.

Da die Bildräume der Orthogonalprojektoren $\hat{P}$ bzw. $\hat{U}$ mit denen von P bzw. U übereinstimmen, sind auch die Projektoren gleich, $\hat{P} = A\hat{A} = P = AA^+$ und $\hat{U} = \hat{A}A = U = A^+A$. Aus (2.2.3,d) folgt daher

$$\hat{A} = (\hat{A}A)\hat{A} = A^+A\hat{A} = A^+AA^+ = A^+.$$

c) Die beiden letzten Aussagen folgen aus der Eindeutigkeit nach Ersetzung von A in (2.2.3) durch A^+ bzw. A^*. ■

Der nächste Abschnitt behandelt die numerische Berechnung der verallgemeinerten Inversen A^+.

2.3 QR-Zerlegung und Berechnung von A^+

In der Numerik I wurde gezeigt, daß die Fehleranfälligkeit eines linearen Gleichungssystems $Bx = b$ durch die Konditionszahl $\kappa(B) = \|B\|\|B^{-1}\|$ bestimmt wird. Die Lösung des Ausgleichsproblems (2.1.1) kann für $\text{Rang}(A) = n$ zwar über die Normalengleichung (2.1.7), $A^*A = A^*y$, berechnet werden. Die Kondition der dabei auftretenden Matrix $B = A^*A$ ist allerdings i.a. sehr schlecht, im invertierbaren Fall $m = n$ gilt $\kappa_2(B) = \kappa_2(A^*A) = \varrho(A^*A)\varrho(A^{-1}A^{*-1}) = \kappa_2(A)^2$. Die Konditionszahl des Ausgangssystems $Ax = y$ wird beim Übergang zur Normalengleichung also quadriert.

Eine Abhilfe bietet hier die Zerlegung der Matrix in eine unitäre Matrix Q und eine obere Dreieckmatrix R, die sogenannte QR-Zerlegung,

$$A = QR, \quad Q \in \mathbf{C}^{m\times m},\ Q^*Q = I, \quad R = \begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1n} \\ & r_{22} & \cdots & r_{2n} \\ & & \ddots & \end{pmatrix} \in \mathbf{C}^{m\times n} \tag{2.3.1}$$

Bei einer regulären, quadratischen Matrix A kann auch diese Zerlegung zur Lösung des Linearen Gleichungssystems $Ax = y$ verwendet werden, es gilt $x = R^{-1}Q^*y$. Diese Darstellung ist oft sogar numerisch stabiler als die übliche LR-Zerlegung, $A = L\tilde{R}$ (vgl. Numerik I), da $\kappa_2(R) = \kappa_2(QR) = \kappa_2(A)$, aber i.a. $\kappa_2(\tilde{R}) \gg \kappa_2(L\tilde{R}) = \kappa_2(A)$. Allerdings ist die Berechnung der QR-Zerlegung etwas teurer als die der LR-Zerlegung.

Die QR-Zerlegung (2.3.1) läßt sich im Prinzip mit dem Orthogonalisierungsverfahren von E.Schmidt (angewendet auf die Spaltenvektoren) berechnen. Dabei können allerdings große Rundungsfehler auftreten, etwa wenn Spaltenvektoren von A fast linear abhängig sind. Es ist numerisch günstiger, die Matrix durch elementare Spiegelungen (Householder-Transformationen) umzuformen. Dabei wird jeweils, analog zum Gauß-Algorithmus, der erste Spaltenvektor der (Rest-) Matrix auf ein Vielfaches des ersten Einheitsvektors abgebildet.

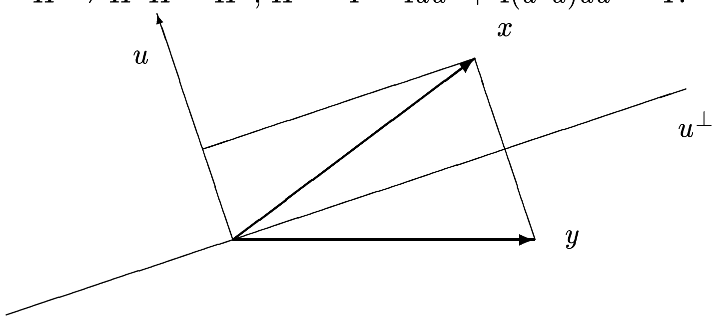
Lemma 2.3.1 *Mit $u \in \mathbf{C}^m$, $\|u\|_2 = 1$, sei*

$$H := I - 2uu^*. \tag{2.3.2}$$

Die Matrix H ist hermitesch ($H^* = H$), unitär ($H^*H = I$) und involutorisch ($H^2 = I$).

Beweis $H^* = I - 2uu^* = H \Rightarrow H^*H = H^2$, $H^2 = I - 4uu^* + 4(u^*u)uu^* = I$. ■

Geometrisch:
$y = Hx = x - 2(u^*x)u$
Spiegelung an $u^\perp$.



Anwendung: u ist so zu bestimmen, daß x auf die $e^{(1)}$-Achse gespiegelt wird, $e^{(1)} = (1,0,0,\dots)^\mathsf{T}$.

$$Hx = x - 2(u^*x)u \overset{!}{=} \beta e^{(1)} \quad (= y). \tag{2.3.3}$$

$$\left.\begin{array}{lll} H \text{ unitär} & \Rightarrow & |\beta| = \|x\|_2 \\ H \text{ hermitesch} & \Rightarrow & \mathbf{R} \ni x^*Hx = \beta x^* e^{(1)} = \beta \bar{x}_1 \end{array}\right\} \Rightarrow \beta = \pm\sigma\|x\|_2,$$

wobei $x_1 = \sigma|x_1|$. Mit (2.3.3) und $\|u\|_2 = 1$ folgt die Darstellung

$$u = \frac{x - \beta e^{(1)}}{\|x - \beta e^{(1)}\|}.$$

In der ersten Komponente von u tritt die Differenz $x_1 - \beta$ auf. Um bei der Berechnung dieser Differenz eine Auslöschung führender Ziffern zu vermeiden, wird das Vorzeichen bei β so gewählt, daß tatsächlich eine Addition der Beträge stattfindet:

$$x_1 - \beta = \sigma|x_1| \mp \sigma\|x\|_2 \overset{!}{=} \sigma(|x_1| + \|x\|_2),$$

also $\beta = -\sigma\|x\|_2$. Damit wird

$$\|x - \beta e^{(1)}\|_2^2 = (|x_1|^2 + \|x\|_2)^2 + |x_2|^2 + \ldots + |x_m|^2 = 2\|x\|(|x_1| + \|x\|).$$

Zusammenfassend ergibt sich

Lemma 2.3.2 *Der Vektor $0 \neq x \in \mathbf{C}^m$ mit $x_1 = \sigma|x_1|$ wird durch die Matrix*

$$H = I - \frac{vv^*}{\|x\|_2(|x_1| + \|x\|_2)}, \qquad v := x + \sigma\|x\|_2 e^{(1)},$$

auf $-\sigma\|x\|_2 e^{(1)}$ abgebildet.

Mit dieser Konstruktion können in der ersten Spalte der Matrix A die Subdiagonalelemente eliminiert werden durch die Wahl $HAe^{(1)} = \beta e^{(1)}$. Wie beim Gaußalgorithmus wird dieser Schritt mit den Restmatrizen wiederholt. Im j-ten Schritt ($A^{(1)} := A$) hat man also

$$A^{(j)} = \left(\begin{array}{ccc|ccc} * & \cdots & * & * & \cdots & * \\ & \ddots & \vdots & \vdots & & \vdots \\ & & * & * & \cdots & * \\ \hline & & & a_{jj}^{(j)} & \cdots & a_{jn}^{(j)} \\ & 0 & & \vdots & & \vdots \\ & & & a_{mj}^{(j)} & \cdots & a_{mn}^{(j)} \end{array}\right) =: \begin{pmatrix} R^{(j)} & B^{(j)} \\ 0 & \tilde{A}^{(j)} \end{pmatrix},$$

mit $R^{(j)} \in \mathbf{C}^{(j-1)\times(j-1)}$, $\tilde{A}^{(j)} \in \mathbf{C}^{(m-j+1)\times(n-j+1)}$. Da Produkte von Spiegelungen unitär sind, kann A unitär auf Dreieckgestalt transformiert werden. Dazu sind die Transformationsmatrizen in der Form

$$H^{(j)} = \left(\begin{array}{c|c} I_{j-1} & 0 \\ 0 & \tilde{H}^{(j)} \end{array}\right) \quad \Leftrightarrow \quad u^{(j)} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ * \\ \vdots \\ * \end{pmatrix} \begin{array}{l} \left.\vphantom{\begin{matrix}0\\ \vdots\\ 0\end{matrix}}\right\} j-1 \\ \left.\vphantom{\begin{matrix}*\\ \vdots\\ *\end{matrix}}\right\} m-j+1 \end{array}$$

zu wählen. Beim Produkt $H^{(j)}A^{(j)} = A^{(j+1)}$ bleiben dann die ersten $j-1$ Zeilen unverändert. Die Anwendung von Lemma 2.3.2 erfolgt mit

$$\tilde{H}^{(j)} \begin{pmatrix} a_{jj}^{(j)} \\ \vdots \\ a_{mj}^{(j)} \end{pmatrix} \stackrel{!}{=} \begin{pmatrix} * \\ 0 \\ \vdots \\ 0 \end{pmatrix}. \tag{2.3.4}$$

In $k := \min\{m-1, n\}$ Schritten ergibt sich so

$$A^{(k+1)} = R^{(k+1)} =: R = H^{(k)} \cdots H^{(2)}H^{(1)}A \quad \Longleftrightarrow \quad A = H^{(1)}H^{(2)} \cdots H^{(k)}R =: QR.$$

Der folgende Satz faßt diese Konstruktion zusammen.

Satz 2.3.3 *Die QR-Zerlegung einer Matrix $A \in \mathbf{C}^{m\times n}$ in das Produkt einer unitären Matrix $Q \in \mathbf{C}^{m\times m}$ und einer oberen Dreieckmatrix $R \in \mathbf{C}^{m\times n}$ ergibt sich in $k = \min\{m-1, n\}$ Schritten in folgender Weise.*

$$A^{(1)} := A, \quad A^{(j)} = (a_{\mu\nu}^{(j)})_{\mu=1,\nu=1}^{m,n}. \tag{2.3.5}$$

Für $j = 1, 2, \ldots, k$ berechne man

$$\left.\begin{array}{l} c_j^2 := \sum_{\mu=j}^m |a_{\mu j}^{(j)}|^2, \; c_j > 0, \; K_j := c_j^2 + |a_{jj}^{(j)}|c_j \\ v^{(j)} := \sigma_j c_j e^{(j)} + \begin{pmatrix} 0 \\ \vdots \\ 0 \\ a_{jj}^{(j)} \\ \vdots \\ a_{mj}^{(j)} \end{pmatrix}, \; \textit{wobei } \sigma_j |a_{jj}^{(j)}| = a_{jj}^{(j)}, \end{array}\right\} \tag{2.3.6}$$

und bilde

$$\left.\begin{array}{l} H^{(j)} := I - \frac{1}{K_j} v^{(j)} v^{(j)*} \\ A^{(j+1)} := H^{(j)}A^{(j)} = A^{(j)} - v^{(j)}\left(\frac{1}{K_j} v^{(j)*}A^{(j)}\right) \end{array}\right\} \tag{2.3.7}$$

Dann ist $A = QR$ mit

$$Q = H^{(1)} \cdots H^{(k)}, \quad R = A^{(k+1)}, \tag{2.3.8}$$

$$\begin{array}{llll} \textit{wobei} & R = \begin{pmatrix} r_{11} & \cdots & r_{1m} & \cdots & r_{1n} \\ & \ddots & \vdots & & \vdots \\ 0 & & r_{mm} & \cdots & r_{mn} \end{pmatrix}, & R = \begin{pmatrix} r_{11} & \cdots & r_{1n} \\ & \ddots & \vdots \\ 0 & & r_{nn} \\ & 0 & \end{pmatrix} \\ \textit{wenn} & m \le n, k := m-1, & m > n, k := n. \end{array}$$

Praktische Durchführung, Rechenaufwand

Aus Effizienzgründen werden keine Produkte von Matrizen ausgerechnet, der Übergang $A^{(j+1)} = H^{(j)}A^{(j)}$ wird der Klammerung in (2.3.7) entsprechend durchgeführt. Auch Q wird nicht explizit berechnet. Bei der Auflösung mehrerer Gleichungssysteme mit Matrix A kann das Produkt Q^*y mit gleichem Aufwand in der Form $H^{(k)}\cdots H^{(1)}y$ ausgewertet werden. Dazu kann man die Komponenten der Vektoren $v^{(j)}$ im freien Platz unter und einschließlich der Hauptdiagonale von R unterbringen. Nur die Elemente r_{jj} werden zur Vereinfachung anderweitig gespeichert. So erfordert die Durchführung von (2.3.6),(2.3.7) folgenden *Rechenaufwand*:

Der Aufwand für c_j, K_j ist von niederer Ordnung und wird vernachlässigt. In (2.3.7) werden berechnet:

$$z^{(j)*} := v^{(j)*}A^{(j)} = (\underbrace{0\dots0}_{j-1}\underbrace{*\dots*}_{m-j+1})\begin{pmatrix} \ddots & \\ \underbrace{0}_{j-1} & \boxed{\underbrace{\tilde{A}^{(j)}}_{n-j+1}} \end{pmatrix}\Big\}_{m-j+1} = (\underbrace{0\dots0}_{j-1}\bullet\underbrace{*\dots*}_{n-j}).$$

Da das Ergebnis in der j-ten Spalte ('$\bullet$') nach Konstruktion (2.3.4) nicht berechnet werden muß, sind für die Produktbildung $2(m-j+1)(n-j)$ Operationen nötig. Der Vektor $v^{(j)}$ besitzt höchstens $m-j+1$, $z^{(j)}$ nicht mehr als $n-j$ nichttriviale Elemente. Daher benötigt die Bildung von $A^{(j+1)} = A^{(j)} - v^{(j)}[\frac{1}{K_j}z^{(j)*}]$ ebenfalls $2(m-j+1)(n-j)$ Operationen (Spalte j bekannt). Im Fall $n < m$ erhält man so den Gesamtaufwand

$$4\sum_{j=1}^{n}(m-j+1)(n-j) = 4\sum_{\nu=0}^{n-1}(m-n+\nu+1)\nu = \frac{4}{3}n^3 + 2(m-n)n^2 + \dots$$

Ein analoges Ergebnis gilt für $n \geq m$ und führt auf

Satz 2.3.4 *Der Rechenaufwand für die QR-Zerlegung einer $m \times n$-Matrix beträgt*

$$\frac{4}{3}\min\{m^3,n^3\} + 2|m-n|\min\{m^2,n^2\} + \dots \text{ Operationen.}$$

Bemerkung: Im regulären Fall mit $m = n$ ist die QR-Zerlegung also ungefähr doppelt so teuer wie eine LR-Zerlegung ohne Pivotisierung und noch 33% teurer als dieselbe mit Spaltenpivotisierung. Dafür wird die Konditionszahl κ_2 von R bei der QR-Zerlegung nicht vergrößert.

Mit Hilfe der QR-Zerlegung kann die *verallgemeinerte Inverse* A^+ bei rechteckigen Matrizen von vollem Rang, $\text{Rang}(A) = \min\{m,n\}$, berechnet werden. Für $m \geq n$, $\text{Rang}(A) = n$, gilt

$$A = QR, \quad \text{mit } R = \begin{pmatrix} R_0 \\ 0 \end{pmatrix},\ R_0 \in \mathbf{C}^{n\times n} \text{ regulär.} \tag{2.3.9}$$

Durch Aufspaltung der transformierten rechten Seite $Q^*y = z = \begin{pmatrix} \hat{z} \\ \tilde{z} \end{pmatrix}$, $\hat{z} = (z_1,\dots,z_n)^{\mathsf{T}}$, ergibt sich direkt in (2.1.1)

$$\|Ax-y\|_2^2 = \|Q(Rx-Q^*y)\|_2^2 = \|Rx-z\|_2^2 = \|R_0x-\hat{z}\|_2^2 + \|\tilde{z}\|_2^2. \tag{2.3.10}$$

Da die zweite Norm unabhängig von x ist, wird das Minimum bei $R_0x = \hat{z}$ angenommen, also ist $\hat{x} = R_0^{-1}\hat{z}$. Zu dieser Zuordnung $y \mapsto \hat{x} = x^+$ gehört die Matrix

$$A^+ = R_0^{-1}E_{nm}Q^*, \quad E_{nm} := \left(\begin{array}{ccc|c} 1 & & & 0\cdots \\ & \ddots & & \vdots \\ & & 1 & 0\cdots \end{array}\right) = (I_n|0) \in \mathbf{R}^{n\times m}. \tag{2.3.11}$$

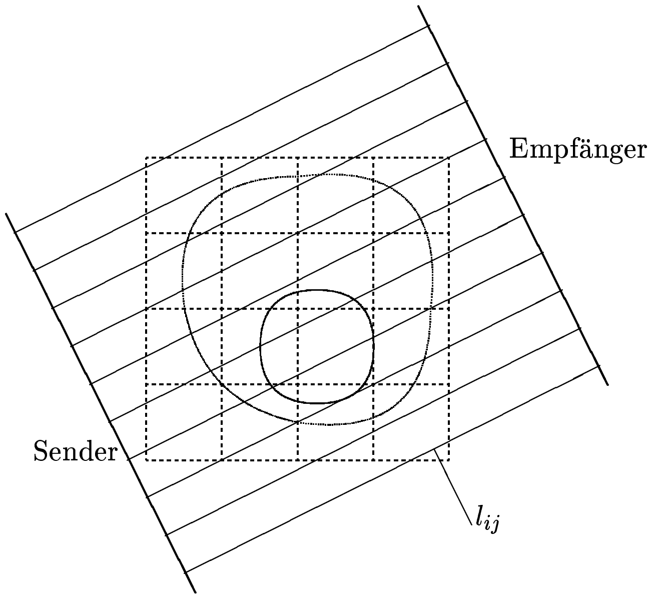
Die Matrix E_{nm} wählt dabei die ersten n Komponenten eines m-Vektors aus. Die Darstellung (2.3.11) ist äquivalent zu $A^+ = (A^*A)^{-1}A^*$, numerisch aber besser konditioniert (vgl. die Diskussion am Anfang des Abschnitts) und billiger! Für die Projektoren aus §2.2 gilt hier übrigens

$$U = A^+A = I_n : \ \mathbf{C}^n \mapsto \mathbf{C}^n = N(A)^\perp,$$
$$P = AA^+ = QE_{nm}^\top R_0 R_0^{-1} E_{nm} Q^* = Q \begin{pmatrix} I_n & 0 \\ 0 & 0 \end{pmatrix} Q^* : \ \mathbf{C}^m \mapsto R(A).$$

Im Fall $m < n$, $\text{Rang}(A) = m$, kann analog vorgegangen werden durch Berechnung der QR-Zerlegung von $A^* = QR$. Hier gilt $A^+ = QE_{mn}^\top (R_0^*)^{-1} : \ \mathbf{C}^m \mapsto \mathbf{C}^n$.

Der allgemeine Fall, $\text{Rang}(A) < \min\{m, n\}$, läßt sich mit Spalten- bzw. Zeilenvertauschungen auf eine Kombination der beiden behandelten Fälle zurückführen.

Beispiel 2.3.5 Computer-Tomographie
Die Dichte-Verteilung in einem Körper kann aus der von außen gemessenen Absorption von (Röntgen-, Gamma-) Strahlen rekonstruiert werden. Zur Rekonstruktion kann, z.B., eine Schicht des Untersuchungsbereichs in Quadrate unterteilt werden. Der Bereich wird dann mit einer Reihe von parallelen Meßstrahlen durchleuchtet und die Messung unter verschiedenen Drehwinkeln der Apparatur wiederholt. Setzt man in jedem Teilquadrat des Untersuchungsbereichs eine konstante Dichte d_j voraus, dann ergibt sich für die Absorption a_i im i-ten Meßstrahl die Beziehung



$$a_i = \sum_j \ell_{ij} d_j.$$

Dabei gibt ℓ_{ij} die Weglänge des i-ten Meßstrahls im j-ten Quadrat an. Nach genügend vielen Messungen erhält man ein (überbestimmtes) Lineares Gleichungssystem für die unbekannten Dichten d_j. Dieses Rekonstruktionsproblem (Radon-Transformation) reagiert allerdings empfindlich auf Störungen. Daher sind in der Praxis Zusatzmaßnahmen erforderlich, um akzeptable Lösungen zu bekommen (Demo-Programm TOMODEMO.PAS).

3 Lineare Optimierung

3.1 Problemstellung, Normalform

Die lineare Optimierung befaßt sich mit einem, v.a. im kaufmännischen Bereich, sehr wichtigen Problem der numerischen Datenverarbeitung, dem der Optimierung einer linearen Funktion unter linearen Nebenbedingungen in Form von Ungleichungen ('Lineare Programme').

Beispiel 3.1.1 Eine gewünschte Metallegierung L_0 aus 30% Blei, 30% Zink und 40% Zinn soll möglichst billig produziert werden. Als Ausgangsmaterial stehen 5 Legierungen $L_1, \dots, L_5$ zur Verfügung mit folgenden Daten:

	Legierung	L_1	L_2	L_3	L_4	L_5	L_0
Gewichts-	Blei	0.1	0.1	0.6	0.4	0.3	0.3
Anteile	Zink	0.5	0.4	0.1	0.2	0.6	0.3
	Zinn	0.4	0.5	0.3	0.4	0.1	0.4
	Preis/kg	40	45	50	52	20	?

Zur mathematischen Formulierung dieses Problems werden die Gewichtsanteile x_i der Legierungen L_i an L_0 eingeführt. Bei der Produktion sind die Nebenbedingungen

$$\begin{aligned} & x_i \geq 0, \quad i = 1, \dots, 5, \\ \text{Blei:}\quad & 0.1x_1 + 0.1x_2 + 0.6x_3 + 0.4x_4 + 0.3x_5 = 0.3 \\ \text{Zink:}\quad & 0.5x_1 + 0.4x_2 + 0.1x_3 + 0.2x_4 + 0.6x_5 = 0.3 \\ \text{Zinn:}\quad & 0.4x_1 + 0.5x_2 + 0.3x_3 + 0.4x_4 + 0.1x_5 = 0.4 \end{aligned}$$

einzuhalten, das Ziel ist aber letztendlich die Minimierung der Kosten

$$40x_1 + 45x_2 + 50x_3 + 52x_4 + 20x_5 \stackrel{!}{=} \min.$$

Viel Optimierungsprobleme besitzen eine ähnliche Form oder können auf diese umgeformt werden. Sie wird als Normalform verwendet.

Definition 3.1.2 *Gegeben sei eine Matrix $A \in \mathbf{R}^{m\times n}, m < n$, mit Rang $A = m$, und $b \in \mathbf{R}^m, c \in \mathbf{R}^n$. Die Aufgabe, ein $\hat{x} \in \mathbf{R}^n$ zu bestimmen, das unter den* Nebenbedingungen

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

(d.h. $x_i \geq 0\ \forall i = 1, \dots, n$) die Zielfunktion

$$c^{\mathsf{T}} x$$

maximiert, heißt **L***ineares* **O***ptimierungsproblem in* **N***ormalform (LON). Kurzformulierung*

$$c^{\mathsf{T}}\hat{x} = \max\{c^{\mathsf{T}}x :\ x \in M\}, \qquad M := \{x \in \mathbf{R}^n :\ Ax = b,\, x \geq 0\}. \tag{3.1.1}$$

Die Menge M wird die der zulässigen Punkte *genannt.*

Andere lineare Probleme können in die Form (3.1.1) übergeführt werden. Außer trivialen Umformungen (Multiplikation mit -1) kommen auch in Frage:

— Ungleichungen als Nebenbedingungen $\alpha_1 x_1 + \ldots + \alpha_n x_n \leq \beta$ werden durch Einführung einer *Schlupfvariablen* y ersetzt durch

$$\alpha_1 x_1 + \ldots + \alpha_n x_n + y = \beta, \quad y \geq 0.$$

— Variablen x ohne Vorzeichenbeschränkung werden ersetzt durch zwei nichtnegative, $x = y_1 - y_2,\ y_1, y_2 \geq 0$.

3.2 Eigenschaften und Grundlagen

Das (schrittweise) Konstruktionsverfahren zur Berechnung einer Lösung von (3.1.1) nutzt die speziellen geometrischen Eigenschaften von M aus.

Satz 3.2.1 *Die Menge M aus (3.1.1) ist konvex. Es treten folgende Fälle auf:*

a) Für $M = \emptyset$ existiert keine Lösung des LON.

b) Für $M \neq \emptyset$, M beschränkt, existiert eine Lösung des LON.

c) Für $M \neq \emptyset$ existiert eine Lösung genau dann, wenn $\sup\{c^{\mathsf{T}} x :\ x \in M\} < \infty$.

Beweis M ist konvex als Durchschnitt des affinen Unterraums $\{x \in \mathbf{R}^n :\ Ax = b\}$ und der Halbräume $H_i = \{x \in \mathbf{R}^n :\ x_i \geq 0\}$. Die Aussage a) ist trivial, b) gilt, da M kompakt und die Zielfunktion stetig ist. Teil c) wird später konstruktiv gezeigt. ∎

Definition 3.2.2 *Ein Punkt $x \in M$ heißt* Ecke, *wenn er nicht konvexe Linearkombination anderer Punkte von M ist, d.h. wenn*

$$x = \lambda y + (1-\lambda) z,\ \lambda \in (0,1),\ y, z \in M \Rightarrow x = y = z.$$

Da die Zielfunktion linear ist, kann man sich auf die Betrachtung der Ecken von M beschränken. Diese können durch jeweils m Spalten von A beschrieben werden. Dazu werden folgende Bezeichnungen benötigt.

Zu einer Indexmenge $J = \{j_1, \ldots, j_k\} \subseteq N := \{1, \ldots, n\}$ sei $|J| := k$ und

$$x_J := (x_{j_1}, \ldots, x_{j_k}), \quad A_J := (a^{(j_1)}, \ldots, a^{(j_k)}). \tag{3.2.1}$$

Bis auf weiteres sei hier angenommen, daß die Komponenten bzw. Spalten nach aufsteigenden Indizes angeordnet sind. Bei der tatsächlichen Rechnung ist diese Festlegung aber i.a. nicht mehr sinnvoll. Dabei sind $a^{(j)} := Ae^{(j)}$, $e^{(j)} := (\delta_{ij})_i$, die Spaltenvektoren der Matrix $A = (a^{(1)}, \ldots, a^{(n)})$. Umgekehrt wird zu einem zulässigen Punkt ($x \in M$) definiert

$$I(x) := \{i \in N :\ x_i > 0\}.$$

Eine besondere Bedeutung besitzen die regulären $m \times m$-Untermatrizen von A.

Definition 3.2.3 *Für $J = \{j_1, \ldots, j_m\} \subseteq N$ heißt A_J* Basis, *wenn A_J regulär ist. Zu A_J wird eine Basislösung $\bar{x}$ definiert durch $\bar{x}_J := A_J^{-1}b$, $\bar{x}_K = 0$, wobei $K = N \setminus J$. Wenn $\bar{x}$ zulässig ist (d.h. für $A_J^{-1}b \geq 0$), heißt A_J zulässige Basis.*

Der Begriff der Basis stellt den Zusammenhang her zwischen dem geometrischen Objekt *Ecke* und der algebraischen Definition der Menge M. Die zulässigen Basislösungen sind nämlich genau die Ecken von M. Hier wird allerdings nur eine Richtung dieser Aussage gezeigt.

Satz 3.2.4 *Jede zulässige Basislösung ist Ecke von M.*

Beweis Von zentraler Bedeutung in diesem Kapitel ist die folgende *Eckendarstellung* von M, bei der M von einer Ecke her parametrisiert wird. Zur Basis A_J, $J \subseteq N$, sei $K := N \setminus J$. Für jedes $x \in M$ gilt

$$b = Ax = \sum_{j=1}^{n} a^{(j)} x_j = \sum_{j \in J} a^{(j)} x_j + \sum_{j \in K} a^{(j)} x_j = A_J x_J + A_K x_K. \tag{3.2.2}$$

Multiplikation mit A_J^{-1} führt auf

$$x_J = A_J^{-1} b - A_J^{-1} A_K x_K = \bar{x}_J - A_J^{-1} A_K \, x_K. \tag{3.2.3}$$

Hier sind die Basisvariablen x_J eines Punktes $x \in M$ in Abhängigkeit der Nichtbasisvariablen x_K parametrisiert.

Zum Eckennachweis seien $\bar{x} = \lambda y + (1 - \lambda) z$, $y, z \in M$, also insbesondere $0 = \bar{x}_K = \lambda y_K + (1 - \lambda) z_K$. Da aber $y, z \geq 0$, $\lambda, 1 - \lambda \geq 0$ ist, folgt $y_K = z_K = 0 = \bar{x}_K$, aus (3.2.3) auch die Gleichheit der J-Komponenten, $y_J = \bar{x}_J - 0, z_J = \bar{x}_J - 0$, also ist $\bar{x} = y = z$ Ecke von M. ■

Da es $\binom{n}{m}$ Untermatrizen von A der Größe $m \times m$ gibt, kann es höchstens so viele Ecken von M geben, also insbesondere nur endlich viele. Das im folgenden beschriebene *Simplexverfahren* arbeitet die Ecken von M zielgerichtet ab, beginnend mit einer Startecke. Wichtig ist dabei die möglichst eindeutige Zuordnung von Basen und Ecken. Daher nennt man eine Basislösung bzw. Ecke *nicht ausgeartet*, wenn $\bar{x}_J > 0$ gilt. Denn eine ausgeartete Ecke x mit $|I(x)| < m$ kann durch verschiedene Basen erzeugt werden. Dies kann im folgenden Algorithmus auf Zyklen führen, die durch zusätzliche Maßnahmen unterdrückt werden müssen (vgl. Collatz/Wetterling). Dieser (seltene) Fall wird hier ausgeschlossen durch die Forderung, daß das LON nicht ausgeartet ist, d.h., keine ausgearteten Ecken enthält.

3.3 Das Simplex-Verfahren

Das Grundprinzip ist ein schrittweises Vorgehen in der Weise, daß in einer Ausgangsecke $\bar{x}$ mit Basis A_J eine Nachbarecke konstruiert wird, in der die Zielfunktion einen echt größeren Wert besitzt. Da M nur endlich viele Ecken besitzt, ist das Verfahren daher endlich.

Es sei also $\bar{x}$ eine nicht ausgeartete Ecke mit Basis A_J und $K := N \setminus J$, also $\bar{x}_J = A_J^{-1}b > 0$, $\bar{x}_K = 0$. Die Basisdarstellung (3.2.3) wird etwas abgekürzt, für jedes $x \in M$ gilt

$$x_J = \bar{x}_J - A_J^{-1}A_K x_K = \bar{x}_J + \sum_{j\in K} v_J^{(j)} x_j, \quad x_j \geq 0\ (j \in K), \tag{3.3.1}$$

wobei $v_J^{(j)} := -A_J^{-1}a^{(j)}$, $j \in K$. Diese Darstellung (3.3.1) gilt aber nur solange wie $x_J \geq 0$, also

$$0 \leq x_i = \underbrace{\bar{x}_i}_{>0} + \sum_{j\in K} v_i^{(j)} \underbrace{x_j}_{\geq 0},\ i \in J.$$

Für feste Indizes $i \in J$, $j \in K$, bedeutet dies

$$v_i^{(j)} \begin{cases} \geq 0 & \Rightarrow \quad \text{Forderung } x_i \geq 0 \text{ ist keine Restriktion für } x_j, \\ < 0 & \Rightarrow \quad x_j \text{ darf als Maximalwert } -\bar{x}_i/v_i^{(j)} \text{ annehmen.} \end{cases}$$

Daher führt die Vorzeichenbedingung $x \geq 0$ in (3.3.1) auf die Einschränkung

$$x_j \leq \min\{-\frac{\bar{x}_i}{v_i^{(j)}} :\ i \in J,\ v_i^{(j)} < 0\} =: \vartheta_j > 0. \tag{3.3.2}$$

Falls $v_J^{(j)} \geq 0$ ist, werde dabei $\vartheta_j = \infty$ gesetzt. Aus (3.3.2) ergibt sich, wie weit von der Ecke $\bar{x}$ aus in jede der Richtungen x_j, $j \in K$, gegangen werden kann innerhalb von M. Zur Auswahl einer Richtung, in der die Zielfunktion ansteigt, wird diese Funktion mit (3.3.1) betrachtet:

$$\begin{aligned} c^\top x &= c_J^\top x_J + c_K^\top x_K = \underbrace{c_J^\top \bar{x}_J}_{\text{alter Wert}} + \sum_{j\in K}(c_J^\top v_J^{(j)} + c_j)x_j \\ &= c^\top \bar{x} + \sum_{j\in K}\gamma_j x_j, \quad \gamma_j := c_J^\top v_J^{(j)} + c_j,\ j \in K. \end{aligned} \tag{3.3.3}$$

Die Parameter γ_j geben an, wie die Zielfunktion proportional zu x_j anwächst. Sie werden daher als *reduzierte Gewinne* (bzw. $-\gamma_j$ als reduzierte Kosten) bezeichnet. Eine Vergrößerung der Variablen x_j, $j \in K$, ist offensichtlich nur für $\gamma_j > 0$ sinnvoll. Außerdem kann an (3.3.3) folgendes abgelesen werden:

a) Wenn $\gamma_j \leq 0\, \forall j \in K$ gilt, ist $\bar{x}$ Maximallösung.

b) Gilt für ein $j \in K$ mit $\gamma_j > 0$ wegen $\vartheta_j = \infty$ an x_j keine Einschränkung, dann gibt es kein Maximum, da $\sup\{c^\top x : x \in M\} = \infty$.

c) Für $\gamma_j > 0$ gibt das Produkt $\gamma_j\vartheta_j$ die in x_j-Richtung erzielbare Vergrößerung der Zielfunktion an. Als 'beste' lokale Strategie liegt hier daher die Wahl derjenigen Richtung mit maximalem Wert von $\gamma_j\vartheta_j$ nahe. Dies erfordert aber einen hohen Rechenaufwand, wie weiter unten überlegt wird.

   Es wird nun ein Vektor $x \in M$ betrachtet, der bei maximaler Vergrößerung einer einzelnen Komponente $x_j :\ 0 \to \vartheta_j$ mit $\gamma_j > 0$ entsteht,

$$x_J^{(1)} := \bar{x}_J + v_J^{(j)}\vartheta_j, \quad x_K^{(1)} := e_K^{(j)}\vartheta_j.$$

Wenn $r \in J$ derjenige Index ist, in dem das Minimum in (3.3.2) angenommen wird, so verschwindet nach Konstruktion genau die Komponente $x_r^{(1)} = 0$. Die Indexmenge $I(x)$ der nichtverschwindenden Komponenten ändert sich beim Übergang zu $x^{(1)}$ daher gemäß

$$I(x^{(1)}) = I(\bar{x}) \setminus \{r\} \cup \{j\}. \tag{3.3.4}$$

Daher ist $x^{(1)}$ eine Nachbarecke von $\bar{x}$. Daß dazu $A_{I(x^{(1)})}$ auch eine Basis ist, wird später gezeigt. Durch die Beziehung (3.3.4) zwischen den Indexmengen kann aber auch umgekehrt das Verfahren vollständig beschrieben werden.

Praktische Durchführung

Die Berechnung aller Vektoren $v^{(j)}$ in (3.3.1) ist recht aufwendig, die Multiplikation $A_J^{-1}A_K$ kostet ca. $2m^2(n-m)$ Operationen, und ist im Falle $\gamma_j \le 0$ auch überflüssig. Der Kostenvektor γ kann vielmehr ohne Kenntnis der $v^{(j)}$ berechnet werden:

$$\gamma_j = c_j - \underbrace{(c_J^\top A_J^{-1})}_{p^\top} a^{(j)} = c_j - p^\top a^{(j)},\ j \in K.$$

Anhand dieser Werte läßt sich feststellen, welche $v^{(j)}$ überhaupt benötigt werden zu Bestimmung des lokal besten Schritts mit $\max_j \gamma_j \vartheta_j$. In der Praxis entscheidet man sich für irgendeine Richtung mit $\gamma_j > 0$ und berechnet nur das zugehörige $v^{(j)}$. Die jetzt beschriebene Verfahrensvariante wird in der Literatur das *revidierte Simplexverfahren* genannt.

In einer Ausgangsecke $x^{(0)} \in M$ wird die zugehörige Basis $A_J,\ J = I(x^{(0)})$, benötigt. Damit kann folgendes Verfahren durchgeführt werden.

Algorithmus 3.3.1 Simplex-Verfahren

Eingabe:	Zulässige Basis $A_J,\ J \subseteq N$	
Schritt 1	$x_J := A_J^{-1}b,\ K := N \setminus J,\ x_K := 0$	
2	berechne $p^\top := c_J^\top A_J^{-1},\ \gamma_j := c_j - p^\top a^{(j)},\ j \in K$.	
3	wenn $\gamma_j \le 0\,\forall j \in K\ \Rightarrow$	**STOP**, Optimum!
4	wähle $s \in K$ mit $\gamma_s > 0$, berechne $v_J^{(s)} := -A_J^{-1}a^{(s)}$.	
5	wenn $v_i^{(s)} \ge 0\,\forall i \in J\ \Rightarrow$	**STOP**, unbeschränkt!
6	Bestimme $r \in J$: $-x_r/v_r^{(s)} = \vartheta_s = \min\{-x_i/v_i^{(s)} :\ v_i^{(s)} < 0, i \in J\}$	
7	$J := J \setminus \{r\} \cup \{s\}$, gehe nach 1	

Einige Eigenschaften des Verfahrens enthält

Satz 3.3.2 *Das LON sei nicht ausgeartet, $x^{(0)} \in M$ eine Ecke mit Basis A_J.*

a) Wenn in Schritt 3 alle $\gamma_j \le 0$ sind, dann ist $x^{(0)}$ Optimallösung.

b) Es existiert keine endliche Lösung, wenn mit einem $\gamma_s > 0$ *auch* $v_J^{(s)} \geq 0,\ s \in K$.

c) Wenn in Schritt 6 gilt $\vartheta_s = -x_r/v_r^{(s)} < \infty$*, dann wird mit* $J^{(1)} := J \setminus \{r\} \cup \{s\}$ *durch* $A_{J^{(1)}}$ *eine zulässige Basis definiert. Für die zugehörige Ecke* $x^{(1)}$ *ist* $x_r^{(1)} = 0,\ x_s^{(1)} = \vartheta_s > 0$.

Beweis Die Aussagen a) und b) wurden schon im Anschluß an (3.3.3) behandelt. Zum Nachweis von c) betrachten wir den Strahl $x(\vartheta)$ mit

$$x_J(\vartheta) := x_J^{(0)} + \vartheta v_J^{(s)}, \quad x_K(\vartheta) := \vartheta e_K^{(s)}.$$

Offensichtlich ist $x(\vartheta_s) = x^{(1)}$, insbesondere $x_s^{(1)} = x_s(\vartheta_s) = \vartheta_s$, und

$$x_r^{(1)} = x_r^{(0)} + \vartheta_s v_r^{(s)} = x_r^{(0)} + (-x_r^{(0)}/v_r^{(s)})v_r^{(s)} = 0.$$

Dies bestätigt die Aussage zu $J^{(1)}$. Als wesentlicher Punkt ist nun noch die Regularität von $A_{J^{(1)}}$ nachzuweisen. Nach Schritt 7 wird bei $A_{J^{(1)}}$ die Spalte $a^{(r)}$ durch $a^{(s)}$ ersetzt. Bei geeigneter Einordnung heißt dies

$$\begin{aligned} A_{J^{(1)}} &= A_J + (a^{(s)} - a^{(r)}){e^{(r)}}^\mathsf{T} \qquad \Longleftrightarrow \\ A_J^{-1}A_{J^{(1)}} &= I + A_J^{-1}(a^{(s)} - a^{(r)}){e^{(r)}}^\mathsf{T}. \end{aligned} \tag{3.3.5}$$

Eine Matrix der Gestalt $I + u{e^{(r)}}^\mathsf{T}$ ist regulär für ${e^{(r)}}^\mathsf{T}u + 1 = u_r + 1 \neq 0$. Denn für beliebiges $y \in \mathbf{R}^n$ folgt aus

$$0 \overset{!}{=} (I + u{e^{(r)}}^\mathsf{T})y = y + uy_r$$

durch Betrachtung der r-ten Komponente zunächst $y_r = 0$ (für $1 + u_r \neq 0$) und damit $y = 0$. Also ist $I + u{e^{(r)}}^\mathsf{T}$ tatsächlich regulär. Speziell bei (3.3.5) lautet diese Regularitätsbedingung

$$0 \neq 1 + {e^{(r)}}^\mathsf{T}A_J^{-1}(a^{(s)} - a^{(r)}) = 1 - v_r^{(s)} - 1 = -v_r^{(s)} < 0,$$

da $A_J^{-1}a^{(r)} = e^{(r)}$. Also ist $A_{J^{(1)}}$ regulär und damit Basis. ■

Da die endlich vielen Ecken von M mit streng wachsender Zielfunktion durchlaufen werden, gilt

Satz 3.3.3 *Für ein nicht ausgeartetes Lineares Optimierungsproblem in Normalform erzeugt das Simplexverfahren, ausgehend von einer Startecke, eine endliche Folge von Ecken mit streng wachsender Zielfunktion. Das Verfahren endet mit einer optimalen Ecke oder der Aussage, daß eine solche nicht existiert.*

Durchführung des Basiswechsels

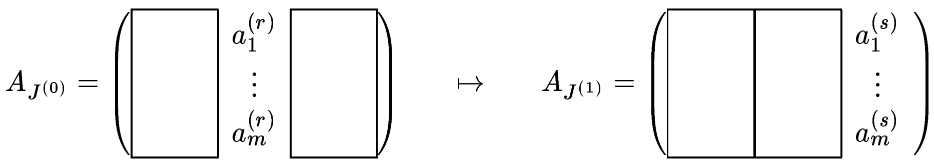
In der Diskussion wurde bisher noch nicht auf die Auflösung der drei Gleichungssysteme

$$A_J x_J = b, \quad p^\mathsf{T} A_J = c_J^\mathsf{T}, \quad A_J v_J^{(s)} = -a^{(s)},$$

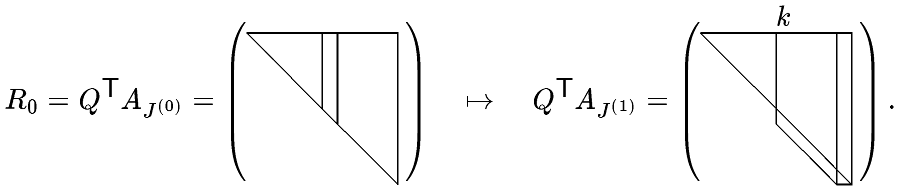
eingegangen, die in jedem Schritt des Simplexverfahrens auftreten. Betrachtete man jeden Schritt isoliert, wäre hierzu ein Aufwand von $O(m^3)$ Operationen notwendig. Allerdings unterscheiden sich die Basen $A_{J^{(0)}}$ und $A_{J^{(1)}}$ aus aufeinanderfolgenden Schritten nur wenig, da $J^{(1)} = J^{(0)} \setminus \{r\} \cup \{s\}$. Dieser Zusammenhang kann auf verschiedene Weise ausgenutzt werden. Beim (für Handrechnung) üblichen Tabellenverfahren werden die Inversen der Basismatrizen mitgeführt und ein Austauschschritt von $A_{J^{(0)}}^{-1}$ zu $A_{J^{(1)}}^{-1}$ verwendet. Hierbei können sich allerdings Rundungsfehler in unkontrollierter Weise ausbreiten.

Alternativ kann eine QR-Zerlegung der Basismatrizen mitgeführt und bei jedem Basiswechsel angepaßt werden. Dieses Verfahren ist i.a. numerisch stabiler. Abweichend von der früheren Vereinbarung wird jetzt die Freiheit in der Zuordnung der Basisindizes auf Spaltenpositionen der Matrizen A_J ausgenutzt. Daher werden die Indizes der Mengen J als m-Tupel geschrieben, d.h. angeordnet. Beim Austausch der Spalte $a^{(r)}$ durch $a^{(s)}$ in der Form

$$J^{(0)} \sim (j_1, \dots, j_k = r, \dots, j_m) \quad \mapsto \quad J^{(1)} \sim (j_1, \dots, j_{k-1}, j_{k+1}, \dots, j_m, s)$$



wird die k-te Spalte von $A_{J^{(0)}}$ gestrichen und $a^{(s)}$ als letzte Spalte neu eingefügt. Bei bekannter QR-Zerlegung von $A_{J^{(0)}} = QR_0$ ergibt sich nach dieser Operation im R-Faktor folgende Struktur.



In der letzten Spalte von $Q^\mathsf{T} A_{J^{(1)}}$ steht als neuer Vektor $Q^\mathsf{T} a^{(s)}$, ab Spalte k wurde jeweils ein neues Subdiagonalelement eingeführt. Diese $m-k$ Elemente können mit ebenso vielen unitären Zeilenumformungen (2×2-Rotationen) wieder eliminiert werden. Daher benötigt ein einzelner Simplexschritt tatsächlich nur $O(m^2)$ Operationen.

Bestimmung einer Startecke

Als Ausgangspunkt des Simplexverfahrens wird eine Ecke $x^{(0)} \in M$ benötigt. Für eine spezielle Klasse von Problemen kann eine solche direkt angegeben werden. Das Optimierungsproblem

$$By \le b,\ y \ge 0, \quad d^\mathsf{T} y \stackrel{!}{=} \max,$$

mit $B \in \mathbf{R}^{m \times k}$ geht nach Einführung von Schlupfvariablen $z \in \mathbf{R}^m$ über in die Normalform

$$By + z = b,\ y \ge 0, z \ge 0, \quad (d^\mathsf{T}, 0^\mathsf{T}) \begin{pmatrix} y \\ z \end{pmatrix} \stackrel{!}{=} \max\,.$$

In den Bezeichnungen von (3.1.1) gilt also

$$A = (B, I) \in \mathbf{R}^{m\times(k+m)},\ x = \begin{pmatrix} y \\ z \end{pmatrix},\ c = \begin{pmatrix} d \\ 0 \end{pmatrix}.$$

Bei diesem Problem ist für $b \geq 0$ der Vektor $x^{(0)} = \begin{pmatrix} 0 \\ b \end{pmatrix}$ eine zulässige Ecke mit Basis

$$A_J = I_m, \quad J = \{k+1, \dots, k+m\}.$$

Diese Eigenschaft läßt sich auch bei allgemeinen Problemen zur Konstruktion einer Startecke im Rahmen einer *Anlaufrechnung* verwenden. Dazu dient folgendes Hilfsproblem für $u \in \mathbf{R}^{n+m}$:

$$(A, I_m)u = b,\ u \geq 0, \quad -u_{n+1} - \ldots - u_{n+m} \stackrel{!}{=} \max. \tag{3.3.6}$$

Der Punkt $u^\mathsf{T} = (0, b^\mathsf{T})$ ist eine zulässige Ecke von (3.3.6), wenn $b \geq 0$. Letzteres kann aber durch geeignete Skalierung der einzelnen Gleichungen im LON erreicht werden. Eine Optimallösung von (3.3.6) existiert, da die Zielfunktion durch Null nach oben beschränkt ist. Der Wert der Hilfs-Zielfunktion in der Optimallösung $\hat{u}$ von (3.3.6) liefert folgende Information

$$-\hat{u}_{n+1} - \ldots - \hat{u}_{n+m} \begin{cases} = 0 & : \quad (\hat{u}_1, \dots, \hat{u}_n)^\mathsf{T} \text{ ist Ecke von } M, \\ < 0 & : \quad M = \emptyset. \end{cases}$$

Zusammen mit dieser Anlaufrechnung ist das Simplexverfahren also in der Lage, das LON vollständig zu behandeln.

Fazit: *Das Simplexverfahren liefert entweder in endlich vielen Schritten eine Lösung des Linearen Optimierungsproblems, oder eine Begründung dafür, weshalb eine solche nicht existiert.*

Zum *Rechenaufwand:* Obwohl Beispiele konstruiert werden können, bei denen das Simplexverfahren alle $\binom{n}{m}$ Ecken des Simplex M absucht und daher einen exponentiellen Aufwand besitzt, tritt diese Situation in praktischen Problemen kaum auf, dort arbeitet das Verfahren sehr effizient. Neuere theoretische Ergebnisse beweisen, daß das LON mit polynomiell wachsendem Aufwand gelöst werden kann, die auf dem konstruktiven Beweisprinzip beruhenden Verfahren sind allerdings (noch) keine Konkurrenz für das Simplexverfahren.

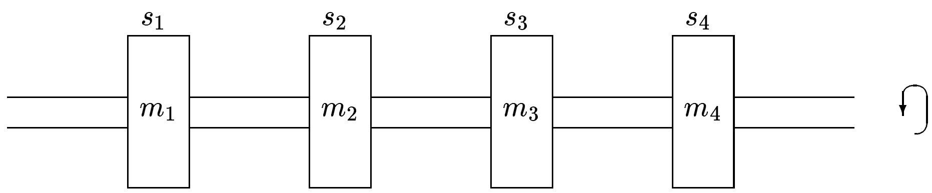
4 Eigenwertprobleme von Matrizen

Bei Systemen von gewöhnlichen Differentialgleichungen mit konstanter Koeffizientenmatrix $A \in \mathbf{R}^{n\times n}$ führt der Standardansatz

$$y'(t) = Ay(t), \text{ Ansatz } y(t) = e^{\lambda t}y_0 \ \Rightarrow \ Ay_0 = \lambda y_0.$$

auf ein Eigenwertproblem. Analog lassen sich Schwingungsprobleme in der Mechanik in der Regel über Eigenwertprobleme behandeln. Von ähnlicher Art ist folgendes Beispiel:

Eine Generatorturbine mit n Schaufelrädern kann als biegsame Achse mit n an den Stellen s_j aufgesetzten Gewichten (Massen m_j) modelliert werden.



Wenn eine Einheitskraft, angebracht an der Stelle s_j, an einer anderen Stelle s_i die Auslenkung b_{ij} verursacht und die Auslenkungen zu verschiedenen Kräften sich linear überlagern, kann die Maximal-Drehzahl $\omega_{\max}$ auf folgende Weise bestimmt werden. Eine Auslenkung der Größe x_j an der Stelle s_j erzeugt bei Rotation mit Drehzahl ω eine Kraft $m_j\omega^2 x_j$ auf die Achse. Die Achse wird dann zerstört, wenn die Kräfte insgesamt die bestehende Auslenkung aufrecht erhalten können, d.h., für

$$\left\{ \begin{array}{rcl} x_1 & = & \omega^2(b_{11}m_1x_1 + \ldots + b_{1n}m_nx_n) \\ & & \cdots \\ x_n & = & \omega^2(b_{n1}m_1x_1 + \ldots + b_{nn}m_nx_n) \end{array} \right. , \text{ d.h.,}$$

$$x = \omega^2 Ax, \quad A = (b_{ij}m_j)_{i,j}.$$

4.1 Grundlagen, Matrix-Normalformen

Definition 4.1.1 *Gegeben sei eine Matrix $A \in \mathbf{C}^{n\times n}$. Ein Wert $\lambda \in \mathbf{C}$ heißt* Eigenwert *von A, wenn ein $x \in \mathbf{C}^n$ existiert mit*

$$Ax = \lambda x, \qquad x \neq 0. \tag{4.1.1}$$

Der Vektor x heißt (Rechts-) Eigenvektor *von A zum Eigenwert λ, der Nullraum $N(A - \lambda I) = \{x : Ax = \lambda x\}$ Eigenraum, seine Dimension die (geometrische) Vielfachheit des Eigenwerts ('Anzahl der linear unabhängigen Eigenvektoren').*

Die Eigenwerte von A sind Nullstellen des *charakteristischen Polynoms*

$$\begin{aligned} p(z) &= \det(A - zI) = (-1)^n[z^n + b_{n-1}z^{n-1} + \ldots + b_0] \qquad (4.1.2)\\ &= (-1)^n(z-\lambda_1)^{\gamma_1}(z-\lambda_2)^{\gamma_2}\cdots(z-\lambda_k)^{\gamma_k}. \end{aligned}$$

Die Vielfachheit γ_j der Nullstelle λ_j heißt *algebraische Vielfachheit* von λ_j. Außer bei Vielfachheit $\gamma_1 = 1$ kann die geometrische Vielfachheit kleiner sein als die algebraische.

Beispiel 4.1.2 Es sei

$$C_m(\lambda) := \begin{pmatrix} \lambda & 1 & & 0 \\ & \lambda & 1 & \\ & & \ddots & \ddots \\ & & & \lambda \end{pmatrix} \in \mathbf{C}^{m\times m}. \tag{4.1.3}$$

Das charakteristische Polynom ist $p(z) = (\lambda - z)^m$, λ ist also m-facher algebraischer Eigenwert von $C_m(\lambda)$. Die Lösungen des Gleichungssystems

$$\Big(C_m(\lambda) - \lambda I\Big)x = \begin{pmatrix} x_2 \\ \vdots \\ x_m \\ 0 \end{pmatrix} = 0$$

besitzen die Form $x = ce^{(1)},\ c \in \mathbf{C}$, die geometrische Vielfachheit von λ ist also nur eins.

Definition 4.1.3 *Für $\lambda \in \mathbf{C},\ k \in \mathbf{N}$ und $x \in \mathbf{C}^n, c \neq 0$, gelte*

$$(A - \lambda I)^k x = 0, \quad \textit{aber } (A - \lambda I)^{k-1} x \neq 0. \tag{4.1.4}$$

Dann heißt x Hauptvektor vom Grad k zum Eigenwert λ.

Satz 4.1.4 *Haupt- und Eigenvektoren zu verschiedenen Eigenwerten sind linear unabhängig. Hauptvektoren verschiedenen Grades sind linear unabhängig. Wenn $x^{(k)}$ ein Hauptvektor vom Grad k ist, dann ist $x^{(k-j)} := (A - \lambda I)^j x^{(k)}$ ein Hauptvektor vom Grad $k - j$. Der Unterraum*

$$U := [x^{(k)}, (A - \lambda I)x^{(k)}, \ldots, (A - \lambda I)^{k-1} x^{(k)}]$$

ist ein invarianter Unterraum *von A, d.h., $AU \subseteq U$.*

Ein Basiswechsel im $\mathbf{C}^n$ entspricht einer *Ähnlichkeitstransformation* der Matrix A mit einer regulären Matrix $T \in \mathbf{C}^{n\times n}$,

$$A \mapsto TAT^{-1} =: B. \tag{4.1.5}$$

B besitzt offensichtlich die gleichen Eigenwerte wie A, während die Haupt- und Eigenvektoren transformiert werden,

$$x \text{ ist Haupt-/Eigenwert von } A \iff Tx \text{ ist Haupt-/Eigenwert von } B.$$

Ähnlichkeitstransformationen übertragen sich auf Funktionen von A, es gilt etwa

$$(TAT^{-1})^{-1} = TA^{-1}T^{-1}, \quad (TAT^{-1})^k = TA^kT^{-1}.$$

Von besonderem Interesse sind Ähnlichkeitstransformationen, die eine Matrix auf einfache Gestalt bringen, an der sich wesentliche Eigenschaften ablesen lassen.

Satz 4.1.5 (Jordan-Normalform) *Zu jeder Matrix $A \in \mathbf{C}^{n\times n}$ gibt es eine reguläre Matrix X so, daß $A = XJX^{-1}$ mit*

$$J = \begin{pmatrix} C_{m_1}(\lambda_1) & & 0 \\ & \ddots & \\ 0 & & C_{m_k}(\lambda_k) \end{pmatrix} = \begin{pmatrix} \lambda_1 & 1 & & & & & & \\ & \ddots & \ddots & & & & & \\ & & \lambda_1 & 1 & & & & \\ & & & \lambda_1 & 0 & & & \\ & & & & \lambda_2 & 1 & & \\ & & & & & \ddots & \ddots & \end{pmatrix}. \tag{4.1.6}$$

Eigenwerte in verschiedenen Jordanblöcken C_m können dabei übereinstimmen. Die Spalten von X enthalten die (Rechts-) Haupt- und Eigenvektoren von A, $X = (x^{(1)}, \ldots, x^{(n)})$. Die Zeilen von X^{-1} sind die Links- Haupt- bzw. Eigenvektoren, $(y^{(1)}, \ldots, y^{(n)})^* = X^{-1}$, daher sind diese zueinander orthogonal, $y^{(i)*}x^{(j)} = \delta_{ij}$.

Eine Matrix, bei der alle Jordanblöcke die Größe eins haben (J hat in der Nebendiagonale keine Einsen), heißt *diagonalisierbar*. Sie besitzt also n linear unabhängige Eigenvektoren. Jede Matrix mit nur einfachen Eigenwerten ist offensichtlich diagonalisierbar.

Ungeachtet ihrer theoretischen Bedeutung wird die Jordan-Normalform bei numerischen Berechnungen kaum verwendet, da sie sehr empfindlich auf Störungen der Matrix reagiert. So ist etwa die Matrix

$$C_2(1) = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

in Jordan-Normalform. Wird aber eines der Hauptdiagonalelemente geringfügig abgeändert, dann ist die Matrix diagonalisierbar und ihre Jordan-Normalform hat eine Null über der Diagonalen. Eine 'stabilere' Normalform ist die folgende .

Satz 4.1.6 (Schur-Normalform) *Jede Matrix $A \in \mathbf{C}^{n\times n}$ kann durch eine unitäre Matrix U, $U^* = U^{-1}$, auf obere Dreieckgestalt transformiert werden,*

$$A = USU^*, \quad S = \begin{pmatrix} \lambda_1 & s_{12} & \ldots & s_{1n} \\ & \lambda_2 & & s_{2n} \\ & & \ddots & \vdots \\ & & & \lambda_n \end{pmatrix}. \tag{4.1.7}$$

In der Hauptdiagonalen von S stehen die Eigenwerte in beliebiger Reihenfolge.

Beweis Mit einer QR-Zerlegung der Matrix $X = UR$ in der Jordan-Normalform zeigt sich

$$A = XJX^{-1} = U(RJR^{-1})U^*.$$

Da $RJR^{-1} =: S$ als Produkt oberer Dreieckmatrizen wieder obere Dreieckmatrix ist, deren Hauptdiagonalelemente die von J sind, folgt die Behauptung. ■

Definition 4.1.7 *Eine Matrix A heißt* normal, *wenn*

$$A^*A = AA^*. \tag{4.1.8}$$

Spezielle normale Matrizen sind die reell symmetrischen ($A^T = A$), die hermiteschen ($A^* = A$), schief-hermiteschen ($A^* = -A$) und unitären ($A^* = A^{-1}$).

Satz 4.1.8 *a) Eine Matrix ist genau dann normal, wenn sie unitär diagonalisierbar ist.*

b) Eine normale Matrix besitzt eine Orthonormalbasis aus Eigenvektoren.

c) Die Eigenwerte einer hermiteschen Matrix sind reell, die einer schief-hermiteschen rein imaginär, die einer unitären liegen auf dem Einheitskreis.

Beweis Trivial bei Einsatz der Schur-Normalform.

Für quantitative Aussagen werden meist Normabschätzungen herangezogen, mit der Schur-Normalform ergibt sich in der Spektralnorm $\|.\|_2$

$$\|A\|_2 = \|USU^*\|_2 = \|S\|_2. \tag{4.1.9}$$

Für normale Matrizen ist daher $\|A\|_2 = \varrho(A) = \max_{i=1}^n |\lambda_i|$. Für allgemeine Matrizen sei an Satz 4.2.1 aus der Numerik I erinnert:

Satz 4.1.9 *a) Für jede Matrixnorm $\|\cdot\|$ gilt $\rho(A) \le \|A\|$.*

b) Für jede Matrix A und jedes $\varepsilon > 0$ existiert eine (spezielle) Norm mit

$$\|A\|_M \le \rho(A) + \varepsilon. \tag{4.1.10}$$

c) Für diagonalisierbare Matrizen A kann in (4.1.10) $\varepsilon = 0$ gewählt werden.

Beispiel 4.1.10 Die wichtigsten Verfahren zur Eigenwert-Bestimmung verwenden (implizit) hohe Potenzen A^k von A. Für

$$C_2(\lambda) = \begin{pmatrix} \lambda & 1 \\ 0 & \lambda \end{pmatrix} \quad \text{ist} \quad C_2^k = \begin{pmatrix} \lambda^k & k\lambda^{k-1} \\ 0 & \lambda^k \end{pmatrix}.$$

Hier sieht man, daß bei $\lambda \neq 0$ die Norm nicht durch $|\lambda|^k$ beschränkt ist, mit $x = e^{(2)}$ etwa gilt

$$|\lambda|^{-k}\|C_2^k x\|_2 = |\lambda|^{-k}|\lambda^{k-1}|\sqrt{k^2 + |\lambda|^2} = \sqrt{1 + k^2/|\lambda|^2} \to \infty \ (k \to \infty).$$

In der Schranke aus Satz 4.1.9 ist also $\varepsilon > 0$ zu setzen.

4.2 Reduktion auf einfachere Gestalt

Die wichtigsten Verfahren zur Behandlung des Eigenwertproblems erzeugen iterativ eine Ähnlichkeitstransformation der Matrix auf obere Dreieck- oder Diagonalgestalt. Der Rechenaufwand pro Iterationsschritt kann sich dabei ganz erheblich verringern, wenn die Struktur der Ausgangsmatrix der angestrebten nahekommt.

Definition 4.2.1 *Die Matrix $A = (a_{ij}) \in \mathbf{C}^{n\times n}$ ist eine*

$$\begin{array}{lcll} \textit{Hessenberg-Matrix} & \Longleftrightarrow & a_{ij} = 0 \text{ für } & j < i-1, \\ \textit{Tridiagonal-Matrix} & \Longleftrightarrow & a_{ij} = 0 \text{ für } & |i-j| > 1. \end{array}$$

Ausführlich:

$$\underset{\text{Hessenberg-Form}}{\begin{pmatrix} a_{11} & a_{12} & \cdot & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdot & & a_{2n} \\ 0 & a_{32} & a_{33} & & \\ & & \ddots & \ddots & \vdots \\ 0 & & & a_{n,n-1} & a_{nn} \end{pmatrix}} \quad \underset{\text{Tridiagonal-Gestalt}}{\begin{pmatrix} a_{11} & a_{12} & 0 & \cdots & 0 \\ a_{21} & a_{22} & a_{23} & & \\ & \ddots & \ddots & \ddots & \\ 0 & & \cdot & \cdot & a_{n-1,n} \\ 0 & \cdots & 0 & a_{n,n-1} & a_{nn} \end{pmatrix}}$$

Satz 4.2.2 *Zu jeder Matrix $A \in \mathbf{C}^{n\times n}$ gibt es eine unitäre Matrix U so, daß $B := UAU^*$ Hessenberg-Form besitzt. U kann als Produkt von Householder-Spiegelungen (2.3.2) dargestellt werden, $U = H^{(n-2)} \cdots H^{(1)}$. Der Rechenaufwand für diese Transformation beträgt $\frac{10}{3}n^3$ Operationen. Wenn A symmetrisch ist, dann ist dies auch B, also tridiagonal. Der Aufwand reduziert sich dann auf $\frac{4}{3}n^3$ Operationen.*

Bemerkung: A kann auch durch eine untere Dreieckmatrix L auf Hessenbergform LAL^{-1} transformiert werden. Diese Methode erfordert nur den halben Rechenaufwand der unitären Reduktion und kann mit Hilfe von Pivotisierungen auch numerisch stabil durchgeführt werden. Allerdings führt dieses Verfahren nicht auf die Schur-Normalform von A bei Anwendung der später folgenden Verfahren (QR-Iteration).

Beweis des Satzes durch Konstruktion: Der einzige Unterschied zum Vorgehen bei der QR-Zerlegung ist die Einschränkung, daß auf $A =: A^{(1)}$ ausschließlich Ähnlichkeitstransformationen anzuwenden sind,

$$A^{(j+1)} := H^{(j)} A^{(j)} H^{(j)}, \quad H^{(j)} = I - 2u^{(j)} u^{(j)*}. \tag{4.2.1}$$

Damit bei der Rechts-Multiplikation mit $H^{(j)}$ die gerade eliminierte j-te Spalte von $H^{(j)}A^{(j)}$ nicht wieder aufgefüllt wird, ist auch $u_j^{(j)} = 0$ zu wählen. Es sei jetzt

$$A^{(j)} = \left(\begin{array}{cccc|ccc} * & \cdots & & * & * & \cdots & * \\ * & \cdot & & & \cdot & \cdot & \cdot \\ & \ddots & \ddots & \vdots & \vdots & & \vdots \\ 0 & & * & * & * & \cdots & * \\ \hline & & & a_{j+1,j}^{(j)} & * & \cdots & * \\ 0 & & & \vdots & \vdots & & \vdots \\ & & & a_{nj}^{(j)} & * & \cdots & a_{nn}^{(j)} \end{array}\right) \begin{array}{l} \left.\vphantom{\begin{array}{c} * \\ * \\ \vdots \\ 0 \end{array}}\right\} j \\ \left.\vphantom{\begin{array}{c} a \\ \vdots \\ a \end{array}}\right\} n-j \end{array} \in \mathbf{C}^{n\times n}$$

Mit Lemma 2.3.2 wird eine Spiegelung (4.2.1) bestimmt mit

$$u^{(j)} = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ u_{j+1}^{(j)} \\ \vdots \\ u_n^{(j)} \end{pmatrix} \quad \text{so, daß} \quad H^{(j)} \begin{pmatrix} * \\ \vdots \\ a_{jj}^{(j)} \\ a_{j+1,j}^{(j)} \\ \vdots \\ a_{nj}^{(j)} \end{pmatrix} = \begin{pmatrix} * \\ \vdots \\ a_{jj}^{(j)} \\ * \\ 0 \\ \vdots \end{pmatrix}.$$

Dann wird diese j-te Spalte beim Übergang zu $A^{(j+1)} = H^{(j)}A^{(j)}H^{(j)}$ nicht mehr verändert. Nach $n-2$ Schritten ist $A^{(n-1)} =: B$ in Hessenbergform. Die Vektoren können wieder, i.w., in den freien Plätzen von B untergebracht werden.

Aufwand: Wie bei Satz 2.3.3 sind nur die Matrix-Umformungen relevant. Die erste,

$$\tilde{A}^{(j)} := H^{(j)}A^{(j)} = A^{(j)} - \underbrace{2u^{(j)}\cdot}_{2(n-j)^2} [\underbrace{u^{(j)*}A^{(j)}}_{2(n-j)^2}]$$

benötigt $4(n-j)^2$ Operationen, die zweite

$$A^{(j+1)} = \tilde{A}^{(j)}H^{(j)} = \tilde{A}^{(j)} - \underbrace{2[\tilde{A}^{(j)}u^{(j)}]}_{2n(n-j)} \underbrace{\cdot u^{(j)*}}_{2(n-j)n}$$

$4n(n-j)$ Operationen, da $\tilde{A}^{(j)}u^{(j)}$ vollbesetzt ist. Der Gesamtaufwand ist somit

$$4\sum_{j=1}^{n-2}(n-j)^2 + 4\sum_{j=1}^{n-2} n(n-j) = \frac{4}{3}n^3 + 2n^3 + \ldots = \frac{10}{3}n^3 + \mathcal{O}(n^2).$$

Die Symmetrie von B folgt aus der von A: $A = A^* \Rightarrow B^* = UA^*U^* = UAU^* = B$. ■

4.3 Das unsymmetrische Eigenwertproblem

Die hier behandelten Verfahren, Vektor-, orthogonale und QR-Iteration, arbeiten nach einem gemeinsamen Prinzip, einer Verallgemeinerung der zunächst behandelten Vektoriteration. Da in der Praxis unterschiedliche Fragestellungen auftreten können, sind alle drei Varianten von Interesse. Das allgemeinste Verfahren ist die QR-Iteration, mit der die volle Schur-Normalform (alle Eigenwerte) berechnet werden kann. Oft spielen aber nur bestimmte Eigenwerte eine wichtige Rolle, z.B., die größten, die (betrags-)kleinsten oder die einer Stelle $\hat{\lambda}$ am nächsten gelegenen (im Turbinenbesipiel die bei der Solldrehzahl liegenden kritischen Drehzahlen). Hier ist die Vektor- oder orthogonale Iteration vorzuziehen, vor allem dann, wenn die Matrix A groß und dünnbesetzt ist, da diese Verfahren A nur über Matrix-Vektor-Multiplikationen verwenden.

Die Vektoriteration beruht darauf, daß sich in der Folge $\{A^k\}_k$ der betragsgrößte Eigenwert, λ_1^k, gegen die anderen durchsetzt. Bei der numerischen Berechnung hoher Potenzen A^k sind allerdings Vorsichtsmaßnahmen zu treffen.

Die Vektor-Iteration (nach von Mises)

Ausgehend von $0 \neq z^{(0)} \in \mathbf{C}^n$ wird eine Vektorfolge berechnet durch

$$\left.\begin{array}{lr} q^{(k-1)} := z^{(k-1)}/|z_{i_k}^{(k-1)}| \quad \text{wobei } |z_{i_k}^{(k-1)}| = \|z^{(k-1)}\|_\infty & \text{(Normierung)} \\ z^{(k)} := Aq^{(k-1)}, & \text{(Iteration)} \\ \lambda^{(k)} := z_{i_k}^{(k)}/q_{i_k}^{(k-1)} & \end{array}\right\} k = 1, 2, \ldots \quad (4.3.1)$$

Bemerkung: a) $q^{(k)}$ ist aus praktischen Gründen auf eins normiert, stimmt aber bis auf die Normierung mit dem Vektor $A^k z^{(0)}$ überein,

$$q^{(k)} = v^{(k)}/\|v^{(k)}\|_\infty, \quad v^{(k)} := A^k z^{(0)}. \tag{4.3.2}$$

b) Bei Iterationsverfahren $x^{(k+1)} := Ax^{(k)} + b$ zur Lösung Linearer Gleichungssysteme $(I - A)\hat{x} = b$ gilt für den Fehler $v^{(k)} := x^{(k)} - \hat{x}$ und den Defekt $d^{(k)} := (I - A)x^{(k)} - b = (I - A)v^{(k)}$ die gleiche Beziehung, $v^{(k)} = Av^{(k-1)} = \ldots = A^k v^{(0)}$. Daher ist der nächste Satz auch in diesem Zusammenhang von grundlegender Bedeutung (vgl. §5).

c) Die Bestimmung von $\lambda^{(k)}$ ind (4.3.1) muß auf eine etwas komplizierte Weise erfolgen, da hierbei gleiche Komponenten von $z^{(k)}$ und $q^{(k-1)}$ verglichen werden müssen. Zahlenbeispiel:

$$A := \begin{pmatrix} 0 & 1 & -3 \\ 1 & 2 & -1 \\ -3 & -1 & 0 \end{pmatrix}, \; z^{(0)} := \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad x^{(1)} = \begin{pmatrix} 1 \\ 1 \\ -1 \end{pmatrix}, \; \lambda_1 = 4, \; \lambda_2 = -3.$$

Vektoriteration, die betragsmaximale Komponente ist unterstrichen:

$k =$	1	2	3	...	10	11	12	13	14
	0	$\underline{3.33}$	0.8		$\underline{3.898}$	3.455	$\underline{3.940}$	3.682	$\underline{3.966}$
$z^{(k)}$	1	1.66	2.1		3.595	3.688	3.762	3.819	3.862
	$\underline{-3}$	-0.33	$\underline{-3.5}$		-3.291	$\underline{-3.922}$	-3.583	$\underline{-3.955}$	-3.759
$\lambda^{(k)}$	0	0.33	0.8		3.291	3.455	3.583	3.682	3.759

Offensichtlich konvergieren $z^{(k)} \to \lambda_1 x^{(1)}$, $\lambda^{(k)} \to \lambda_1$, allerdings recht langsam $|\lambda^{(k)} - \lambda_1|/|\lambda^{(k-1)} - \lambda_1| \cong \frac{3}{4} = |\lambda_2/\lambda_1|$.

Da Eigenvektoren nur bis auf Konstanten bestimmt sind, müssen in der folgenden Konvergenzaussage die Vektoren $q^{(k)}$ und $x^{(1)}$ gleichartig normiert werden.

Satz 4.3.1 *Für die Eigenwerte der Matrix A gelte $|\lambda_1| > |\lambda_2| \geq \ldots \geq |\lambda_n|$. An dem Startvektor $z^{(0)}$ sei der Eigenvektor $x^{(1)}$ beteiligt, d.h., $z^{(0)} = \alpha_1 x^{(1)} + \ldots$ mit $\alpha_1 \neq 0$. Dann konvergiert die Vektor-Iteration in dem Sinn, daß eine skalare Folge $\sigma_k \in \mathbf{C}$ existiert mit*

$$\sigma_k q^{(k)} \to x^{(1)}, \quad \lambda^{(k)} \to \lambda_1, \quad k \to \infty.$$

Für eine quantitative Aussage sei $0 < \varepsilon < |\lambda_1| - |\lambda_2|$. Zu einem festen Index i mit $x_i^{(1)} \neq 0$ sei $\sigma_k = x_i^{(1)}/q_i^{(k)}$. Dann existieren Konstanten c, k_0 so, daß

$$\left.\begin{array}{c} |\lambda^{(k)} - \lambda_1| \\ \|\sigma_k q^{(k)} - x^{(1)}\| \end{array}\right\} \leq c \left(\frac{|\lambda_2| + \varepsilon}{|\lambda_1|} \right)^k, \ k \geq k_0.$$

Beweis Zur Vereinfachung des Beweises wird zusätzlich die Diagonalisierbarkeit von A vorausgesetzt. Der Startvektor besitze die Entwicklung

$$z^{(0)} = \sum_{j=1}^{n} \alpha_j x^{(j)}, \ \alpha_1 \neq 0.$$

Dann gilt für die Vektoren $v^{(k)}$ aus (4.3.2) die Darstellung

$$v^{(k)} = \sum_{j=1}^{n} \lambda_j^k \alpha_j x^{(j)} = \lambda_1^k \alpha_1 \Big(x^{(1)} + \sum_{j=2}^{n} \frac{\alpha_j}{\alpha_1} \Big(\frac{\lambda_j}{\lambda_1} \Big)^k x^{(j)} \Big). \tag{4.3.3}$$

Mit dem normierten Vektor $q^{(k-1)} := v^{(k-1)}/\|v^{(k-1)}\|$ folgt dann

$$\lambda^{(k)} = \frac{z_{i_k}^{(k)}}{q_{i_k}^{(k-1)}} = \frac{v_{i_k}^{(k)}}{v_{i_k}^{(k-1)}} = \lambda_1 \frac{x_{i_k}^{(1)} + \sum_{j=2}^{n} \frac{\alpha_j}{\alpha_1} \big(\frac{\lambda_j}{\lambda_1} \big)^k x_{i_k}^{(j)}}{x_{i_k}^{(1)} + \sum_{j=2}^{n} \frac{\alpha_j}{\alpha_1} \big(\frac{\lambda_j}{\lambda_1} \big)^{k-1} x_{i_k}^{(j)}}$$

Nach Konstruktion ist $|q_{i_k}^{(k-1)}| = 1$, da $(\lambda_j/\lambda_1)^k \to 0 \, \forall j > 1$ ist daher auch $x_{i_k}^{(1)} \neq 0$ für $k \geq k_0$. Daher folgt $|\lambda^{(k)} - \lambda_1| \leq c |\frac{\lambda_2}{\lambda_1}|^k, \ k \geq k_0$.

Für $\sigma := \lambda_1/|\lambda_1| \neq 1$ ist asymptotisch $q^{(k)} \simeq \sigma^k x^{(1)}$. Konvergenz gilt also nur für die von $q^{(k)}$ aufgespannten Unterräume. Eine übliche Konvergenzaussage kann nach Einfrieren einer Komponente gemacht werden. Für $x_i^{(1)} \neq 0$ gilt auch $v_i^{(k)} \neq 0$ für $k \geq k_0$ nach (4.3.3). Daher ist

$$\frac{q^{(k)}}{q_i^{(k)}} = \frac{v^{(k)}}{v_i^{(k)}} = \frac{x^{(1)} + \sum_{j=2}^{n} \frac{\alpha_j}{\alpha_1} \big(\frac{\lambda_j}{\lambda_1} \big)^k x^{(j)}}{x_i^{(1)} + \sum_{j=2}^{n} \frac{\alpha_j}{\alpha_1} \big(\frac{\lambda_j}{\lambda_1} \big)^k x_i^{(j)}} = \frac{x^{(1)}}{x_i^{(1)}} + \mathcal{O}\Big(|\frac{\lambda_2}{\lambda_1}|^k \Big).$$

Bei einer nicht diagonalisierbaren Matrix kann das Wachstum der nichttrivialen Jordanblöcke nach einer Aufspaltung

$$J = X^{-1}AX = \begin{pmatrix} \lambda_1 & 0 \\ 0 & M \end{pmatrix}$$

mit Satz 4.1.9 abgeschätzt werden. Dann tritt in der Aussage tatsächlich ein $\varepsilon > 0$ auf. ■

Der zum betragsgrößten Eigenwert λ_1 gehörige Eigenvektor $x^{(1)}$ setzt sich im Verlauf der Iteration also gegenüber den anderen Eigenvektoren durch. Daher werden λ_1 bzw. $x^{(1)}$ als *dominanter* Eigenwert bzw. Eigenvektor bezeichnet. Die Voraussetzung $\alpha_1 = y^{(1)*}x^{(1)} \neq 0$ kann oft aus theoretischen Gründen garantiert werden. Aber selbst ohne Vorkenntnis wird durch Rundungsfehler i.a. ein Beitrag mit $\alpha_1 \neq 0$ eingeschleppt.

Die Konvergenz der einfachen Vektoriteration ist nur linear mit dem Konvergenzfaktor $|\lambda_2/\lambda_1|$. Wenn λ_1 und λ_2 nur geringe Betragsunterschiede aufweisen, konvergiert die Iteration nur sehr langsam. Folgende Beobachtungen führen aber zu merklichen Verbesserungen:

- Die Eigenwerte der Matrix $A - \lambda I$ sind $\lambda_j - \lambda$. Eine solche *Spektralverschiebung* kann die Konvergenz von (4.3.1) verbessern, z.B., wenn $\lambda_1\lambda_2 < 0$, oder $0 < \ldots \leq \lambda_2 < \lambda_1$ bei reellen Eigenwerten.

- Die Eigenwerte von A^{-1} sind λ_j^{-1}, die von $(A - \lambda I)^{-1}$ also $(\lambda_j - \lambda)^{-1}$. Der etwa zur Berechnung von λ_1 relevante Quotient

  $$\max_{j \neq 1} \left| \frac{\lambda_1 - \lambda}{\lambda_j - \lambda} \right|$$

  wird bei geeigneter Wahl von λ beliebig klein.

Dies nutzt folgendes Verfahren aus.

Inverse Iteration (nach Wielandt)

$\lambda \in \mathbf{C}$ sei kein Eigenwert von A, mit $z^{(0)} \in \mathbf{C}^n$ wird iterativ berechnet

$$\left.\begin{array}{ll} q^{(k-1)} := z^{(k-1)}/|z_{i_k}^{(k-1)}| & \text{wobei } |z_{i_k}^{(k-1)}| = \|z^{(k-1)}\|_\infty \\ z^{(k)} := (A - \lambda I)^{-1} q^{(k-1)}, & \\ \lambda^{(k)} := \lambda + q_{i_k}^{(k-1)}/z_{i_k}^{(k)} & \end{array}\right\} k = 1, 2, \ldots \qquad (4.3.4)$$

Für die Konvergenz dieses Verfahrens gilt

Satz 4.3.2 *Der Eigenwert λ_m von A sei in Bezug auf λ isoliert, d.h., es gelte*

$$|\lambda_m - \lambda| < |\lambda_j - \lambda| \; \forall j \neq m.$$

Wenn der Eigenvektor $x^{(m)}$ an $z^{(0)}$ beteiligt ist, also $y^{(m)*}z^{(0)} \neq 0$ gilt, dann konvergieren die Werte $\lambda^{(k)}$ gegen λ_m und (im verallgemeinerten Sinn) die Vektoren $q^{(k)}$ gegen $x^{(m)}$. Quantitativ gilt für jedes $\varepsilon > 0$ mit $|\lambda_m - \lambda| + \varepsilon < |\lambda_j - \lambda|\ \forall j \neq m$ eine Schranke

$$\left.\begin{array}{r} |\lambda^{(k)} - \lambda_m| \\ \min_{\sigma\in\mathbf{C}} \|\sigma q^{(k)} - x^{(m)}\| \end{array}\right\} \leq c\left(\frac{|\lambda_m - \lambda|}{\min_{j\neq m}|\lambda_j - \lambda| - \varepsilon}\right)^k, \quad k \geq k_0.$$

Durch geeignete Wahl von λ läßt sich also gezielt ein einzelner Eigenwert der Matrix bestimmen. Zur weiteren Beschleunigung der Konvergenz kann in (4.3.4) für λ jeweils die aktuellste Eigenwert-Schätzung, $\lambda^{(k-1)}$, zur Spektralverschiebung verwendet werden:

$$z^{(k)} := (A - \lambda^{(k-1)}I)^{-1}q^{(k-1)}.$$

Bei dieser Variante ist nun in jedem Schritt eine LR- oder QR-Zerlegung durchzuführen. Daher sollte hier die Matrix vorher in Hessenbergform gebracht werden, da jede Zerlegungen dann nur einen $\mathcal{O}(n^2)$-Aufwand erfordern. Diese Variante der Inversen Iteration hat in gutartigen Situationen hervorragende Eigenschaften (vgl. symmetrisches Eigenwertproblem). In schwierigen Fällen kann es aber vorkommen, daß nur der erste Schritt eine Verbesserung bringt. Die wichtigste Einsatzmöglichkeit der Inversen Iteration ist die Berechnung einzelner Eigenvektoren, wenn Eigenwert-Approximationen mit einem anderen Verfahren (z.B. QR-Verfahren, Bisektion) geliefert wurden. Zu den problematischen Fällen gilt

Beispiel 4.3.3 Es sei $A = \begin{pmatrix} 1 & 1 \\ \varepsilon & 1 \end{pmatrix}$ mit kleinem ε, z.B., $\varepsilon = 10^{-10}$. Eine solche Matrix kann etwa bei Maschinenrechnung anstelle eines 2×2-Jordanblocks entstehen. Sie hat die Eigenwerte $1 \pm \sqrt{\varepsilon}$ und die Eigenvektoren $x^{(1/2)} = \begin{pmatrix} 1 \\ \pm\sqrt{\varepsilon} \end{pmatrix}$. Als Eigenwert-Näherung der Inversen Iteration werde $\lambda = 1$ verwendet. Dann ist

$$(A - \lambda I)^{-1}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ \varepsilon & 0 \end{pmatrix}^{-1}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} x_2/\varepsilon \\ x_1 \end{pmatrix}.$$

Für die 'meisten' Startvektoren $q^{(0)} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$, die mit $\varepsilon|x_1| < |x_2|$, ist $q^{(1)} = \begin{pmatrix} 1 \\ \varepsilon x_1/x_2 \end{pmatrix}$ eine gute Näherung der Eigenvektoren, denn $\|Aq^{(1)} - \lambda q^{(1)}\| = \varepsilon\sqrt{1 + |x_1/x_2|^2}$ ist klein. Dagegen ist die nächste Iterierte $z^{(2)} = \begin{pmatrix} x_1/x_2 \\ 1 \end{pmatrix}$ wieder weit davon entfernt, da $\|Az^{(2)} - \lambda z^{(2)}\| = \|q^{(1)}\| = 1$.

Die Vektor-Iteration (in der besprochenen Form) ist nur bei einfachen Eigenwerten mit genügend großem gegenseitigen Abstand praktisch verwendbar. Zur Behandlung mehrfacher, oder eng zusammenliegender Eigenwerte muß die Iteration gleichzeitig mit mehreren Vektoren durchgeführt werden. Bei der Vektor-Iteration wird die im Vektor $z^{(k)}$ enthaltene Information aufgetrennt in einen Richtungsvektor $q^{(k)}$ mit Norm eins und einen Multiplikator, die Eigenwert-Schätzung $\lambda^{(k)}$. Bei Iteration mit $p > 1$ Vektoren, d.h., mit Matrizen $Z^{(k)} \in \mathbf{C}^{n\times p}$, in der Form $Z^{(k)} = AQ^{(k-1)}$ kann diese Aufspaltung

$$z^{(k)} = q^{(k)}\|z^{(k)}\| \quad \text{durch die QR-Zerlegung} \quad Z^{(k)} = Q^{(k)}R^{(k)}$$

verallgemeinert werden. In Satz 2.3.3 war dabei Q als quadratische und R als $n \times p$-Matrix in Dreieckform betrachtet worden. Da aber die letzten $n-p$ Zeilen von R verschwinden, ist es hier ($p \ll n$) geschickter, R als quadratische $p \times p$-Matrix und Q als $n \times p$-Matrix zu betrachten. Zur Vermeidung von Doppelbezeichnungen (vgl. QR-Iteration) wird der unitäre Faktor in diesem Verfahren U genannt. Die so verallgemeinerte Vektor-Iteration bietet außerdem einen einfachen Zugang zum wichtigsten Eigenwert-Verfahren, der QR-Iteration.

Orthogonale Iteration

Ausgehend von einer unitären Matrix $U^{(0)} \in \mathbf{C}^{n\times p}$ werden Matrizen $Z^{(k)}, U^{(k)} \in \mathbf{C}^{n\times p}$, $R^{(k)} \in \mathbf{C}^{p\times p}$ berechnet nach

$$\left.\begin{array}{rcll} Z^{(k)} & := & AU^{(k-1)} & \\ U^{(k)}R^{(k)} & := & Z^{(k)} & \text{(QR-Zerlegung)} \end{array}\right\} k = 1, 2, \ldots \tag{4.3.5}$$

Da die Spalten von $U^{(k)}$ zueinander orthogonal sind, können sie i.a. natürlich nicht alle gegen Eigenvektoren von A konvergieren. Es zeigt sich aber, daß der von ihnen aufgespannte Unterraum $R(U^{(k)})$ gegen den *invarianten Unterraum* H_p von A mit $AH_p \subseteq H_p$ konvergiert, der von den Eigenvektoren zu den p betragsgrößten Eigenwerten aufgespannt wird, $H_p = [x^{(1)}, \ldots, x^{(p)}]$. Daher heißt (4.3.5) auch Unterraum-Iteration. Die orthogonale Iteration ist vor allem dann effektiv, wenn nur ein Teil ($p \ll n$) der Eigenwerte einer großen Matrix berechnet werden soll. Wenn A dünn besetzt ist, kann dies bei der Multiplikation AU direkt ausgenutzt werden. Dagegen zerstört das im folgenden besprochene QR-Verfahren, das eng mit (4.3.5) zusammenhängt, eine eventuell vorhandene Struktur von A.

Bemerkung: Alle bei der orthogonalen Iteration (4.3.5) auftretenden Matrizen können für beliebige $p < n$ als vordere Spalten quadratischer $n \times n$-Matrizen betrachtet werden. Denn

— beim Produkt AU sind die einzelnen Spalten voneinander völlig unabhängig,

— bei der QR-Zerlegung nach Householder hängen die j-ten Spalten von R und U nicht von den nachfolgenden Spalten von Z ab, sie werden durch die Transformationen $H^{(j+1)}, H^{(j+2)}, \ldots$ nicht mehr verändert, denn $Ue^{(j)} = H^{(1)} \cdots H^{(n-1)}e^{(j)} = H^{(1)} \cdots H^{(j)}e^{(j)}$.

Aufgrund dieser Bemerkung muß, etwa bei Konvergenzaussagen, nur der Fall $p = n$ behandelt werden. Da diese Aussagen dann für die orthogonale Iteration und das QR-Verfahren aufgrund des folgenden Zusammenhangs völlig analog sind, werden sie beim QR-Verfahren behandelt. Zu diesem kommt man zwangsläufig, wenn man (im Fall $p = n$) die Matrizen

$$S^{(k)} := U^{(k)*}AU^{(k)} \tag{4.3.6}$$

betrachtet, die die Eigenwert-Information der orthogonalen Iteration enthalten. Überlegt man nämlich, wie $S^{(k)}$ aus $S^{(k-1)}$ hervorgeht, so ergibt sich aus (4.3.5) $R^{(k)} = U^{(k)*}AU^{(k-1)}$ und daher einerseits

$$S^{(k)} = U^{(k)^*} A U^{(k-1)} U^{(k-1)^*} U^{(k)} = R^{(k)} \underbrace{U^{(k-1)^*} U^{(k)}}_{=: Q^{(k)}}, \tag{4.3.7}$$

aber auch

$$S^{(k-1)} = U^{(k-1)^*} A U^{(k-1)} = U^{(k-1)^*} Z^{(k)} = \underbrace{U^{(k-1)^*} U^{(k)}}_{Q^{(k)}} R^{(k)}.$$

Also bekommt man $S^{(k)}$ durch Berechnung einer QR-Zerlegung von $S^{(k-1)}$ und anschließender Multiplikation der beiden Faktoren in umgekehrter Reihenfolge. Dies ist

Das QR-Verfahren

Mit $S^{(0)} := A$ werden Matrizen $Q^{(k)}, S^{(k)}, R^{(k)} \in \mathbf{C}^{n\times n}$ berechnet nach

$$\left.\begin{array}{rcll} Q^{(k)}R^{(k)} & := & S^{(k-1)} & \text{(QR-Zerlegung)} \\ S^{(k)} & := & R^{(k)}Q^{(k)} & \end{array}\right\} \; k = 1, 2, \ldots \tag{4.3.8}$$

Für die Subdiagonalelemente von $S^{(k)}$ läßt sich im allgemeinsten Fall zeigen, daß bei der Anordnung $|\lambda_1| \geq \ldots \geq |\lambda_n|$ gilt

$$s_{ij}^{(k)} = \mathcal{O}\Big(\Big|\frac{\lambda_i}{\lambda_j}\Big|^k\Big), \; j < i, k \to \infty. \tag{4.3.9}$$

In der einfachen Form ist das Verfahren (4.3.8) zu aufwendig, da jeder Schritt $\mathcal{O}(n^3)$ Operationen kostet (Satz 2.3.3). Die Effizienz des Verfahrens kann durch die folgenden drei Maßnahmen dramatisch gesteigert werden.

- **Satz 4.3.4** *Besitzt $S^{(0)} = A$ Hessenbergform, dann gilt dies auch für alle Iterierten $S^{(k)}$. In diesem Fall beträgt der Aufwand für einen Schritt (4.3.8) nur noch $8n^2$ Operationen.*

  Beweis Die für die QR-Zerlegung (2.3.7) benutzten Spiegelungen $H^{(j)} = I - 2u^{(j)}u^{(j)^*}$ verwenden Vektoren $u^{(j)}$, in denen nur die j-te und $j+1$-te Komponente nicht null sind. Bei der Rückmultiplikation $S = RH^{(1)} \ldots H^{(n-1)}$ werden in S nur Elemente direkt links von der Hauptdiagonale eingeführt. ■

- **Satz 4.3.5** *Bei Spektralverschiebung*

$$\left.\begin{array}{rcl} Q^{(k)}R^{(k)} & := & S^{(k-1)} - \lambda I \\ S^{(k)} & := & R^{(k)}Q^{(k)} + \lambda I \end{array}\right\} \; k = 1, 2, \ldots \tag{4.3.10}$$

  verbessert sich die Konvergenzaussage (4.3.9) auf

$$s_{ij}^{(j)} = \mathcal{O}\Big(\Big|\frac{\lambda_i - \lambda}{\lambda_j - \lambda}\Big|^k\Big), \; j < i,$$

  mit der neuen Anordnung $|\lambda_1 - \lambda| \geq \ldots \geq |\lambda_n - \lambda|$.

Beweis Die Eigenwerte von $S^{(k)}$ werden nicht verändert, denn

$$S^{(k)} = R^{(k)}Q^{(k)} + \lambda I = Q^{(k)^*}(S^{(k-1)} - \lambda I)Q^{(k)} + \lambda I = Q^{(k)^*}S^{(k-1)}Q^{(k)}, \tag{4.3.11}$$

konvergenzbestimmend sind aber die verschobenen Eigenwerte $\lambda_j - \lambda$. ∎

- **Satz 4.3.6** *Ist S Hessenberg-Matrix mit $s_{n,n-1} = 0$, dann ist s_{nn} ein Eigenwert von S. Die restlichen Eigenwerte sind die von S_1 in*

$$S = \left(\begin{array}{c|c} & * \\ S_1 & \vdots \\ & * \\ \hline 0 \cdots 0 & s_{nn} \end{array}\right).$$

Beweis $e^{(n)}$ ist Links-Eigenvektor von S mit Eigenwert s_{nn}. Die Links-Eigen- bzw. Haupt-Vektoren von S_1 können mit einer geeigneten n-ten Komponente zu solchen von S erweitert werden. ∎

Wenn im Verlauf der Iteration das letzte Subdiagonalelement $s_{n,n-1}^{(k)}$ sehr klein wird, kann daher die QR-Iteration mit der kleineren Matrix S_1 fortgesetzt werden (*Deflation*).

Bei der praktischen Durchführung wird zur Anwendung von Satz 4.3.5 noch eine Strategie zur Wahl der (des) Spektralparameter(s) benötigt. Im allgemeinen ist die Konvergenz des letzten Diagonalelements gegen den betragskleinsten Eigenwert, $s_{nn}^{(k)} \to \lambda_n$, am schnellsten. Nach einer Anlaufphase, etwa wenn

$$\left|1 - \frac{s_{nn}^{(k-2)}}{s_{nn}^{(k-1)}}\right| < \frac{1}{3}$$

gilt, kann dieses Verhalten durch die Wahl

$$\lambda^{(k)} := s_{nn}^{(k-1)} \tag{4.3.12}$$

für λ in Satz 4.3.5 unterstützt werden.

Bei **reellen** Matrizen mit nicht-reellen Eigenwerten treten diese in **konjugiert-komplexen Paaren** auf. Wenn genau zwei Eigenwerte gleichen Betrag haben, hat $S^{(k)}$ für $k \to \infty$ die Gestalt

$$S^{(k)} \cong \begin{pmatrix} \ddots & \cdot & \cdot & \cdots \\ & * & * & \cdots \\ & * & * & \cdots \\ & & & \ddots \end{pmatrix}.$$

Auch bei reeller Rechnung können die komplexen Eigenwerte aus dem 2×2-Block

$$\begin{pmatrix} s_{pp} & s_{p,p+1} \\ s_{p+1,p} & s_{p+1,p+1} \end{pmatrix}$$

berechnet werden, wenn $s_{p+1,p}$ das auch asymptotisch nicht verschwindende Subdiagonalelement in $S^{(k)}$ ist. Die Spektralverschiebung nach Satz 4.3.5 bewirkt aber nur dann eine erhebliche Konvergenzbeschleunigung, wenn komplexe Spektralparameter λ verwendet werden. Die (aufwendige) komplexe Rechnung läßt sich dabei umgehen, wenn in zwei aufeinander folgenden Schritte konjugiert komplexe Parameter $\lambda^{(k+1)} = \bar{\lambda}^{(k)}$ verwendet werden. Dazu bieten sich an die beiden Eigenwerte von

$$\begin{pmatrix} s_{n-1,n-1}^{(k-1)} & s_{n-1,n}^{(k-1)} \\ s_{n,n-1}^{(k-1)} & s_{nn}^{(k-1)} \end{pmatrix}.$$

Man überlegt sich leicht, daß dann die Matrizen $S^{(k+1)}, Q^{(k+1)}$ wieder reell sind. Sie können mit reeller Rechnung aus $S^{(k-1)}, Q^{(k-1)}$ durch einen *QR-Doppelschritt* berechnet werden. Dieser Spezialfall wird wegen seiner Komplexität nicht behandelt.

Ein umfassender, globaler Konvergenzbeweis für das QR-Verfahren ist nur beim symmetrischen Eigenwertproblem möglich. Im unsymmetrischen Fall gilt

Satz 4.3.7 *Die Eigenwerte der Matrix A seien betragsmäßig verschieden und $|\lambda_1| > \ldots > |\lambda_n|$. Für die Matrix $Y := X^{-1}$ aus der Jordan-Normalform $A = XJX^{-1}$ existiere die LR-Zerlegung. Dann konvergieren die Matrizen $S^{(k)}$ des einfachen QR-Verfahrens (4.3.8) nach (4.3.9) gegen eine obere Dreieckmatrix, insbesondere gilt*

$$s_{ii}^{(k)} \to \lambda_i, \ k \to \infty.$$

Die Konvergenz der QR-Iteration (4.3.10) mit Spektralverschiebung (4.3.12) ist lokal quadratisch, d.h.,

$$|s_{n,n-1}^{(k+1)}| \le c|s_{n,n-1}^{(k)}|^2,$$

sobald $|s_{n,n-1}^{(k)}|$ 'klein genug' ist.

Beweis Zur Hervorhebung des grundlegenden Beweisprinzips wird der Beweis nur unter Einschränkungen und nur für das einfache Verfahren (4.3.8) geführt. Es sei A reell und $|\lambda_n| > 0$, A also regulär. Ein wesentliches Argument im Beweis ist die weitgehende Eindeutigkeit und stetige Abhängigkeit der QR-Zerlegung. Dazu sind folgende Vorbemerkungen erforderlich:

— Die QR-Zerlegung nach Satz 2.3.3 enthält eine Vorzeichenentscheidung zur Vermeidung von Rundungsfehlern. Wählt man davon abweichend in R positive Hauptdiagonalelemente ($\beta := \|x\|_2$ in (2.3.3)), dann ist die QR-Zerlegung eindeutig.

— Diese QR-Zerlegung hängt auch stetig von der zerlegten Matrix ab, z.B., gilt für die QR-Faktoren Q_k, R_k einer Folge $R + F_k = Q_k R_k$, mit einer oberen Dreieckmatrix R (positive Hauptdiagonale) und $F_k \to 0, \ k \to \infty$, die Aussage $Q_k \to I, \ R_k \to R$. Zur Unterscheidung von den Matrizen des QR-Verfahrens sind diese hier ausnahmsweise unten indiziert.

a) Zunächst seien

$$U^{(k)} := Q^{(1)} Q^{(2)} \cdots Q^{(k)}, \ k > 0, \ U^{(0)} := I, \tag{4.3.13}$$

die akkumulierten Transfomationsmatrizen der QR-Iteration (4.3.8), vgl. auch (4.3.7). Mit diesen gilt wegen (4.3.11)

$$\begin{aligned} AU^{(k-1)} &= AQ^{(1)}\cdots Q^{(k-1)} = Q^{(1)}S^{(1)}Q^{(2)}\cdots Q^{(k-1)} \\ &= Q^{(1)}\cdots Q^{(k-1)}S^{(k-1)} = U^{(k-1)}S^{(k-1)} && (4.3.14) \\ &= U^{(k-1)}Q^{(k)}R^{(k)} = U^{(k)}R^{(k)}. && (4.3.15) \end{aligned}$$

und daher

$$A^k = A^kU^{(0)} = A^{k-1}U^{(1)}R^{(1)} = \ldots = U^{(k)}\underbrace{R^{(k)}\cdots R^{(2)}R^{(1)}} =: U^{(k)}\hat{R}^{(k)}. \qquad (4.3.16)$$

$U^{(k)}$ ist also der eindeutige (s.o.) unitäre Faktor der QR-Zerlegung von A^k.

b) In der Jordan-Normalform von A sei $UR_X = X$ die QR-Zerlegung von X (U ist die unitäre Matrix der Schur-Normalform, vgl. Beweis von Satz 4.1.6) und $L_YR_Y = Y$ die LR-Zerlegung von $Y = X^{-1}$. Dann gilt auch

$$A^k = XJ^kX^{-1} = UR_XJ^kL_YR_Y = UR_X\underbrace{(J^kL_YJ^{-k})}_{=I+F_k}J^kR_Y. \qquad (4.3.17)$$

Da J diagonal ist, hat die untere Dreieckmatrix $J^kL_YJ^{-k}$ Elemente $l_{ij}(\lambda_i/\lambda_j)^k$. Die Subdiagonalen dieser Matrix verschwinden für $k\to\infty$, da $\lambda_i/\lambda_j \le q < 1$ für $j < i$. Daher gilt in (4.3.17) tatsächlich

$$J^kL_YJ^{-k} = I + F_k \quad \text{mit} \quad \|F_k\| \le cq^k \to 0,\ k\to\infty.$$

Nach den einleitenden Bemerkungen hat die Folge der QR-Zerlegungen von $R_X(I+F_k) = Q_kR_k$ die Eigenschaft $Q_k \to I, R_k \to R_X$. Als letzter Schritt sei

$$D := \operatorname{diag}(\sigma_i),\ \sigma_i := \operatorname{sign}(\lambda_i),$$

mit $D^2 = I$. Aus (4.3.17) wird nun

$$A^k = UQ_kR_kJ^kR_Y = (UQ_kD^k)(D^kR_kJ^kR_Y). \qquad (4.3.18)$$

Dabei ist $D^kR_kJ^kR_Y$ eine obere Dreieckmatrix, deren Diagonalelemente als Produkte der Diagonalen ihrer Faktoren positiv sind, und UQ_kD^k ist unitär.

c) Durch Vergleich der beiden (eindeutigen) QR-Zerlegungen (4.3.16) und (4.3.18) von A^k folgt somit

$$U^{(k)} = UQ_kD^k \quad \text{und} \quad \hat{R}^{(k)} = D^kR_kJ^kR_Y. \qquad (4.3.19)$$

Wegen $Q_k \to I$ konvergiert daher $U^{(k)}D^k \to U$, die Konvergenz von $U^{(k)}$ gegen U aus der Schur-Normalform tritt also bis auf Vorzeichen der Spalten ein. Diese Vorzeichen sind aber irrelevant, für die Matrizen $S^{(k)}$ gilt nach (4.3.14) und (4.3.19) nämlich

$$S^{(k)} = U^{(k)*}AU^{(k)} = D^k\underbrace{Q_k^*}_{\to I}\underbrace{U^*AU}_{=S}\underbrace{Q_k}_{\to I}D^k.$$

Demnach konvergiert $S^{(k)}$ gegen eine obere Dreieckmatrix, präzise gilt $D^k S^{(k)} D^k \to S$ mit der Matrix S aus der Schur-Normalform. In der Hauptdiagonale kompensieren sich die alternierenden Faktoren: $\sigma_i^k s_{ii}^{(k)} \sigma_i^k = s_{ii}^{(k)} \to s_{ii} = \lambda_i$, $k \to \infty$. ∎

Beispiel 4.3.8 Das QR-Verfahren wird mit Spektralverschiebung (4.3.12) ab Schritt 5 auf die Hessenbergmatrix

$$A = \begin{pmatrix} 3 & -2 & -0.5 & 0 & 0 \\ -2 & 3 & -2 & -0.5 & 0 \\ 0 & -2 & 3 & -2 & -0.5 \\ 0 & 0 & -2 & 3 & -2 \\ 0 & 0 & 0 & -2 & 3 \end{pmatrix}$$

angewendet. Die Tabelle zeigt die Diagonale der Matrizen $S^{(k)}$ und das Subdiagonalelement $s_{m,m-1}^{(k)}$, wobei $m \le n$ die aktuelle Dimension der Restmatrix ist (vgl. Satz 4.3.6).

k	s_{11}	s_{22}	s_{33}	s_{44}	s_{55}	$s_{m,m-1}$
1	4.846154	4.088911	3.139369	1.843980	1.081586	−1.653266
2	5.401216	4.602085	4.064473	0.835122	0.097104	0.740867
3	5.698527	4.970917	3.902994	0.882349	−0.454788	−0.965393
4	5.878893	5.162691	3.570136	0.797329	−0.409049	0.467165
5	5.991053	5.211672	3.418541	1.126695	−0.747961	−0.132251
6	6.064557	5.203066	3.356221	1.095888	−0.719731	−0.001994
7	6.114333	5.179253	3.331381	1.094180	−0.719147	−0.000001
8	6.154318	5.149222	3.322309	1.093298	−0.719147	−0.000002
9	6.171288	5.130994	3.323566	1.093298	−0.719147	−0.000000
10	6.171301	5.122938	3.331609	1.093298	−0.719147	−0.000354
11	6.156498	5.137690	3.331661	1.093298	−0.719147	−0.000000
12	6.103079	5.191109	3.331661	1.093298	−0.719147	−0.007077
13	6.098903	5.195285	3.331661	1.093298	−0.719147	−0.000032
14	6.098883	5.195305	3.331661	1.093298	−0.719147	−0.000000

Das QR-Verfahren ist **das** Standardverfahren zur Lösung des vollständigen unsymmetrischen Eigenwertproblems. Wie im Beispiel werden nach der Anlaufphase für jeden Eigenwert meist ca. 2 QR-Schritte benötigt. Daraus leitet man einen *empirischen* Mittelwert für den Rechenaufwand zur Bestimmung aller Eigenwerte (ohne Transformationsmatrix U) von $16n^3$ Operationen her, die volle Schur-Normalform (S und U) benötigt

$$\cong 30n^3 \text{Operationen.}$$

Das QR-Verfahren vereinigt in sich die anderen Iterationsverfahren. Es kann nämlich leicht gezeigt werden, daß hier u.a. sowohl eine Vektoriteration als auch eine Inverse Iteration ausgeführt werden.

a) Zunächst entspricht nach (4.3.14) ein QR-Schritt einem Schritt der orthogonalen Iteration (4.3.5) (vgl. auch (4.3.7)). Da $R^{(k)}$ obere Dreieckmatrix ist, wird somit in der ersten Spalte von $U^{(k)}$ eine Vektoriteration durchgeführt: $AU^{(k-1)}e^{(1)} = r_{11}^{(k)} U^{(k)} e^{(1)}$.

b) Bei Spektralverschiebung ändert sich (4.3.15) zu

$$AU^{(k-1)} = U^{(k)}R^{(k)} + \lambda^{(k-1)}U^{(k-1)} \quad \Longleftrightarrow \quad (A - \lambda^{(k-1)}I)U^{(k-1)} = U^{(k)}R^{(k)}.$$

Da $U^{(k)}$ unitär ist, folgt nach Inversion und Transposition

$$(A^* - \bar{\lambda}^{(k-1)}I)^{-1}U^{(k-1)} = U^{(k)}(R^{(k-1)})^{-1*}.$$

Hier ist $(R^{(k-1)})^{-*}$ ein untere Dreieckmatrix und daher gilt $(R^{(k-1)})^{-*}e^{(n)} = e^{(n)}/\bar{r}_{nn}^{(k)}$. Analog zum ersten Teil folgt

$$U^{(k)}e^{(n)} = \bar{r}_{nn}^{(k)}\,(A^* - \bar{\lambda}^{(k-1)}I)^{-1}\,(U^{(k-1)}e^{(n)}).$$

Dies ist aber ein Schritt der Inversen Iteration (4.3.4).

Das QR-Verfahren führt also in der ersten Spalte der akkummulierten Transformationsmatrizen $U^{(k)}$ eine Vektor-Iteration und in deren letzter Spalte eine Inverse Iteration durch.

Der *Vorläufer* des QR-Verfahrens war das *LR-Verfahren*

$$A^{(0)} := A, \quad L^{(k)}R^{(k)} := A^{(k-1)}, \quad A^{(k)} := R^{(k)}L^{(k)},$$

bei dem statt der QR-Zerlegung die LR-Zerlegung der Iterierten verwendet wird. Auch dieses Verfahren bewahrt die Hessenbergform von $A^{(0)}$ und erfordert pro Schritt sogar weniger Aufwand als das QR-Verfahren. Allerdings kann die Existenz der LR-Zerlegungen (ohne Pivotisierung) nicht garantiert werden, Pivotisierung kann andererseits die Konvergenz zerstören. Daher ist das Verfahren weniger robust als das QR-Verfahren.

4.4 Das symmetrische Eigenwertproblem

Für diesen Spezialfall können bei den besprochenen Verfahren verbesserte Varianten angegeben und präzisere Aussagen gemacht werden. Außerdem sind hier einige Spezialverfahren einsetzbar, die nicht oder nur schwer auf den unsymmetrischen Fall verallgemeinert werden können. Zwei Eigenschaften des hermiteschen Eigenwert-Problems

$$Ax = \lambda x, \quad A = A^*, \tag{4.4.1}$$

$0 \neq x \in \mathbf{C}^n$, erleichtern den Zugang erheblich. Eine hermitesche Matrix besitzt nämlich

- nur reelle Eigenwerte,
- eine Orthonormalbasis von Eigenvektoren.

Dies ergibt sich direkt aus der Schur-Normalform, vgl. Satz 4.1.8. Etwas direkter ist allerdings der Zugang über die quadratische Form.

Definition 4.4.1 *Zur Matrix $A \in \mathbf{C}^{n\times n}$ und $0 \neq x \in \mathbf{C}^n$ heißt*

$$R_A[x] := \frac{x^*Ax}{x^*x} \tag{4.4.2}$$

der Rayleigh-Quotient *an der Stelle x.*

Wenn x, y auf eins normierte Eigenvektoren zu verschiedenen Eigenwerten $\lambda \neq \mu$ sind, folgen die beiden Aussagen aus

$$\lambda = x^*Ax = (x^*Ax)^\mathsf{T} = \overline{x^*A^*x} = \overline{\lambda} \qquad \text{und}$$

$$0 = y^*Ax - (Ay)^*x = \lambda y^*x - \mu y^*x = (\lambda - \mu)y^*x.$$

Mit Hilfe des Rayleigh-Quotienten kann bei der Vektor-Iteration eine wesentlich bessere Eigenwert-Näherung als im unsymmetrischen Fall berechnet werden.

Satz 4.4.2 *Es sei $x \in \mathbf{C}^n$ mit $\|x\|_2 = 1$.*

a) Für eine beliebige Matrix $A \in \mathbf{C}^{n\times n}$ ist

$$\|Ax - R_A[x]x\|_2 = \min_{\lambda\in\mathbf{C}} \|Ax - \lambda x\|_2.$$

b) Wenn $A \in \mathbf{C}^{n\times n}$ normal ist, gilt für jeden Index k mit $d_k := \min_{j\neq k} |\lambda_j - \lambda_k| > 0$ die Aussage

$$\Big|R_A[x] - \lambda_k\Big| \le \frac{1}{d_k}\|Ax - \lambda_k x\|_2^2.$$

Bemerkung: Wenn x als eine Näherung für den Eigenvektor $x^{(k)}$ angesehen werden kann, weil $\hat\varepsilon := \|Ax - \lambda_k x\|$ klein ist, dann ist nach Teil a) der Rayleighquotient dazu die beste Eigenwert-Näherung. Der Fehler dieser Eigenwert-Approximation entspricht nach Teil b) bei einfachen Eigenwerten und normalen Matrizen dem Quadrat des Fehlers $\hat\varepsilon$ der Eigenvektor-Näherung x.

Beweis a) Die Aussage folgt direkt aus folgender quadratischer Ergänzung

$$\begin{aligned}\|Ax - \lambda x\|_2^2 &= \|Ax\|_2^2 - 2\Re(\bar\lambda x^*Ax) + |\lambda|^2\|x\|_2^2\\ &= |\lambda - R_A[x]|^2\|x\|_2^2 + \|Ax\|_2^2 - |R_A[x]|^2\|x\|_2^2.\end{aligned}$$

b) Da die Eigenvektoren $\{x^{(j)}\}$ einer normalen Matrix eine Orthonormalbasis bilden, gilt

$$x = \sum_{j=1}^n \xi_j x^{(j)}, \quad Ax = \sum_{j=1}^n \lambda_j\xi_j x^{(j)}, \quad R_A[x] = \sum_{j=1}^n \lambda_j|\xi_j|^2,$$

wenn $\|x\|_2^2 = \sum_{j=1}^n |\xi_j|^2 = 1$. Die Behauptung folgt aus einem Vergleich der beiden auftretenden Ausdrücke

$$\begin{aligned}\|Ax - \lambda_k x\|_2^2 &= \sum_{j\neq k} |\lambda_j - \lambda_k|^2|\xi_j|^2,\\ \Big|R_A[x] - \lambda_k\Big| &= \Big|\sum_{j\neq k}(\lambda_j - \lambda_k)|\xi_j|^2\Big| \le \frac{1}{d_k}\sum_{j\neq k}|\lambda_j - \lambda_k|^2|\xi_j|^2. \qquad \blacksquare\end{aligned}$$

Insbesondere bei hermiteschen Matrizen kann die Eigenwert-Näherung bei der *Vektor-Iteration* durch Verwendung des Rayleigh-Quotienten verbessert werden. Für die modifizierte Iteration

$$\left.\begin{aligned} z^{(k)} &:= Aq^{(k-1)},\\ \lambda^{(k)} &:= q^{(k-1)*}z^{(k)} \quad = R_A[q^{(k-1)}],\\ q^{(k)} &:= z^{(k)}/\|z^{(k)}\|_2 \end{aligned}\right\}\ k = 1, 2, \ldots \tag{4.4.3}$$

mit $\|q^{(0)}\|_2 = 1$ steht in der Konvergenzaussage

$$|\lambda^{(k)} - \lambda_1| \le c \left|\frac{\lambda_2}{\lambda_1}\right|^{2k}, \ k \ge k_0,$$

im Vergleich zu Satz 4.3.1 der doppelte Exponent $2k$. Die verbesserte Eigenwert-Schätzung wirkt sich noch viel günstiger auf die *Inverse Iteration* (4.3.4) aus, wenn der Rayleigh-Quotient bei der Spektralverschiebung eingesetzt wird. Es sei $\|q^{(0)}\|_2 = 1,\ \lambda^{(0)} := R_A[q^{(0)}]$, und

$$\left.\begin{array}{rcl} z^{(k)} & := & (A - \lambda^{(k-1)}I)^{-1} q^{(k-1)}, \\ \lambda^{(k)} & := & \lambda^{(k-1)} + z^{(k)*} q^{(k-1)} / \|z^{(k)}\|_2^2, \\ q^{(k)} & := & z^{(k)} / \|z^{(k)}\|_2 \end{array}\right\} \ k = 1, 2, \ldots \tag{4.4.4}$$

Dabei ist in der zweiten Zeile

$$\lambda^{(k)} = R_A[z^{(k)}] = \frac{z^{(k)*} A z^{(k)}}{\|z^{(k)}\|_2^2} = \frac{1}{\|z^{(k)}\|_2^2} z^{(k)*} (A - \lambda^{(k-1)} I) z^{(k)} + \lambda^{(k-1)} = \frac{z^{(k)*} q^{(k-1)}}{\|z^{(k)}\|_2^2} + \lambda^{(k-1)}.$$

Dieses Verfahren (4.4.4) hat jetzt extrem gute Konvergenzeigenschaften.

Satz 4.4.3 *Bei einer hermiteschen Matrix A konvergiert die Inverse Iteration (4.4.4) für fast alle Startvektoren $q^{(0)}$ gegen einen Eigenvektor x von A mit Eigenwert λ (globale Konvergenz). Bei einfachen Eigenwerten ist die Konvergenz in der Grenze kubisch, d.h., es gibt $k_0 \in \mathbf{N}, c \in \mathbf{R}$ und $\sigma_k \in \mathbf{C}$ so, daß für $k \ge k_0$ gilt*

$$\begin{array}{rcl} |\lambda^{(k)} - \lambda| & \le & c|\lambda^{(k-1)} - \lambda|^3 \\ \|\sigma_k q^{(k)} - x\|_2 & \le & c\|\sigma_{k-1} q^{(k-1)} - x\|_2^3 \end{array}$$

Beweis [Parlett, Math. Comp. 28(74), 679-693]

Zur Erläuterung wird ein einfacher Spezialfall und eine Querverbindung zum Newtonverfahren betrachtet.

Beispiel 4.4.4 $n = 2,\ A = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}$ (oBdA), $q^{(k)} = \begin{pmatrix} c_k \\ s_k \end{pmatrix}$ mit $c_k^2 + s_k^2 = 1$. Dann ist (vgl. Beweis von Satz 4.4.2)

$$\lambda^{(k)} = R_A[z^{(k)}] = R_A[q^{(k)}] = \lambda_1 c_k^2 + \lambda_2 s_k^2,$$

$$z^{(k+1)} = \begin{pmatrix} c_k/(\lambda_1 - \lambda^{(k)}) \\ s_k/(\lambda_2 - \lambda^{(k)}) \end{pmatrix} = \frac{1}{\lambda_1 - \lambda_2} \begin{pmatrix} c_k/s_k^2 \\ -s_k/c_k^2 \end{pmatrix},$$

da, z.B., $\lambda_1 - \lambda^{(k)} = \lambda_1(1 - c_k^2) - \lambda_2 s_k^2 = (\lambda_1 - \lambda_2) s_k^2$. Die Normierung führt dann auf

$$q^{(k+1)} = \frac{\pm 1}{\sqrt{c_k^6 + s_k^6}} \begin{pmatrix} c_k^3 \\ -s_k^3 \end{pmatrix} \longrightarrow \pm \left\{ \begin{array}{ll} e^{(1)} & \text{für } |c_0| > |s_0| \\ e^{(2)} & \text{für } |c_0| < |s_0| \end{array} \right. .$$

Außer in den Ausnahmefällen mit $|c_0| = |s_0|$ konvergiert die Inverse Iteration also kubisch gegen einen der beiden Einheits- und Eigenvektoren.

Beispiel 4.4.5 Wird zur Eigenwert-Gleichung (4.4.1) noch eine Normierungsbedingung für den Eigenvektor hinzugefügt, ergibt sich ein nichtlineares Gleichungssystem für die $n+1$ Unbekannten x, λ. Zu

$$F(x,\lambda) := \begin{pmatrix} Ax - \lambda x \\ \frac{1}{2}(1 - x^* x) \end{pmatrix} = 0, \qquad \text{ist } F'(x,\lambda) = \begin{pmatrix} A - \lambda I & -x \\ -x^* & 0 \end{pmatrix}.$$

Das Gleichungssystem eines Newton-Schritts,

$$F'\left(x^{(k)}, \lambda^{(k)}\right) \begin{pmatrix} x^{(k+1)} - x^{(k)} \\ \lambda^{(k+1)} - \lambda^{(k)} \end{pmatrix} = -F\left(x^{(k)}, \lambda^{(k)}\right) \quad \Leftrightarrow$$

$$\begin{cases} (A - \lambda^{(k)} I) x^{(k+1)} & -\lambda^{(k+1)} x^{(k)} & = -\lambda^{(k)} x^{(k)} \\ \qquad x^{(k)^*} x^{(k+1)} & & = \frac{1}{2} + \frac{1}{2}\|x^{(k)}\|_2^2 \end{cases}$$

kann in folgender Weise gelöst werden

$$\begin{cases} z^{(k+1)} & := & (A - \lambda^{(k)} I)^{-1} x^{(k)}, \\ \lambda^{(k+1)} & := & \lambda^{(k)} + \frac{1}{2}(1 + \|x^{(k)}\|^2)/(x^{(k)*} z^{(k+1)}), \\ x^{(k+1)} & := & (\lambda^{(k+1)} - \lambda^{(k)}) z^{(k+1)}. \end{cases}$$

Das (quadratisch konvergente) Newtonverfahren unterscheidet sich hier also nur durch die fehlende exakte Normierung (i.a. $\|x^{(k)}\| \neq 1$) von der kubisch konvergenten Inversen Iteration.

Auch beim *QR-Verfahren* können die Aussagen gegenüber dem unsymmetrischen Fall verbessert werden. Nach Satz 4.2.2 kann oBdA für A Tridiagonalgestalt angenommen werden. Daher reduziert sich schon der Rechenaufwand pro QR-Schritt auf ca. $28n$ Operationen und n Quadratwurzeln.

Wegen des engen Zusammenhangs zwischen Vektoriteration und QR-Verfahren überträgt sich die lokal kubische Konvergenz auf (4.3.10),(4.3.12), denn

$$\lambda^{(k)} = s_{nn}^{(k)} = R_{S^{(k)}}[e^{(n)}].$$

Allerdings kann bei dieser Wahl in bestimmten Fällen zu Beginn nur lineare Konvergenz eintreten. In der Startphase läßt sich die Konvergenz noch weiter verbessern dadurch, daß als Spektralparameter derjenige Eigenwert des letzten 2×2-Blocks

$$\begin{pmatrix} s_{n-1,n-1}^{(k)} & s_{n-1,n}^{(k)} \\ s_{n,n-1}^{(k)} & s_{n,n}^{(k)} \end{pmatrix}$$

gewählt wird, der näher an $s_{nn}^{(k)}$ liegt. Mit $\delta_k := \frac{1}{2}(s_{nn}^{(k)} - s_{n-1,n-1}^{(k)})$ ist dies

$$\lambda^{(k)} = \frac{1}{2}(s_{nn}^{(k)} + s_{n-1,n-1}^{(k)}) + sign(\delta_k)\sqrt{\delta_k^2 + {s_{n-1,n}^{(k)}}^2}. \tag{4.4.5}$$

Für diese Wahl kann sogar folgende globale Konvergenzaussage gemacht werden. Eine Tridiagonalmatrix heißt dabei *unzerlegbar*, wenn $a_{i,i-1} \neq 0 \;\forall i = 2, \ldots, n$. Unzerlegbare Tridiagonalmatrizen besitzen nur einfache Eigenwerte.

Satz 4.4.6 (Wilkinson) *Für eine unzerlegbare symmetrische Tridiagonalmatrix A konvergiert das QR-Verfahren (4.3.10) mit Spektralverschiebung (4.4.5) global, also ohne Startschritte. Die Konvergenz ist mindestens quadratisch, d.h.,* $\exists \varepsilon_0 > 0$ *so, daß*

$$|s^{(k)}_{n-1,n}| \le \varepsilon \le \varepsilon_0 \quad \Rightarrow \quad |s^{(k+1)}_{n-1,n}| \le c\frac{\varepsilon^3}{s^{(k)}_{n-1,n-1} - \lambda^{(k)}} \le c\varepsilon^2.$$

Mit dieser QR-Variante können (als empirischer Wert) alle Eigenwerte mit *ca.* $\frac{4}{3}n^3$ Operationen und die volle Schur-Normalform mit ca. $10n^3$ Operationen berechnet werden.

Außer den Modifikationen der allgemein einsetzbaren Verfahren gibt es weitere, deren Anwendungsbereich auf das symmetrische Eigenwert-Problem beschränkt ist. Diese besitzen in bestimmten Situationen Vorteile gegenüber den Standardverfahren und sind daher ebenfalls von Interesse.

Bisektionsverfahren

Mit diesem Verfahren ist es möglich, gezielt einzelne Eigenwerte der Matrix zu bestimmen. Dabei kann die Nummer oder die Lage des Eigenwerts vorgegeben werden. Die Grundlage des Verfahrens ist die Tatsache, daß die Unterdeterminanten (Haupt-Minoren)

$$p_k(\lambda) := \det \begin{pmatrix} a_{11}-\lambda & a_{12} & & & \\ a_{21} & a_{22}-\lambda & a_{23} & & \\ & \ddots & \ddots & \ddots & \\ & & \cdot & \cdot & a_{k-1,k} \\ & & & a_{k,k-1} & a_{kk}-\lambda \end{pmatrix}, \quad k = 1, \ldots, n, \tag{4.4.6}$$

$p_0 :\equiv 1$, einer symmetrischen Tridiagonalmatrix eine sogenannte *Sturmsche Kette* bilden.

Definition 4.4.7 *Die Folge* $p_0 \equiv 1, p_1, \ldots, p_n$ *von reellen Polynomen wachsenden Grads bildet eine* Sturmsche Kette, *wenn folgende Bedingungen gelten:*

1. *Die Nullstellen von* p_n *sind einfach.*
2. *Für jede reelle Nullstelle* ξ *von* p_n *gilt sign* $p_{n-1}(\xi) = -$*sign* $p_n'(\xi)$ $(\neq 0)$.
3. *Für jede reelle Nullstelle* ξ *von* p_i, $i = 1, \ldots, n-1$, *gilt*

$$p_{i-1}(\xi)p_{i+1}(\xi) < 0.$$

Für die Polynome einer Sturmschen Kette kann die Verteilung der reellen Nullstellen an der Vorzeichenverteilung der Folge $\{p_j(\hat{x})\}_j$ an einer festen Stelle $\hat{x}$ abgelesen werden. Dazu sei

$$w(\hat{x}) := \text{ Anzahl der Vorzeichenwechsel in } (p_0(\hat{x}), p_1(\hat{x}), \ldots, p_n(\hat{x})) \tag{4.4.7}$$

mit der Verabredung, daß zuvor alle verschwindenden Werte $p_j(\hat{x}) = 0$ aus der Folge entfernt werden. Dann gilt

Satz 4.4.8 *Die Anzahl der reellen Nullstellen des Polynoms p_n einer Sturmschen Kette im Intervall $[a,b]$ ist gleich*

$$w(b) - w(a).$$

Beweis [Stoer, Numer. Math. 1]

Um den Satz auf das Eigenwertproblem anwenden zu können, ist nachzuweisen, daß die Haupt-Minoren p_k aus (4.4.6) tatsächlich die Bedingungen aus Definition 4.4.7 erfüllen. Dazu, und zur numerischen Auswertung dient folgende Rekursionsformel.

$$\begin{aligned} p_0(\lambda) &:= 1,\ p_1(\lambda) := a_{11} - \lambda, \\ p_{k+1}(\lambda) &:= (a_{k+1,k+1} - \lambda)p_k(\lambda) - |a_{k+1,k}|^2 p_{k-1}(\lambda),\ k = 1, \dots, n-1. \end{aligned} \tag{4.4.8}$$

Beweis Entwicklung nach der letzten Zeile in (4.4.6):

$$p_{k+1} = (a_{k+1,k+1} - \lambda)p_k - a_{k+1,k} \begin{vmatrix} \ddots & & \vdots \\ & a_{k-1,k-1} - \lambda & 0 \\ \cdots & a_{k,k-1} & a_{k,k+1} \end{vmatrix}. \qquad \blacksquare$$

Also kann insbesondere der Wert $p_n(\lambda)$ mit $4n$ Operationen bestimmt werden.

Beispiel 4.4.9

$$T = \begin{pmatrix} 1 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 3 & -1 \\ 0 & 0 & -1 & 4 \end{pmatrix} \Rightarrow \{\lambda_j\} \doteq \{0.25, 1.82, 3.18, 4.74\}.$$

In $\hat{x} = 2$ ist $(p_0(2), \dots, p_4(2)) = (1, -1, -1, 0, 1)$. Bei der Zählung der Vorzeichenwechsel wird die Null gestrichen, es ergibt sich $w(2) = 2$. Wegen $w(-\infty) = 0$ liegen 2 Eigenwerte links von $\hat{x} = 2$.

Satz 4.4.10 *Die Matrix $A \in \mathbf{C}^{n\times n}$ sei eine unzerlegbare, hermitesche Tridiagonalmatrix. Dann gelten für die Polynome p_k, $k = 0, \dots, n$, aus (4.4.6), (4.4.8) die Aussagen*

a) $p_k \in \Pi_k$, $p_k(\lambda) = (-\lambda)^k + \dots$, *d.h.* $p_k(\lambda) \to \begin{cases} \infty, & \lambda \to -\infty \\ (-1)^k\infty, & \lambda \to \infty \end{cases}$

b) Das Polynom p_k, $k > 0$, hat k reelle, einfache Nullstellen $\lambda_1^{(k)} < \dots < \lambda_k^{(k)}$, die durch die Nullstellen von p_{k-1} getrennt werden,

$$\lambda_1^{(k)} < \lambda_1^{(k-1)} < \lambda_2^{(k)} < \dots < \lambda_{k-1}^{(k-1)} < \lambda_k^{(k)}.$$

c) $p_k(\xi) = 0 \ \Rightarrow\ p_{k-1}(\xi)p_{k+1}(\xi) < 0.$

d) $\mathit{sign}\, p_{k-1}(\lambda_j^{(k)}) = -\mathit{sign}\, p_k'(\lambda_j^{(k)}),\ j = 1, \dots, k.$

Beweis a) Folgt aus (4.4.8).

c) Aus (4.4.8) folgt für $p_k(\xi) = 0$ direkt $p_{k+1}(\xi) = -|a_{k+1,k}|^2 p_{k-1}(\xi)$. Dabei ist $a_{k+1,k} \neq 0$ n.V., also gilt $p_{k+1}(\xi)p_{k-1}(\xi) \leq 0$. Keiner der beiden Werte kann aber verschwinden, da sonst aus (4.4.8) $p_0(\xi) = 0$ folgen würde.

b) Die Nullstellen sind reell, da die p_k charakteristische Polynome symmetrischer Matrizen sind. Die Trennungseigenschaft folgt induktiv dadurch, daß für jedes Polynom die Vorzeichenverteilung in und zwischen den Nullstellen aufgelistet wird:

$$\left\{\begin{array}{lccccccc}
p_0: & & & & + & & & \\
p_1: & & & + & 0 & - & & \\
p_2: & & + & 0 & - & 0 & + & \\
p_3: & + & 0 & - & 0 & + & 0 & -
\end{array}\right. \tag{4.4.9}$$

Dabei folgt die Vorzeichenverteilung am linken und rechten Rand aus a), die vertikalen Pfeile deuten die Folgerung c) an, auf die Existenz der Nullstellen schließlich kann aus den Vorzeichenwechseln geschlossen werden. Da bei p_k jeweils k Nullstellen auftreten, sind alle einfach.

d) Folgt ebenfalls aus (4.4.9): sign $p_{k-1}(\lambda_j^{(k)}) = \text{sign}\,\Big(p_k(\lambda_j^{(k)} - \epsilon) - p_k(\lambda_j^{(k)} + \epsilon)\Big)$. ∎

Mit Hilfe von Satz 4.4.8 kann ein sehr einfaches Verfahren zur Bestimmung eines beliebigen Eigenwerts λ_m, $1 \leq m \leq n$, angegeben werden. Ausgehend von einem Startintervall $[a, b]$, z.B., $[a, b] = [-\|A\|, \|A\|]$, wird die Einschließung von λ_m durch *Bisektion* anhand der Vorzeichenwechsel w schrittweise verbessert:

wiederhole:
  $\lambda := (a + b)/2$;
  falls $w(\lambda) \geq m$, dann $b := \lambda$
    sonst: $a := \lambda$;
bis $|a - b| \leq eps(|a| + |b|)$;   (4.4.10)

Bessere a-priori-Einschließungen der Eigenwerte werden im nächsten Teilparagraphen behandelt. Jedes der erzeugten Teilintervalle enthält wegen $w(-\infty) = 0$ den Eigenwert λ_m. Die Konvergenz des Bisektionsverfahrens (4.4.10) ist im Vergleich zur Endkonvergenz der anderen Verfahren recht langsam, da die Intervallänge in jedem Schritt nur halbiert wird ($\Rightarrow$ Konvergenzfaktor $= \frac{1}{2}$). Es bietet sich aber an, gerade bei nahe zusammenliegenden Eigenwerten, zur Bestimmung eines Startintervalls, das nur λ_m enthält. Die genaue Berechnung dieses Eigenwertes und des zugehörigen Eigenvektors kann dann schneller mit der Inversen Iteration vorgenommen werden.

Jacobi-Verfahren

Das Grundprinzip dieses Verfahrens besteht darin, durch einfache unitäre Ähnlichkeitstransformationen einzelne Nebendiagonalelemente der gesamten Ausgangsmatrix (d.h. ohne Vortransformation auf Tridiagonalgestalt) zu Null zu machen. Das Verfahren ist zwar mit ca. $32n^3$ Operationen aufwendiger als das QR-Verfahren mit $\frac{4}{3}n^3$ Operationen, hat aber den Vorteil eines einfachen Aufbaus und der Parallelisierbarkeit ($\rightarrow$ Superrechner). Zur Transformation werden bei reellen Matrizen elementare Rotationen der Gestalt

$$J(p,q;\vartheta) := \begin{pmatrix} 1 & & & & & & & \\ & \ddots & & & & & & \\ & & 1 & & & & & \\ & & & c & & s & & \\ & & & & 1\ \ddots\ 1 & & & \\ & & & -s & & c & & \\ & & & & & & 1 & \\ & & & & & & & \ddots\ 1 \end{pmatrix}, \tag{4.4.11}$$

mit $c := \cos\vartheta,\ s := \sin\vartheta$, benutzt. Die Einträge c und $\pm s$ stehen dabei in den Zeilen und Spalten p und q. Der Winkel ϑ wird so bestimmt, daß die Elemente $b_{pq} = b_{qp}$ in der transformierten Matrix

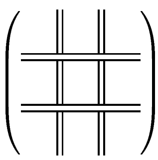
$$B := J^* A J \tag{4.4.12}$$

verschwinden. Die Wirkung der beiden Multiplikationen ist, mit $A' := J^* A$,

$$\left.\begin{aligned} a'_{pj} &= c a_{pj} - s a_{qj} \\ a'_{qj} &= s a_{pj} + c a_{qj} \end{aligned}\right\} \ j = 1,\dots,n, \quad a'_{ij} = a_{ij} \text{ für } i \neq p,q,$$

$$\left.\begin{aligned} b_{ip} &= c a'_{ip} - s a'_{iq} \\ b_{iq} &= s a'_{ip} + c a'_{iq} \end{aligned}\right\} \ i = 1,\dots,n, \quad b_{ij} = a'_{ij} \text{ für } j \neq p,q,$$

Die Transformation (4.4.12) ändert A also nur im schraffierten Bereich

$B :=$  .

Insbesondere ergeben sich

$$\begin{aligned} b_{pp} &= c^2 a_{pp} - 2cs a_{pq} + s^2 a_{qq}, \\ b_{qq} &= s^2 a_{pp} + 2cs a_{pq} + c^2 a_{qq}, \\ b_{qp} = b_{pq} &= (a_{pp} - a_{qq})cs + a_{pq}(c^2 - s^2) \stackrel{!}{=} 0. \end{aligned} \tag{4.4.13}$$

Natürlich gilt $b_{pp} + b_{qq} = a_{pp} + a_{qq}$. Das Verschwinden des Ausdrucks in der dritten Gleichung ist äquivalent mit

$$\frac{1}{2}(a_{pp} - a_{qq}) \sin 2\vartheta + a_{pq} \cos 2\vartheta = 0.$$

Für den nichttrivialen Fall $a_{pq} \neq 0$ definiert dies den Winkel ϑ (oBdA $|\vartheta| \leq \frac{\pi}{4}$), nämlich

$$\cot 2\vartheta = \frac{a_{qq} - a_{pp}}{2a_{pq}} =: z. \tag{4.4.14}$$

Für die Transformation werden nicht ϑ, sondern $\cos\vartheta$ und $\sin\vartheta$ benötigt. Für $t := \tan\vartheta$ ist

$$\cot 2\vartheta = \frac{1-t^2}{2t} = z \qquad \Longleftrightarrow \qquad t^2 - 2zt - 1 = 0.$$

Die betragskleinste Lösung dieser Gleichung ist

$$t = \begin{cases} 1, & z = 0, \\ \dfrac{\text{sign } z}{|z| + \sqrt{1+z^2}}, & z \neq 0, \end{cases}$$

aus der sich die Transformationsparameter ableiten,

$$c = \frac{1}{\sqrt{1+t^2}} = \cos\vartheta, \quad s = tc = \sin\vartheta.$$

Die Transfomationsformeln (4.4.13) können noch vereinfacht werden,

$$b_{pp} = a_{pp} - ta_{pq}, \; b_{qq} = a_{qq} + ta_{pq}, \; b_{pq} = b_{qp} = 0, \tag{4.4.15}$$

während sich für die restlichen Elemente wegen der Symmetrie ergibt

$$\left.\begin{array}{rcl} b_{pi} = b_{ip} & = & ca_{ip} - sa_{iq} \\ b_{qi} = b_{iq} & = & sa_{ip} + ca_{iq} \end{array}\right\} \begin{array}{c} i = 1, \ldots, n, \\ i \neq p, q. \end{array} \tag{4.4.16}$$

<u>**Rechenaufwand**</u> pro Jacobi-Schritt: $6n$ Operationen.
Zur iterativen Durchführung des Jacobi-Verfahrens ist eine Auswahl der Matrix-Elemente a_{pq} durchzuführen. In Frage kommen (u.a.) zwei Strategien

- klassische Strategie, das betragsgrößte Nebendiagonalelement wird auf Null transformiert,

  $$a_{pq} = \max_{j \neq i} |a_{ij}|.$$

  Die Maximumbestimmung kann allerdings aufwendiger sein als die Transformation.

- Zyklische Strategie, alle Nebendiagonalelemente werden der Reihe nach bearbeitet, $p = 1, 2, \ldots, n-1, \; q = p+1, \ldots, n$. Modifikation: zu kleine Elemente werden übersprungen.

Eine Analyse der **Konvergenz** des Jacobi-Verfahrens ist einfach möglich durch Betrachtung der Quadratsumme aller Nebendiagonalelemente,

$$S(A) := \sum_{j>i} a_{ij}^2. \tag{4.4.17}$$

Wegen der Orthogonalität der Matrix J gilt nach (4.4.16)

$$\left.\begin{array}{rcl} b_{ip}^2 + b_{iq}^2 & = & a_{ip}^2 + a_{iq}^2 \\ b_{pj}^2 + b_{qj}^2 & = & a_{pj}^2 + a_{qj}^2 \end{array}\right\} i, j \neq p, q.$$

Da durch die Transformation a_{pq} eliminiert wird, ist

$$S(B) = S(J^*AJ) = S(A) - a_{pq}^2. \tag{4.4.18}$$

Insbesondere wird die nichtnegative Größe S verkleinert. Daraus folgt

Satz 4.4.11 *Beim Jacobi-Verfahren,* $A^{(0)} := A$,

$$A^{(k+1)} := J^{(k)^*} A^{(k)} J^{(k)}, \quad J^{(k)} := J(p, q, \vartheta),$$

mit klassischer oder zyklischer Strategie konvergiert die Folge $A^{(k)}$ *gegen eine Diagonalmatrix der Eigenwerte von* A. *Beim klassischen Verfahren gilt die a-priori-Schranke*

$$S(A^{(k)}) \le \Big(1 - \frac{2}{n(n-1)}\Big)^k S(A).$$

Beweis (klassische Strategie): Nach (4.4.18) ist $S(A^{(k)})$ eine streng monoton fallende, nach unten durch null beschränkte Folge. Außerdem gilt hier, bei Wahl des maximalen Matrixelements

$$S(A^{(k)}) \le \frac{n(n-1)}{2} a_{pq}^{(k)} \;\Rightarrow\; S(A^{(k+1)}) \le \Big(1 - \frac{2}{n(n-1)}\Big) S(A^{(k)}).$$

Nach (4.4.15) ist die Änderung der Hauptdiagonalen pro Schritt durch $\sqrt{S(A^{(k)})}$ beschränkt. ■

Bei einer Matrix mit nur einfachen Eigenwerten, bei denen also

$$d := \min_{i \ne j} |\lambda_i - \lambda_j| > 0$$

positiv ist, tritt bei beiden Strategien sogar quadratische Konvergenz ein, wenn volle Zyklen, d.h., $N := \frac{1}{2} n(n-1)$ Schritte betrachtet werden. Dann gilt

$$S(A^{(k+N)}) \le \frac{c_n}{d^2} \, S(A^{(k)})^2.$$

Beispiel 4.4.12 Die Eigenwerte der Matrix

$$A := \begin{pmatrix} 20 & -7 & 3 & -2 \\ -7 & 5 & 1 & 4 \\ 3 & 1 & 3 & 1 \\ -2 & 4 & 1 & 2 \end{pmatrix}$$

sind $\{\lambda_j\} \doteq \{23.5, 6.46, 1.17, -1.16\}$, also $d \doteq 2.3$. Das zyklische Verfahren liefert

$k =$	0	6	12	18	24
$S(A^{(k)}) =$	80	2.9	$6.9_{10}-3$	$5.5_{10}-10$	$1.9_{10}-31$

4.5 Überblick der Eigenwert-Verfahren

Die folgende Tabelle faßt die Eigenschaften und die Einsatzbereiche der behandelten Verfahren zusammen. Das am Ende erwähnte Lanczos-Verfahren wird im nächsten Kapitel besprochen.

Verfahren für Eigenwertprobleme

Verfahren	Unsym. EWP	Symm. EWP	für Poblemklasse
Vektor-Iteration	+	+	größter EW u. EV, v.a. dünnbesetzte Matrix
Inverse Iteration	evtl. problematisch	kubisch u. global konvergent	zu λ_0 nächster EW u. EV, z.B. nach QR-Verfahren, Bisektion
orthogonale Iteration	+	+	Gruppe größter EWe, invarianter Unterraum
QR-Verfahren	+	kubisch u. global konvergent	Standardverfahren für volle Matrix: Schur-Normalform
Bisektion	$-$	lineare Konvergenz	für k-ten EW, EW-Haufen, kein EV ($\rightarrow$ Inverse Iteration)
Jacobi-Verfahren	$-$	quadratische Konvergenz	alle EWe: Schurnormalform, parallelisierbar
Lanczos-Verfahren	$-$	+	größte EWe, schnelle Approximation bei dünnbesetzter Matrix

4.6 Eigenwert- und Fehler-Schranken

Wie in der Numerik I wird ein Algorithmus als *gutartig* bezeichnet, wenn er die exakte Lösung eines nur leicht gestörten Ausgangsproblems liefert. Große Fehler in den berechneten Lösungen sind bei solchen gutartigen Verfahren auf eine zu große Fehleranfälligkeit (schlechte Kondition) des Ausgangsproblem zurückzuführen.

In diesem Sinn sind die behandelten Verfahren gutartig. Bei den Transformationsverfahren mit unitären Matrizen (Hessenberg-Transf., QR-, Jacobi-Verfahren) etwa gilt, daß das Ergebnis bei numerischer Rechnung statt der exakten Matrix $T = Q^*AQ$ eine Matrix $\tilde{T}$ ist mit

$$\tilde{T} = Q^*(A + F)Q, \quad \|F\|_2 \leq c \cdot eps\|A\|_2. \tag{4.6.1}$$

Dabei ist *eps* die Maschinengenauigkeit und c eine harmlose Konstante, die i.w. von der Zahl der Rechenoperationen abhängt. Diese überschaubaren Störungen des *Problems* können sehr unterschiedliche Störungen der *Lösungen* beim Eigenwertproblem zur Folge haben. Diese Auswirkungen hängen von den Eigenschaften der Matrix ab. Das symmetrische Eigenwertproblem ist dabei sehr unempfindlich gegen Störungen, also gut konditioniert. Dies folgt aus

Satz 4.6.1 *(Courantsches Maximum-Minimum-Prinzip) Die Eigenwerte der hermiteschen Matrix $A \in \mathbf{C}^{n\times n}$ seien geordnet nach $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$. Wenn S lineare Unterräume des $\mathbf{C}^{n\times n}$ bezeichnet, dann ist*

$$\lambda_j = \max_{\dim S=j} \min_{0\neq y\in S} R_A[y], \quad j = 1, \ldots, n.$$

Sobald in den folgenden Aussagen verschiedene Matrizen $A, B, \ldots$ auftreten, werden deren Eigenwerte mit $\lambda_j(A), \lambda_j(B), \ldots$ bezeichnet, sie seien wie in Satz 4.6.1 fallend angeordnet. Eine direkte Folgerung des Minimax-Prinzips ist

Satz 4.6.2 *Sind A und $A+F$ hermitesch, dann gilt*

$$\lambda_j(A) + \lambda_n(F) \le \lambda_j(A+F) \le \lambda_j(A) + \lambda_1(F), \;\; j = 1, \ldots, n.$$

Beweis für die untere Schranke: Da $R_{A+F}[y] = R_A[y] + R_F[y]$ ist

$$\begin{aligned}\lambda_j(A+F) &= \max_{\dim S=j} \min_{y\in S} \Big(R_A[y] + R_F[y]\Big) \ge \max_{\dim S=j} \Big(\min_{y\in S} R_A[y] + \min_{z\in S} R_F[z]\Big) \\ &\ge \max_{\dim S=j} \min_{y\in S} R_A[y] + \min_{z\in \mathbf{C}^n} R_F[z].\end{aligned}$$

Die letzte Größe stimmt mit der linken Seite der Behauptung überein. ∎

Die Störung der Eigenwerte von A hat insbesondere höchstens den Betrag $\varrho(F) = \|F\|_2$, es gilt also $|\lambda_j(A+F) - \lambda_j(A)| \le \|F\|_2$. Bei den Eigenvektoren kann die Störung allerdings größer ausfallen. Im folgenden Satz wird davon ausgegangen, daß eine Näherung x für einen Eigenvektor und λ für den zugehörigen Eigenwert (z.B. durch den Rayleigh-Quotienten) bekannt ist.

Satz 4.6.3 *Gegeben sei eine hermitesche Matrix A und ein Paar (x, λ) mit $x \ne 0$. Damit sei*

$$\|Ax - \lambda x\|_2 \le \varepsilon \|x\|_2 \quad bzw. \quad \|Ax - \lambda x\|_2 \le \delta \|Ax\|_2. \tag{4.6.2}$$

a) Dann gilt

$$\min_j |\lambda_j - \lambda| \le \varepsilon \quad bzw. \quad \min_{\lambda_j \ne 0} |1 - \frac{\lambda}{\lambda_j}| \le \delta.$$

b) Ist λ_k der zu λ nächstgelegene Eigenwert, bei dem also $\min_j |\lambda_j - \lambda|$ angenommen wird, und ist dieser einfach mit $d_k := \min_{j\ne k} |\lambda_j - \lambda_k| > 0$, dann gilt

$$\min_{\xi\in\mathbf{R}} \|x - \xi x^{(k)}\|_2 \le 2\frac{\varepsilon}{d_k}\|x\|_2.$$

Beweis Da die Eigenvektoren eine Orthonormalbasis bilden, gilt für $x = \sum_{j=1} \xi_j x^{(j)}$ in Teil a):

$$\min_j (\lambda_j - \lambda)^2 \sum_j |\xi_j|^2 \le \sum_{j=1}^n (\lambda_j - \lambda)^2 |\xi_j|^2 = \|Ax - \lambda x\|_2^2 \le \varepsilon^2 \|x\|_2^2 = \varepsilon^2 \sum_j |\xi_j|^2.$$

Für die zweite Ungleichung gilt mit $\eta_j := \xi_j \lambda_j$:

$$\begin{aligned}\min_{\lambda_j\ne 0} (1 - \lambda/\lambda_j)^2 \sum_j |\eta_j|^2 &\le \sum_{\lambda_j \ne 0} (1 - \lambda/\lambda_j)^2 |\lambda_j \xi_j|^2 + \lambda^2 \sum_{\lambda_j = 0} |\xi_j|^2 \\ &= \|Ax - \lambda x\|_2^2 \le \delta^2 \|Ax\|_2^2 = \delta^2 \sum_j |\eta_j|^2.\end{aligned}$$

b) Nach Teil a) folgt

$$\varepsilon\|x\|_2 \geq \|Ax - \lambda_k x\|_2 - |\lambda - \lambda_k|\|x\|_2 \geq \|Ax - \lambda_k x\|_2 - \varepsilon\|x\|_2.$$

Dies bedeutet

$$4\varepsilon^2\|x\|_2^2 \geq \|Ax - \lambda_k x\|_2^2 = \sum_{j\neq k}(\lambda_j - \lambda_k)^2|\xi_j|^2 \geq d_k^2 \sum_{j\neq k}|\xi_j|^2 = d_k^2\|x - \xi_k x^{(k)}\|_2^2.$$

Wegen der Orthogonalität der Eigenvektor-Basis wird $\min_\xi \|x - \xi x^{(k)}\|_2$ in ξ_k angenommen. ■

Bemerkung: 1) Nach Satz 4.4.2 ist der Rayleighquotient $R_A[x]$ die beste Eigenwert-Schätzung zur Eigenvektor-Näherung x. Aus dem Beweis dieses Satzes folgt (bei $\|x\|_2 = 1$)

$$\varepsilon^2 = \|Ax - R_A[x]x\|_2^2 = \|Ax\|_2^2 - R_A[x]^2,$$

und für beliebige λ die Beziehung $\|Ax - \lambda x\|_2^2 = \varepsilon^2 + |\lambda - R_A[x]|^2$. Daher gilt

$$\min_j |\lambda_j - R_A[x]| \leq \frac{1}{d_k}\|Ax - \lambda_k x\|_2^2 = \frac{1}{d_k}(|\lambda_k - R_A[x]|^2 + \varepsilon^2) \leq 2\frac{\varepsilon^2}{d_k}. \qquad (4.6.3)$$

Für $\varepsilon < d_k/2$ ist diese Schranke besser als die aus Satz 4.6.3.

2) Die in der ersten Bemerkung verwendete Größe ε ist bei der Vektoriteration (4.4.3) bzw. (4.4.4) einfach verfügbar. Bei der Inversen Iteration (4.4.4) etwa gilt mit $\lambda^{(k)} = R_A[x^{(k)}]$ und $B_{k-1} := A - \lambda^{(k-1)}I$:

$$\begin{aligned}
\varepsilon^2\|z^{(k)}\|_2^2 &= \|Az^{(k)} - \lambda^{(k)}z^{(k)}\|_2^2 = \|Az^{(k)} - \lambda^{(k-1)}z^{(k)} - (\lambda^{(k)} - \lambda^{(k-1)})z^{(k)}\|_2^2 \\
&= \|B_{k-1}z^{(k)} - R_{B_{k-1}}[z^{(k)}]z^{(k)}\|_2^2 = \|q^{(k-1)}\|_2^2 - (z^{(k)*}q^{(k-1)}/\|z^{(k-1)}\|_2)^2 \\
&= 1 - (z^{(k)*}q^{(k-1)}/\|z^{(k-1)}\|_2)^2.
\end{aligned}$$

Alle hier auftretenden Größen werden bei der Inversen Iteration sowieso berechnet, mit Satz 4.6.3 steht also ein einfaches Abbruchkriterium zur Verfügung.

3) Beim symmetrischen QR-Verfahren sind die Einheitsvektoren Näherungs-Eigenvektoren der Matrizen $S^{(k)}$. Für $x = e^{(n)}$ ist $R_S[x] = s_{nn}$ und $\|Sx - s_{nn}x\|_2 = |s_{n-1,n}| =: \varepsilon$. Ein Eigenwert von S unterscheidet sich von s_{nn} also höchstens um den Betrag $|s_{n-1,n}|$.

Beispiel 4.6.4 Die Eigenwerte von

$$A = \begin{pmatrix} 1 & 2 & 7/3 \\ 2 & 2 & 3 \\ 7/3 & 3 & 3 \end{pmatrix}$$

sind $\{\lambda_j\} \doteq \{7.09, -0.5293, -0.562\}$, daher $d_2 \cong 1/30$. Für $x = (0.529, 0.439, -0.723)^T$ ist $R_A[x] \doteq -0.529326$ eine Eigenwert-Näherung der Genauigkeit $\varepsilon = 7.3_{10}-4$. Für den Eigenvektor ergibt sich die gröbere Schranke $\|x - x^{(2)}\| \leq 1.2_{10} - 2$.

Im Gegensatz zum symmetrischen Eigenwertproblem können bei *unsymmetrischen* (genauer: nicht-normalen) Matrizen die Eigenwerte und -Vektoren sehr empfindlich auf Störungen reagieren. Zunächst wird aber eine globale Aussage über die Lage der Eigenwerte behandelt.

Satz 4.6.5 (Gerschgorin) *Zur Matrix $A \in \mathbf{C}^{n\times n}$ sei definiert*

$$r_i := \sum_{\substack{j=1\\ j\neq i}}^{n} |a_{ij}|,\ i = 1,\dots,n. \tag{4.6.4}$$

Dann liegen alle Eigenwerte von A in der Vereinigung der Kreisscheiben

$$K_{r_i}(a_{ii}) := \{z \in \mathbf{C}:\ |z - a_{ii}| \le r_i\},\ i = 1,\dots,n.$$

Beweis Es werde angenommen, ein Eigenwert λ liege außerhalb aller Kreisscheiben, sein Eigenvektor sei x mit $1 = \|x\|_\infty = |x_k|$. Dann folgt

$$0 = |(a_{kk}-\lambda)x_k + \sum_{j\neq k} a_{kj}x_j| \ge |a_{kk}-\lambda||x_k| - \sum_{j\neq k}|a_{kj}|\|x\|_\infty = (|a_{kk}-\lambda| - r_k)\|x\|_\infty > 0.$$

Dies ist offensichtlich ein Widerspruch. ■

Beispiel 4.6.6 Für die Matrix des letzten Beispiels ist $r_1 \doteq 4.33, r_2 = 5, r_3 \doteq 5.33$. Im Reellen gilt $K_1 \doteq [-3.33, 5.33], K_2 = [-3, 7], K_2 \doteq [-2.33, 8.33]$. Die Vereinigung dieser drei Intervalle ist $[-3.33, 8.33]$.

Bemerkung: 1) Wenn die Menge $\bigcup_{i=1}^n K_{r_i}(a_{ii})$ in mehrere, nicht zusammenhängende Teile zerfällt, kann gezeigt werden, daß in jeder Komponente der Menge soviele Eigenwerte liegen, wie Kreise dazugehören.

2) Durch Anwendung des Satzes auf A^T oder

$$D^{-1}AD = \Big(\frac{d_j}{d_i}a_{ij}\Big)_{i,j},\ D = \mathrm{diag}(d_i),$$

lassen sich die Schranken eventuell (für einzelne Eigenwerte) verbessern.

Besonders empfindlich können mehrfache Eigenwerte auf Störungen reagieren.

Beispiel 4.6.7 Der doppelte Eigenwert 1 der Matrix $A = \begin{pmatrix}1 & 1\\ 0 & 1\end{pmatrix}$ spaltet sich bei Addition der Matrix $F = \begin{pmatrix}0 & 0\\ \varepsilon & 0\end{pmatrix}$ auf in die beiden Eigenwerte $\lambda_{1/2}(A+F) = 1 \pm \sqrt{\varepsilon}$. Die Eigenwerte hängen hier also nicht-differenzierbar von ε ab!

Zur Verallgemeinerung der obigen Störungssätze kann im unsymmetrischen Fall die Jordan- oder die Schur-Normalform herangezogen werden.

Satz 4.6.8 *Die Matrix A sei diagonalisierbar mit $A = XJX^{-1}$. Dann gilt für $j = 1,\dots,n$ die Schranke*

$$\min_i |\lambda_j(A+F) - \lambda_i(A)| \le \|F\|\kappa(X).$$

Diese Aussage gilt in der $1, 2, \infty$-Norm mit der zugehörigen Konditionszahl $\kappa(X) = \|X\|\|X^{-1}\|$.

Beweis Es sei λ Eigenwert von $A+F$ und x ein zugehöriger Eigenvektor mit Norm eins, $\|x\|=1$. λ sei aber (oBdA) kein Eigenwert von A. Dann gilt

$$\lambda x - Ax = Fx \iff x = (\lambda I - A)^{-1}Fx \quad \Rightarrow$$

$$\begin{aligned} 1 &= \|x\| \leq \|(\lambda I - A)^{-1}F\| \leq \|(\lambda I - A)^{-1}\|\|F\| \\ &\leq \|(\lambda I - J)^{-1}\|\kappa(X)\|F\| = \frac{1}{\min_j|\lambda_j - \lambda|}\kappa(X)\|F\| \quad \blacksquare \end{aligned} \tag{4.6.5}$$

Der allgemeinste Fall wird erst durch den folgenden Satz abgedeckt, bei dem von der Schur-Normalform $A = USU^*$ ausgegangen wird. Dabei wird die Dreieckmatrix S zerlegt in die Diagonale $\Lambda = \operatorname{diag}(\lambda_j)$ und den Rest M, $S = \Lambda + M$. Die Norm $\|M\|_2$ ist ein Maß dafür, wieweit sich A von einer normalen Matrix unterscheidet.

Satz 4.6.9 *Es sei $U^*AU = S = \Lambda + M$ die Schur-Normalform der Matrix A und $p \leq n$ so, daß $|M|^p = 0$ ist. Es sei*

$$\vartheta := \|F\|_2 \sum_{j=0}^{p-1} \|M\|_2^j.$$

Dann gilt für $j = 1, \ldots, n$ die Schranke

$$\min_i |\lambda_j(A+F) - \lambda_i(A)| \leq \max\{\vartheta, \vartheta^{1/p}\}.$$

Beweis Hier wird in (4.6.5) eine andere Abschätzung der Inversen verwendet,

$$\|(\lambda I - A)^{-1}\|_2 = \|(\lambda I - \Lambda - M)^{-1}\|_2 \leq \|(\lambda I - \Lambda)^{-1}\|\|\Big(I - (\lambda I - \Lambda)^{-1}M\Big)^{-1}\|_2.$$

Mit $d := \|(\lambda I - \Lambda)^{-1}\|_2^{-1} = \min_j |\lambda_j - \lambda|$ und dem Satz von Neumann folgt

$$\|\Big(I - (\lambda I - \Lambda)^{-1}M\Big)^{-1}\|_2 \leq \sum_{j=0}^{p-1} d^{-j}\|M\|_2^j,$$

da mit $|M|^p = 0$ auch $[(\lambda I - \Lambda)^{-1}M]^p = 0$. Die Beziehung (4.6.5) führt daher hier auf

$$1 \leq \|F\|_2 \frac{1}{d} \sum_{j=0}^{p-1} \frac{1}{d^j}\|M\|_2^j \leq \|F\|_2 \sum_{j=1}^{p-1} \|M\|_2^j \max\{d^{-1}, d^{-p}\}.$$

Also folgt mit $\min\{d, d^p\} \leq \vartheta \iff d \leq \max\{\vartheta, \vartheta^{1/p}\}$ die Behauptung. $\blacksquare$

Beide Sätze zeigen, daß für große Werte von $\kappa(X)$ bzw. $\|M\|_2$ kleine Änderungen der Matrix A große Störungen in den Eigenwerten zur Folge haben können. Nach (4.6.1) ist bei allen numerischen Algorithmen mit einer Störmatrix mindestens in der Größenordnung der Maschinengenauigkeit *eps* zu rechnen. Nach Satz 4.6.9 können daraus Eigenwertstörungen der Größe $(eps)^{1/p}$ erwachsen, die wesentlich über der Maschinengenauigkeit liegen.

Allerdings können verschiedene Eigenwerte der gleichen Matrix sehr unterschiedlich auf Störungen reagieren. So läßt sich zeigen, daß für einen einfachen Eigenwert $\lambda(A)$ der Matrix A bei Störung durch eine Matrix εF, $\|F\|_2 = 1$, gilt

$$\lim_{\varepsilon \to 0} \left| \frac{\lambda(A + \varepsilon F) - \lambda(A)}{\varepsilon} \right| \leq \frac{1}{|y^* x|}, \tag{4.6.6}$$

wobei x und y die Rechts- und Links-Eigenvektoren der Länge eins zum Eigenwert $\lambda(A)$ sind. Der Eigenwert λ ist dann gut (schlecht) konditioniert, wenn x und y einen kleinen (großen) Winkel miteinander bilden. Da bei normalen Matrizen gilt $x = y$, folgt auch aus diesem Ergebnis die Robustheit des entsprechenden Eigenwertproblems.

Beispiel 4.6.10 $A = \begin{pmatrix} 1 & 1 \\ \varepsilon & 1 \end{pmatrix}$, $\lambda_{1/2} = 1 \pm \sqrt{\varepsilon}$.

$$x^{(1/2)} = \frac{1}{\sqrt{1+\varepsilon}} \begin{pmatrix} 1 \\ \pm\sqrt{\varepsilon} \end{pmatrix}, \quad y^{(1/2)} = \frac{1}{\sqrt{1+\varepsilon}} \begin{pmatrix} \pm\sqrt{\varepsilon} \\ 1 \end{pmatrix}.$$

Die Eigenwerte sind sehr schlecht konditioniert, denn

$$\frac{1}{|y^* x|} = \frac{1+\varepsilon}{2\sqrt{\varepsilon}}.$$

Die um ε veränderte Matrix $\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ besitzt den um $\sqrt{\varepsilon}$ gestörten doppelten Eigenwert 1.

Vor dem Hintergrund des Satzes 4.6.5 und des letzten Beispiels kann im unsymmetrischen Fall i.a. nicht erwartet werden, daß Eigenwerte oder Eigenvektoren bis auf Maschinengenauigkeit exakt berechnet werden können. Sie werden aber *numerisch akzeptabel* genannt, wenn sie *exakte* Eigenwerte/Eigenvektoren einer Matrix $A + F$ sind mit $\|F\| \cong eps\|A\|$. In diesem Sinn liefern die behandelten Transformationsverfahren (Hessenberg-, QR-, Jacobi-) wegen (4.6.1) numerisch akzeptable Approximationen.

5 Iterationsverfahren für große Matrizen

Die stetige Entwicklung bei Computern zu immer höheren Rechenleistungen und Speicherkapazitäten hat dazu geführt, daß immer feinere und komplexere Modelle realer Prozesse aufgestellt und numerisch gelöst werden können. Gerade dann machen sich aber auch Effizienzunterschiede verschiedener Verfahren (unterschiedliche Komplexitätsordnungen) immer stärker bemerkbar. Außerdem spielen Struktureigenschaften ein stärkere Rolle. Bei vielen Modellen treten lineare Gleichungssysteme oder Eigenwertprobleme mit sehr großen Matrizen als Unterprobleme auf. Wegen der meist lokal beschränkten Kopplung zwischen den Unbekannten (z.B. nur lokale Nachbarschaften) sind diese Matrizen oft dünn besetzt.

Bei einer Reihe von Iterationsverfahren kann diese Eigenschaft einfach ausgenutzt werden, etwa dann, wenn die Matrix A nicht explizit bekannt sein muß, sondern nur über Matrix-Vektor-Produkte Ax eingeht. Andere Verfahren wurden in der Numerik I behandelt, z.B. das Gesamtschritt-, Einzelschritt- und SOR-Verfahren. Unglücklicherweise wird bei den meisten Iterationsverfahren für wichtige Problemklassen die Konvergenz bei wachsender Problemgröße n immer schlechter (d.h. $\varrho(B_n) \to 1,\ n \to \infty$). Das früher ebenfalls behandelte Relaxationsverfahren läßt sich aber auf einfache Weise verbessern.

5.1 Polynomielle Beschleunigungsverfahren

Es wird die Fixpunktgleichung

$$x = Bx + r \qquad \iff \qquad Ax = r \quad \text{mit } A = I - B, \tag{5.1.1}$$

$x, r \in \mathbf{R}^n, A, B \in \mathbf{R}^{n\times n}$ betrachtet, die Lösung sei z. In der Numerik I wurde zu deren Lösung das *Relaxationsverfahren*

$$x^{(k+1)} = (1-\omega)x^{(k)} + \omega(Bx^{(k)} + r) = x^{(k)} + \omega(r - Ax^{(k)}),\ k = 0, 1, \ldots, \tag{5.1.2}$$

betrachtet mit dem Parameter $\omega \in \mathbf{R}$. Die Iterationsmatrix des Verfahrens war

$$B(\omega) = (1-\omega)I + \omega B = I - \omega A$$

und besaß bei reellen Eigenwerten $0 < \lambda_n(A) \leq \ldots \leq \lambda_1(A)$ einen minimalen Spektralradius

$$\varrho(B(\hat\omega)) = \frac{\lambda_1(A) - \lambda_n(A)}{\lambda_1(A) + \lambda_n(A)} \quad \text{in} \quad \hat\omega = \frac{2}{\lambda_1(A) + \lambda_n(A)}. \tag{5.1.3}$$

Der (asymptotische) Konvergenzfaktor $\varrho(B(\hat\omega))$ hängt alleine von dem Verhältnis λ_1/λ_n der Eigenwerte von A ab. In Anlehnung an die Numerik I wird daher die *Spektral-Konditionszahl* eine regulären Matrix definiert durch

$$\hat\kappa(A) := \varrho(A)\varrho(A^{-1}) = \frac{\max\{|\lambda_j(A)| :\ j = 1, \ldots, n\}}{\min\{|\lambda_j(A)| :\ j = 1, \ldots, n\}}. \tag{5.1.4}$$

Offensichtlich gilt $1 \le \hat{\kappa}(A) \le \kappa_p(A),\ p \ge 1$. Für symmetrische Matrizen, $A = A^{\mathsf{T}}$, gilt sogar $\hat{\kappa}(A) = \kappa_2(A)$. Der Konvergenzfaktor aus (5.1.3) läßt sich also in der Form

$$\min_{\omega \in \mathbf{R}} \varrho(B(\omega)) = \frac{\hat{\kappa}(A) - 1}{\hat{\kappa}(A) + 1} = 1 - \frac{2}{\hat{\kappa}(A) + 1} \tag{5.1.5}$$

schreiben. Die Optimalitätsaussage gilt genau genommen nur bei isolierter Betrachtung eines einzelnen Schrittes (5.1.2). Faßt man dagegen mehrere Schritte mit *wechselnden* Parametern ω_k zusammen, ergibt sich für den Fehler $x^{(m)} - z$ nach m Schritten

$$\begin{aligned} x^{(m)} - z &= x^{(m-1)} - z + \omega_m(Az - Ax^{(m-1)}) = (I - \omega_m A)(x^{(m-1)} - z) \\ &= (I - \omega_m A)(I - \omega_{m-1} A) \cdots (I - \omega_1 A)(x^{(0)} - z). \end{aligned} \tag{5.1.6}$$

Der Vorfaktor des Startfehlers $x^{(0)} - z$ ist also ein *Polynom* der Matrix A. Dessen Parameter ω_k sollen im folgenden so bestimmt werden, daß dieses Matrixpolynom "möglichst klein" wird. Zu nächst gilt

Satz 5.1.1 *Bei einer Matrix A mit reellen Eigenwerten*

$$0 < a \le \lambda_n \le \ldots \le \lambda_1 \le b$$

gilt für den Fehler bei der Richardson-Iteration

$$x^{(k)} := x^{(k-1)} + \omega_k(r - Ax^{(k-1)}),\ k = 1, \ldots, m, \tag{5.1.7}$$

mit Parametern ω_k nach m Schritten die Beziehung

$$x^{(m)} - z = p_m(A)(x^{(0)} - z). \tag{5.1.8}$$

Dabei ist $p_m \in \Pi_m$, $p_m(0) = 1,\ p_m(t) = (1 - \omega_1 t) \cdots (1 - \omega_m t)$ und

$$\varrho(p_m(A)) \le \max\{|p_m(t)| :\ t \in [a, b]\}. \tag{5.1.9}$$

Beweis Die Identität (5.1.8) entspricht (5.1.6), die Reihenfolge der Faktoren ist beliebig. Für jeden Eigenvektor x von A mit Eigenwert λ gilt

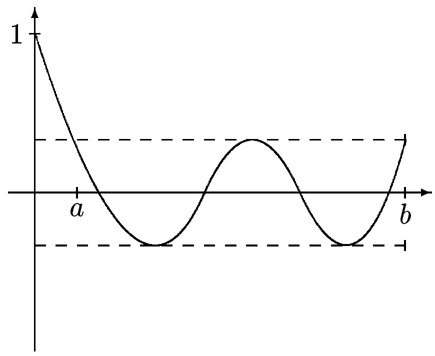
$$p_m(A)x = (I - \omega_m A) \ldots (I - \omega_2 A)(1 - \omega_1 \lambda)x = \ldots = p_m(\lambda)x.$$

Aus der Voraussetzung $\lambda_j \in [a, b]$ folgt die Schranke (5.1.9) daher für diagonalisierbare Matrizen, sie gilt aber auch allgemein. ∎

Statt für einen einzelnen Schritt kann der Konvergenzfaktor jetzt für einen Zyklus von m Schritten minimiert werden. Da die Eigenwerte der Matrix i.a. nicht bekannt sind, Eigenwert-Schranken aber mit akzeptablem Aufwand bestimmt werden können (vgl., z.B., Satz 4.6.5), minimiert man die Schranke (5.1.9),

$$\max\{|p_m(t)| :\ t \in [a, b]\} \stackrel{!}{=} \min,\ p_m \in \Pi_m,$$

unter der Nebenbedingung $p_m(0) = 1$. Die Lösung dieses Problems ist aus der Numerik I bekannt. Es geht auch hier darum, einen möglichst kleinen Funktionswert von p_m im Intervall $[a, b]$ zu erreichen, nur die Normierungsbedingung unterscheidet sich von der bei der Suche nach optimalen Interpolationsstützstellen.



Satz 5.1.2 *Unter allen Polynomen $p_m \in \Pi_m$ mit $p_m(0) = 1$ wird*

$$\|p\|_{[a,b]} := \max\{|p(t)| : \ t \in [a, b]\}$$

minimal bei

$$\hat{p}_m(t) := T_m\Big(\frac{b + a - 2t}{b - a}\Big)\Big/T_m\Big(\frac{b + a}{b - a}\Big), \tag{5.1.10}$$

wobei $T_m(s)$ das Tschebyscheff-Polynom vom Grad m ist. Es wird durch die Rekursion $T_0 = 1,\ T_1(s) := s$,

$$T_{k+1}(s) := 2s\,T_k(s) - T_{k-1}(s), \ k = 1, 2, \ldots \tag{5.1.11}$$

definiert und kann explizit dargestellt werden durch

$$T_m(s) = \begin{cases} \cos(m \arccos s), & |s| \le 1 \\ \cosh(m \operatorname{arcosh} s), & s > 1. \end{cases}$$

Die Parameter $\hat{\omega}_k$ von $\hat{p}_m(t) = \prod_{k=1}^{m}(1 - \hat{\omega}_k t)$ sind daher

$$\frac{1}{\hat{\omega}_k} = \frac{a + b}{2} - \frac{b - a}{2}\cos\frac{2k - 1}{2m}\pi, \ k = 1, \ldots, m. \tag{5.1.12}$$

Der Minimalwert der Maximumnorm ist

$$\|\hat{p}_m\|_{[a,b]} = 1\Big/T_m\Big(\frac{b + a}{b - a}\Big) \le 2\left(\frac{\sqrt{b/a} - 1}{\sqrt{b/a} + 1}\right)^m. \tag{5.1.13}$$

Beweis Nach Satz 2.6.1 aus der Numerik I minimiert $\hat{q}_m(t) := T_m\left(\frac{b+a-2t}{b-a}\right)$ die Norm $\|q_m\|_{[a,b]}$ unter allen Polynomen $q_m(t) = c_m t^m + \ldots$ mit festem höchsten Koeffizienten $c_m = -2^{m+1}/(b - a)^m$. Die neue Nebenbedingung der Form $p_m(0) = 1$ wird durch die Wahl $\hat{p}_m = \hat{q}_m(t)/\hat{q}_m(0)$ erfüllt. Die Kehrwerte $1/\hat{\omega}_k$ entsprechen daher den Nullstellen von $\hat{p}_m$. Die erste Schranke (5.1.13) gilt nun wegen $\|\hat{q}_m\|_{[a,b]} = 1$.

Zur Normierung wird in (5.1.10) ein Funktionswert $T_m(s)$ mit $s > 1$ verwendet. Um auch für diesen Fall eine explizite Darstellung der Polynome aus (5.1.11) zu erhalten, wird $s = \cosh\sigma$ gesetzt. Wegen

$$\cosh(k \pm 1)\sigma = \cosh k\sigma\ \cosh\sigma \pm \sinh k\sigma\ \sinh\sigma,$$

sowie $\cosh 0 = 1$ gilt daher hier die explizite Beziehung $T_m(s) = \cosh(m \operatorname{arcosh} s)$. Die letzte Ungleichung in (5.1.13) folgt aus einer weiteren Darstellung der Tschebyscheff-Polynome mit Argument $q = (b+a)/(b-a) = (\kappa+1)/(\kappa-1)$, $\kappa = b/a$, nämlich

$$T_m\Big(\frac{\kappa+1}{\kappa-1}\Big) = \frac{1}{2}\Big[\Big(\frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1}\Big)^m + \Big(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\Big)^m\Big] \geq \frac{1}{2}\Big(\frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1}\Big)^m. \qquad \blacksquare$$

Der Konvergenz-Vorteil bei Wahl der Relaxationsparameter nach Satz 5.1.2 für einen Zyklus der Länge $m > 1$ gegenüber dem einfachen Relaxationsverfahren (5.1.2) der Zykluslänge $m = 1$ soll jetzt bei $[a,b] = [\lambda_1, \lambda_n]$ analysiert werden. Dann hängt (5.1.13) wieder nur von der Konditionszahl $\hat{\kappa}(A)$ ab und für große Werte derselben, $\hat{\kappa}(A) \gg 1$, ist

$$\|p_m\|_{[a,b]} = 1/T_m\Big(\frac{\hat{\kappa}+1}{\hat{\kappa}-1}\Big) \doteq 1/T_m\Big(1 + \frac{2}{\hat{\kappa}}\Big).$$

Wegen $\operatorname{arcosh}(1 + 2/\hat{\kappa}) \doteq 2/\sqrt{\hat{\kappa}}$ ist für nicht zu große m

$$\|p_m\|_{[a,b]} \doteq \frac{1}{\cosh(2m/\sqrt{\hat{\kappa}})} \doteq 1 - \frac{2}{\hat{\kappa}} m^2. \tag{5.1.14}$$

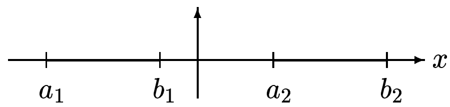
Im Vergleich dazu liefern bei gleichem Aufwand m Schritte mit festem $\hat{\omega}$ nach (5.1.5) nur den Kontraktionsfaktor

$$\varrho(B(\hat{\omega}))^m \doteq (1 - \frac{2}{\hat{\kappa}})^m \doteq 1 - \frac{2}{\hat{\kappa}} m.$$

Im betrachteten, praktisch wichtigen Fall $\hat{\kappa} \gg 1$ wird die Konvergenz also erheblich verbessert. Ein m-Zyklus der Richardson-Iteration (5.1.7) mit optimalen Parametern (5.1.12) entspricht ca. m^2 Schritten der einfachen Relaxation!

Optimale Parameter für die Richardson-Iteration können auch noch bei komplexen Eigenwerten angegeben werden, wenn man Ellipsen in der komplexen Ebene zur Einschließung verwendet. Auch wenn im reellen positive und *negative* Eigenwerte auftreten, lassen sich oft noch gute Parameter konstruieren. Gilt etwa

$$\lambda_j(A) \subseteq [a_1, b_1] \cup [a_2, b_2] =: S, \; a_1 \leq b_1 < 0 < a_2 \leq b_2,$$



wobei die beiden Teilintervalle gleiche Länge besitzen, $b_1 - a_1 = b_2 - a_2$, können diese mit der Abbildung $t \mapsto y = (t-\alpha)^2$, $\alpha = (a_1 + b_2)/2 = (b_1 + a_2)/2$ auf das positive Intervall $[(a_2 - \alpha)^2, (b_2 - \alpha)^2]$ abgebildet werden. Nun sei $q_m(y) = (1 - \omega_1 y) \cdots (1 - \omega_m y)$ das normierte Tschebyscheff-Polynom zu diesem positiven Intervall. Dann ist

$$p_{2m}(t) := \frac{q_m((t-\alpha)^2)}{q_m(\alpha^2)} = \prod_{j=1}^m \frac{1 - \omega_j (t-\alpha)^2}{1 - \omega_j \alpha^2} = \prod_{j=1}^m \Big(1 - \frac{\sqrt{\omega_j}}{1 + \sqrt{\omega_j}\alpha} t\Big)\Big(1 + \frac{\sqrt{\omega_j}}{1 - \sqrt{\omega_j}\alpha} t\Big)$$

ein Polynom mit

$$\|p_{2m}\|_S \leq 2\Big(\frac{\kappa - 1}{\kappa + 1}\Big)^m, \; \kappa := \frac{b_2 - a_1}{a_2 - b_1} > 1.$$

Als Parameter der Iteration sind hier also $\{\sqrt{\omega_j}/(1 + \sqrt{\omega_j}\alpha), -\sqrt{\omega_j}/(1 - \sqrt{\omega_j}\alpha)\}_{j=1}^m$ zu wählen für einen Zyklus der Länge $2m$.

Ein wesentlicher Nachteil des Richardson-Verfahrens ist der, daß die Konvergenzaussage (5.1.9), (5.1.13) erst nach dem Durchlaufen eines vollen m-Zyklus zutrifft, bei periodischer Wahl $\omega_{k+jm} := \omega_k$ also nur für $x^{(m)}, x^{(2m)}, \ldots$. Der Fehler der restlichen Iterierten kann sogar erheblich über dem Startfehler liegen. Daher muß man sich vor Beginn für eine Mindestzahl m von Iterationen entscheiden.

Dies läßt sich im Fall des positiven reellen Spektrums (und einigen anderen Fällen) auf folgende Weise umgehen. Mit den skalierten Tschebyscheff-Polynomen war der Fehler (5.1.8)

$$x^{(m)} - z = \hat{p}_m(A)(x^{(0)} - z).$$

Für *Orthogonal-Polynome* gilt aber eine zweistufige *Rekursionsformel* (vgl. Numerik I, §2.4)

$$\hat{p}_{m+1}(t) = \alpha_m(t - \gamma_m)\hat{p}_m(t) + \beta_m\hat{p}_{m-1}(t), \; m = 1, 2, \ldots \tag{5.1.15}$$

Betrachtet man nun für beliebiges k die Vektoren

$$y^{(k)} := z + \hat{p}_k(A)(x^{(0)} - z) = \hat{p}_k(A)x^{(0)} + (I - \hat{p}_k(A))z, \; k = 0, 1, \ldots,$$

die nur für $k = 0$ bzw. $k = m$ mit denen der Richardson-Iteration übereinstimmen, so folgt aus (5.1.15)

$$\begin{aligned}
y^{(k+1)} &= \hat{p}_{k+1}(A)(x^{(0)} - z) + z \\
&= \alpha_k(A - \gamma_k I)\hat{p}_k(A)(x^{(0)} - z) + \beta_k\hat{p}_{k-1}(A)(x^{(0)} - z) + z \\
&= \alpha_k(A - \gamma_k I)(y^{(k)} - z) + \beta_k(y^{(k-1)} - z) + z \\
&= \alpha_k(Ay^{(k)} - r) - \alpha_k\gamma_k y^{(k)} + \beta_k y^{(k-1)} + \underbrace{(1 - \beta_k + \alpha_k\gamma_k)}_{=0} z.
\end{aligned}$$

Der Vorfaktor bei z verschwindet dabei, da $\hat{p}_k(0) = 1 \, \forall k$, vgl. (5.1.8). Dies entspricht aber genau $1 = \beta_k - \alpha_k\gamma_k$ in (5.1.15). Der Vektor $y^{(k+1)}$, der nun in jedem Schritt $k+1$ einen minimalen Fehlerfaktor $\varrho(p_{k+1}(A))$ besitzt, kann aus den Vektoren $y^{(k)}$ und $y^{(k-1)}$ berechnet werden. Für die Koeffizienten der Rekursion (5.1.15) der Polynome (5.1.10) ergeben sich mit

$$q := \frac{b + a}{b - a}, \quad d := \frac{2}{b + a}, \quad \varrho_k := 2q\frac{T_k(q)}{T_{k+1}(q)} \tag{5.1.16}$$

die Beziehungen $\alpha_k = -\varrho_k d, \; \alpha_k\gamma_k = -\varrho_k, \; \beta_k = 1 - \varrho_k$. Denn es ist $\hat{p}_k(t) = T_k(q(1 - dt))/T_k(q)$ und aus der Rekursion (5.1.11) folgt

$$\begin{aligned}
\hat{p}_{k+1}(t) &= \frac{T_{k+1}(q(1 - dt))}{T_{k+1}(q)} = 2q(1 - dt)\frac{T_k(q(1 - dt))}{T_{k+1}(q)} - \frac{T_{k-1}(q(1 - dt))}{T_{k+1}(q)} \\
&= 2q(1 - dt)\frac{T_k(q)}{T_{k+1}(q)}\,\hat{p}_k(t) - \frac{T_{k-1}(q)}{T_{k+1}(q)}\,\hat{p}_{k-1}(t) \\
&= \varrho_k(1 - dt)\hat{p}_k(t) + (1 - \varrho_k)\hat{p}_{k-1}(t).
\end{aligned}$$

Beim letzten Koeffizienten $1 - \varrho_k$ wurde (5.1.11) noch einmal verwendet. Mit diesen Bezeichnungen wird nun die *Tschebyscheff-Iteration* formuliert. Wegen der Norm-Konvergenzaussage wird dabei nur der symmetrische Fall betrachtet.

Satz 5.1.3 *Die Eigenwerte der symmetrischen $n \times n$-Matrix A seien im Intervall $[a, b]$, $0 < a \leq b$, enthalten, damit ist $\kappa := b/a \geq \hat{\kappa}(A)$. Für die Vektoren der* Tschebyscheff-Iteration *mit Startvektor $y^{(0)} \in \mathbf{R}^n$,*

$$\left\{\begin{array}{rcl} y^{(1)} & := & y^{(0)} + d(r - Ay^{(0)}), \\ y^{(k+1)} & := & \varrho_k\Big(y^{(k)} + d(r - Ay^{(k)})\Big) + (1 - \varrho_k)y^{(k-1)}, \ k = 1, 2, \ldots, \end{array}\right. \tag{5.1.17}$$

und den in (5.1.16) definierten Parametern, gilt die Fehlerschranke

$$\|y^{(k)} - z\|_2 \leq \frac{1}{T_k(q)}\|y^{(0)} - z\|_2 \leq 2\Big(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1}\Big)^k \|y^{(0)} - z\|_2, \ k \geq 1. \tag{5.1.18}$$

Beweis Nach obiger Konstruktion gilt für jedes k

$$y^{(k)} - z = \hat{p}_k(A)\,(y^{(0)} - z).$$

Wegen der Symmetrie von A ist $\|\hat{p}_k(A)\|_2 = \varrho(\hat{p}_k(A)) \leq 1/T_k(q)$, vgl. (5.1.13). ■

Bemerkung: Die Tschebyscheff-Iteration (5.1.17) besteht offensichtlich aus einer zusätzlichen Extrapolation ($1 < \varrho_k \leq 2$, vgl. Übung) bei der einfachen, optimalen Relaxation (5.1.2), (5.1.3):

$$\tilde{y}^{(k+1)} := y^{(k)} + \frac{2}{a+b}(r - Ay^{(k)}), \quad y^{(k+1)} = \varrho_k \tilde{y}^{(k+1)} + (1 - \varrho_k)y^{(k-1)}.$$

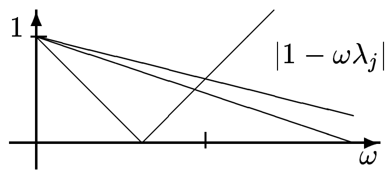
Praktische Durchführung

Sowohl bei der Richardson- als auch der Tschebyscheff-Iteration werden zur Durchführung gute (Ab-) Schätzungen der extremalen Eigenwerte $\lambda_1(A), \lambda_n(A)$ benötigt. Nach den Ergebnissen von §3 bietet sich dazu die Vektor-Iteration an. Diese kann sogar ohne zusätzlichen Aufwand mit der einfachen Relaxation (5.1.2) kombiniert werden. Sowohl bei Betrachtung des Fehlers $x^{(k)} - z$ als auch des Defekts $d^{(k)} := r - Ax^{(k)}$ entspricht diese nämlich einer Vektor-Iteration,

$$d^{(k+1)} := r - Ax^{(k+1)} = r - A(x^{(k)} + \omega d^{(k)}) = (I - \omega A)d^{(k)} \tag{5.1.19}$$

mit der Matrix $I - \omega A$. Diese Beziehung kann im Fall positiver, reeller Eigenwerte $0 < a \leq \lambda_n \leq \ldots \leq \lambda_1 \leq b$ ausgenutzt werden. Dazu sei an die Aussage

$$\varrho(I - \omega A) = \left\{\begin{array}{ll} \omega\lambda_1 - 1 & \text{für} \quad \omega > \\ 1 - \omega\lambda_n & \text{für} \quad \omega \leq \end{array}\right. \hat{\omega} = \frac{2}{\lambda_n + \lambda_1}.$$



erinnert. Durch Variation von ω können daher λ_n und λ_1 bei (5.1.19) geschätzt werden:

a) Für $\omega := 1/\|A\| \le 1/\lambda_1$ ist $\varrho(I-\omega A) = 1-\omega\lambda_n > 0$. Analog zu (4.4.3) bekommt man aus (5.1.19) eine Schätzung für λ_n.

b) Für beliebiges $x \in \mathbf{R}^n$ ist $\omega := 2/(R_A[x]+\lambda_n) \ge \hat{\omega}$. Mit diesem ω ist daher $\varrho(I-\omega A) = \omega\lambda_1 - 1$ und aus der Eigenwert-Schätzung bei (5.1.19) kann eine Schätzung von λ_1 abgeleitet werden. Bei ungünstiger Wahl von x kann allerdings ω zu groß und daher $\varrho(I-\omega A) > 1$ sein. Dann würde der Fehler bei der Iteration gegenüber der Startnäherung anwachsen.

c) Mit diesen Eigenwert-Schätzungen für λ_n und λ_1 kann jetzt auf die Richardson- oder Tschebyscheff-Iteration umgeschaltet werden.

Wegen des engen Zusammenhangs mit dem *Eigenwert-Problem* können Richardson- und Tschebyscheff-Methode auch zur Beschleunigung der Vektor-Iteration eingesetzt werden. Statt der Folge (4.3.2)

$$v^{(m)} := A^m z^{(0)} \quad \text{wird betrachtet } v^{(m)} := p_m(A)z^{(0)},\ p_m \in \Pi_m.$$

Dabei wird p_m im Fall $0 \le \lambda_n \le \ldots \lambda_1$ so optimiert, daß der Konvergenzfaktor

$$\|p_m\|_{[0,\lambda_2]} \Big/ |p_m(\lambda_1)|$$

minimal wird. Dazu wird allerdings eine Schätzung für λ_2 benötigt. Bei Verwendung der optimalen Tschebyscheff-Polynome verbessert sich die Konvergenz von

$$\left|\frac{\lambda_2}{\lambda_1}\right|^m \quad \text{auf} \quad 1\Big/\left|T_m\Big(2\frac{\lambda_1}{\lambda_2}-1\Big)\right| \le \left|\frac{\lambda_2}{\lambda_1}\right|^m \frac{2}{(1+\sqrt{1-\lambda_2/\lambda_1})^{2m}}$$

Ein der Tschebyscheff-Iteration ähnliches Verfahren mit 'eingebauter' Parameterbestimmung wird im folgenden Abschnitt besprochen.

5.2 Das Verfahren der Konjugierten Gradienten

Bei einer postiv definiten symmetrischen Matrix kann die Lösung z des linearen Gleichungssystems $Ax = b$ als eindeutiges Minimum eines konvexen Funktionals $\varphi:\ \mathbf{R}^n \mapsto \mathbf{R}$ charakterisiert und daher durch ein Abstiegsverfahren approximiert werden.

Satz 5.2.1 *Die Matrix $A \in \mathbf{R}^{n\times n}$ sei symmetrisch und positiv definit. Dann wird durch*

$$(x,y)_A := y^\mathsf{T} Ax, \quad \|x\|_A := \sqrt{(x,x)_A} = \sqrt{x^\mathsf{T} Ax}, \tag{5.2.1}$$

$x, y \in \mathbf{R}^n$ ein Innenprodukt und eine Norm definiert. Die Lösung z des linearen Gleichungssystems $Ax = b$ ist eindeutiges Minimum von

$$\varphi(x) := \frac{1}{2}x^\mathsf{T} Ax - b^\mathsf{T} x, \quad x \in \mathbf{R}^n. \tag{5.2.2}$$

Beweis Wegen der Definitheit von A ist $(x,x)_A = x^\top Ax > 0\ \forall x \neq 0$ und $\|x\|_A$ daher eine Norm. Für φ gilt mit $b = Az$ die Darstellung

$$\begin{aligned} 2\varphi(x) &= x^\top Ax - 2b^\top x = x^\top Ax - 2z^\top Ax = (x,x)_A - 2(x,z)_A \\ &= \|x-z\|_A^2 - \|z\|_A^2. \end{aligned} \tag{5.2.3}$$

Also ist z eindeutiges Minimum von φ. ∎

Die Funktion φ ist streng konvex. An der Stelle x ist die Richtung des steilsten Abstiegs

$$-\mathrm{grad}\varphi(x) = b - Ax =: d,$$

also der Defekt von x. Bei einer Liniensuche in dieser Richtung kann das Minimum auf dem Strahl $s(t) := x + td,\ t \in \mathbf{R}$, exakt bestimmt werden,

$$\begin{aligned} \varphi(s(t)) &= \frac{1}{2}(x+td)^\top A(x+td) - b^\top(x+td) \\ &= \frac{1}{2}t^2 d^\top Ad + td^\top Ax + \frac{1}{2}x^\top Ax - b^\top x - td^\top b \\ &= \frac{1}{2}t^2 d^\top Ad - td^\top d + \varphi(x) \\ &= \frac{1}{2}d^\top Ad\Big(t - \frac{d^\top d}{d^\top Ad}\Big)^2 + \varphi(x) - \frac{1}{2}\frac{(d^\top d)^2}{d^\top Ad}. \end{aligned} \tag{5.2.4}$$

Das Minimum wird also in $\hat{t} = d^\top d/d^\top Ad = 1/R_A[d]$ angenommen. Diese *Methode des steilsten Abstiegs* kann auch iterativ angewendet werden

$$d^{(k)} := b - Ax^{(k)}, \quad x^{(k+1)} := x^{(k)} + \frac{1}{R_A[d^{(k)}]}d^{(k)}, \quad k = 0,1,\ldots\ . \tag{5.2.5}$$

Allerdings ist ihre Konvergenz oft sehr schlecht. Dazu wird die verschobene Zielfunktion $2\varphi(y) + \|z\|_A^2 = \|y-z\|_A^2$ im Minimum von (5.2.4) betrachtet. Hier ist

$$2\varphi(s(\hat{t})) + \|z\|_A^2 = \|s(\hat{t}) - z\|_A^2 = \|x-z\|_A^2 - \frac{(d^\top d)^2}{d^\top Ad}.$$

Da zwischen der A-Norm und der Euklid-Norm der Zusammenhang

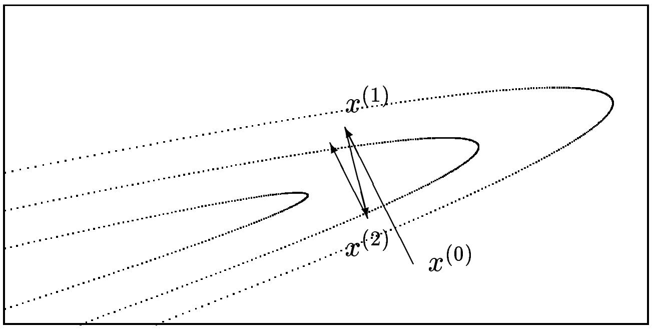
$$\lambda_{\min}(A)\|x\|_2^2 \le \|x\|_A^2 \le \lambda_{\max}(A)\|x\|_2^2 \tag{5.2.6}$$

(Satz 4.6.1) besteht, folgt mit $d^{(k)} = b - Ax^{(k)} = A(z - x^{(k)})$ weiter

$$\begin{aligned} \|x^{(k+1)} - z\|_A^2 &= \|x^{(k)} - z\|_A^2 - \frac{\|A(x^{(k)} - z)\|_2^4}{\|A(x^{(k)} - z)\|_A^2} \le \|x^{(k)} - z\|_A^2 - \frac{\|A(x^{(k)} - z)\|_2^2}{\|A\|_2} \\ &\le \Big(1 - \frac{\lambda_{\min}(A)}{\lambda_{\max}(A)}\Big)\|x^{(k)} - z\|_A^2 = \Big(1 - \frac{1}{\hat{\kappa}(A)}\Big)\|x^{(k)} - z\|_A^2. \end{aligned} \tag{5.2.7}$$

Für $\hat{\kappa}(A) \gg 1$ ergibt sich somit eine ähnlich langsame Konvergenz wie beim einfachen Relaxationsverfahren, (5.1.5).

Geometrischer Hintergrund: Die Niveau-Flächen von φ sind Ellipsoide mit den Halbachsen $1/\sqrt{\lambda_j(A)}$ und Zentrum z. Für $\hat{\kappa} \gg 1$ sind die Ellipsoide sehr flach. Das Verfahren springt dann von einer Wand des Tals (im Bild: $\mathbf{R}^2$) zur anderen, kommt aber längs des Tals nur langsam voran.



Ein anschaulicher Grund für die langsame Konvergenz ist das fehlende 'Gedächtnis' des Verfahrens, da in späteren Schritten beinahe in schon früher benutzte Richtungen gegangen wird. Die Suchrichtung müßte also zur Verbesserung des Verfahrens modifiziert werden, diese wird jetzt mit $p^{(k)}$ bezeichnet im k-ten Schritt des Verfahrens. Ein Verfahren mit vollem Gedächtnis sollte im k-ten Schritt mit Startvektor $x^{(0)} = 0$ sogar das Minimalproblem

$$\min\{\varphi(x) : \ x \in [p^{(1)}, \ldots, p^{(k)}]\} \tag{5.2.8}$$

über dem von allen bisherigen Suchrichtungen aufgespannten Unterraum berechnen. Dazu seien

$$P_j := (p^{(1)}, \ldots, p^{(j)}) \in \mathbf{R}^{n \times j}, \ j \le k,$$

die Matrizen der Suchrichtungen. Mit $t \in \mathbf{R}$, $y \in \mathbf{R}^{k-1}$ sei

$$s(t, y) := P_{k-1}y + tp^{(k)}$$

der Suchvektor. Für den Wert φ gilt hier

$$\begin{aligned} \varphi(s(t,y)) &= \frac{1}{2}(P_{k-1}y + tp^{(k)})^\top A(P_{k-1}y + tp^{(k)}) - b^\top(P_{k-1}y + tp^{(k)}) \\ &= \varphi(P_{k-1}y) + t({p^{(k)}}^\top AP_{k-1}y - b^\top p^{(k)}) + \frac{1}{2}t^2(p^{(k)}, p^{(k)})_A. \end{aligned} \tag{5.2.9}$$

In dieser Beziehung kommt nur durch den Ausdruck $t{p^{(k)}}^\top AP_{k-1}y = t(p^{(k)}, P_{k-1}y)_A$ eine Kopplung der Unbekannten t und y zustande. Gilt dagegen

$$(p^{(k)}, P_{k-1}y)_A = 0 \ \forall y \quad \iff \quad (p^{(k)}, p^{(j)})_A = 0 \ \forall j < k, \tag{5.2.10}$$

dann zerfällt das Minimalproblem (5.2.8) in zwei unabhängige Teilprobleme, dessen Minimum bzgl. t in

$$t_k = b^\top p^{(k)} / {p^{(k)}}^\top Ap^{(k)} \tag{5.2.11}$$

angenommen wird. Als erstes Zwischenergebnis läßt sich also zusammenfassen:

Wenn schon $x^{(k-1)} \in [p^{(1)}, \ldots, p^{(k-1)}]$ das $k-1$-dimensionale Minimalproblem (5.2.8) löst und $p^{(k)}$ A-orthogonal zu $p^{(1)}, \ldots, p^{(k-1)}$ ist, vgl. (5.2.10), dann führt die exakte, eindimensionale Liniensuche

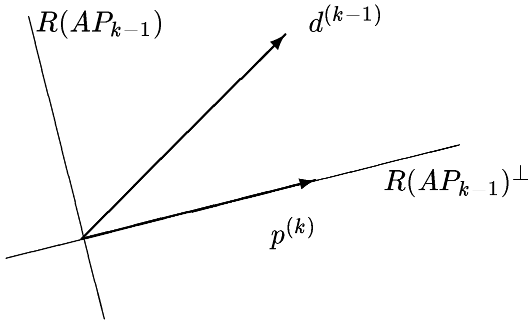
$$x^{(k)} := x^{(k-1)} + t_k p^{(k)} \tag{5.2.12}$$

auf die k-dimensionale Minimallösung in (5.2.8).

Die Abnahme der Zielfunktion φ ist dabei

$$\varphi(x^{(k)}) = \varphi(x^{(k-1)}) - \frac{1}{2}\frac{(b^\top p^{(k)})^2}{p^{(k)\top} A p^{(k)}} = \varphi(x^{(k-1)}) - \frac{1}{2}\frac{(d^{(k-1)\top} p^{(k)})^2}{p^{(k)\top} A p^{(k)}},$$

mit $d^{(k-1)} = b - Ax^{(k-1)}$, denn nach (5.2.10) ist $p^{(k)\top} A x^{(k-1)} = 0$. Damit dieser Iterationsschritt einen möglichst tiefen Abstieg erbringt, sollte die Suchrichtung $p^{(k)}$ in die Richtung von $d^{(k-1)}$ zeigen, unter Beachtung der Nebenbedingung (5.2.10). Diese Richtung kann folgendermaßen charakterisiert werden.



Lemma 5.2.2 *Der Vektor $p^{(k)}$, in dem*

$$\min\{\|p - d^{(k-1)}\|_2 : \ p \in \mathbf{R}^n, \, p^\top A P_{k-1} = 0\}$$

angenommen wird, hat die Form

$$p^{(k)} = d^{(k-1)} - AP_{k-1} v^{(k-1)}, \ v^{(k-1)} \in \mathbf{R}^{k-1}, \tag{5.2.13}$$

mit der Lösung $v^{(k-1)}$ des linearen Ausgleichsproblems

$$\min\{\|d^{(k-1)} - AP_{k-1} v\|_2 : \ v \in \mathbf{R}^{k-1}\}. \tag{5.2.14}$$

Beweis Nach Satz 2.1.2 erfüllt die Lösung von (5.2.14) die Normalengleichung

$$0 = (AP_{k-1})^\top (d^{(k-1)} - AP_{k-1} v^{(k-1)}) = (AP_{k-1})^\top p^{(k)},$$

d.h., (5.2.10). Daher ist auch $p^{(k)}$ die Orthogonalprojektion von $d^{(k-1)}$ auf $R(AP_{k-1})^\perp$, also der nächstgelegene Vektor. ∎

Damit $p^{(k)}$ aus (5.2.13), (5.2.14) als Suchrichtung verwendet werden kann, muß geprüft werden, wann dieser Vektor null wird. Es ist (für $x^{(k-1)} = P_{k-1} y$) $p^{(k)} = 0$ äquivalent mit

$$\begin{aligned} 0 &= \min_v \|d^{(k-1)} - AP_{k-1} v\|_2 = \min_v \|b - Ax^{(k-1)} - AP_{k-1} v\|_2 \\ &= \min \|b - AP_{k-1}(y + v)\|_2. \end{aligned}$$

Dann ist aber $b \in R(AP_{k-1}) = [Ap^{(1)}, \ldots, Ap^{(k-1)}] \iff z = A^{-1} b \in [p^{(1)}, \ldots, p^{(k-1)}]$. Dies bedeutet

$$p^{(k)} = 0 \iff d^{(k-1)} = 0 \iff x^{(k-1)} = z. \tag{5.2.15}$$

Die Konstruktion (5.2.14) der Suchrichtung bricht also nur zusammen, wenn die Lösung z schon durch $x^{(k-1)}$ angenommen wird ('günstiger Abbruch'). Insbesondere liefert (bei exakter Rechnung) spätestens die n-te Iterierte die exakte Lösung $x^{(n)} = z$. Die bisherigen Ergebnisse werden nun zur Formulierung einer 'Nullversion' des Verfahrens benutzt, bevor später überlegt wird, wie die Suchrichtungen $p^{(k)}$ effizient zu berechnen sind.

CG-0-Version:

$$
\boxed{\begin{array}{l}
x^{(0)} := 0;\ d^{(0)} := b;\ k := 0;\\
\text{solange } d^{(k)} \neq 0:\\
\quad k := k+1;\\
\quad \text{falls } k = 1 \text{ dann } p^{(1)} := d^{(0)}\\
\qquad \text{sonst } p^{(k)} := d^{(k-1)} - AP_{k-1}v^{(k-1)} \text{ nach (5.2.14)};\\
\quad t_k := {p^{(k)}}^{\mathsf T} d^{(k-1)} / {p^{(k)}}^{\mathsf T} A p^{(k)};\\
\quad x^{(k)} := x^{(k-1)} + t_k p^{(k)};\\
\quad d^{(k)} := b - Ax^{(k)};
\end{array}} \tag{5.2.16}
$$

Für das Verfahren der Konjugierten Gradienten (conjugate-gradient-method, cg-method) können die wichtigsten theoretischen Eigenschaften schon aus der Nullversion abgeleitet werden.

Satz 5.2.3 *Nach Ausführung der Schritte $j = 1, \ldots, k$ in (5.2.16) mit $d^{(j)} \neq 0$ gilt*

$$d^{(j)} = d^{(j-1)} - t_j A p^{(j)}, \quad P_j^{\mathsf T} d^{(j)} = 0,\ j = 1, \ldots, k. \tag{5.2.17}$$

Die verwendeten Suchrichtungen und Defekte bilden Basen der Krylov-Räume

$$\mathcal{K}_j := [b, Ab, A^2 b, \ldots, A^{j-1} b],$$

$$[p^{(1)}, \ldots, p^{(j)}] = [d^{(0)}, \ldots, d^{(j-1)}] = \mathcal{K}_j,\ j = 1, \ldots, k. \tag{5.2.18}$$

Beweis a) Aus $x^{(j)} = x^{(j-1)} + t_j p^{(j)}$ folgt $d^{(j)} = b - Ax^{(j)} = d^{(j-1)} - t_j A p^{(j)}$. Im Unterraum $[p^{(1)}, \ldots, p^{(j)}] = R(P_j)$ hat die Funktion φ die Gestalt

$$\varphi(P_j y) = \frac{1}{2} y^{\mathsf T} (P_j^{\mathsf T} A P_j) y - y^{\mathsf T} (P_j^{\mathsf T} b),\ y \in \mathbf{R}^j.$$

Die Minimalstelle $\hat{y}$ dieser Funktion erfüllt nach Satz 5.2.1, angewendet auf die ebenfalls positiv definite Matrix $P_j^{\mathsf T} A P_j$, das Gleichungssystem $0 = P_j^{\mathsf T} A P_j \hat{y} - P_j^{\mathsf T} b = P_j^{\mathsf T} (A P_j \hat{y} - b)$. Nach Konstruktion ist $P_j \hat{y} = x^{(j)}$, also gilt $P_j^{\mathsf T} d^{(j)} = 0$.

b) Für $j = 1$ ist (5.2.18) offensichtlich erfüllt. Unter der Annahme, (5.2.18) gelte für $j < k$, folgt durch Induktion aus (5.2.17) $d^{(j)} \in \mathcal{K}_{j+1}$ und daher nach (5.2.13)

$$p^{(j+1)} = \underbrace{d^{(j)}}_{\in \mathcal{K}_{j+1}} - A \underbrace{P_j v^{(j)}}_{\in \mathcal{K}_j} \in \mathcal{K}_{j+1}.$$

Da $d^{(j)} \perp \mathcal{K}_j$ und $p^{(j+1)} \perp_A \mathcal{K}_j$ haben $[d^{(0)}, \ldots, d^{(j)}]$ und $[p^{(1)}, \ldots, p^{(j+1)}]$ jeweils Dimension $j+1$ und stimmen daher mit $\mathcal{K}_{j+1}$ überein. ∎

Die folgende Eigenschaft gab dem Konjugierte-Gradienten-Verfahren seinen Namen ('orthogonal'='konjugiert', $-\mathrm{grad}\,\varphi$=Defekt).

Satz 5.2.4 *Nach k Schritten von (5.2.16) (mit $d^{(j)} \neq 0 \forall j \leq k$) sind die Defekte $d^{(0)}, \dots, d^{(k)}$ paarweise orthogonal.*

Beweis Nach (5.2.17) und (5.2.18) gilt für jedes $j = 1, \dots, k$,

$$d^{(j)\mathsf{T}} P_j = 0 \iff d^{(j)} \perp [p^{(1)}, \dots, p^{(j)}] = \mathcal{K}_j = [d^{(0)}, \dots, d^{(j-1)}]. \quad \blacksquare$$

Somit konstruiert das cg-Verfahren sowohl Orthogonalbasen $\{d^{(0)}, \dots, d^{(j-1)}\}$ als auch A-Orthogonalbasen $\{p^{(1)}, \dots, p^{(j)}\}$ für $\mathcal{K}_j$, $j \leq k$. Ersichtlich ist (5.2.16) unabhängig von der Länge $\|p^{(k)}\|_2$. Daher kann im k-ten Schritt *ein* $p^{(k)}$ in dem durch (5.2.13) festgelegten eindimensionalen Unterraum $\mathcal{K}_k \cap (A\mathcal{K}_{k-1})^{\perp}$ aus dem schon bekannten $d^{(k-1)}$, $\mathcal{K}_k \ni d^{(k-1)} \perp_2 \mathcal{K}_{k-1}$ durch Projektion berechnet werden bei Verwendung des A-Innenprodukts:

$$p^{(k)} := d^{(k-1)} - \sum_{j=1}^{k-1} \frac{(d^{(k-1)}, p^{(j)})_A}{(p^{(j)}, p^{(j)})_A} p^{(j)} = d^{(k-1)} - \frac{d^{(k-1)\mathsf{T}} A p^{(k-1)}}{p^{(k-1)\mathsf{T}} A p^{(k-1)}} p^{(k-1)}. \tag{5.2.19}$$

Dabei tritt nur noch $p^{(k-1)}$ auf, da $d^{(k-1)} \perp \mathcal{K}_{k-1} \supseteq A\mathcal{K}_{k-2} \ni Ap^{(j)}$, $j < k-1$. Mit (5.2.19) wird (5.2.16) ein sehr effizientes Verfahren, da pro Schritt nur zwei Matrix-Vektor-Multiplikationen benötigt werden bei $d^{(k)} = b - Ax^{(k)}$ und $Ap^{(k)}$. Die erste davon kann ebenfalls noch eingespart werden durch Berechnung des Defekts über (5.2.17). Wegen (5.2.19) gilt

$$t_k = \frac{p^{(k)\mathsf{T}} d^{(k-1)}}{p^{(k)\mathsf{T}} A p^{(k)}} = \frac{\|d^{(k-1)}\|_2^2}{p^{(k)\mathsf{T}} A p^{(k)}}.$$

Aus (5.2.17) folgt für den Koeffizienten in (5.2.19) außerdem

$$\frac{t_{k-1} d^{(k-1)\mathsf{T}} A p^{(k-1)}}{t_{k-1} p^{(k-1)\mathsf{T}} A p^{(k-1)}} = -\frac{\|d^{(k-1)}\|_2^2}{\|d^{(k-2)}\|_2^2} =: -\beta_{k-1}.$$

Diese Vereinfachungen führen schließlich auf die endgültige Form des *Verfahrens der Konjugierten Gradienten.*

CG-1-Version:

$$\boxed{\begin{array}{l} x^{(0)} := 0;\ d^{(0)} := b;\ p^{(1)} := b;\ k := 0; \\ \text{solange } d^{(k)} \neq 0: \\ \quad k := k+1; \\ \quad t_k := \|d^{(k-1)}\|_2^2 / p^{(k)\mathsf{T}} A p^{(k)}; \\ \quad x^{(k)} := x^{(k-1)} + t_k p^{(k)}; \\ \quad d^{(k)} := d^{(k-1)} - t_k A p^{(k)}; \\ \quad \beta_k := \|d^{(k)}\|_2^2 / \|d^{(k-1)}\|_2^2; \\ \quad p^{(k+1)} := d^{(k)} + \beta_k p^{(k)}; \end{array}} \tag{5.2.20}$$

In dieser Form benötigt das Verfahren eine Matrix-Vektor-Multiplikation, drei Vektor-Linearkombinationen und zwei Innenprodukte, außerdem Platz für drei n-Vektoren. Zusammenfassend werden die wichtigsten Eigenschaften im nächsten Satz aufgeführt. Zusätzlich enthält dieser eine Fehlerschranke für den praktisch interessantesten Fall $k < n$.

Satz 5.2.5 *Nach k Schritten mit $d^{(j)} \neq 0, j < k$, gelten für die im CG-Verfahren (5.2.20) berechneten Größen folgende Aussagen.*

a) Die Suchrichtungen $\{p^{(1)}, \dots, p^{(k)}\}$ sind A-orthogonal, die Defekte $\{d^{(0)}, \dots, d^{(k-1)}\}$ orthogonal und jeweils Basen des Krylovraums $\mathcal{K}_k$,

$$[p^{(1)}, \dots, p^{(k)}] = [d^{(0)}, \dots, d^{(k-1)}] = \mathcal{K}_k = [b, Ab, A^2 b, \dots, A^{k-1} b].$$

b) Die Näherung $x^{(k)}$ minimiert φ über $\mathcal{K}_k$, d.h.,

$$\varphi(x^{(k)}) = \min_{x \in \mathcal{K}_k} \varphi(x).$$

Daher ist $x^{(k)}$ exakt, wenn $z \in \mathcal{K}_k$, also spätestens für $k = n$.

c) In der A-Norm (5.2.1) gilt mit $q := (\hat{\kappa}(A) + 1)/(\hat{\kappa}(A) - 1)$ die Fehlerschranke

$$\begin{aligned} \|x^{(k)} - z\|_A &= \min\{\, \|g_k(A)(x^{(0)} - z)\|_A : \; g_k \in \Pi_k,\, g_k(0) = 1\} \\ &\leq \frac{1}{T_k(q)} \|x^{(0)} - z\|_A \leq 2 \left(\frac{\sqrt{\hat{\kappa}(A)} - 1}{\sqrt{\hat{\kappa}(A)} + 1} \right)^k \|x^{(0)} - z\|_A. \end{aligned} \tag{5.2.21}$$

Beweis Die Behauptungen a) und b) fassen die Sätze 5.2.3-5.2.4 zusammen.

c) Nach (5.2.3) ist die Minimumsuche bei $\varphi(x)$ äquivalent zur Minimierung von $\|x - z\|_A^2 = 2\varphi(x) + \|z\|_A^2$. Da die Elemente von $\mathcal{K}_k$ die Gestalt $\sum_{j=0}^{k-1} y_j A^j b = f_{k-1}(A) b,\; y_j \in \mathbf{R},\; f_{k-1} \in \Pi_{k-1}$, besitzen, folgt aus b)

$$\|x^{(k)} - z\|_A = \min_{f_{k-1} \in \Pi_{k-1}} \|f_{k-1}(A) b - z\|_A.$$

Mit $b = Az$ kann die letzte Differenz auch in der Form $f_{k-1}(A)Az - z = -(I - f_{k-1}(A)A)z = -g_k(A)z$ geschrieben werden mit $g_k \in \Pi_k,\; g(0) = 1$, da $g_k(t) = 1 - t f_{k-1}(t)$. Daher entspricht die letzte Charakterisierung der ersten Zeile in (5.2.21), da auch $x^{(0)} = 0$.

Da A symmetrisch positiv definit ist, läßt es sich zeigen, daß A eine ebensolche Quadratwurzel $A^{1/2}$ besitzt. Mit dieser ist $\|x\|_A^2 = x^\top A x = (A^{1/2} x)^\top A^{1/2} x = \|A^{1/2} x\|_2^2$. Damit kann die A-Norm bei (5.2.21) umgeformt werden,

$$\|g_k(A) z\|_A = \|A^{1/2} g_k(A) z\|_2 = \|g_k(A) A^{1/2} z\|_2 \leq \|g_k(A)\|_2 \|z\|_A = \varrho(g_k(A)) \|z\|_A.$$

Die letzte Gleichung gilt wegen der Symmetrie von A, vgl. (4.1.9). Die Fehlerschranken in (5.2.21) ergeben sich nun dadurch, daß für das zu minimierende Polynom g_k in

$$\|x^{(k)} - z\|_A \leq \min\{\max\{|g_k(\lambda)| : \lambda \in [\lambda_n, \lambda_1]\} : \; g_k \in \Pi_k, g_k(0) = 1\} \|x^{(0)} - z\|_A$$

analog zu Satz 5.1.2 das Tschebyscheff-Polynom zum optimalen Intervall $[\lambda_n, \lambda_1]$ eingesetzt wird. Die Schranken (5.2.21) folgen damit aus den entsprechenden Nachweisen von (5.1.13). ■

Die Fehlerschranke für das CG-Verfahren (in der A-Norm) stimmt mit derjenigen für die Tschebyscheff-Iteration überein, wenn dort ebenfalls das kleinstmögliche Einschließungsintervall für die Eigenwerte verwendet wird. Die Interpretation des CG-Verfahrens als eine Tschebyscheff-Iteration mit *eingebauter* Parameterbestimmung wird durch eine etwas andere Formulierung noch hervorgehoben, bei der die Suchrichtungen $p^{(j)}$ aus (5.2.20) eliminiert wurden.

CG-2-Version:

$$\boxed{\begin{array}{l}
x^{(-1)} := 0;\ x^{(0)} := 0;\ d^{(0)} := b;\ k := 0;\\
\text{solange } d^{(k)} \neq 0:\\
\quad k := k+1;\\
\quad \gamma_{k-1} := \|d^{(k-1)}\|_2^2 / \|d^{(k-1)}\|_A^2;\\
\quad \text{falls } k = 1 \text{ dann } \omega_1 := 1 \text{ sonst}\\
\qquad \beta_{k-1} := \|d^{(k-1)}\|_2^2 / \|d^{(k-2)}\|_2^2;\\
\qquad \omega_k := 1 \Big/ \Big(1 - \frac{\gamma_{k-1}}{\gamma_{k-2}} \beta_{k-1} / \omega_{k-1}\Big);\\
\quad x^{(k)} := \omega_k (x^{(k-1)} + \gamma_{k-1} d^{(k-1)}) + (1-\omega_k) x^{(k-2)};\\
\quad d^{(k)} := r - Ax^{(k)};
\end{array}} \tag{5.2.22}$$

Der eigentliche Iterationsschritt hat die gleiche Struktur wie bei der Tschebyscheff-Iteration (5.1.17), denn (5.2.22) besteht aus

$$\left\{ \begin{array}{lcll}
\tilde{x}^{(k)} & := & x^{(k-1)} + d^{(k-1)}/R_A[d^{(k-1)}], & \text{einfacher Gradientenschritt (5.2.5)}\\
x^{(k)} & := & \omega_k \tilde{x}^{(k)} + (1-\omega_k) x^{(k-2)}, & \text{+Extrapolationsschritt}
\end{array} \right.$$

Der wesentliche Unterschied zwischen beiden Verfahren ist also der, daß bei der Tschebyscheff-Iteration die Parameter aus explizit erforderlichen Eigenwert-Schranken berechnet werden müssen, während das CG-Verfahren alle Parameter automatisch bestimmt. Dabei minimieren die Parameter der Tschebyscheff-Iteration die Fehler*schranke* (5.1.9), während sich die CG-Koeffizienten an den *tatsächlichen* Fehler anpassen. Daher konvergiert das CG-Verfahren in der Praxis meist schneller als die Tschebyscheff-Iteration, die Fehlerschranke (5.2.21) ist ziemlich pessimistisch (vgl. Demo-Beispiel am Ende). Sie liefert aber den Ansatzpunkt für weitere Verbesserungen.

5.3 Vorkonditionierung

In der Numerik I wurde bei der Konstruktion von Iterationsverfahren der Begriff der regulären Zerlegung $A = M - N$ einer Matrix A eingeführt. Der reguläre *Hauptteil* M war dabei so zu wählen, daß mit dem Rest N die Iterationsmatrix $B = M^{-1}N$ kontraktiv war, $\varrho(M^{-1}N) < 1$. In der Sprechweise der letzten Abschnitte entspricht eine solche Zerlegung dem Übergang vom

Linearen Gleichungssystem $Ax = b$ mit Matrix A zu dem äquivalenten mit der Matrix $I - B = I - M^{-1}N = M^{-1}A$, also zu

$$(M^{-1}A)\,x = M^{-1}b. \tag{5.3.1}$$

Der alten Forderung an die Matrix M bei der Fixpunktiteration, daß der Konvergenzfaktor $\varrho(M^{-1}N)$ möglichst klein sein soll, entspricht bei den hier besprochenen Verfahren aufgrund der Konvergenzaussagen der Sätze 5.1.2, 5.1.3, 5.2.5 die Forderung, daß die Kondition $\hat{\kappa}(M^{-1}A)$ den Wert eins möglichst wenig überschreitet. Daher wird der Übergang zum System (5.3.1) auch als *Vorkonditionierung* bezeichnet.

Schwierigkeiten bei der Umsetzung dieses Ansatzes ergeben sich aus den für den Einsatz der Verfahren erforderlichen Voraussetzungen an $M^{-1}A$. Bei Richardson- und Tschebyscheff-Iteration wurden reelle Eigenwerte gefordert, das CG-Verfahren setzt eine symmetrisch, positiv-definite Matrix voraus. Diese Voraussetzungen können durch geeignete Umformulierungen erfüllt werden, wenn gilt

$$M \text{ in (5.3.1) ist symmetrisch, positiv definit.} \tag{5.3.2}$$

Dann besitzt M nämlich eine *Cholesky-Zerlegung* $M = LL^{\mathsf{T}}$ mit einer unteren Dreieckmatrix L, die positive Diagonalelemente besitzt (vgl. Numerik I, Satz 4.3.4). Durch Multiplikation von (5.3.1) mit L^{T} ergibt sich das neue Gleichungssystem $L^{-1}Ax = L^{-1}b \iff$

$$(L^{-1}AL^{-\mathsf{T}})(L^{\mathsf{T}}x) = L^{-1}b$$

für die Hilfsvariable $y = L^{\mathsf{T}}x$. Die Matrix dieses Systems, $L^{-1}AL^{-\mathsf{T}} = L^{\mathsf{T}}(M^{-1}A)L^{-\mathsf{T}}$, ist symmetrisch, positiv definit und ähnlich zu $M^{-1}A$. Daher sind unter der Voraussetzung (5.3.2) Richardson- und Tschebyscheff-Iteration bei (5.3.1) einsetzbar. Die *vorkonditionierte Tschebyscheff-Iteration* etwa lautet

$$\begin{array}{rcl} M(\tilde{y}^{(k+1)} - y^{(k)}) & := & \dfrac{2}{\alpha+\beta}(r - Ay^{(k)}) \\ y^{(k+1)} & := & \varrho_k \tilde{y}^{(k+1)} + (1-\varrho_k) y^{(k-1)} \end{array}. \tag{5.3.3}$$

Dabei muß M so beschaffen sein, daß die Lösung des Gleichungssystems für $\tilde{y}^{(k+1)} - y^{(k)}$ (nicht M^{-1}!) einfach berechnet werden kann. Die Koeffizienten sind jetzt aus Eigenwert-Schranken $[\alpha, \beta] \ni \lambda_j(M^{-1}A)\,\forall j$ zu bestimmen, die Konvergenz hängt von β/α ab.

Beim CG-Verfahren hat die Matrix des Gleichungssystems aber unbedingt symmetrisch und definit zu sein. Diese Eigenschaft besitzt auch $M^{-1}A$ unter der Voraussetzung (5.3.2), wenn das ℓ_2-Innenprodukt durch das Produkt $(\cdot,\cdot)_M$ ersetzt wird, denn

$$(x, M^{-1}Ay)_M = (y^{\mathsf{T}}AM^{-1})Mx = y^{\mathsf{T}}Ax = y^{\mathsf{T}}MM^{-1}Ax = (M^{-1}Ax, y)_M.$$

In der Fehlerabschätzung (5.2.21) kann dann auch statt $\hat{\kappa}(A)$ der Wert $\hat{\kappa}(M^{-1}A)$ eingesetzt werden.

Präkonditioniertes CG-Verfahren:

$$\boxed{\begin{array}{l}
x^{(0)} := 0;\ d^{(0)} := b;\ \text{löse } Mp^{(1)} := b;\ k := 0; \\
\text{solange } d^{(k)} \neq 0\text{:} \\
\quad \text{löse } Mc^{(k)} = d^{(k)}; \\
\quad k := k + 1; \\
\quad t_k := d^{(k-1)^{\mathsf{T}}} c^{(k-1)} / p^{(k)^{\mathsf{T}}} A p^{(k)}; \\
\quad x^{(k)} := x^{(k-1)} + t_k p^{(k)}; \\
\quad d^{(k)} := d^{(k-1)} - t_k A p^{(k)}; \\
\quad \beta_k := d^{(k)^{\mathsf{T}}} c^{(k)} / d^{(k-1)^{\mathsf{T}}} c^{(k-1)}; \\
\quad p^{(k+1)} := c^{(k)} + \beta_k p^{(k)};
\end{array}} \qquad (5.3.4)$$

Strategien zur Wahl von M:

• Diagonale von A, $M = D := \operatorname{diag}(a_{ii})$, analog zum Gesamtschrittverfahren (Numerik I). Dies ist nur sinnvoll, wenn die Diagonalelemente von A sehr unterschiedliche Größen besitzen.

• Die Wahl '$M = D + L$' aus der Zerlegung

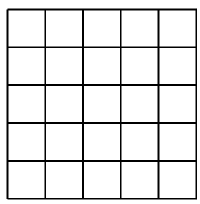
$$A = \left(\begin{array}{ccc} & & R \\ & D & \\ L & & \end{array}\right),$$

wie beim Einzelschrittverfahren scheidet aus, da $D + L$ nicht symmetrisch ist. Das Einzelschrittverfahren ergab sich bei schrittweiser Auflösung der i-ten Gleichung nach $x_i^{(k+1)}$, $i = 1, \ldots, n$. Wird daran ein umgekehrter Durchlauf $i = n, n-1, \ldots, 1$, angeschlossen, ergibt sich wegen $R = L^{\mathsf{T}}$ doch wieder eine symmetrische Iterationsmatrix. Gleiches gilt beim beschleunigten Einzelschrittverfahren, dem SOR-Verfahren, bei dem $D + L$ ersetzt wird durch $D + \omega L$ mit einem Parameter $0 < \omega < 2$. Ein Vorwärts- Rückwärts- Zyklus führt auf die symmetrische Matrix

$$M_\omega = (D + \omega L) D^{-1} (D + \omega L^{\mathsf{T}}). \qquad (5.3.5)$$

Dieses Verfahren heißt SSOR ('symmetric successive overrelaxation'). Da dessen Konvergenz auf einem anderen Prinzip beruht als Tschebyscheff- und CG-Iteration, kann es durch letztere weiter beschleunigt werden.

Beispiel 5.3.1 Zur Approximation einer Minimalfläche (Seifenhaut), die auf den Rändern eines Rechtecks gegebene Werte annimmt, kann diese Fläche durch ein elastisches Rechtecknetz angenähert werden. Die unbekannte Höhe am Netzknoten in Zeile i und Spalte j sei u_{ij}, dabei werden je $m + 2$ Zeilen und Spalten $i, j \in \{0, \ldots, m + 1\}$ verwendet. Im Knoten (i, j) erfüllt die Höhe bei der (approximierten) Minimalfläche die Bedingung



$$u_{ij} = \text{Mittelwert der Nachbarn.}$$

Insgesamt gilt also das Lineare Gleichungssystem

$$\begin{aligned} u_{ij} - \tfrac{1}{4}(u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1}) &= 0, & i,j = 1,\ldots,m, \\ u_{ij} &= \text{Randwert}, & i \text{ oder } j \in \{0, m+1\}, \end{aligned}$$

mit $n = m^2$ Unbekannten. Die Spektral-Kondition der zugehörigen Matrix ist $\hat{\kappa} \doteq 4m^2/\pi^2 \cong 0.4n$. In der folgenden Tabelle sind Kontraktionsfaktoren (bezogen auf einen Iterationsschritt) bei diesem Beispiel für die verschiedenen Verfahren aufgeführt. Dabei stehen in der ersten Zeile Approximationsformeln (für große m) der theoretisch hergeleiteten Schranken, in den folgenden Zeilen die bei zwei verschiedenen Problemgrößen praktisch beobachteten Faktoren (gemittelt über die Iterationsschritte). Die so berechnete Näherung u stimmt umso besser mit der tatsächlichen Minimalfläche überein, je größer m ist. Für genaue Approximationen sind daher große Gleichungssystem mit großer Kondition zu lösen. Auf diese Tatsache reagieren die verschiedenen Verfahren unterschiedlich empfindlich.

Verfahren:	Relaxat.	SSOR	Richards. 32-Zyklus	Tscheb.	CG-Verf.	SSOR+ Tscheb.
Theorie:	$1-\frac{\pi^2}{2m^2}$	$1-\frac{\pi}{m}$	$1-\frac{16\pi^2}{m^2}$	$1-\frac{\pi}{m}$	$1-\frac{\pi}{m}$	$1-2\sqrt{\frac{\pi}{m}}$
$n=m^2=400$:	0.9775	0.8845	0.9278	0.9277	0.7460	0.6501
$n=m^2=1600$:	0.9887	0.9288	0.9564	0.9559	0.9159	0.7293

Im Vergleich der drei diskutierten Verfahren, Richardson- und Tschebyscheff-Iteration sowie CG-Verfahren gibt es im betrachteten Fall der symmetrisch, definiten Matrizen eine klare Abstufung nach Effizienz und Flexibilität. Hier ist auf jeden Fall das CG-Verfahren der Tschebyscheff-Iteration, und diese der Richardson-Iteration vorzuziehen (außer bei Speicherplatzbeschränkungen). Allerdings sind die Verfahren unterschiedlich gut in allgemeineren Situationen einsetzbar, die effizienteren Verfahren sind die weniger flexiblen. Die Entwicklung eines verallgemeinerten CG-Verfahrens für nicht-definite oder unsymmetrische Probleme ist ein sehr aktuelles Forschungsgebiet, keiner der vielen Ansätze kommt aber bisher ohne gravierende Nachteile aus. Die Tschebyscheff-Iteration ist auf Fälle verallgemeinerbar, in denen das Spektrum in einer Halbebene der komplexen Zahlen liegt, die die Null nicht enthält. Nur für die Richardson-Iteration gibt es keine solchen prinzipiellen Einschränkungen, die Konstruktion guter Parameter kann im Einzelfall allerdings schwierig sein.

Gegenüberstellung

	Sym.Pos.Def.	einseit. Spektrum	allgemein
Richardson	+	+	+
Tschebysch.	++	++	—
CG	+++	—	—

Schwierigkeitsgrad $\longrightarrow$

$\downarrow$ Effizienz

5.4 Lanczos-Verfahren zur Eigenwert-Berechnung

Das Verfahren der Konjugierten Gradienten wurde im letzten Abschnitt als Minimierungsverfahren eingeführt. Diese Iteration erzeugt aber gleichzeitig auch ein *Modell* der Matrix A von niederer Dimension, das dazu geeignet ist, *Eigenwert-Approximationen* für A zu berechnen ohne diese Matrix explizit zu kennen. Zur Herleitung werden k Schritte der CG-Iteration (5.2.20) ohne die Näherung $x^{(j)}$ betrachtet,

$$\begin{aligned} d^{(0)} &= b, & p^{(1)} &= b, \\ d^{(j)} &= d^{(j-1)} - t_j A p^{(j)}, & p^{(j+1)} &= d^{(j)} + \beta_j p^{(j)}, \quad j = 1, 2, \dots, k. \end{aligned}$$

Darin können die Suchrichtungen $p^{(j)}$ eliminiert werden. Aus der ersten Gleichung folgt

$$Ap^{(j)} = \frac{1}{t_j}(d^{(j-1)} - d^{(j)}). \tag{5.4.1}$$

Wird diese Gleichung mit dem folgenden Index verwendet, kann dort $p^{(j+1)}$ mit der zweiten Rekursionsformel ersetzt werden,

$$\frac{1}{t_{j+1}}(d^{(j)} - d^{(j+1)}) = Ap^{(j+1)} = Ad^{(j)} + \beta_j Ap^{(j)} = Ad^{(j)} + \frac{\beta_j}{t_j}(d^{(j-1)} - d^{(j)}).$$

Für $j = 0$ ist $Ad^{(0)} = \frac{1}{t_1}(d^{(0)} - d^{(1)})$, da $p^{(1)} = d^{(0)}$. Setzt man $\beta_0 := 0$, gilt also

$$Ad^{(j)} = -\frac{\beta_j}{t_j}d^{(j-1)} + \Big(\frac{\beta_j}{t_j} + \frac{1}{t_{j+1}}\Big)d^{(j)} - \frac{1}{t_{j+1}}d^{(j+1)}, \quad j = 0, \dots, k-1. \tag{5.4.2}$$

Nach Satz 5.2.4 bilden die Defekte $\{d^{(j)}\}_{j=0}^{k}$ eine Orthogonalbasis des Krylovraums $\mathcal{K}_{k+1}$. Faßt man die normierten Defekte zusammen zu einer orthogonalen Matrix Q_k, $Q_k^{\mathsf{T}} Q_k = I_k$, d.h.,

$$q^{(j)} := \frac{d^{(j-1)}}{\|d^{(j-1)}\|_2}, \ j = 1, \dots, k, \quad Q_k := (q^{(1)}, \dots, q^{(k)}) \in \mathbf{R}^{n \times k},$$

so kann die Beziehung (5.4.2) in Matrixschreibweise durch

$$AQ_k = Q_k M_k + \gamma_k q^{(k+1)} e^{(k)^{\mathsf{T}}} \tag{5.4.3}$$

formuliert werden. Die Matrix M_k ist einfach aufgebaut. Die Spalte $j > 0$ dieser Identität zeigt wegen (5.4.2), daß der Vektor $Aq^{(j)}$ eine Linearkombination von $q^{(j-1)}, q^{(j)}, q^{(j+1)}$ ist und M_k somit eine *Tridiagonalmatrix*. Die Symmetrie von A überträgt sich wegen

$$\underbrace{Q_k^{\mathsf{T}} A Q_k^{\mathsf{T}}}_{\text{sym.}} = \underbrace{Q_k^{\mathsf{T}} Q_k}_{=I_k} M_k + \gamma_k \underbrace{Q^{\mathsf{T}} q^{(k+1)}}_{=0} e^{(k)^{\mathsf{T}}} = M_k,$$

auf M_k. Daher gilt (5.4.3) mit

$$M_k = \begin{pmatrix} \alpha_1 & \gamma_1 & & \\ \gamma_1 & \alpha_2 & \gamma_2 & \\ & \ddots & \ddots & \ddots \\ & & \gamma_{k-1} & \alpha_k \end{pmatrix} \in \mathbf{R}^{k \times k}. \tag{5.4.4}$$

Diese Beziehung (5.4.3) kann wegen der Orthonormalität der $q^{(j)}$ auch direkt zur Berechnung von M_k und Q_k dienen. Dazu wird die j-te Spalte von (5.4.3) betrachtet und nach $q^{(j+1)}$ aufgelöst:

$$\gamma_j q^{(j+1)} = (A - \alpha_j I)q^{(j)} - \gamma_{j-1} q^{(j-1)}.$$

Für die Koeffizienten dieser Rekursion ergeben sich die Werte

$$\begin{aligned} \alpha_j &= {q^{(j)}}^\top A q^{(j)}, \quad \text{da } 0 = \gamma_j {q^{(j)}}^\top q^{(j+1)} = {q^{(j)}}^\top A q^{(j)} - \alpha_j, \\ \gamma_j &= \|(A - \alpha_j I)q^{(j)} - \gamma_{j-1} q^{(j-1)}\|_2. \end{aligned}$$

Daher liefert das folgende Verfahren (bis auf Vorzeichen) die Größen aus der Identität (5.4.3), (5.4.4). Der Startvektor ist nun beliebig, $d^{(0)} \neq 0$.

Lanczos-Verfahren:

$$\boxed{\begin{array}{l} \gamma_0 := \|d^{(0)}\|_2;\ q^{(0)} := 0;\ j := 0; \\ \text{solange } \gamma_j \neq 0: \\ \quad q^{(j+1)} := d^{(j)}/\gamma_j;\ j := j+1;\ \alpha_j := {q^{(j)}}^\top A q^{(j)}; \\ \quad d^{(j)} = (A - \alpha_j I)q^{(j)} - \gamma_{j-1} q^{(j-1)}; \\ \quad \gamma_j := \|d^{(j)}\|_2; \end{array}} \tag{5.4.5}$$

Falls dieses Verfahren erst mit $\gamma_n = 0$ abbricht, liefert es (bei exakter Rechnung) eine *Tridiagonalisierung* $M_n = Q_n^\top A Q_n$ von A nach einem zu Satz 4.2.2 unterschiedlichen Prinzip. Da die Vektoren $\{q^{(1)}, \ldots, q^{k)}\}$ aber eine Basis des Krylovraums $\mathcal{K}_k$ bilden, was an (5.4.5) direkt ablesbar ist, sind schon für $k \ll n$ die Eigenwerte der kleineren Matrix M_k gute Approximationen der *extremalen* Eigenwerte von A.

Der Grund dafür ist das Courantsche Maximum-Minimum-Prinzip, Satz 4.6.1. Zur Diskussion seien $\lambda_1 \geq \ldots \geq \lambda_n$ die Eigenwerte von A und

$$\lambda_1^{(k)} \geq \ldots \geq \lambda_k^{(k)} \quad \text{die Eigenwerte von } M_k \text{ aus (5.4.4),(5.4.5).}$$

Letztere können mit den Verfahren aus §4.4 effizient berechnet werden. Wegen $M_k = Q_k^\top A Q_k$, $Q_k \mathbf{R}^k = \mathcal{K}_k$, gilt z.B. für den maximalen Eigenwert

$$\lambda_1^{(k)} = \max_{y \in \mathbf{R}^k} \frac{y^\top M_k y}{y^\top y} = \max_{y \in \mathbf{R}^k} \frac{y^\top Q_k^\top A Q_k y}{y^\top Q_k^\top Q_k y} = \max_{x \in \mathcal{K}_k} \frac{x^\top A x}{x^\top x} \leq \max_{x \in \mathbf{R}^n} \frac{x^\top A x}{x^\top x} = \lambda_1. \tag{5.4.6}$$

Ähnlich wie beim CG-Verfahren minimiert $\lambda_k^{(k)}$ bzw. maximiert $\lambda_1^{(k)}$ den Rayleighquotienten $R_A[x]$ auf dem Krylov-Unterraum. Eine Fehlerschranke für diese beiden Eigenwerte läßt sich daher analog zu Satz 5.2.5 herleiten.

Satz 5.4.1 *Die symmetrische Matrix A besitze die Eigenwerte $\lambda_1 \geq \ldots \geq \lambda_n$ und die orthonormalen Eigenvektoren $x^{(1)}, \ldots, x^{(n)}$. Nach k Schritten des Lanczos-Verfahrens (5.4.5) gilt für* *den größten Eigenwert* $\lambda_1^{(k)}$ *der Matrix* M_k *aus (5.4.4) die Fehlerschranke*

$$\lambda_1 \geq \lambda_1^{(k)} \geq \lambda_1 - (\lambda_1 - \lambda_n)\frac{\tan^2\varphi}{T_{k-1}^2(1+2\varrho_1)}, \tag{5.4.7}$$

wobei $\cos\varphi_1 = |{q^{(1)}}^{\mathsf{T}} x^{(1)}|$, $\varrho_1 = (\lambda_1 - \lambda_2)/(\lambda_2 - \lambda_n)$ *und* T_{k-1} *das Tschebyscheff-Polynom (5.1.11) vom Grad* $k-1$ *ist.*

Beweis Jedes Element des Krylovraums $x \in \mathcal{K}_k$ hat die Gestalt $x = p(A)d^{(0)} = \sum_{j=1}^n p(\lambda_j)\xi_j x^{(j)}$, $p \in \Pi_{k-1}$, wobei $d^{(0)} = \sum_{j=1}^n \xi_j x^{(j)}$. Nach (5.4.6) ist

$$\lambda_1^{(k)} = \max_{p\in\Pi_{k-1}} \frac{{d^{(0)}}^{\mathsf{T}} p(A)Ap(A)d^{(0)}}{{d^{(0)}}^{\mathsf{T}} p(A)^2 d^{(0)}} = \max_{p\in\Pi_{k-1}} \frac{\sum_{j=1}^n \xi_j^2 p(\lambda_j)^2\lambda_j}{\sum_{j=1}^n \xi_j^2 p(\lambda_j)^2}.$$

Mit beliebigen Polynomen $p \in \Pi_{k-1}$ gilt also

$$\frac{{d^{(0)}}^{\mathsf{T}} p(A)Ap(A)d^{(0)}}{{d^{(0)}}^{\mathsf{T}} p(A)^2 d^{(0)}} = \lambda_1 - \frac{\sum_{j=2}^n \xi_j^2 p(\lambda_j)^2(\lambda_1-\lambda_j)}{\sum_{j=1}^n \xi_j^2 p(\lambda_j)^2} \geq \lambda_1 - (\lambda_1-\lambda_n)\frac{\sum_{j=2}^n \xi_j^2 p(\lambda_j)^2}{\xi_1^2 p(\lambda_1)^2}. \tag{5.4.8}$$

Das Verhältnis $|p(\lambda_j)/p(\lambda_1)|$, $j > 1$, läßt sich dadurch auf einen minimalen Wert bringen, daß man p als Tschebyscheff-Polynom zum Intervall $[\lambda_n, \lambda_2]$ wählt. Nach Satz 5.1.2 gilt mit

$$p(\lambda) = T_{k-1}\Big(\frac{\lambda_2+\lambda_n-2\lambda}{\lambda_2-\lambda_n}\Big) \;\Rightarrow\; |p(\lambda_j)| \leq 1,\; j = 2,\ldots,n.$$

Da $|p(\lambda_1)| = |T_{k-1}(-1-2\varrho_1)| = |T_{k-1}(1+2\varrho_1)|$ ist, folgt damit $|p(\lambda_j)/p(\lambda_1)| \leq 1/|T_{k-1}(1+2\varrho_1)|$, $j > 1$. Außerdem ist

$$\frac{\sum_{j=2}^n \xi_j^2}{\xi_1^2} = \frac{\|d^{(0)}\|_2^2}{({x^{(1)}}^{\mathsf{T}} d^{(0)})^2} - 1 = \frac{1}{\cos^2\varphi_1} - 1 = \tan^2\varphi_1.$$

Mit diesen Schranken folgt aus (5.4.8) die Behauptung. ■

Für $k \ll n$ fällt der Aufwand zur Berechnung des größten Eigenwerts von M_k gegenüber (5.4.5) nicht ins Gewicht, der Aufwand für k Schritte ist daher ungefähr so groß wie bei der Vektor-Iteration (4.4.3). Bis auf Konstanten ist der Fehler bei definiten Matrizen ($\lambda_n \geq 0 \Rightarrow 1+2\varrho_1 \geq 2\lambda_1/\lambda_2 - 1$) nach k Matrix-Vektor-Multiplikationen jeweils

$$\begin{array}{ll} \Big(\frac{\lambda_2}{\lambda_1}\Big)^{2k} & \text{bei sym. Vektoriteration,} \\ \frac{1}{T_k(2\lambda_1/\lambda_2-1)^2} \leq 4\Big(\frac{\lambda_2}{\lambda_1}\Big)^{2k}\Big(1+\sqrt{1-\lambda_2/\lambda_1}\Big)^{-4k} & \text{bei Lanczos.} \end{array} \tag{5.4.9}$$

Zum Vergleich der Schranken aus (5.4.9) seien Zahlenwerte für $k = 20$ und verschiedene Verhältnisse λ_1/λ_2 angegeben:

$\lambda_1/\lambda_2 =$	1.5	1.1	1.01
Vekt-It.	$9_{10}-8$	$2_{10}-2$	0.7
Lanczos	$5_{10}-23$	$6_{10}-11$	$1_{10}-3$

Das Lanczos-Verfahren benutzt i.w. die gleiche Information wie die Vektoriteration, nämlich die Krylov-Vektoren $\{A^j q^{(1)}\}$, leitet daraus aber wesentlich bessere Eigenwert-Schätzungen ab. Beim praktischen Einsatz ist das Lanczos-Verfahren aber recht anfällig gegen Rundungsfehler.

6 Fortsetzungsverfahren für nichtlineare Probleme

6.1 Die numerische Verfolgung von Lösungskurven

Das Standardverfahren zur Lösung des nichtlinearen Gleichungssystems

$$F(x) = 0, \qquad F: \mathbf{R}^n \mapsto \mathbf{R}^n, \tag{6.1.1}$$

ist das *Newton-Verfahren* (vgl. Numerik I, §5.2): $x^{(0)} \in \mathbf{R}^n$,

$$F'(x^{(k)})\Big(x^{(k+1)} - x^{(k)}\Big) = -t_k F(x^{(k)}),\ k = 0, 1, \ldots, \tag{6.1.2}$$

mit Parametern $t_k > 0$. Beim klassischen Newtonverfahren mit $t_k \equiv 1$ kann die Konvergenz nur in einer, evtl. sehr kleinen, Umgebung der Lösung z von (6.1.1) garantiert werden. Eine Verbesserung war die Einführung des Parameters t_k, der mit Hilfe einer *Liniensuche* so bestimmt wird, daß der Funktionswert in $x^{(k+1)}$ kleiner wird,

$$\varphi(x^{(k+1)}) < \varphi(x^{(k)}) \ \text{ mit } \ \varphi(x) := \|F(x)\|_2^2.$$

Das Prinzip entspricht dem des einfachen Gradientenverfahrens (5.2.5). Aber auch bei diesem Verfahren ist die Konvergenz gegen z nicht sicher, da die Folge $\{x^{(k)}\}$ aus (5.2.2) gegen ein lokales Minimum $\bar{x}$ mit $\varphi(\bar{x}) > 0$ konvergieren kann.

Ein alternativer Zugang nutzt die lokal garantierte Konvergenz des Newtonverfahrens aus. Man kann nämlich erwarten, daß die Lösung eines leicht *deformierten* Problems (6.1.1) eine gute Startnäherung für die Iteration (6.1.2) liefert. Konkret wird ein zusätzlicher Parameter λ eingeführt, etwa $\lambda \in [0, 1]$, und die *Problemschar*

$$0 = G(x, \lambda) := \lambda F(x) + (1 - \lambda) F_0(x),\ \lambda \in [0, 1], \tag{6.1.3}$$

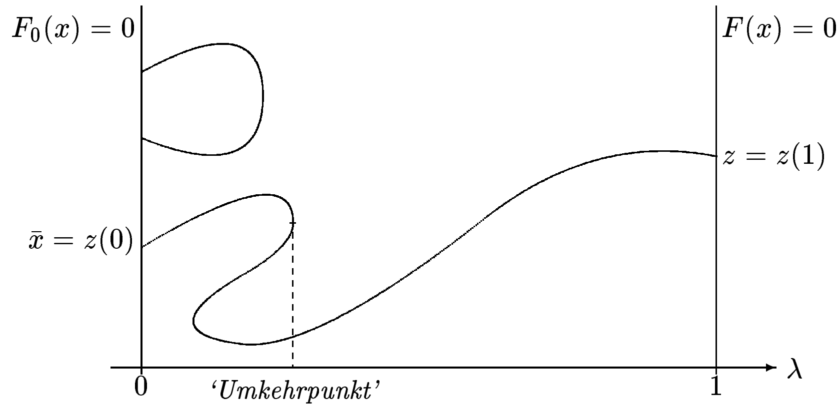
betrachtet ('Einbettung'). Dabei ist F_0 eine Abbildung, für die eine Nullstelle $\bar{x}$, $F_0(\bar{x}) = 0$, bekannt ist, z.B.,

$$\begin{aligned} F_0(x) &:= x &&\Rightarrow \quad \bar{x} = 0, \\ F_0(x) &:= F(x) - F(\tilde{x}) &&\Rightarrow \quad \bar{x} = \tilde{x}. \end{aligned}$$

Unter geeigneten Voraussetzungen hängen die Lösungen von (6.1.3) stetig vom Parameter λ ab, sie bilden eine *Lösungskurve*. Daher kann man versuchen, dieser Kurve zu folgen,

vom bekannten Startpunkt $\quad z(0) = \bar{x} \quad$ mit $\quad G(x, 0) = F_0(x) = 0,$
zum gesuchten Endpunkt $\quad z(1) = z \quad$ mit $\quad G(x, 1) = F(x) = 0.$

Im folgenden Bild sind mögliche Situationen skizziert, insbesondere muß $z(\lambda)$ keine (eindeutige) Funktion von λ sein.



Parameterabhängige Gleichungssysteme $G(x, \lambda) = 0$ müssen in der Praxis oft auch unabhängig von der Motivation (6.1.3) behandelt werden, etwa als Abhängigkeitsanalysen über Material- oder Modellparameter. Daher wird nun das abstrakte Problem

$$G(x, \lambda) = 0, \quad G: \ D \times [0,1] \mapsto \mathbf{R}^n, \ D \subset \mathbf{R}^n, \tag{6.1.4}$$

betrachtet. Der Satz über implizite Funktionen liefert hier ein einfaches Kriterium für die Existenz einer glatten und bezüglich λ eindeutigen Lösungskurve.

Satz 6.1.1 *D sei offen und $G \in C^1(D \times [0,1])$. Die Ableitung G_x sei überall invertierbar und es gelte*

$$\|G_x(x, \lambda)^{-1}\| \le \alpha, \ \|G_\lambda(x, \lambda)\| \le \beta_1 \ \forall (x, \lambda) \in D \times [0,1]. \tag{6.1.5}$$

Es sei $z_0 \in D$ eine Lösung von $G(x, 0) = 0$ so, daß

$$\{x \in \mathbf{R}^n : \ \|x - z_0\| \le \alpha\beta_1\} \subseteq D.$$

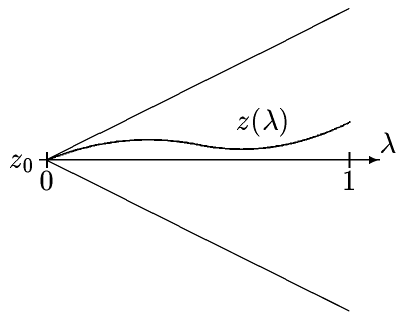
Dann gibt es eine in z_0 beginnende, stetig differenzierbare Lösungsfunktion $z(\lambda)$ mit $G(z(\lambda), \lambda) \equiv 0$, $z(0) = z_0$, für $\lambda \in [0,1]$. Ihre Ableitung z' erfüllt die Beziehung

$$z'(\lambda) = \frac{dz}{d\lambda}(\lambda) = -G_x(z(\lambda), \lambda)^{-1} G_\lambda(z(\lambda), \lambda). \tag{6.1.6}$$

Beweis Aus dem Satz über implizite Funktionen folgt die lokale Existenz und die Gültigkeit von (6.1.6), da

$$G(z(\lambda), \lambda) \equiv 0 \ \Rightarrow \ G_x z' + G_\lambda \equiv 0.$$

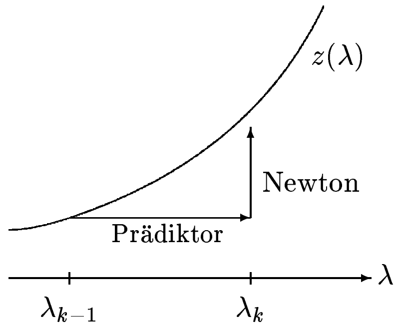
Die daraus abgeleitete Gleichung (6.1.6) ist eine Differentialgleichung $z' = f(z, \lambda)$ mit stetiger und beschränkter rechter Seite f, $\|f\| \le \alpha\beta_1$ in $D \times [0,1]$. Nach dem Existenssatz von Peano existiert daher eine Lösung $z(\lambda)$, die in dem Kegel $\|z(\lambda) - z(0)\| \le \alpha\beta_1\lambda$ verläuft. Wenn dieser ganz in $D \times [0,1]$ liegt, erreicht die Lösung $z(\lambda)$ den rechten Rand, $\lambda = 1$. ∎



Unter der Voraussetzung (6.1.5) können insbesondere keine *Umkehrpunkte* wie im obigen Bild auftreten, da in diesen die Ableitung G_x singulär wird. Zunächst wird die Verfolgung von Kurven ohne Umkehrpunkte behandelt. Tatsächlich berechnet werden dabei allerdings nur einzelne Punkte der Lösungskurve in Parameterwerten $0 = \lambda_0, \lambda_1, \ldots$. Mögliche

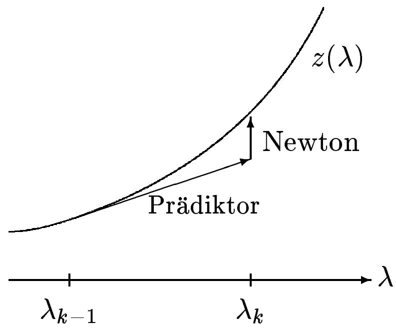
Strategien

- *Klassische Fortsetzung*:

  Der Wert $z(\lambda_{k-1})$ wird als Startwert der Newton-Iteration zur Lösung von $G(x, \lambda_k) = 0$ im Nachbarpunkt $\lambda_k > \lambda_{k-1}$ verwendet. Wenn z' groß werden kann, sind hierbei sehr kleine Schritte $h_k = \lambda_k - \lambda_{k-1}$ notwendig.

- *Tangentiale Fortsetzung*:

  Wegen (6.1.6) ist in λ_{k-1} mit $z(\lambda_{k-1})$ auch die Ableitung $z'(\lambda_{k-1})$ bekannt. Eine Startnäherung $x^{(0)}(\lambda_k)$ für $z(\lambda_k)$ im Nachbarpunkt $\lambda_k = \lambda_{k-1} + h_k$ kann daher mit der Tangente in λ_{k-1} konstruiert werden. Dies ergibt ein zweistufiges Verfahren

$$\begin{array}{ll} \text{Prädiktor-} \\ \text{Schritt:} \end{array} \left\{ \begin{array}{l} x^{(0)}(\lambda_k) := z(\lambda_{k-1}) + h_k z'(\lambda_{k-1}), \quad \text{mit} \\ G_x\Big(z(\lambda_{k-1}), \lambda_{k-1}\Big) z'(\lambda_{k-1}) = -G_\lambda\Big(z(\lambda_{k-1}), \lambda_{k-1}\Big) \end{array} \right. \tag{6.1.7}$$

$$\begin{array}{ll} \text{Korrektor-} \\ \text{Schritt:} \end{array} \left\{ \begin{array}{l} G_x\Big(x^{(j)}(\lambda_k), \lambda_k\Big)\Big(x^{(j+1)}(\lambda_k) - x^{(j)}(\lambda_k)\Big) = -G\Big(x^{(j)}(\lambda_k), \lambda_k\Big), \\ j = 0, 1, \ldots\ . \end{array} \right. \tag{6.1.8}$$

Im folgenden wird nur die zweite, bessere Strategie untersucht. Auch bei dieser ist grundsätzlich die Schrittweite h_k so klein zu wählen, daß die Newton-Iteration (6.1.8) mit dem Startwert $x^{(0)}(\lambda_k)$ konvergiert. Da man andererseits nur selten am genauen Verlauf der Kurve $z(\lambda)$ interessiert ist, sind möglichst große Schrittweiten anzustreben. Die größte Schrittweite, für die noch Konvergenz garantiert werden kann, ist im folgenden Satz charakterisiert.

Satz 6.1.2 *Es sei D offen, $G \in C^2(D \times [0,1])$ und*

$$\|G_x(x,\lambda)^{-1}\| \le \alpha, \ \|G_\lambda\| \le \beta_1, \ \max\{\|G_{xx}\|, \|G_{x\lambda}\|, \|G_{\lambda\lambda}\|\} \le \beta_2 \tag{6.1.9}$$

$\forall (x,\lambda) \in D \times [0,1]$. *Damit sei*

$$\bar{h} := \frac{1}{\alpha(1 + \alpha\beta_1)\beta_2}. \tag{6.1.10}$$

Die Lösung $z(\lambda_{k-1})$, $\lambda_{k-1} \in [0,1)$, von (6.1.4) sei einschließlich der Kugel

$$\{x \in \mathbf{R}^n : \ \|x - z(\lambda_{k-1})\| \le \alpha\beta_1\bar{h}\}$$

in D enthalten. Dann konvergiert das Prädiktor-Korrektor-Verfahren (6.1.7), (6.1.8) gegen $z(\lambda_k)$ wenn $\lambda_k \le 1$ und

$$h_k = \lambda_k - \lambda_{k-1} \le \bar{h}.$$

Beweis Nach Satz 5.2.2 aus der Numerik I konvergiert die Newton-Iteration (6.1.8), wenn für den Startwert $x^{(0)}$ gilt

$$\|x^{(0)}(\lambda_k) - z(\lambda_k)\| < \frac{2}{3\alpha\beta_2}. \tag{6.1.11}$$

Um diese Aussage hier anzuwenden, ist die Abweichung von Kurve und Tangente abzuschätzen, der Satz von Taylor zeigt dazu

$$\|z(\lambda_{k-1}) + h_k z'(\lambda_{k-1}) - z(\lambda_k)\| \le \frac{1}{2} h_k^2 \max_{s\in[0,h]} \|z''(\lambda_{k-1} + s)\|.$$

Daher wird zunächst eine Schranke für z'' in D benötigt. Nach Satz 6.1.1 ist z differenzierbar und $\|z'\| \le \alpha\beta_1$, daher liegen $x^{(0)}(\lambda_k)$ und $z(\lambda_k)$ in D. Wegen $G \in C^2$ ist die Ableitung z'' stetig und ergibt sich aus der Kettenregel

$$G_x z' + G_\lambda \equiv 0 \;\Rightarrow\; G_x z'' + G_{xx} z' z' + 2G_{x\lambda} z' + G_{\lambda\lambda} \equiv 0,$$

durch Multiplikation mit G_x^{-1}. Daher ist

$$\|z''\| \le \alpha\beta_2 (1 + \|z'\|)^2 \le \alpha\beta_2 (1 + \alpha\beta_1)^2.$$

Somit läßt sich die Konvergenzbedingung (6.1.11) mit der Taylor-Schranke durch die Forderung

$$\|x^{(0)}(\lambda_k) - z(\lambda_k)\| \le \frac{1}{2} h_k^2 \alpha\beta_2 (1 + \alpha\beta_1)^2 \overset{!}{<} \frac{2}{3\alpha\beta_2}$$

erzwingen. Letztere ist aber unter der Voraussetzung $h_k \le \bar{h}$ erfüllt. ■

Dieser Satz garantiert, daß man mit Schrittweiten $h_k \le \bar{h}$ die Kurve $z(\lambda)$ tatsächlich durch das Intervall $[0, 1]$ verfolgen kann, also mit ca. $1/\bar{h}$ Schritten. Die Aussage des Satzes hat aber kaum eine praktische Bedeutung, da die Schranken (6.1.9) selten bekannt sind. Andererseits ist es aber auch aus Effizienzgründen nicht sinnvoll, mit maximalen Schrittweiten $h_k \cong \bar{h}$ vorzugehen. Dies liegt an der quadratischen Konvergenz des Newtonverfahrens, da der Startfehler einer guten Näherung wesentlich stärker reduziert wird als der einer schlechten. Für die Fehler $e_j := \|x^{(j)}(\lambda_k) - z(\lambda_k)\|$ lautet die Konvergenzaussage von Satz I.5.2.2

$$e_j \le \frac{3}{2}\alpha\beta_2\, e_{j-1}^2. \tag{6.1.12}$$

Für $h \cong \bar{h}$ wird der Fehler am Anfang nur langsam kleiner, da $e_1 \le (\frac{3}{2}\alpha\beta_2 e_0)e_0 \cong e_0$ nach Konstruktion im Beweis des letzten Satzes. Um wieder nahe genug an die Lösungskurve zu kommen, werden daher viele Newton-Schritte benötigt. Da der Prädiktor-Schritt (6.1.7) den gleichen Aufwand wie eine Newton-Korrektor (6.1.8) erfordert, ist es günstiger, und sicherer, mehrere kleine λ-Schritte mit wenig Korrekturen pro Schritt zu machen. Dazu muß eine gute *Kontraktion* der Newton-Iteration gesichert werden, etwa $\frac{3}{2}\alpha\beta_2 e_0 \overset{!}{\le} \frac{1}{10} \;\Rightarrow\; e_1 \le$

$\frac{1}{10}e_0$, $e_2 \leq \frac{1}{1000}e_0, \ldots$. Die unbekannte tatsächliche Kontraktion e_1/e_0 kann dabei aus den Newton-Korrekturen $\|x^{(j+1)} - x^{(j)}\|$ geschätzt werden. Wegen der quadratischen Konvergenz ist $\|x^{(j+1)} - x^{(j)}\| = \|(x^{(j)} - z) - (x^{(j+1)} - z)\| = e_j + \mathcal{O}(e_j^2)$. Mit $q \ll 1$ kann daher aus

$$\frac{e_1}{e_0} \cong \frac{\|x^{(2)} - x^{(1)}\|}{\|x^{(1)} - x^{(0)}\|} \overset{!}{\leq} q \quad (q = \frac{1}{10} \cdots \frac{1}{2}) \tag{6.1.13}$$

auf eine akzeptable Konvergenz der Newton-Korrektur (6.1.8) geschlossen werden.

In der Praxis läßt sich diese Forderung mit Hilfe einer *Schrittweitensteuerung* realisieren. Dazu werden zunächst mit einer geschätzten Schrittweite h_k (z.B. der aus dem letzten λ-Schritt) der Prädiktor- (6.1.7) und zwei Korrektorschritte (6.1.8) durchgeführt. Entspricht dabei die Kontraktion nicht der Bedingung (6.1.13), wird dieser Versuch verworfen und die Schrittweite verkleinert, etwa $h_k \to h_k/2$ (z.B., 'in Kurven bremsen!'). Umgekehrt kann bei einer zu guten Kontraktion $\|x^{(2)} - x^{(1)}\|/\|x^{(1)} - x^{(0)}\| < \frac{1}{4}q$ die Schrittweite für den nächsten Schritt vergrößert werden, $h_{k+1} := 2h_k$ ('beschleunigen auf gerader Strecke'). Der folgende Algorithmus skizziert dieses Vorgehen.

Fortsetzungsverfahren mit Schrittweitensteuerung

(6.1.14)

```
λ := 0; h := h_Start; z := z_0,
löse G_x(z_0,0)t = −G_λ(z_0,0);                {Tangente}
wiederhole:
  y := z + ht; j := 0;                          {Prädiktor}
  wiederhole:                                   {Newton-Iteration}
    löse G_x(y,λ+h)d = −G(y,λ+h);
    y := y + d; j := j + 1; c_j := ||d||;
  bis (c_j ≤ ε) oder (j ≥ j_max);               {z.B. j_max = 2}
  falls (c_j > ε) oder (c_2/c_1 > q) dann       {Schritt verwerfen!}
    h := h/2; falls h < h_min dann Abbruch;
  sonst                                         {Schritt akzeptieren!}
    λ := λ + h; z := y;
    löse G_x(z,λ)t = −G_λ(z,λ);                 {neue Tangente}
    falls c_2/c_1 < q/4 dann h := 2h;
    falls λ + h > 1 dann h := 1 − λ;
bis λ ≥ 1;
```

Im Prädiktor-Schritt kann übrigens die QR-Zerlegung von G_x aus dem letzten Newton-Schritt verwendet werden.

Zu einem Abbruch des Verfahrens kommt es in der Regel nur dann, wenn die Voraussetzungen von Satz 6.1.1 verletzt sind, bei stetig differenzierbarem G also i.w., wenn G_x singulär wird. Ein einfaches Beispiel dazu ist die Kreisgleichung

$$0 = G(x, \lambda) := x^2 + \lambda^2 - 1$$

im $\mathbf{R}^1$. Eine Parameterdarstellung der Lösung durch $(1,0)$ nach λ, wie $z(\lambda) = \sqrt{1-\lambda^2}$, stößt im Punkt $(x,\lambda) = (0,1)$ auf Schwierigkeiten, da z' dort nicht beschränkt ist. Parametrisiert man den Kreis aber nach der *Bogenlänge* s, z.B.,

$$(z(s), \lambda(s)) = (\cos s, \sin s),$$

hat der *Umkehrpunkt* $(0,1)$ keinerlei besondere Bedeutung mehr.

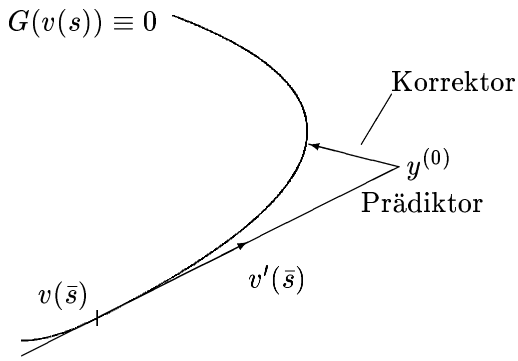
Beim allgemeinen Problem kann analog die Hervorhebung des $n+1$-ten Parameters λ fallengelassen werden dadurch, daß alle Komponenten zu einem $n+1$-Vektor y zusammengefaßt werden, $y_i = x_i$ $(i \le n)$, $y_{n+1} := \lambda$. Das Problem lautet dann

$$G(y) = 0, \quad G : D \subseteq \mathbf{R}^{n+1} \mapsto \mathbf{R}^n. \tag{6.1.15}$$

Wenn G in einer Umgebung eines Punktes $\bar{v}$ mit $G(\bar{v}) = 0$ stetig differenzierbar ist und $G'(\bar{v}) = \frac{\partial G}{\partial y}$ den *Maximalrang* n besitzt, garantiert der Satz über implizite Funktionen die lokale Existenz einer Lösungskurve $v(s)$, $s \in \mathbf{R}$. Diese ist sogar stetig differenzierbar und es gilt

$$G(v(s)) \equiv 0 \quad \Rightarrow \quad G'(v(s))v'(s) \equiv 0. \tag{6.1.16}$$

Die Ableitung und Tangentenrichtung $v' \in \mathbf{R}^{n+1}$ ist somit ein Element des eindimensionalen Nullraums von $G' : \mathbf{R}^{n+1} \mapsto \mathbf{R}^n$. Wenn insbesondere als Kurvenparameter die Bogenlänge s gewählt wird mit $\|\partial v/\partial s\|_2 \equiv 1$, ist v' durch (6.1.16) bis auf das Vorzeichen (d.h. die Kurvenrichtung) eindeutig bestimmt. In Verallgemeinerung des Verfolgungsverfahrens (6.1.7), (6.1.8) kann also mit Hilfe von (6.1.16) ein *Prädiktor*-Schritt in Richtung der Kurventangente durchgeführt werden.



In Verallgemeinerung von (6.1.8) sollte der *Korrektor* idealerweise zum nächstgelegenen Punkt, $v(\hat{s})$, auf der Kurve führen. Die Korrektur $d^{(0)} = y^{(1)} - y^{(0)}$ des durch den Prädiktor berechneten Startvektors $y^{(0)}$ sollte daher senkrecht zu $v'(\hat{s})$ sein, d.h. $d^{(0)} \in N\Big(G'(v(\hat{s}))\Big)^{\perp}$. Die Gleichung für die Newtonkorrektur, $G'(y^{(0)})d^{(0)} = -G(y^{(0)})$ ist ein *unterbestimmtes* Gleichungssystem. Ändert man jedoch die erwähnte Orthogonalitätsbedingung etwas ab, so wird durch

$$G'(y^{(0)})d^{(0)} = -G(y^{(0)}), \quad d^{(0)} \in N\Big(G'(y^{(0)})\Big)^{\perp}$$

die eindeutige *Kleinste-Quadrate-Lösung* (vgl. Def. 2.1.1) definiert,

$$d^{(0)} = -G'(y^{(0)})^{+} G(y^{(0)}),$$

$(G')^+$ ist dabei die *verallgemeinerte Inverse*. So ergibt sich das

Prädiktor-Korrektor-Verfahren für (6.1.15):

$$\begin{array}{ll}\text{Prädiktor-Schritt:} & \left\{\begin{array}{l} s_k := s_{k-1} + h_k, \\ y^{(0)}(s_k) := v(s_{k-1}) + h_k t_{k-1}, \quad \text{mit} \\ G'\Big(v(s_{k-1})\Big) t_{k-1} = 0, \ \|t_{k-1}\|_2 = 1, \end{array}\right. \end{array} \tag{6.1.17}$$

$$\begin{array}{ll}\text{Korrektor-Schritt:} & \left\{\begin{array}{l} y^{(j+1)}(s_k) = y^{(j)}(s_k) - G'\Big(y^{(j)}(s_k)\Big)^+ G(y^{(j)}(s_k)), \\ j = 0, 1, \ldots \ . \end{array}\right. \end{array} \tag{6.1.18}$$

Die Richtung der Tangente t_k in (6.1.17) wird am einfachsten durch $t_{k-1}^\top t_{k-2} > 0$ festgelegt. Zur Berechnung der Lösungen in (6.1.17) und (6.1.18) dient die *QR-Zerlegung* (vgl. Satz 2.3.3) der Matrix

$$(G')^\top = QR = Q\begin{pmatrix} R_0 \\ 0 \end{pmatrix}, \quad Q \in \mathbf{R}^{(n+1)\times(n+1)}, \ R_0 \in \mathbf{R}^{n\times n},$$

wobei R_0 eine reguläre obere Dreieckmatrix ist. Die Tangente t ist dann nämlich gerade der $n+1$-te Spaltenvektor von Q, denn

$$0 = G't = (R_0^\top, 0)\underbrace{Q^\top t}_{=:r}, \ r \in \mathbf{R}^{n+1} \iff r_i = \pm\delta_{i,n+1} \ \Rightarrow \ t = \pm Q e^{(n+1)}.$$

Die Kleinste-Quadrate-Lösung zu $G'd = -G$ ergibt sich ebenfalls aus dieser QR-Zerlegung $d = -QE_{n,n+1}^\top (R_0^\top)^{-1} G$, vgl. (2.3.11 ff). Auch hier kann i.a. (außer im Anfangspunkt) für den Prädiktorschritt (6.1.17) die QR-Zerlegung des letzten Korrektorschritts zu Stelle s_{k-1} benutzt werden.

Bei einer Schrittweitensteuerung wie in (6.1.14) für das Verfahren (6.1.17),(6.1.18) ist es zweckmäßig, außer der anfänglichen Kontraktion $\|y^{(2)} - y^{(1)}\|/\|y^{(1)} - y^{(0)}\| \overset{!}{\le} q < 1$ auch den Winkel zwischen aufeinanderfolgenden Tangenten zu überwachen, $t_{k-1}^\top t_{k-2} \overset{!}{\ge} \tau > 0$.

Beispiel 6.1.3 Ein beliebtes Testbeispiel, dessen Schwierigkeit mit wachsender Dimension größer wird, ist

$$F(x) = 0, \quad F_i(x) := x_i - \exp\cos(i\sum_{j=1}^n x_j), \ i = 1, \ldots, n.$$

Mit $G(x,\lambda) = \lambda F(x) + (1-\lambda)x$ ergibt sich dafür die Einbettung

$$G(x,\lambda) = 0, \quad G_i(x,\lambda) := x_i - \lambda \exp\cos(i\sum_{j=1}^n x_i), \ i = 1, \ldots, n,$$

mit dem Startwert $z(0) = 0$. Die Bogenlängen der Lösungskurven wachsen mit der Dimension (stark) an von ca. $1.6\,(n = 2)$ auf ca. $60\,(n = 9)$ (Demo-Programm FORTSETZ.PAS).

Index

Ähnlichkeitstransformation, 41
Anlaufrechnung, 39

B-Spline, 12
Bernstein, 5
Beziér, 5
Bisektion, 62
Bogenlänge, 97

CG-Verfahren, 83
Cholesky-Zerlegung, 86
Courant
    Maximum-Minimum-Prinzip, 66, 90

Defekt, 23
Deflation, 52
diagonalisierbar, 42
Dreieck-Matrix, 27–29, 42, 44

Ecke, 33
    ausgeartet, 34
Eigenvektor, 40
Eigenwert, 40
Einbettung, 92
Einheitswurzel, primitive, 17

Faltungsprodukt, 22
Fourier-Entwicklung, 21
Fourier-Transformation, 19

Gauß,C.F., 23
Gerschgorin, Satz von, 69
Gleichungssystem
    linear, 27, 46
    nichtlinear, 92

Haupt-Minoren, 60
Hessenberg-Matrix, 44, 49, 51
Householder-Spiegelung, 27, 44

Inverse Iteration, 56, 58, 62
Inverse, Verallgemeinerte, 24, 97
    Berechnung, 30

Kleinste-Quadrate
    Lösung, 23, 97
    Methode, 23
Konditionszahl, 72
Konjugierte-Gradienten-Verfahren, 83, 89
Kontrollpunkte, 5
Konvergenz, 47, 48
    kubisch, 58
    quadratisch, 53
Korrektor, 94, 97, 98
Krylov-Raum, 82

Lösungskurve, 92
Leibniz-Regel, 13
Liniensuche, 79, 92
LR-Verfahren, 56
LR-Zerlegung, 27, 30, 56

Messung, 23

Newton-Verfahren, 92
normale Matrix, 43
Normalform
    Jordan-, 42
    Lineare Optimierung, 32
    Schur-, 42, 55

Optimierungsproblem, lineares, 32

Polynom
    charakteristisches, 40
    Orthogonal-, 76
Prädiktor, 94, 97, 98
Programm, lineares, 32
Pseudo-Inverse, 24

QR-Doppelschritt, 53
QR-Verfahren, 51, 56, 59

QR-Zerlegung, 27, 29, 30, 38, 42, 49–51, 53, 98

Rayleigh-Quotient, 56
Rekursionsformel, 76
Relaxationsverfahren, 72
Richardson-Iteration, 73, 77
  optimale Parameter, 74

Schlupfvariable, 33
Schrittweitensteuerung, 96
Simplex-Verfahren, 34, 36
  revidiertes, 36
Spektralverschiebung, 48, 51
Spline
  -Approximation, lokale, 15
  -Funktion, 3
  kubischer, 3
  Minimaleigenschaft, 3
  natürlicher, 3
SSOR-Verfahren, 87
Sturmsche Kette, 60

Tomographie, 31
Tridiagonal-Matrix, 7, 44, 89
  unzerlegbare, 59
Tridiagonalisierung, 90
Tschebyscheff
  Iteration, 77
  Polynom, 74–76, 91

Umkehrpunkt, 94, 97
Unterraum
  -Iteration, 50
  invarianter, 41, 50

Vektor-Iteration, 46, 55, 57, 77, 78, 91
Versuchsparameter, 23
Vielfachheit
  algebraische, 41
  geometrische, 40
Von-Mises-Iteration, 46
Vorkonditionierung, 86

Wielandt-Iteration, 48

Zerlegung der Eins, 12, 14
zulässige Punkte, 32
zyklische Matrix, 22