W. Oevel

Numerik I

Inhalt

Literatur

[Scw93] H.R. Schwarz: Numerische Mathematik. Stuttgart: Teubner 1993.
(hervorragende, leicht lesbare Einführung, sehr empfehlenswert)

[Sto93] J. Stoer: Numerische Mathematik 1. Berlin: Springer 1993.
[StB90] J. Stoer und R. Bulirsch: Numerische Mathematik 2. Berlin: Springer 1990.
(Klassiker, sehr umfangreich und vollständig, als erste Einführung weniger leicht lesbar als [Scw93], da sehr detailliert)

[ScW92] R. Schaback und H. Werner: Numerische Mathematik. Berlin: Springer 1992.
(gut lesbar, kompakter als [Scw93],[Sto93],[StB90])

[Oev96] W. Oevel: Einführung in die Numerische Mathematik. Heidelberg: Spektrum Akademischer Verlag 1996.
(leicht lesbare Einführung, sehr breit geschrieben, reduzierte Themenauswahl im Vergleich zu [Scw93],[Sto93],[StB90],[ScW93]. Ist eine sehr detaillierte Ausarbeitung der in dieser Vorlesung behandelten Themen.)

Sammlung weiterer Literatur:

1. Numerische Mathematik

Abramowitz, M.; Stegun, I. A. (Eds.): Handbook of Mathematical Functions. Washington: National Bureau of Standards 1965; New York: Dover 1982.

Atkinson, K. E.: Introduction to Numerical Analysis. New York: Wiley 1989.

Bunse, W.; Bunse-Gerstner, A.: Numerische lineare Algebra. Stuttgart: Teubner 1985.

Conte, S. D.; de Boor, C.: Elementary Numerical Analysis. An Algorithmic Approach. Singapore: McGraw–Hill 1986.

Deuflhard, P.; Hohmann, A.: Numerische Mathematik I. Eine algorithmisch orientierte Einführung. Berlin: de Gruyter 1993.

Deuflhard, P.; Bornemann, F.: Numerische Mathematik II. Integration gewöhnlicher Differentialgleichungen. Berlin: de Gruyter 1994.

Engeln–Müllges, G.; Reutter, F.: Numerische Mathematik für Ingenieure. Mannheim: Bibliographisches Institut 1994.

Golub, G. H.; Van Loan, C. F.: Matrix Computations. Baltimore: Johns Hopkins University Press 1989.

Golub, G. H.; Ortega, J. M.: Wissenschaftliches Rechnen und Differentialgleichungen. Eine Einführung in die Numerische Mathematik. Lemgo: Heldermann 1995.

Golub, G. H.; Ortega, J. M.: Scientific Computing. Eine Einführung in das wissenschaftliche Rechnen und Parallele Numerik. Stuttgart: Teubner 1996.

Hämmerlin, G.; Hoffmann, K.-H.: Numerische Mathematik. Berlin: Springer 1994.

Hager, W. W.: Applied Numerical Linear Algebra. Englewood Cliffs. Prentice Hall 1988.

Hart, J. F.; et al.: Computer Approximations. New York: Wiley 1978.

Isaacson, E.; Keller, H. B.: Analyse numerischer Verfahren. Frankfurt: Deutsch 1973.
– Nachdruck der amerikanischen Originalausgabe: Analysis of Numerical Methods.
New York: Dover 1994.

Kiełbasiński, A.; Schwetlick, H.: Numerische lineare Algebra. Berlin: Deutscher Verlag der Wissenschaften 1988.

Köckler, N.: Numerische Algorithmen in Softwaresystemen – unter besonderer Berücksichtigung der NAG-Bibliothek. Stuttgart: Teubner 1990.

Locher, F.: Numerische Mathematik für Informatiker. Berlin: Springer 1993.

Maess, G.: Vorlesungen über Numerische Mathematik. Band I: Lineare Algebra. Band II: Analysis. Basel: Birkhäuser 1985, 1987.

Maron, M. J.: Numerical Analysis: A Practical Approach. New York: Macmillan 1991.

Oevel, W.: Einführung in die Numerische Mathematik. Heidelberg: Spektrum 1996.

Opfer, G.: Numerische Mathematik für Anfänger. Braunschweig: Vieweg 1994.

Schaback, R.; Werner, H.: Numerische Mathematik. Berlin: Springer 1993.

Schwarz, H. R.: Numerische Mathematik. Stuttgart: Teubner 1993.

Späth, H.: Numerik. Eine Einführung für Mathematiker und Informatiker. Braunschweig: Vieweg 1994.

Stoer, J.: Numerische Mathematik 1, 2 (Band 2 gemeinsam mit R. Bulirsch). Berlin: Springer 1994, 1991 .

Stummel, F.; Hainer, K.: Praktische Mathematik. Stuttgart: Teubner 1982.

Törnig, W.; Spellucci, P.: Numerische Mathematik für Ingenieure und Physiker. Band 1: Numerische Methoden der Algebra. Band 2: Eigenwertprobleme und numerische Methoden der Analysis. Berlin: Springer 1988, 1990.

Überhuber, C.: Computer–Numerik 1, 2. Berlin: Springer 1995.

Werner, J.: Numerische Mathematik. Band 1: Lineare und nichtlineare Gleichungssysteme, Interpolation, numerische Integration. Band 2: Eigenwertaufgaben, lineare Optimierungsaufgaben, unrestringierte Optimierungsaufgaben. Braunschweig: Vieweg 1992.

2. Programmsammlungen

Anderson, E.; Bai, Z.; Bischof, C.; et al.: LAPACK Users' Guide. Philadelphia: SIAM 1995.

Dongarra, J. J.; Bunch, J. R.; Moler, C. B.; Stewart, G. W.: LINPACK Users' Guide. Philadelphia: SIAM 1979.

Engeln–Müllges, G.; Uhlig, G.: Numerical Algorithms with Fortran. Berlin: Springer 1996.

Garbow, B. S.; Boyle, J. M.; Dongarra, J. J.; Moler, C. B.: Matrix Eigensystem Routines – EISPACK Guide Extension. Berlin: Springer 1990.

NAG Fortran Library. Oxford: Numerical Algorithms Group.
– Introductory Guide, Mark 16. 1993.
– Manual, Mark 16, Vols. 1–12. 1993.
– Mark 16, Contents Summary. 1993.

Piessens, R.; de Doncker–Kapenga, E.; Überhuber, C.W.; Kahaner, D. K.: QUADPACK, A Subroutine Package for Automatic Integration. Berlin: Springer 1983.

Press, W. H.; Flannery, B. P.; Teukolsky, S. A.: Numerical Recipes in FORTRAN: The Art of Scientific Computing. Cambridge: Cambridge University Press 1993.

Rice, J.; Boisvert, R. F.: Solving Elliptic Problems Using ELLPACK. Berlin: Springer 1985.

Schwarz, H. R.: FORTRAN–Programme zur Methode der finiten Elemente. Stuttgart: Teubner 1991.

Smith, B.T.; Boyle, J.M.; Dongarra, J.J.; Garbow, B.S.; Ikebe, Y.; Klema, V. C.; Moler, C. B.: Matrix Eigensystem Routines – EISPACK Guide. Berlin: Springer 1990.

Abkürzung SIAM : Society for Industrial and Applied Mathematics

Kapitel 1

Kurzer Ausblick

Numerik: approximative Lösung eines mathem. Problems, Zahlenwerte als Ergebnis $\boxed{\downarrow 14.10.97}$

Aspekte:

1) Theorie: finde praktisch durchführbares Verfahren zur Approximation des Problems
2) Verfahrensfehler
3) Effizienz
4) Stabilität (Verstärkung von Fehlern im Lauf der Rechnung)

Beispiel 1.1: Berechne $I := \int_0^1 e^{-x^2} dx$. Verfahren: approximiere $\int_a^b f(x)dx$ durch

$$
\begin{array}{rlcl}
 & I_{Riemann} & = & h \sum_{i=0}^{n-1} f(x_i)\ , \quad x_i = a + i\,h\ , \quad h = \dfrac{b-a}{n} \\
\text{oder} & I_{Trapez} & = & h\left(\dfrac{f(a)}{2} + \sum_{i=1}^{n-1} f(x_i) + \dfrac{f(b)}{2} \right) \\
\text{oder} & I_{Simpson} & \stackrel{(n\ gerade)}{=} & \dfrac{h}{6}\Big(f(x_0) + 4f(x_1) + 2f(x_2) + 4f(x_3) + 2f(x_4) + \cdots \\
 & & & \cdots + 4f(x_{n-3}) + 2f(x_{n-2}) + 4f(x_{n-1}) + f(x_n) \Big)
\end{array}
$$

1) Theorie: für $n \to \infty$ konvergieren diese Werte gegen I.

2) Verfahrensfehler: die Theorie liefert Abschätzungen, aus denen (in der Regel)

$$Simpson \quad besser\ als \quad Trapez \quad besser\ als \quad Riemann$$

folgt. In der Tat:

$$n = 10 \quad \to \quad \begin{cases} |I - I_{Riemann}| & \approx & 0.031 \\ |I - I_{Trapez}| & \approx & 0.00061 \\ |I - I_{Simpson}| & \approx & 8.2 \times 10^{-7} \end{cases}$$

3) zur Effizienz:

Beispiel 1.2: Berechne für gegebene Werte $f_0, \ldots, f_{N-1}$

$$d_k = \sum_{j=0}^{N-1} f_j \; e^{\sqrt{-1}\; jk\, 2\pi/N} \;, \quad k = 0, 1, \ldots, N-1$$

("diskrete Fourier-Transformation" $(f_0, \ldots, f_{N-1}) \mapsto (d_0, \ldots, d_{N-1})$).

Naiv:	man braucht $\approx N^2$ Rechenoperationen
FFT-Algorithmus:	es geht mit $\approx N \log_2 N$ Rechenoperationen

4) zur Stabilität:

Beispiel 1.3: Betrachte lineares Gleichungssystem

$$\begin{pmatrix} 10 & 7.1 \\ 2 & \sqrt{2} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 10 \\ 4 \end{pmatrix}$$

mit exakter Lösung

$$x = 2\,\frac{71 - 25\,\sqrt{2}}{71 - 50\,\sqrt{2}} \approx 246.401\ldots\,, \quad y = -\,\frac{100}{71 - 50\,\sqrt{2}} \approx -345.635\ldots\,.$$

Problem: ersetze $\sqrt{2}$ durch numerische Approximation $\sqrt{2} \mathrel{\widehat{=}} 1.41421...$, etwa:

$$\begin{array}{lll} \sqrt{2} \mathrel{\widehat{=}} 1.41 & \to \; x = 143.000\ldots & y = -200.000\ldots \\ \sqrt{2} \mathrel{\widehat{=}} 1.414 & \to \; x = 237.666\ldots & y = -333.333\ldots \\ \sqrt{2} \mathrel{\widehat{=}} 1.4142 & \to \; x = 245.827\ldots & y = -344.827\ldots \end{array}$$

Phänomen: Ungenauigkeiten können sich dramatisch verstärken (das Problem ist "schlecht konditioniert", "instabil"). Hier wird ein Fehler in $\sqrt{2}$ in x um einen Faktor $\approx 42\,410$, in y um einen Faktor $\approx 59\,732$ verstärkt (Aufgabe 4). Verbesserung:

$$y = -\,\frac{100}{71 - 50\,\sqrt{2}} = -\,\frac{100\,(71 + 50\,\sqrt{2})}{(71 - 50\,\sqrt{2})(71 + 50\,\sqrt{2})} = \frac{-100\,(71 + 50\,\sqrt{2})}{41}$$

$$= -\frac{7100}{41} - \frac{5000}{41}\,\sqrt{2} = \begin{cases} -345.129\ldots & \text{für } \sqrt{2} \mathrel{\widehat{=}} 1.41 \\ -345.609\ldots & \text{für } \sqrt{2} \mathrel{\widehat{=}} 1.414 \\ -345.634\ldots & \text{für } \sqrt{2} \mathrel{\widehat{=}} 1.4142 \end{cases}$$

Jetzt nur noch Fehlerverstärkung um den Faktor $5000/41 \approx 122$.

Fazit: Aufgabe der Numerik ist die Entwicklung effizienter und stabiler Algorithmen zur zahlenmäßigen Approximation mathematischer Probleme.

Nützliche Notation:

Definition 1.4: (Landau-Symbole)
Für Funktionen $f, g : \mathbb{R} \mapsto \mathbb{R}$ (oder $\mathbb{N} \mapsto \mathbb{R}$) bedeutet

$$f(x) = O(g(x)) \quad \textit{im Limes} \quad x \to x_0 \ ,$$

daß $f(x)/g(x)$ in einer Umgebung von x_0 beschränkt ist. Die Symbolik

$$f(x) = o(g(x)) \quad \textit{im Limes} \quad x \to x_0$$

steht für die Aussage $\lim\limits_{x \to x_0} f(x)/g(x) = 0$.

Beispiele:

$$\frac{n^3 + n^2}{3} = O(n^3) \quad (\text{im Limes } n \to \infty)$$

$$1 - \cos(x) = o(x) \quad (\text{im Limes } x \to 0)$$

Kapitel 2

Fehleranalyse

$\boxed{\downarrow 15.10.97}$

$$\text{Eingabeparameter } \vec{x} \longrightarrow \boxed{\begin{array}{c}\text{Numerisches}\\ \text{Verfahren}\\ \text{(Algorithmus)}\end{array}} \longrightarrow \text{Ausgabewerte } \vec{y}$$

Fehlerquellen:

- **Eingabefehler:** $\vec{x}$ ist mit Fehlern behaftet (z.B. Meßdaten, durch numerische Rechnung entstanden, ...)
- **Verfahrensfehler:** das Verfahren ist nicht exakt (z.B. Integralapproximation durch eine endliche Riemann-Summe)
- **Rundungsfehler:** Zahlen (z.B. $\pi = 3.1415...$) werden nicht exakt dargestellt, diese Fehler pflanzen sich beim Rechnen fort.

Eingabefehler: hängen vom Kontext des konkreten Problems ab.
Verfahrensfehler: sind mit dem jeweiligen Algorithmus zu diskutieren (Theorie).
Rundungsfehler: mathematisch kaum *exakt* zu kontrollieren. (Zukunft: Ergebnisverifikation, Intervallrechnung durch Computer)

In diesem Kapitel werden heuristische Hilfsmittel zur approximativen Analyse von Rundungsfehlern und ihrer Fortpflanzung bereitgestellt.

2.1 Gleitpunktdarstellung (floating point numbers)

Im Rechner wird nur begrenzter Speicherplatz zur Darstellung reeller Zahlen reserviert $\Longrightarrow$ die Zahlenwelt des Rechners ("Maschinenzahlen") besteht nur aus *endlich vielen rationalen Zahlen.* Praktisch alle $x \in \mathbb{R}$ müssen hierdurch approximiert werden und es kommt *unvermeidlich* zu **Darstellungs- (=Rundungs-)fehlern**.

Ziel: approximiere "große" Zahlen (z.B. $\pi^{10} \times e^{30}$) und "kleine" Zahlen (z.B. $e^{-20}/40!$) mit derselben Güte. Dies wird durch Gleitpunktdarstellung erreicht. Man wählt eine Basis $E \in \{2, 3, 4, \ldots\}$ und stellt eine reelle Zahl dar durch:

Definition 2.1:

Die **Gleitpunktdarstellung** *von* $x \in \mathbb{R} \setminus \{0\}$ *zur* *Basis* E *ist*

$$x \; = \; \mathrm{sign}(x) \; a \times E^b$$

mit dem *Vorzeichen* $\mathrm{sign}(x) = \pm$, *der* *Mantisse* $a \in [E^{-1}, 1) \subset \mathbb{R}$ *und dem* *Exponenten* $b \in \mathbb{Z}$.

Beispiel 2.2: Zur Basis $E = 10$:

$$\begin{aligned} x &= -\pi^{10}\, e^{30} &&= -\,0.100076... \times 10^{19} \\ x &= e^{-20}/40! &&= +\,0.2526... \times 10^{-56} \\ x &= 10 &&= +\,0.100... \times 10^{2} \end{aligned}$$

Bemerkung 2.3: *Bei gegebener Basis ist diese Darstellung eindeutig:*

$$b = \lfloor \log_E(|x|) + 1 \rfloor \; , \quad a = |x| \; E^{-b}$$

(mit $\lfloor y \rfloor$ *= Gauß-Klammer = größte ganze Zahl* $\leq y$*).*

Die Mantisse wird in Ziffern zerlegt:

Lemma 2.4:

Zu $a \in [E^{-1}, 1)$ *existieren* **Ziffern** $a_1, a_2, \ldots \in \{0, 1, \ldots, E-1\}$ *mit* $a_1 \neq 0$, *sodaß*

$$a = a_1\, E^{-1} + a_2\, E^{-2} + a_3\, E^{-3} + \cdots \; .$$

Beweis: Algorithmische Konstruktion

$$a_1 = \lfloor a\, E \rfloor \; , \quad a_2 = \lfloor (a - a_1 E^{-1})\, E^2 \rfloor \; , \quad a_3 = \lfloor (a - a_1 E^{-1} - a_2 E^{-2})\, E^3 \rfloor$$

usw. Nach Aufgabe 1 gilt $0 \leq a - (a_1 E^{-1} + \cdots + a_n E^{-n}) < E^{-n}$, also

$$a = \lim_{n \to \infty} \sum_{i=1}^{n} a_i\, E^{-i} \; .$$

Q.E.D.

Für $E = 10$ ist dies unsere gewohnte Dezimalschreibweise:

$$\begin{array}{ccccccccc} a & = & 0 & . & 1 & 2 & 3 & 4 & 5 \ \ldots \\ & & & & \uparrow & \uparrow & \uparrow & & \\ & & & & a_1 & a_2 & a_3 & \ldots & \end{array}$$

Achtung! $\quad 0.19999... \equiv 0.20000...$.
Allgemein gilt:

$$\sum_{i=n_0+1}^{\infty} (E-1)E^{-i} = (E-1)E^{-(n_0+1)} \sum_{i=0}^{\infty} E^{-i} = \frac{(E-1)E^{-(n_0+1)}}{1-1/E} = E^{-n_0} \ .$$

Vereinbarung 2.5:

Eine Ziffernfolge der Form $a_{n_0+1} = a_{n_0+2} = \ldots = E-1$ (mit $a_{n_0} < E-1$) ist nicht zulässig und wird umgeschrieben:

$$(a_1, \ldots, a_{n_0}, E-1, E-1, \ldots) = (a_1, \ldots, a_{n_0}+1, 0, 0, \ldots) \ .$$

Lemma 2.6:

Mit Vereinbarung 2.5 ist die Ziffernfolge für $a \in [E^{-1}, 1)$ eindeutig.

Beweis: Betrachte zwei Ziffernfolgen

$$a = \sum_{i=1}^{n-1} a_i E^{-i} + \sum_{i=n}^{\infty} a_i E^{-i} = \sum_{i=1}^{n-1} a_i E^{-i} + \sum_{i=n}^{\infty} \tilde{a}_i E^{-i}$$

mit $a_n \neq \tilde{a}_n$, o.B.d.A. $a_n > \tilde{a}_n$. Es folgt

$$(a_n - \tilde{a}_n)E^{-n} = \sum_{i=n+1}^{\infty} (\tilde{a}_i - a_i)E^{-i} \overset{(Vereinb.\ 2.5)}{<} \sum_{i=n+1}^{\infty} (E-1)E^{-i} = E^{-n} \ .$$

Widerspruch zu $(a_n - \tilde{a}_n)E^{-n} \geq E^{-n}$.

Q.E.D.

Bei "N-stelliger Darstellung" werden nur die ersten N Ziffern einer Mantisse gespeichert, der Rest wird abgeschnitten (bzw. gerundet):

Definition 2.7: (Rundung, "round")

Definiere $\mathrm{rd} : [E^{-1}, 1) \mapsto [E^{-1}, 1) \cap \mathbb{Q}$:

$$\begin{aligned} \mathrm{rd}(a) &= \mathrm{rd}(a_1 E^{-1} + \cdots + a_N E^{-N} + a_{N+1} E^{-(N+1)} + \cdots) \\ &:= a_1 E^{-1} + \cdots + a_N E^{-N} \qquad (\text{“Abrundung”}) \ . \end{aligned}$$

Alternativ ersetzt man a_N durch $a_N + 1$, falls

$$\sum_{i=N+1}^{\infty} a_i E^{-i} \geq \frac{1}{2} E^{-N}$$

gilt ("eigentliche Rundung"). Hierbei sind eventuell rekursiv Überträge abzuarbeiten, falls sich $a_N + 1 = E$ ergibt:

z.B. $\quad \mathrm{rd}(0.4999678) = 0.5000 \quad$ bei Rundung auf $N = 4$ Dezimalziffern.

Die sich durch Abrundung oder eigentliche Rundung ergebende rationale Zahl $\mathrm{rd}(a)$ approximiert $a \in [E^{-1}, 1)$:

$$|a - \mathrm{rd}(a)| \quad \begin{cases} < E^{-N} & \text{(Abrundung)} \\ \leq \frac{1}{2} E^{-N} & \text{(eigentliche Rundung)} \end{cases}$$

Mit $a \geq E^{-1}$ folgt

$$\left| \frac{a - \mathrm{rd}(a)}{a} \right| \quad \begin{cases} < E^{1-N} & \text{(Abrundung)} \\ \leq \frac{1}{2} E^{1-N} & \text{(eigentliche Rundung)} \end{cases}$$

Definition 2.8:

Eine Zahl $x \in \mathbb{R} \backslash \{0\}$ wird im Rechner durch die **N-stellige Gleitpunktapproximation zur Basis E**

$$\mathrm{rd}(x) = \mathrm{rd}(\mathrm{sign}(x)\, a \times E^b) := \mathrm{sign}(x)\, \mathrm{rd}(a) \times E^b \,\widehat{=}\, (\pm, a_1, \ldots, a_N, b)$$

dargestellt. Es gilt

$$\left| \frac{x - \mathrm{rd}(x)}{x} \right| = \left| \frac{a - \mathrm{rd}(a)}{a} \right| \leq \tau := \begin{cases} E^{1-N} & \textit{(Abrundung)} \\ \frac{1}{2} E^{1-N} & \textit{(eigentliche Rundung)} \end{cases}$$

mit der **(relativen) Maschinengenauigkeit** τ.

Grob gesprochen ist $\mathrm{rd}(x)$ die $x \in \mathbb{R}$ nächstgelegene rationale Zahl, die auf dem Rechner dargestellt werden kann. Typisch ist $E = 2$ (binär) oder $E = 16$ (hexa-dezimal).

16.10.97↓ **Beispiel 2.9:** 64-bit Variablen, $E = 2$:

$$\overbrace{\underset{\substack{\nearrow\\ \text{Vorzeichen-}\\ \text{bit}}}{\pm}\,|\,\underbrace{\quad\ldots\ldots\ldots\quad}_{\substack{\text{52 bits für die}\\ \text{Mantisse } a}}\,|\,\underbrace{\;\ldots\;}_{\substack{\text{11 bits für den}\\ \text{Exponenten } b}}}^{\text{64 bits}}$$

Exponentenbereich $b_{min}, \ldots, b_{max}$**:** Mit 11 bits lassen sich die ganzen Zahlen $\underbrace{(0,\ldots,0)}_{10} \widehat{=} 0$ bis $\underbrace{(1,\ldots,1)}_{10} \widehat{=} 2^0 + 2^1 + \cdots + 2^9 = 1023$ und ein Vorzeichen darstellen, also $b \in \{-1023, \ldots, 1023\}$, also $E^b \in \{2^{-1023}, \ldots, 2^{1023}\}$. Es gilt $2^{\pm 1023} \approx 10^{\pm 308}$.

Mantissengenauigkeit: Mit 52 bits lassen sich

$$\underbrace{a_1}_{=1} \;,\; a_2 \;,\; a_3 \;,\; \ldots \;,\; a_{53} \;\in \{0,1\} \qquad \text{(braucht nicht gespeichert zu werden)}$$

speichern, also $N = 53$, d.h., bei eigentlicher Rundung: $\tau = \frac{1}{2} 2^{1-53} \approx 1.1 \times 10^{-16}$.

Ergebnis: Mit 64 bits lassen sich alle reellen Zahlen x im Bereich $10^{-308} \le |x| \le 10^{308}$ bis auf etwa 16 Dezimalstellen genau approximieren.

Einige FORTRAN-90 Compiler bieten 3 Datentypen für reelle Variablen an:

Typ	bits	τ	$E^{b_{min/max}}$	
REAL (KIND=4)	32	$\approx 10^{-7}$	$\approx 10^{\pm 38}$	(REAL (normal))
REAL (KIND=8)	64	$\approx 10^{-16}$	$\approx 10^{\pm 308}$	(DOUBLE PRECISION)
REAL (KIND=16)	128	$\approx 10^{-31}$	$\approx 10^{\pm 308}$	

Definition 2.10:

Für eine gegebene Basis E, eine Stellenzahl N und einen Exponentenbereich $b_{min}, \ldots, b_{max}$ ist

$$\mathcal{M} = \{0\} \;\cup\; \left\{ \pm\Big(\sum_{i=1}^{N} a_i E^{-i} \Big) E^b \;;\quad a_i \in \{0, 1, \ldots, E-1\} \;, \right.$$
$$\left. a_1 \neq 0 \;,\quad b \in \{b_{min}, \ldots, b_{max}\} \right\}$$

die Menge der vom Rechner darstellbaren **Maschinenzahlen**.

Sie ist durch die Software (Compiler) bzw. die Hardware (floating-point-Prozessor) vorgegeben. Sollte in einer Rechnung der Exponentenbereich in $\mathcal{M}$ verlassen werden (overflow/underflow), so wird in der Regel das Symbol `NaN` (=Not a Number) ausgegeben.

Merke: Die Maschinengenauigkeit τ ist *die Größe*, welche die Rundungsfehler beschreibt. Sie ist der Beschreibung des Compilers zu entnehmen und hängt von der internen Darstellung E und der Speicherlänge N ab. Innerhalb des Exponentenbereichs $b_{min}, \ldots, b_{max}$ können alle reellen Zahlen $x \in \mathbb{R}$ mit einer durch τ beschränkten *relativen* Genauigkeit durch $\mathrm{rd}(x) \in \mathcal{M}$ approximiert werden:

$$\mathrm{rd}(x) = x + x\,\tau_x \quad \text{mit} \quad |\tau_x| \leq \tau = \textit{Maschinengenauigkeit},$$

wobei

$$\tau_x := \frac{\mathrm{rd}(x) - x}{x}$$

(relativer) Darstellungsfehler der Zahl $x \in \mathbb{R}$ heißt.

2.2 Arithmetik

Als **Grundarithmetik** wird bezeichnet:

- die 4 Grundrechenarten $+, -, *, /$ und Exponentiation $**$ ($x{**}\,y = x^y$),
- einfache Funktionsauswertungen der dem System bekannten Funktionen: `exp,log,sin,cos,arctan,sqrt,...` .

Problem: die Grundarithmetik verläßt i.a. die Maschinenzahlen, z.B.

$$x, y \in \mathcal{M} \not\Rightarrow x + y \in \mathcal{M} \ .$$

(z.B.: $x = 1, y = 10^{-20}$ sind im Dezimalsystem exakt mit einer Ziffer darstellbar, $x + y$ braucht 21 Dezimalstellen.)

Die Arithmetik der meisten Sprachen ist nach folgendem Rechenmodell implementiert:

Rechenmodell 2.11:

Die von der Maschine ausgeführte Operation ist die mathematisch exakte Operation gefolgt von der Rundung auf die nächste Maschinenzahl:

Programmzeile	*berechnet wird*
`z:=x+y;`	$z = x \oplus y := \mathrm{rd}(x+y) = x + y \ + \ (x+y)\,\tau_z$
`z:=exp(x);`	$z = \widehat{\exp}(x) := \mathrm{rd}(\exp(x)) = \exp(x) \ + \ \exp(x)\,\tau_z$

mit $|\tau_z| \leq \tau =$ Maschinengenauigkeit.

Merke: Jede Grundoperation liefert einen durch die Maschinengenauigkeit beschränkten *relativen* Fehler.

Bemerkung 2.12: *τ ist die kleinste positive Zahl, sodaß $1+\tau$ nicht durch Rundung zu 1 wird. Damit kann τ leicht ausgetestet werden:*

```
tau:=1;
while 1+tau>1 do tau:=tau/2;
```

Häßlich: die Ersatzarithmetik $\mathcal{M} \times \mathcal{M} \mapsto \mathcal{M}$ wie z.B. $\oplus = \mathrm{rd}(\,.+.\,)$ ist nicht mehr assoziativ, z.B. bei $\tau \approx 10^{-16}$:

$$\underbrace{(\underbrace{10^{-20} \oplus 1}_{1}) \oplus (-1)}_{0} \quad \neq \quad \underbrace{10^{-20} \oplus (\underbrace{1 \oplus (-1)}_{0})}_{10^{-20}}$$

2.3 Fehlerfortpflanzung

Gegeben glattes $f : \vec{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n \mapsto \mathbb{R}$. Wie ändert sich f bei Änderung von $\vec{x}$ auf $\vec{x} + \Delta\vec{x}$? Taylorentwicklung:

$$f(\vec{x} + \Delta\vec{x}) = f(\vec{x}) + \sum_{i=1}^{n} \frac{\partial f}{\partial x_i}\, \Delta x_i + \frac{1}{2!} \sum_{i,j=1}^{n} \frac{\partial^2 f}{\partial x_i \partial x_j}\, \Delta x_i \Delta x_j \; + \; \cdots \; .$$

Interpretiere Δx_i als "kleine Fehler" und vernachlässige höhere Terme:

Definition 2.13: (Differentielle Fehlerfortpflanzung)

$$\Delta f := \sum_{i=1}^{n} \frac{\partial f}{\partial x_i}\Big|_{\vec{x}}\, \Delta x_i \; + \; f(\vec{x})\, \tau_f$$

heißt **differentieller Fehler** *der Abbildung $f : \vec{x} \in \mathbb{R}^n \mapsto \mathbb{R}$ (am Punkt $\vec{x}$). Er besteht aus der Fortpflanzung der Eingabefehler Δx_i, die über die* **absoluten Konditionszahlen** *$\partial f / \partial x_i$ verstärkt werden. Hinzu kommt der Darstellungsfehler τ_f beim Abspeichern des berechneten Wertes $f(\vec{x})$, für den $|\tau_f| \leq \tau =$* Maschinengenauigkeit *gilt.*

Interpretation: ohne τ_f ist in linearer Näherung $\Delta f = f(\vec{x} + \Delta\vec{x}) - f(\vec{x})$ der von den Fehlern der Eingangsparameter stammende Fehler. Die höheren Terme der Taylorentwicklung werden ignoriert:

Achtung: der differentielle Fehler ist nur eine Näherung des wirklichen Fehlers (aber eine in der Regel sehr gute).

Beispiel 2.14: Berechne bei Maschinengenauigkeit τ den Wert $z = \sqrt{3}/\sqrt{2}$. Wie genau ist das Ergebnis? Setze $x = \sqrt{3}$, $y = \sqrt{2}$ mit den Darstellungsfehlern $\Delta x = \mathrm{rd}(x) - x = x\,\tau_x$, $\Delta y = \mathrm{rd}(y) - y = y\,\tau_y$ (nach Rechenmodell 2.11). Mit $z = f(x,y) := x/y$ folgt die differentielle Fehlerfortpflanzung ↓28.10.97

$$\Delta z = \frac{\partial f}{\partial x}\Delta x + \frac{\partial f}{\partial y}\Delta y \ + \ z\,\tau_z = \frac{1}{y}\,\Delta x - \frac{x}{y^2}\,\Delta y \ + \ z\,\tau_z$$

$$\Longrightarrow \quad \frac{\Delta z}{z} = \frac{\Delta x}{x} - \frac{\Delta y}{y} + \tau_z = \tau_x - \tau_y + \tau_z \ .$$

Nur bekannt: $|\tau_x|, |\tau_y|, |\tau_z| \leq \tau$. Ergebnis: $\left|\frac{\Delta z}{z}\right| \leq 3\,\tau$.

Bezeichnung 2.15:

Die **absoluten Fehler** *von Zahlen* $x, y, \ldots$ *werden mit* $\Delta x, \Delta y, \ldots$ *bezeichnet, die* **relativen Fehler** *mit* $\epsilon_x = \frac{\Delta x}{x}$, $\epsilon_y = \frac{\Delta y}{y}$, $\ldots$.

Relative Fehler sind meist aussagekräftiger als absolute Fehler, Rundungsfehler sind relative Fehler.

2.3.1 Fehlerfortpflanzung in der Grundarithmetik

Gegeben x, y mit den Fehlern $\Delta x, \Delta y$ bzw. ϵ_x, ϵ_y. Es sei τ_z mit $|\tau_z| \leq \tau =$ *Maschinengenauigkeit* der Darstellungsfehler des aus x, y berechneten Ergebnisses.

Addition zweier Zahlen mit gleichem Vorzeichen:

$$z = f(x,y) = x + y \ \Rightarrow \ \Delta z = \Delta x + \Delta y + z\,\tau_z \ \Rightarrow \ \epsilon_{x+y} = \frac{x}{x+y}\,\epsilon_x + \frac{y}{x+y}\,\epsilon_y + \tau_z$$

Gleiches Vorzeichen $\Rightarrow \ 0 \leq \dfrac{x}{x+y} \leq 1$, $0 \leq \dfrac{y}{x+y} \leq 1$, Fehler der Summanden werden also nicht verstärkt. Offensichtlich gilt

$$\boxed{|\epsilon_{x+y}| \ \leq \ \max(|\epsilon_x|, |\epsilon_y|) + |\tau_z|} \qquad \text{(Addition ist harmlos!)}$$

Subtraktion zweier Zahlen mit gleichem Vorzeichen: ersetze y durch $-y$:

$$z = x - y : \quad \boxed{\epsilon_{x-y} = \frac{x}{x-y}\,\epsilon_x - \frac{y}{x-y}\,\epsilon_y \ + \ \tau_z}$$

Die relativen Fehler werden um die Faktoren $|x/(x-y)|$, $|y/(x-y)|$ verstärkt, von denen mindestens einer ≥ 1 ist. Kritisch, wenn

$$x \approx y \ , \ \text{d.h.} \ \frac{x}{y} \approx \frac{y}{x} \approx 1 \ , \ \text{d.h.} \ \left|1 - \frac{x}{y}\right| \approx \left|1 - \frac{y}{x}\right| \ll 1 \ , \ \text{d.h.} \ |x - y| \ll |x| \approx |y| \ ,$$

da dann $\left|\frac{x}{x-y}\right| \approx \left|\frac{y}{x-y}\right| \gg 1$.

Merke: Bei **Auslöschung** $|x-y| \ll |x| \approx |y|$ werden relative Fehler in x und y extrem verstärkt. Dies ist die Hauptursache für Rechenfehler.

Beispiel:

x	0.123456???	(auf 6 Stellen genau)
$-\ y$	−0.123444???	(auf 6 Stellen genau)
$=\ z$	0.000012???	(auf 2 Stellen genau)

Damit sinnvoll:

Vereinbarung 2.16:

In Zukunft hat "≈" eine konkrete Bedeutung:

$$x \approx y \text{ heißt: } \quad |x-y| \ll |x| \ , \quad \text{also} \quad \left|1-\frac{y}{x}\right| \ll 1$$

$$\text{und damit auch} \quad |x-y| \ll |y| \ , \quad \text{also} \quad \left|1-\frac{x}{y}\right| \ll 1 \ .$$

Achtung: $x \approx 0$ *macht keinen Sinn!*

Weiter:

Multiplikation $z = x\,y$: $\boxed{\epsilon_z = \epsilon_x + \epsilon_y + \tau_z}$ (harmlos)

Division $z = x/y$: $\boxed{\epsilon_z = \epsilon_x - \epsilon_y + \tau_z}$ (harmlos)

Exponentiation $z = x^y$: $\boxed{\epsilon_z = y\,\epsilon_x + \ln(z)\,\epsilon_y + \tau_z}$ (kritisch für $|y| \gg 1$ oder $|\ln(z)| \gg 1$)

Funktionsauswertung $z = f(x)$: $\boxed{\epsilon_z = \dfrac{x f'(x)}{f(x)}\,\epsilon_x + \tau_z}$

z.B.

$z = \exp(x)$: $\boxed{\epsilon_{\exp(x)} = x\,\epsilon_x + \tau_z}$ (kritisch für $|x| \gg 1$)

$z = \ln(x)$: $\boxed{\epsilon_{\ln(x)} = \dfrac{\epsilon_x}{\ln(x)} + \tau_z}$ (kritisch für $x \approx 1$)

$z = \sqrt{x}$: $\boxed{\epsilon_{\sqrt{x}} = \dfrac{\epsilon_x}{2} + \tau_z}$ (harmlos)

$\vdots \qquad\qquad \vdots \qquad\qquad \vdots$

Noch einmal:

Zusammenfassung: Wesentliche Ursache für Rechenfehler ist die Differenzbildung zweier fast gleicher Zahlen $x \approx y$ ("Auslöschung"), genauer:

> *Falls x, y fehlerbehaftet sind, so ist für $|x-y| \ll |x| \approx |y|$ die Differenz $x - y$ mit einem extrem verstärkten relativen Fehler behaftet.*

Es tritt dann allerdings kein *zusätzlicher* Darstellungsfehler der Differenz auf (Aufgabe 3).

2.3.2 Differentielle Fehleranalyse

In jedem Rechenschritt(Grundarithmetik) entstehen Fehler. Wie analysiert man deren Effekt beim Weiterrechnen? Also: was passiert bei einer Hintereinanderschaltung mehrerer Grundoperationen?

Definition 2.17:

Ein **Algorithmus** $\vec{f} : \mathbb{R}^n \to \mathbb{R}^m$ *, der Eingabedaten $\vec{x} \in \mathbb{R}^n$ die Ausgabedaten $\vec{y} = \vec{f}(\vec{x}) \in \mathbb{R}^m$ zuordnet, ist eine Komposition $\vec{f} = \vec{f}^{(k)} \circ \cdots \circ \vec{f}^{(1)}$ von* **Elementaroperationen** $\vec{f_i} : \mathbb{R}^{n_{i-1}} \mapsto \mathbb{R}^{n_i}$ *der Grundarithmetik:*

$$\vec{x} \in \mathbb{R}^n \xrightarrow{input} \vec{x}^{(0)} := \vec{x} \xrightarrow{\vec{f}^{(1)}} \vec{x}^{(1)} \in \mathbb{R}^{n_1} \xrightarrow{\vec{f}^{(2)}} \ldots$$

$$\ldots \xrightarrow{\vec{f}^{(k)}} \vec{x}^{(k)} \in \mathbb{R}^m \xrightarrow{output} \vec{y} := \vec{x}^{(k)} .$$

Hierbei seien alle Komponenten der Zwischenergebnisse $\vec{x}^{(i)} = \vec{f}^{(i)}(\vec{x}^{(i-1)})$ durch Operationen der Grundarithmetik aus den Komponenten von $\vec{x}^{(i-1)}$ berechenbar.

Abbildungen, die sich nicht in diesem Sinne in Elementaroperationen zerlegen lassen, können nicht auf dem Rechner implementiert werden.

Beispiel 2.18: Berechne $y = \sqrt{x+1} - \sqrt{x-1}$. (Achtung: Auslöschungsproblem für $x \gg 1$!)
Algorithmus:

$$\begin{aligned} x \xrightarrow{input} x^{(0)} := x \quad &\longrightarrow \quad \vec{x}^{(1)} = \begin{pmatrix} x^{(0)} + 1 \\ x^{(0)} - 1 \end{pmatrix} = \begin{pmatrix} x+1 \\ x-1 \end{pmatrix} \\ &\longrightarrow \quad \vec{x}^{(2)} = \begin{pmatrix} \sqrt{x_1^{(1)}} \\ \sqrt{x_2^{(1)}} \end{pmatrix} = \begin{pmatrix} \sqrt{x+1} \\ \sqrt{x-1} \end{pmatrix} \\ &\longrightarrow \quad x^{(3)} = x_1^{(2)} - x_2^{(2)} = \sqrt{x+1} - \sqrt{x-1} \xrightarrow{output} y := x^{(3)} . \end{aligned}$$

Heuristisches Verfahren:

Methode der Differentiellen Fehleranalyse: benutze die differentielle Fehlerfortpflanzung 2.13 Schritt für Schritt (analog zu Beispiel 2.14), um den relativen differentiellen Fehler des Endergebnisses als *Linearkombination* des Eingangsfehlers und der Darstellungsfehler aller Zwischenergebnisse zu schreiben. Es interessieren dabei die Verstärkungsfaktoren (Konditionszahlen).

Beispiel 2.19: Es seien $\tau_i^{(j)}$ die relativen Darstellungsfehler der Zwischenergebnisse $x_i^{(j)}$ des Algorithmus

$$x^{(0)} = x\ , \qquad \begin{array}{ll} x_1^{(1)} = x^{(0)} + 1\ , & x_1^{(2)} = \sqrt{x_1^{(1)}}\ , \\ x_2^{(1)} = x^{(0)} - 1\ , & x_2^{(2)} = \sqrt{x_2^{(1)}}\ , \end{array} \qquad x^{(3)} = x_1^{(2)} - x_2^{(2)}\ , \quad y = x^{(3)}$$

aus Beispiel 2.18. Anwendung von 2.13 liefert

$$\begin{aligned}
\Delta x^{(0)} &= \Delta x \quad \text{(Eingabefehler)} \\
\Delta x_1^{(1)} &= \Delta x^{(0)} + x_1^{(1)}\,\tau_1^{(1)} && = \Delta x + (x+1)\,\tau_1^{(1)} \\
\Delta x_2^{(1)} &= \Delta x^{(0)} + x_2^{(1)}\,\tau_2^{(1)} && = \Delta x + (x-1)\,\tau_2^{(1)} \\
\Delta x_1^{(2)} &= \frac{\Delta x_1^{(1)}}{2\sqrt{x_1^{(1)}}} + x_1^{(2)}\,\tau_1^{(2)} && = \frac{\Delta x}{2\sqrt{x+1}} + \frac{\sqrt{x+1}}{2}\,\tau_1^{(1)} + \sqrt{x+1}\,\tau_1^{(2)} \\
\Delta x_2^{(2)} &= \frac{\Delta x_2^{(1)}}{2\sqrt{x_2^{(1)}}} + x_2^{(2)}\,\tau_2^{(2)} && = \frac{\Delta x}{2\sqrt{x-1}} + \frac{\sqrt{x-1}}{2}\,\tau_2^{(1)} + \sqrt{x-1}\,\tau_2^{(2)} \\
\Delta x^{(3)} &= \Delta x_1^{(2)} - \Delta x_2^{(2)} + x^{(3)}\,\tau^{(3)} && = \left(\frac{1}{\sqrt{x+1}} - \frac{1}{\sqrt{x-1}}\right)\frac{\Delta x}{2} \\
& && \quad + \sqrt{x+1}\left(\frac{\tau_1^{(1)}}{2} + \tau_1^{(2)}\right) \\
& && \quad - \sqrt{x-1}\left(\frac{\tau_2^{(1)}}{2} + \tau_2^{(2)}\right) + x^{(3)}\,\tau^{(3)}\ .
\end{aligned}$$

Ergebnis $y = x^{(3)}$ mit $\Delta y = \Delta x^{(3)}$: $\boxed{\downarrow 30.10.97}$

$$\frac{\Delta y}{y} = -\,\frac{x}{2\sqrt{x^2-1}}\,\frac{\Delta x}{x} + \frac{\sqrt{x+1}}{y}\left(\frac{\tau_1^{(1)}}{2} + \tau_1^{(2)}\right) - \frac{\sqrt{x-1}}{y}\left(\frac{\tau_2^{(1)}}{2} + \tau_2^{(2)}\right) + \tau^{(3)}\ .$$

Annahme $x \gg 1$ und Vereinfachung $\sqrt{x+1} \approx \sqrt{x-1} \approx \sqrt{x}\ ,\ \sqrt{x^2-1} \approx x\ ,$

$$y = \frac{(\sqrt{x+1} - \sqrt{x-1})\,(\sqrt{x+1} + \sqrt{x-1})}{\sqrt{x+1} + \sqrt{x-1}} = \frac{2}{\sqrt{x+1} + \sqrt{x-1}} \approx \frac{1}{\sqrt{x}}$$

$$\Longrightarrow \quad \boxed{\frac{\Delta y}{y} \approx -\frac{1}{2}\,\frac{\Delta x}{x} + x\left(\frac{\tau_1^{(1)}}{2} + \tau_1^{(2)} - \frac{\tau_2^{(1)}}{2} - \tau_2^{(2)}\right) + \tau^{(3)}\ .}$$

Der Eingangsfehler $\Delta x/x$ und $\tau^{(3)}$ sind harmlos, alle anderen Fehler werden um $x \gg 1$ verstärkt. Mit $|\tau_1^{(1)}|, \ldots, |\tau_3| \leq \tau = $ *Maschinengenauigkeit*:

$$\left|\frac{\Delta y}{y}\right| \overset{(approx.)}{\leq} \frac{1}{2}\left|\frac{\Delta x}{x}\right| + (3x+1)\,\tau\ .$$

Beispiel: $x = 1.234 \times 10^{14}$ mit 16-Stellen-Arithmetik

$$\begin{aligned} y_{exakt} &= 9.00207... \times 10^{-8} && (\approx 1/\sqrt{x}\,) \\ y_{Rechner} &= 8.94069... \times 10^{-8} && \text{(PASCAL)}\ , \end{aligned}$$

also

$$\frac{y_{exakt} - y_{Rechner}}{y_{exakt}} \approx \frac{6 \times 10^{-10}}{9 \times 10^{-8}} \approx 7 \times 10^{-3}\ .$$

Die Fehleranalyse sagt mit $|\Delta x/x| \leq \tau \approx 10^{-16}$ die obere Schranke

$$\left|\frac{\Delta y}{y}\right| \overset{(approx)}{\leq} \left(3\,x + \frac{3}{2}\right)\tau \ \leq\ 4 \times 10^{-2}$$

voraus.

Anmerkung: es gilt die alternative Darstellung

$$y = \sqrt{x+1} - \sqrt{x-1} = \frac{2}{\sqrt{x+1} + \sqrt{x-1}}\ .$$

Sie ist für $x \gg 1$ numerisch viel besser, da Auslöschung vermieden wird.

Man sieht: differentielle Fehleranalyse ist schon für kleine Beispiele aufwendig. Anwendung: analysiere kritische Bausteine in komplexen Algorithmen.

Formalisierung dieser Analysetechnik:

Für $\vec{y} = \begin{pmatrix} y_1(x_1,\ldots,x_n) \\ \vdots \\ y_m(x_1,\ldots,x_n) \end{pmatrix}$ gilt nach Definition 2.13 des differentiellen Fehlers $\Delta y_i = \sum_j \frac{\partial y_i}{\partial x_j}\,\Delta x_j + \tau_{y_i}\,y_i$, also

$$\Delta\vec{y} = \begin{pmatrix} \Delta y_1 \\ \vdots \\ \Delta y_m \end{pmatrix} = \begin{pmatrix} \frac{\partial y_1}{\partial x_1} & \cdots & \frac{\partial y_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial y_m}{\partial x_1} & \cdots & \frac{\partial y_m}{\partial x_n} \end{pmatrix} \begin{pmatrix} \Delta x_1 \\ \vdots \\ \Delta x_n \end{pmatrix} + \begin{pmatrix} \tau_{y_1} & & \\ & \ddots & \\ & & \tau_{y_m} \end{pmatrix} \begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix}$$

$$= \frac{\partial\vec{y}}{\partial\vec{x}}\,\Delta\vec{x} + \tau_{\vec{y}}\,\vec{y}$$

mit der **diagonalen Fehlermatrix** $\tau_{\vec{y}} = \mathrm{diag}(\tau_{y_1}, \ldots, \tau_{y_m})$ und der üblichen Jacobi-Matrix $\frac{\partial \vec{y}}{\partial \vec{x}} = \left(\frac{\partial y_i}{\partial x_j}\right)$ der Abbildung $\vec{x} \mapsto \vec{y}(\vec{x})$. Für die Komposition zweier Abbildungen $\vec{x} \mapsto \vec{y}(\vec{x}) \mapsto \vec{z}(\vec{y})$ folgt

$$\Delta \vec{z} \;=\; \frac{\partial \vec{z}}{\partial \vec{y}}\, \Delta \vec{y} \;+\; \tau_{\vec{z}}\, \vec{z} \;=\; \frac{\partial \vec{z}}{\partial \vec{y}} \frac{\partial \vec{y}}{\partial \vec{x}}\, \Delta \vec{x} \;+\; \frac{\partial \vec{z}}{\partial \vec{y}}\, \tau_{\vec{y}}\, \vec{y} \;+\; \tau_{\vec{z}}\, \vec{z}$$

mit $\frac{\partial \vec{z}}{\partial \vec{y}} \frac{\partial \vec{y}}{\partial \vec{x}} = \frac{\partial \vec{z}}{\partial \vec{x}}$ (Jacobi-Matrix der Abbildung $\vec{x} \mapsto \vec{z}(\vec{y}(\vec{x}))$) und den Darstellungsfehlern $\tau_{\vec{z}} = \mathrm{diag}(\tau_{z_1}, \tau_{z_2}, \ldots)$ der Komponenten von $\vec{z}$. Für einen allgemeinen Algorithmus

$$\vec{x} \stackrel{(input)}{\rightarrow} \vec{x}^{(0)} = \vec{x} \rightarrow \vec{x}^{(1)}(\vec{x}^{(0)}) \rightarrow \;\; \cdots \;\; \rightarrow \vec{x}^{(k)}(\vec{x}^{(k-1)}) \stackrel{(output)}{\rightarrow} \vec{y} = \vec{x}^{k}$$

ergibt sich die rekursive Fehlerfortpflanung

$$\Delta \vec{x}^{(i)} \;=\; \frac{\partial \vec{x}^{(i)}}{\partial \vec{x}^{(i-1)}}\, \Delta \vec{x}^{(i-1)} \;+\; \tau^{(i)}\, \vec{x}^{(i)}$$

mit den Darstellungsfehlern $\tau^{(i)} = \mathrm{diag}(\tau_1^{(i)}, \tau_2^{(i)}, \ldots)$ der Komponenten des Zwischenergebnisses $\vec{x}^{(i)} = (x_1^{(i)}, x_2^{(i)}, \ldots)^T$ (T=Transposition, alle Vektoren sind Spaltenvektoren!). Die offensichtliche Lösung dieser Rekursion

$$\Delta \vec{x}^{(i)} \;=\; \frac{\partial \vec{x}^{(i)}}{\partial \vec{x}^{(i-1)}} \cdots \frac{\partial \vec{x}^{(1)}}{\partial \vec{x}^{(0)}}\, \Delta \vec{x}^{(0)}$$

$$+\; \frac{\partial \vec{x}^{(i)}}{\partial \vec{x}^{(i-1)}} \cdots \frac{\partial \vec{x}^{(2)}}{\partial \vec{x}^{(1)}}\, \tau^{(1)} \vec{x}^{(1)} \;+\; \frac{\partial \vec{x}^{(i)}}{\partial \vec{x}^{(i-1)}} \cdots \frac{\partial \vec{x}^{(3)}}{\partial \vec{x}^{(2)}}\, \tau^{(2)} \vec{x}^{(2)} \;+\; \cdots$$

führt zu:

Definition 2.20:

Gegeben Algorithmus $\vec{f} = \vec{f}^{(k)} \circ \cdots \circ \vec{f}^{(1)}$ *mit Eingabeparametern* $\vec{x}^{(0)} = \vec{x}$ *und Zwischenergebnissen* $\vec{x}^{(i)} = \vec{f}^{(i)}(\vec{x}^{(i-1)})$, $i = 1, \ldots, k$. *Der* **absolute differentielle Fehler** *der Ausgabewerte* $\vec{y} = \vec{x}^{(k)} = \vec{f}(\vec{x})$ *ist*

$$\Delta \vec{y} \;=\; \frac{\partial \vec{y}}{\partial \vec{x}}\, \Delta \vec{x} \;+\; \sum_{j=1}^{k-1} \frac{\partial \vec{y}}{\partial \vec{x}^{(j)}}\, \tau^{(j)}\, \vec{x}^{(j)} \;+\; \tau^{(k)}\, \vec{y}$$

$$
\begin{aligned}
= \quad & \frac{\partial \vec{x}^{(k)}}{\partial \vec{x}^{(k-1)}} \frac{\partial \vec{x}^{(k-1)}}{\partial \vec{x}^{(k-2)}} \cdots \frac{\partial \vec{x}^{(2)}}{\partial \vec{x}^{(1)}} \frac{\partial \vec{x}^{(1)}}{\partial \vec{x}^{(0)}} \; \Delta \vec{x} \quad && \Big\} \text{(fortgepflanzter Eingangsfehler)} \\
+ \quad & \frac{\partial \vec{x}^{(k)}}{\partial \vec{x}^{(k-1)}} \frac{\partial \vec{x}^{(k-1)}}{\partial \vec{x}^{(k-2)}} \cdots \frac{\partial \vec{x}^{(2)}}{\partial \vec{x}^{(1)}} \; \tau^{(1)} \vec{x}^{(1)} \\
+ \quad & \ldots \\
+ \quad & \frac{\partial \vec{x}^{(k)}}{\partial \vec{x}^{(k-1)}} \frac{\partial \vec{x}^{(k-1)}}{\partial \vec{x}^{(k-2)}} \; \tau^{(k-2)} \vec{x}^{(k-2)} && \Bigg\} \text{(fortgepflanzte Darstellungsfehler)} \\
+ \quad & \frac{\partial \vec{x}^{(k)}}{\partial \vec{x}^{(k-1)}} \; \tau^{(k-1)} \vec{x}^{(k-1)} \\
+ \quad & \tau^{(k)} \vec{x}^{(k)} \, .
\end{aligned}
$$

mit $\tau^{(i)} = \mathrm{diag}(\tau_1^{(i)}, \tau_2^{(i)}, \ldots)$ *und* $|\tau_j^{(i)}| \le \tau = \text{Maschinengenauigkeit}.$

Interpretation: Der so definierte differentielle Fehler ist die lineare Näherung an den tatsächlich anfallenden Fehler (bei kleinen Fehlern eine sehr gute Approximation). Gehalt der Formel: Endfehler ist als Linearkombination der Eingabefehler $\Delta\vec{x}$ und aller Rundungsfehler $\tau_j^{(i)}$ dargestellt und kann über die Verstärkungsfaktoren als Vielfaches der Maschinengenauigkeit abgeschätzt werden (siehe Beispiel 2.19).

Definition 2.21:

Unter Vernachlässigung von Rundungsfehlern gilt

$$
\Delta y_i = \sum_j \frac{\partial y_i}{\partial x_j} \, \Delta x_j \; , \qquad \frac{\Delta y_i}{y_i} = \sum_j \frac{x_j}{y_i} \, \frac{\partial y_i}{\partial x_j} \, \frac{\Delta x_j}{x_j} \; .
$$

Die Größen $\frac{\partial y_i}{\partial x_j}$ *bzw.* $\frac{x_j}{y_i}\frac{\partial y_i}{\partial x_j}$ *heißen* **absolute** *bzw.* **relative Konditionszahlen** *der Abbildung* $\vec{x} \to \vec{y}(\vec{x})$. *Sie sind die Verstärkungsfaktoren der absoluten bzw. relativen* Eingangs*fehler.*

Bemerkung 2.22: *Zwei verschiedene Algorithmen*

$$
\vec{y} = \Big(\vec{f}^{(k)} \circ \cdots \circ \vec{f}^{(1)}\Big)(\vec{x}) = \Big(\vec{F}^{(K)} \circ \cdots \circ \vec{F}^{(1)}\Big)(\vec{x})
$$

für dasselbe Ergebnis $\vec{y}(\vec{x})$ *haben dieselben Konditionszahlen* $\frac{\partial y_i}{\partial x_j}$ *bezüglich des Eingabefehlers* $\Delta\vec{x}$. *Sie können sich trotzdem numerisch völlig unterschiedlich verhalten, da* $\Delta\vec{y}$ *auch durch die von den* **Teilalgorithmen** $\vec{f}^{(k)} \circ \cdots \circ \vec{f}^{(2)}$, $\vec{f}^{(k)} \circ \cdots \circ \vec{f}^{(3)}$ *usw. verstärkten Fehlern der Zwischenergebnisse bestimmt wird.*

Ein Algorithmus ist numerisch nur dann gutmütig, wenn **alle** *Teilalgorithmen gut konditioniert sind. Beispiel:*

$$y = \underbrace{\sqrt{x+1} - \sqrt{x-1}}_{schlecht} = \underbrace{\frac{2}{\sqrt{x+1} + \sqrt{x-1}}}_{gut} .$$

In der ersten Darstellung ist für $x \gg 1$ *der die Differenz der Wurzeln bildende Teilalgorithmus schlecht konditiniert.*

Kapitel 3

Nichtlineare Gleichungen (skalar)

Aufgabe: finde Lösung von $f(x) = 0$ mit $f : \mathbb{R} \to \mathbb{R}$ (eine Gleichung für eine Unbekannte). Allgemeine Idee: Umformulierung in ein äquivalentes Fixpunktproblem

$$f(x) = 0 \quad \Longleftrightarrow \quad \Phi(x) = x$$

und Anwendung des Banachschen Fixpunktsatzes.

3.1 Der Banachsche Fixpunktsatz

Definition 3.1:

Eine Abbildung $\Phi : A \subset \mathbb{R} \to \mathbb{R}$ *heißt* **Kontraktion**, *wenn eine* **Kontraktionskonstante** $k \in [0, 1)$ *existiert, so daß* $|\Phi(x) - \Phi(y)| \leq k\,|x - y|$ $\forall x, y \in A$.

Bildpunkte liegen dichter zusammen als die Urbilder ("Kontraktion").

Bemerkung 3.2: *Kontraktionen sind automatisch (Lipschitz-)stetig.*

Satz 3.3: (Banachscher Fixpunktsatz ("BFS"), skalare Version)

Sei $\Phi : A \to A$ *Kontraktion auf abgeschlossenem* $A \subset \mathbb{R}$ *mit einer Kontraktionskonstanten* $k < 1$. *Dann*

a) *existiert ein eindeutiger Fixpunkt* $x^* = \Phi(x^*) \in A$,

b) *konvergiert jede Folge* $x_{i+1} = \Phi(x_i)$ *mit beliebigem Startwert* $x_0 \in A$ *gegen* x^*, *c) gelten für jede solche Folge die Abschätzungen*

$$|x_i - x^*| \;\leq\; \underbrace{\frac{k}{1-k}\,|x_i - x_{i-1}|}_{\text{“a posteriori”}} \;\leq\; \underbrace{\frac{k^i}{1-k}\,|x_1 - x_0|}_{\text{“a priori”}} \quad .$$

Beweis: Zeige: (x_i) ist Cauchy-Folge.

$$\begin{aligned} |x_{i+n} - x_i| &= |x_{i+n} - x_{i+n-1} + x_{i+n-1} \mp \cdots - x_{i+1} + x_{i+1} - x_i| \\ &\leq |x_{i+n} - x_{i+n-1}| \;+\; |x_{i+n-1} - x_{i+n-2}| \;+\; \cdots \;+\; |x_{i+1} - x_i| \end{aligned}$$

für jedes $i, n \geq 0$. Aus $|x_j - x_{j-1}| = |\Phi(x_{j-1}) - \Phi(x_{j-2})| \leq k\,|x_{j-1} - x_{j-2}|$, d.h.,

$$|x_j - x_{j-1}| \;\leq\; k\,|x_{j-1} - x_{j-2}| \;\leq\; k^2\,|x_{j-2} - x_{j-3}| \;\leq\; \ldots$$

folgt

$$\begin{aligned} |x_{i+n} - x_i| &\leq k^{n-1}\,|x_{i+1} - x_i| \;+\; k^{n-2}\,|x_{i+1} - x_i| \;+\; \cdots \;+\; |x_{i+1} - x_i| \\ &= (1 + k + k^2 + \cdots + k^{n-1})\;|x_{i+1} - x_i| \;=\; \frac{1-k^n}{1-k}\;|x_{i+1} - x_i| \end{aligned}$$

$$\leq \frac{|x_{i+1} - x_i|}{1-k} \overset{(\#)}{\leq} \frac{k\,|x_i - x_{i-1}|}{1-k} \leq \frac{k^2\,|x_{i-1} - x_{i-2}|}{1-k} \leq \ldots \overset{(\#\#)}{\leq} \frac{k^i\,|x_1 - x_0|}{1-k} \,.$$

Mit $k^i \to 0$ folgt die Cauchy-Eigenschaft. Es existiert somit ein Grenzwert x^*. Mit $x_0 \in A$ und $\Phi : A \to A$ folgt $x_i \in A \Rightarrow x^*$ ist Häufungspunkt von $A \Rightarrow x^* \in A$ (abgeschlossen). Da Φ stetig:

$$x^* \;=\; \lim_{i\to\infty} x_{i+1} \;=\; \lim_{i\to\infty} \Phi(x_i) \;=\; \Phi(\lim_{i\to\infty} x_i) \;=\; \Phi(x^*) \;.$$

Eindeutigkeit: für weiteren Fixpunkt $x^{**} \neq x^*$ folgt der Widerspruch

$$|x^* - x^{**}| \;=\; |\Phi(x^*) - \Phi(x^{**})| \;\leq\; k\,|x^* - x^{**}| \;<\; |x^* - x^{**}| \;.$$

Die Abschätzungen c) ergeben sich aus (#) und (##):

$$\lim_{n\to\infty} |x_{i+n} - x_i| \;=\; |x^* - x_i| \;\overset{(\#)}{\leq}\; \frac{k}{1-k}\;|x_i - x_{i-1}| \;\overset{(\#\#)}{\leq}\; \frac{k^i}{1-k}\;|x_1 - x_0| \;.$$

Q.E.D.

Bemerkung 3.4: *Je kleiner die Kontraktionskonstante k, umso schneller wird der Fehler beim Übergang $x \to \Phi(x)$ kleiner:*

$$|\Phi(x) - x^*| = |\Phi(x) - \Phi(x^*)| \leq k\,|x - x^*| \ .$$

Also: $k \ll 1$ bedeutet schnelle Konvergenz, für $k \approx 1$ ist die Konvergenz langsam.

Bemerkung 3.5: *Es gilt folgende angenehme Interpretation der a posteriori Abschätzung: für $k \leq 1/2$ gilt $k/(1-k) \leq 1$, also*

$$|x_i - x^*| \leq |x_i - x_{i-1}| \ .$$

Im Klartext: die erreichte Genauigkeit nach i Schritten ist besser als die letzte beobachtete Änderung der Folgenpunkte. Also (für $k \leq 1/2$): die führenden Stellen, die sich in der Iteration nicht mehr ändern, sind bereits exakt! Dasselbe gilt im wesentlichen auch mit Rundungsfehlern, siehe Satz 3.12. In einem Programm kann man damit ein Abbruchkriterium der Form

if $|x_i - x_{i-1}| < \mathit{gewünschteAbsoluteGenauigkeit}$ **then** *Abbruch*

implementieren, falls $k \leq 1/2$ gilt.

Bemerkung 3.6: *Für stetig diff'bares Φ gilt der Mittelwertsatz $\Phi(x) - \Phi(y) = \Phi'(\xi)(x-y)$ mit Zwischenwert ξ zwischen x und y. Für Intervalle liegt mit x und y auch ξ in A, womit sich die beste (= kleinste) Kontraktionskonstante durch $k = \sup\limits_{\xi \in A} |\Phi'(\xi)|$ ergibt.*

Es gibt viele Möglichkeiten zur Umformulierung $f(x) = 0 \Longleftrightarrow \Phi(x) = x$ (z.B. $\Phi(x) = x + \alpha(x)\,f(x)$ mit beliebigem $\alpha(x)$). Man muß Φ so wählen, daß der BFS 3.3 anwendbar ist.

Satz 3.7:

Es sei $\Phi : \mathbb{R} \mapsto \mathbb{R}$ stetig diff'bar mit Fixpunkt $x^* = \Phi(x^*)$. Es existiert eine abgeschlossene Umgebung A von x^* mit

- $\Phi : A \mapsto A$
- *Φ ist Kontraktion auf A*

genau dann, wenn $|\Phi'(x^*)| < 1$ gilt.

Beweis: 1) Es gelte $|\Phi'(x^*)| < 1$. Dann existiert $U_\epsilon(x^*) = \{\xi; |\xi - x^*| < \epsilon\}$ mit $|\Phi'(\xi)| < 1 \ \forall \xi \in U_\epsilon$. Für $x, y \in A := \{\xi; |\xi - x^*| \le \epsilon/2\} \subset U_\epsilon$ folgt

$$|\Phi(x) - \Phi(y)| \overset{(MWS)}{=} |\Phi'(\xi)||x - y| \le k\,|x - y|$$

mit $k := \max\limits_{\xi \in A} |\Phi'(\xi)| < 1$. Weiterhin gilt $\forall x \in A$:

$$|\Phi(x) - x^*| = |\Phi(x) - \Phi(x^*)| \le k\,|x - x^*| < |x - x^*| \le \frac{\epsilon}{2}\ ,$$

also $\Phi(A) \subset A$.

2) Sei $|\Phi'(x^*)| \ge 1$. Setze $\tilde{k}(x, y) := \left|\dfrac{\Phi(x) - \Phi(y)}{x - y}\right| \overset{(MWS)}{=} |\Phi'(\xi(x, y))|$.
Für $x, y \to x^*$ und damit $\xi \to x^*$ folgt $\tilde{k}(x, y) \to |\Phi'(x^*)| \ge 1$, so daß nicht $|(\Phi(x) - \Phi(y))/(x - y)| \le k$ für ein festes $k < 1$ gelten kann.

Q.E.D.

Bemerkung 3.8: *Intuitiv ist dies klar. Für $x \approx x^*$, also*

$$\Phi(x) - x^* = \Phi(x) - \Phi(x^*) = \Phi'(\xi)(x - x^*) \approx \Phi'(x^*)(x - x^*)$$

folgt

$$\begin{array}{lcll}
|\Phi'(x^*)| > 1 & \Rightarrow & |\Phi(x) - x^*| > |x - x^*| & (\text{“abstoßender” Fixpunkt}) \\
|\Phi'(x^*)| < 1 & \Rightarrow & |\Phi(x) - x^*| < |x - x^*| & (\text{“anziehender” Fixpunkt}) \\
|\Phi'(x^*)| = 1 & : & \text{alles kann passieren} & (\text{“hyperbolischer” Fixpunkt}).
\end{array}$$

Damit gilt die **allgemeine Strategie**: zur Lösung von $f(x) = 0$ finde äquivalentes Fixpunktproblem $x = \Phi(x)$, so daß an der Lösung $|\Phi'(x^*)| < 1$ gilt ($\Rightarrow$ BFS 3.3 ist lokal anwendbar).

Beispiel 3.9: Problem $f(x) = x + x^3 - 1 = 0$. Da f monton und $f(\pm\infty) = \pm\infty \Rightarrow$ es existiert eindeutige Lösung.
Einsetzen: z.B. $f(0.6) = -0.184$, $f(0.7) = 0.043 \Rightarrow x^* \in (0.6, 0.7)$.

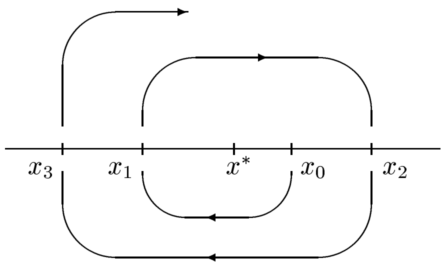
Versuch 1: $x + x^3 - 1 = 0 \Leftrightarrow x = 1 - x^3 =: \Phi(x)$. Aber:

$$|\Phi'(x)| = 3x^2 \in (3 \times 0.6^2, 3 \times 0.7^2) = (1.08, 1.47)\ \forall x \in (0.6, 0.7)$$

$$\Rightarrow \quad |\Phi'(x^*)| > 1.08 \quad (\text{abstoßender Fixpunkt}).$$

In der Tat:

$$\begin{aligned} x_0 &= 0.700... \\ x_1 &= 0.657... \\ x_2 &= 0.716... \\ x_3 &= 0.632... \\ &\vdots \end{aligned}$$



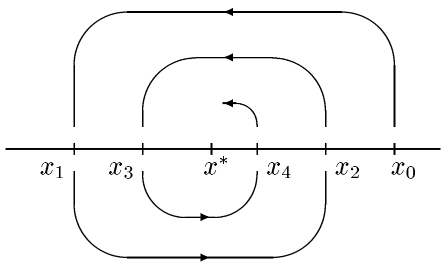
Versuch 2: $x + x^3 - 1 = 0 \;\Leftrightarrow\; x = (1 + x - x^3)/2 =: \Phi(x)$. Mit $\Phi'(x) = (1 - 3\,x^2)/2$ (monoton fallend für $x > 0$) folgt

$$\Phi'(x) \in (\Phi'(0.7), \Phi'(0.6)) = (-0.235, -0.04)\ \forall x \in (0.6, 0.7)$$

$$\Rightarrow \quad |\Phi'(x^*)| < 0.235 \quad \text{(anziehender Fixpunkt).}$$

In der Tat:

$$\begin{aligned} x_0 &= 0.70000... \\ x_1 &= 0.67850... \\ x_2 &= 0.68307... \\ x_3 &= 0.68217... \\ x_4 &= 0.68235... \\ &\vdots \end{aligned}$$



Φ ist eine Kontraktion auf $A := [0.6, 0.7]$, denn

- $\Phi([0.6, 0.7]) = [\Phi(0.7), \Phi(0.6)] = [0.6785..., 0.692...] \subset A$ (Φ ist auf A monoton fallend, da $\Phi'(x) \in [-0.235, -0.04]$)
- Kontraktionskonstante $k = \max\limits_{\xi \in A} |\Phi'(\xi)| = 0.235 < 1$.

Wie gut approximiert x_4 die Lösung?

$$\begin{array}{llcll} \text{a priori:} & |x_4 - x^*| & \leq & \dfrac{k^4}{1-k}\,|x_1 - x_0| & = & 8.57... \times 10^{-5} \\ \text{a posteriori:} & |x_4 - x^*| & \leq & \dfrac{k}{1-k}\,|x_4 - x_3| & = & 5.45... \times 10^{-5}\ . \end{array}$$

Bemerkung 3.10: *Alternative Fehlerkontrolle: für ein Nullstellenproblem $f(x) = 0$ mit stetig diff'barem f gilt <u>stets</u> (unabhängig vom verwendeten Verfahren $\Phi : A \mapsto A$):*

$$|x_i - x^*| \;\leq\; \frac{|f(x_i)|}{\min\limits_{\xi \in A} |f'(\xi)|}\ .$$

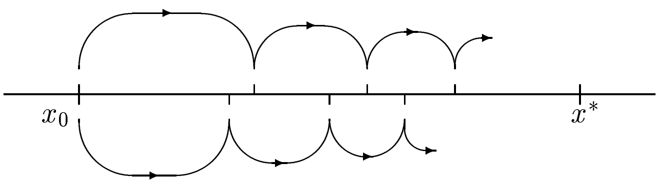
Beweis: Taylorentwicklung mit Zwischenwert $\xi \in A$:

$$f(x_i) = f(x^*) \;+\; f'(\xi)\,(x_i - x^*) \quad \Rightarrow \quad x_i - x^* = \frac{f(x_i)}{f'(\xi)}\;. \qquad \text{Q.E.D.}$$

Dies liefert bei einfachen Nullstellen und hinreichend kleinem Intervall A sehr realistische Fehlerabschätzungen.

6.11.97↓ **Bemerkung 3.11:** *Zum Einfluß von Rechenfehlern: erfreulich!*

exakte x_i



fehlerhaft berechnete $\tilde{x}_i$

Banach-Iterationen sind "selbstkorrigierend": *fehlerhaft berechnete Punkte werden beim Weiterrechnen wieder vom Fixpunkt angezogen. Fehler können sich hierdurch nicht akkumulieren!! Lediglich die a priori Abschätzung wird bei langen Iterationen verfälscht.*

Genauer:

Satz 3.12: (Rundungsfehler bei Banach-Iterationen)

Es sei $x^* = \Phi(x^*)$ der exakte Fixpunkt einer Kontraktion Φ mit der Kontraktionskonstanten $k < 1$. Es sei $\tilde{\Phi}(x)$ die mit einem Auswertungsfehler $\Delta\Phi(x) := \tilde{\Phi}(x) - \Phi(x)$ behaftete numerische Iterationsfunktion. Die Iteration $\tilde{x}_i = \tilde{\Phi}(\tilde{x}_{i-1})$ wird abgebrochen, sobald $|\tilde{x}_i - \tilde{x}_{i-1}| \leq \epsilon$ (= vorgegebene absolute Genauigkeit) gilt. Akzeptiert man das Folgenglied $\tilde{x}_i$ als numerischen Fixpunkt, so gilt

$$|\tilde{x}_i - x^*| \leq \underbrace{\frac{k\,\epsilon}{1-k}}_{\text{Verfahrensfehler}} + \underbrace{\frac{|\Delta\Phi(\tilde{x}_{i-1})|}{1-k}}_{\text{Rundungsfehler}}\;.$$

Beweis: Aufgabe 8.

Interpretation: bis auf den Faktor $1/(1-k)$ ist die prinzipiell erreichbare Genauigkeit ($\epsilon = 0$) durch die Auswertungsgenauigkeit $\Delta\Phi$ <u>eines</u> (des letzten) Iterationsschritts gegeben (**unabhängig von der Zahl der tatsächlich ausgeführten Iterationsschritte!**). In der Praxis muß allerdings $\epsilon > |\Delta\Phi|$ gewählt werden, um Abbruch garantieren zu können. Für $k \ll 1$ gilt dann

$$|\tilde{x}^* - x^*| \leq \frac{k\,\epsilon + |\Delta\Phi(\tilde{x}_{i-1})|}{1-k} \approx k\,\epsilon + |\Delta\Phi(\tilde{x}_{i-1})| < (k+1)\,\epsilon \approx \epsilon\;.$$

Zusammenfassung: finde Nullstelle x^* von f, d.h., löse $f(x) = 0$.

Algorithmus 3.13: ("Banach-Iteration")

a) *Wähle stetig diff'bares $\Phi : \mathbb{R} \mapsto \mathbb{R}$, so daß*
   - $\Phi(x^*) = x^* \;\Rightarrow\; f(x^*) = 0$,
   - $|\Phi'(x^*)| < 1$.

b) *Identifiziere abgeschlossenes Intervall A um x^*, so daß*

$$\left.\begin{array}{l} \bullet\quad \Phi(A) \subset A \\ \bullet\quad k := \max\limits_{\xi \in A} |\Phi'(\xi)| \; < 1 \end{array}\right\} \quad \textit{A existiert nach Satz 3.7.}$$

c) *Iteriere $x_{i+1} = \Phi(x_i)$ mit beliebigem $x_0 \in A$. Es gilt*

$$|x_i - x^*| \;\leq\; \frac{k}{k-1}\, |x_i - x_{i-1}| \;\leq\; \frac{k^i}{1-k}\, |x_1 - x_0| \; .$$

Alternativ: Fehlerkontrolle nach Bemerkung 3.10.

Verbleibendes Problem: *systematische* Konstruktion eines "guten" (schnell konvergierenden) Φ aus gegebenem f. Schnelle Konvergenz ergibt sich bei kleinen Kontraktionskonstanten $k \ll 1$. Im Sinne des folgenden Kapitels kann man Iterationen betrachten, deren Kontraktionskonstanten "effektiv" 0 sind.

3.2 Superlineare Konvergenz

Gegeben $A_0 = [a_0, b_0]$, $\Phi : A_0 \mapsto A_0$ stetig diff'bar und $k_0 := \max\limits_{\xi \in A_0} |\Phi'(\xi)| < 1$ (d.h., alle Voraussetzungen des BFS 3.3 gelten). Betrachte Intervallfolge

$$\cdots \subset A_i := \Phi(A_{i-1}) \subset A_{i-1} \subset \cdots \subset A_0 \;\; :$$

$$\begin{array}{ccc} & & \text{optimale} \\ & & \text{Kontraktionskonstante:} \\ a_0 \;\underline{\qquad\qquad\qquad}\; b_0 & & k_0 := \max\limits_{\xi \in A_0} |\Phi'(\xi)| \\ \downarrow\; \Phi & & \\ a_1 \;\underline{\qquad\qquad}\; b_1 & & k_1 := \max\limits_{\xi \in A_1} |\Phi'(\xi)| \\ \downarrow\; \Phi & & \\ a_2 \;\underline{\qquad}\; b_2 & & k_2 := \max\limits_{\xi \in A_2} |\Phi'(\xi)| \\ \downarrow\; \Phi & & \\ \cdots & & \cdots \end{array}$$

Es gilt $A_i = [a_i, b_i] = [\Phi(\alpha_{i-1}), \Phi(\beta_{i-1})]$ mit gewissen $\alpha_{i-1}, \beta_{i-1} \in A_{i-1}$, so daß

$$b_i - a_i = \Phi(\beta_{i-1}) - \Phi(\alpha_{i-1}) = \Phi'(\xi)(\beta_{i-1} - \alpha_{i-1}) \le k_{i-1}\,(b_{i-1} - \alpha_{i-1}) \ .$$

Die Intervalle schrumpfen zum Fixpunkt zusammen, es gilt

- $\Phi : A_i \mapsto A_i$ (da $\Phi(A_i) \subset \Phi(A_{i-1}) = A_i$),
- Φ ist Kontraktion auf A_i mit **"lokaler" Kontraktionskonstante** k_i:

$$|\Phi'(x^*)| \le k_i := \max_{\xi \in A_i} |\Phi'(\xi)| \le k_{i-1} \le k_{i-2} \le \ldots \le k_0 < 1.$$

Damit kann die Iteration für Indizes $\ge i$ als Banach-Iteration über dem kleineren Interval A_i angesehen werden, sodaß effektiv die kleinere Kontraktionskonstante $k_i \le k_0$ gültig ist.
Ergebnis: mit $x_0 \in A_0$, $x_i = \Phi(x_{i-1}) \in A_i$ gilt

$$|x_{i+1} - x^*| = |\Phi(x_i) - \Phi(x^*)| \le k_i\,|x_i - x^*|$$

mit den lokalen Kontraktionskonstanten $1 > k_0 \ge k_1 \ge \ldots \ge k_i \overset{i\to\infty}{\longrightarrow} |\Phi'(x^*)|$.
Frage: was passiert bei $\Phi'(x^*) = 0$? Es ist extrem schnelle Konvergenz zu erwarten.

Satz 3.14: (Kriterium für höhere Konvergenz)
A sei abgeschlossenes Intervall, $\Phi : A \mapsto A$ sei q-fach stetig diff'bar mit Fixpunkt $x^* = \Phi(x^*) \in A$. Es gelte

$$\Phi'(x^*) = \Phi''(x^*) = \ldots = \Phi^{(q-1)}(x^*) = 0 \ .$$

Dann gilt für die Iteration $x_{i+1} = \Phi(x_i)$ mit $x_0 \in A$ die Fehlerfortpflanzung

$$|x_{i+1} - x^*| \ \le \ \frac{\max\limits_{\xi \in A} |\Phi^{(q)}(\xi)|}{q!} \ |x_i - x^*|^q \ .$$

Beweis: Taylorentwicklung um x^*:

$$\Phi(x) = \underbrace{\Phi(x^*)}_{x^*} \ + \ \underbrace{\sum_{k=1}^{q-1} \frac{\Phi^{(k)}(x^*)}{k!}\,(x - x^*)^k}_{0} \ + \ \frac{\Phi^{(q)}(\xi)}{q!}\,(x - x^*)^q$$

mit Zwischenpunkt $\xi \in A$, also $|\Phi(x) - x^*| = \dfrac{|\Phi^{(q)}(\xi)|}{q!}\,|x - x^*|^q$.

Q.E.D.

Definition 3.15: (Konvergenzordnung)

Eine gegen x^* konvergierende Folge (x_i) heißt **konvergent von der Ordnung** q, *wenn eine Konstante* const *existiert, so daß*

$$|x_i - x^*| \le \mathrm{const}\, |x_{i-1} - x^*|^q \quad \forall i \ge \text{ geeignetes } i_0 \ .$$

Superlineare Konvergenz *heißt* $\displaystyle \lim_{i\to\infty} \frac{|x_i - x^*|}{|x_{i-1} - x^*|} = 0$ *(z.B. bei $q > 1$).*

Bemerkung 3.16: *Superlineare Konvergenz ist ein lokales Phänomen, das nur in einer (kleinen) Umgebung der Lösung relevant ist. Die Abschätzung*

$$\textit{neuer Fehler} \le \textit{const (Potenz des letzten Fehlers)}$$

sagt nur dann etwas über die schnelle Verkleinerung von Fehlern aus, wenn der "letzte Fehler" bereits klein ist. Damit hilft superlineare Konvergenz nur in der Endphase einer Banach-Iteration.

Bemerkung 3.17: *Bei superlinearer Konvergenz ist die a priori Abschätzung des BFS sehr unrealistisch (viel zu grob), da sie auf dem Konzept einer festen Kontraktionskonstanten beruht. Die a posteriori Abschätzung ist wesentlich besser (aber auch noch recht grob). Die Kontrolle gemäß Bemerkung 3.10 liefert i.a. realistische Werte.*

3.3 Das Newton-Verfahren

Ansatz: $\Phi(x) = x + \alpha(x)\, f(x)$, womit $\Phi(x^*) = x^* \Leftrightarrow f(x^*) = 0$. Ziel: superlineare Konvergenz, also

$$\Phi'(x^*) = 1 + \alpha'(x^*)\, f(x^*) + \alpha(x^*)\, f'(x^*) \overset{(!)}{=} 0 \quad \Rightarrow \quad \alpha(x^*) = -\frac{1}{f'(x^*)} \ .$$

Wähle $\alpha(x) = -1/f'(x)$: das **Newton-Verfahren** für $f(x) = 0$ ist die Banach-Iteration mit

$$\Phi(x) = x - \frac{f(x)}{f'(x)} \ .$$

Satz 3.18: (Das Newton-Verfahren)

Es sei x^* einfache Nullstelle der 3-fach stetig diff'baren Funktion f. Dann existiert eine Umgebung $U(x^*)$, so daß die **Newton-Iteration**

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

für alle $x_0 \in U(x^*)$ quadratisch gegen x^* konvergiert.

Beweis: Wegen $f'(x^*) \neq 0$ folgt $f'(x) \neq 0$ auf einer Umgebung von x^*, womit $\Phi(x) = x - f(x)/f'(x)$ dort definiert und 2-fach stetig diff'bar ist. Mit

$$\Phi' = \frac{f f''}{f'^2}\ , \quad \Phi'' = \frac{2ff''^2 - ff'f''' - f'^2 f''}{f'^3}$$

folgt $\Phi'(x^*) = 0$, $\Phi''(x^*) = -f''(x^*)/f'(x^*)$. Nach Satz 3.7 ist lokale Konvergenz garantiert, die nach Satz 3.14 quadratisch ist.

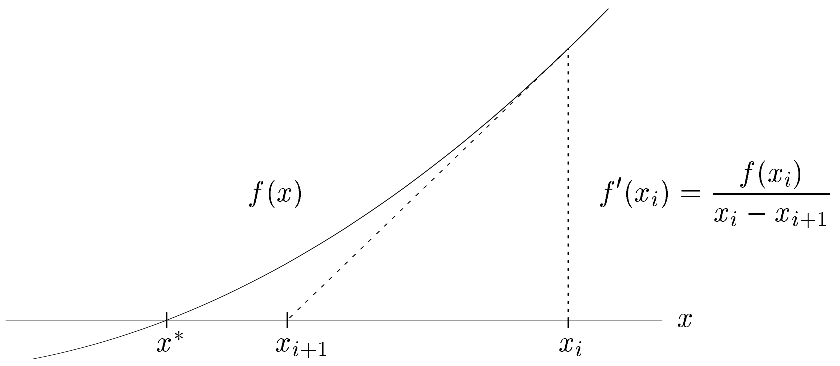
Q.E.D.

11.11.97↓ **Bemerkung 3.19:** *"f einfach stetig diff'bar" reicht für superlineare Konvergenz. Mit $f(x) = f'(\xi)(x - x^*)$ (Zwischenwert ξ) folgt*

$$|\Phi(x) - x^*| = \left| x - \frac{f(x)}{f'(x)} - x^* \right| = \left| 1 - \frac{f'(\xi)}{f'(x)} \right| \ |x - x^*|$$

Mit $x \to x^*$ und somit $\xi \to x^*$ folgt $|\Phi(x) - x^*| \leq k(x)\,|x - x^*|$, wobei $k(x) = |1 - f'(\xi)/f'(x)| \stackrel{x \to x^*}{\longrightarrow} 0$.

Bemerkung 3.20: *Geometrisch: x_{i+1} ist die Nullstelle der Tangente am Punkt x_i:*



Bemerkung 3.21: *Prinzipielle (aber sehr grobe) Fehlerkontrolle: identifiziere Umgebung A von x^* mit $\Phi : A \mapsto A$ und*

$$k := \sup_{\xi \in A} |\Phi'(\xi)| = \sup_{\xi \in A} \left| \frac{f(\xi) f''(\xi)}{(f'(\xi))^2} \right| < 1$$

und benutze die Fehlerabschätzungen des BFS 3.3.

Besser: a posteriori Kontrolle nach Bemerkung 3.10 oder:

Satz 3.22: (realistische Fehlerabschätzungen für das Newton-Verfahren)
Es sei f 2-fach stetig diff'bar und A ein abgeschlossenes Intervall um eine einfache Nullstelle x^*, es gelte $\Phi(A) \subset A$ (mit $\Phi(x) = x - f(x)/f'(x)$) und $f'(x) \neq 0 \ \forall x \in A$. Mit

$$c \;=\; \frac{\max\limits_{\xi\in A} |f'(\xi)|}{\min\limits_{\xi\in A} |f'(\xi)|} \;, \qquad d \;=\; \frac{1}{2}\, \frac{\max\limits_{\xi\in A} |f''(\xi)|}{\min\limits_{\xi\in A} |f'(\xi)|}$$

gelten für das Newton-Verfahren $x_i = \Phi(x_{i-1})$ für jedes $x_0 \in A$ die Abschätzungen

$$\begin{aligned}
|x_i - x^*| &\leq \frac{1}{d} \; (\, d\,|x_0 - x^*|\,)^{(2^i)} \\
|x_i - x^*| &\leq \frac{1}{d} \; \left(\, \sqrt{c}\, d\, |x_1 - x_0|\,\right)^{(2^i)} && \textit{(a priori)} \\
|x_i - x^*| &\leq c\,d\, |x_i - x_{i-1}|^2 && \textit{(a posteriori)}\;.
\end{aligned}$$

Beweis: Taylor-Entwicklung um x_{i-1}:

$$f(x^*) \;=\; 0 \;=\; f(x_{i-1}) \;+\; f'(x_{i-1})(x^* - x_{i-1}) \;+\; \frac{1}{2}\, f''(\xi_{i-1})\; (x^* - x_{i-1})^2 \;.$$

Nach Division durch $f'(x_{i-1})$:

$$x_i - x^* \;\equiv\; x_{i-1} \;-\; \frac{f(x_{i-1})}{f'(x_{i-1})} \;-\; x^* \;=\; \frac{1}{2}\, \frac{f''(\xi_{i-1})}{f'(x_{i-1})}\; (x_{i-1} - x^*)^2 \tag{3.1}$$

mit Zwischenpunkt ξ_{i-1} zwischen x_{i-1} und x^*. Es folgt

$$\begin{aligned}
c_i \;&:=\; \frac{x_i - x^*}{(x_i - x_{i-1})^2} \;=\; \frac{x_i - x^*}{(\, f(x_{i-1})/f'(x_{i-1})\,)^2} \\
&=\; \frac{f''(\xi_{i-1})\, f'(x_{i-1})\, (x_{i-1} - x^*)^2}{2\, f(x_{i-1})^2} \;.
\end{aligned}$$

Taylor-Entwicklung mit anderem Zwischenpunkt η_{i-1} liefert $f(x_{i-1}) = f'(\eta_{i-1})\,(x_{i-1} - x^*)$, sodaß

$$|c_i| \;=\; \left| \frac{f''(\xi_{i-1})\, f'(x_{i-1})}{2 f'(\eta_{i-1})^2} \right| \;\leq\; \frac{\max\limits_{\xi\in A} |f''(\xi)| \;\; \max\limits_{x\in A} |f'(x)|}{2\; (\min\limits_{\eta\in A} |f'(\eta)|\,)^2} \;\leq\; cd\;.$$

Es folgt die a posteriori Abschätzung

$$|x_i - x^*| \;=\; |c_i|\; (x_i - x_{i-1})^2 \;\leq\; c\,d\; (x_i - x_{i-1})^2 \;.$$

Aus (3.1) ergibt sich $|x_i - x^*| \le d\,|x_{i-1} - x^*|^2$, wodurch rekursiv die erste der angegebenen Abschätzungen folgt:

$$\begin{aligned} |x_i - x^*| &\le d\,|x_{i-1} - x^*|^2 \le d\,d^2\,|x_{i-2} - x^*|^4 \le \ldots \\ &\le d\,d^2 \cdots d^{(2^{i-2})}\,|x_1 - x^*|^{(2^{i-1})} \stackrel{(\#)}{=} d^{(2^{i-1}-1)}\,|x_1 - x^*|^{(2^{i-1})} \\ &\le d\,d^2 \cdots d^{(2^{i-1})}\,|x_0 - x^*|^{(2^i)} = d^{(2^i-1)}\,|x_0 - x^*|^{(2^i)} . \end{aligned}$$

Mit (#) und $|x_1 - x^*| \le cd\,|x_1 - x_0|^2$ (a posteriori Abschätzung, $i = 1$) erhält man die a priori Abschätzung:

$$|x_i - x^*| \le d^{(2^{i-1}-1)}\,c^{(2^{i-1})}\,d^{(2^{i-1})}\,|x_1 - x^*|^{(2^i)} = \frac{1}{d}\,\left(\sqrt{c}\,d\,|x_1 - x_0|\right)^{(2^i)} .$$

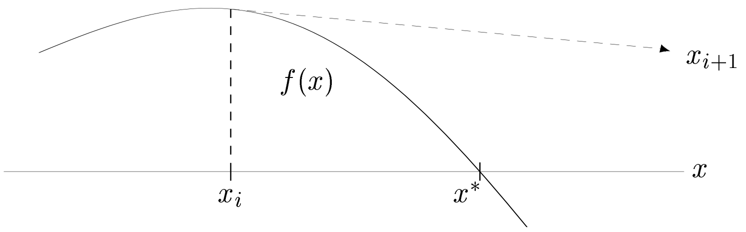
Q.E.D.

Bemerkung 3.23: *Was passiert bei q-fachen Nullstellen, d.h.,*

$$f(x^*) = f'(x^*) = \ldots = f^{(q-1)}(x^*) = 0 \;,\; f^{(q)}(x^*) \ne 0 \;?$$

Antwort (Aufgabe 12): das Newton-Verfahren konvergiert immer noch, aber nur linear mit asymptotischer Kontraktionskonstante $|\Phi'(x^*)| = 1 - 1/q$. Ist q bekannt, so kann mit der Modifikation $x_{i+1} = \Phi(x_i)$ mit $\Phi(x) = x - q\,f(x)/f'(x)$ wieder quadratische Konvergenz erreicht werden.

Bemerkung 3.24: *Das Newton-Verfahren konvergiert lokal immer (d.h., für hinreichend gute Startwerte). Global gibt es große Probleme, wenn die Funktion f lokale Extrema oder Sattelpunkte hat (Φ wird singulär):*



Bemerkung 3.25: *In der* **Steffensen-Iteration** $x_{i+1} = \Phi(x_i)$ *mit*

$$\Phi(x) = x - \frac{(f(x))^2}{f(x + f(x)) - f(x)} \qquad \left(= x - \frac{f(x)}{\frac{f(x+f(x)) - f(x)}{f(x)}} \right)$$

wird die Ableitung $f'(x)$ im Nenner des Newton-Verfahrens durch

$$\frac{f(x + f(x)) - f(x)}{f(x)} \qquad (\approx f'(x) \text{ für } |f(x)| \ll 1)$$

ersetzt. In unmittelbarer Nähe einer Nullstelle verhält sich das Steffensen-Verfahren damit genauso wie das Newton-Verfahren. Z.B., quadratische Konvergenz:

$$\lim_{x \to x^*} \Phi'(x) \ = \ 0 \ , \quad \lim_{x \to x^*} \Phi''(x) \ = \ f''(x^*) \left(1 \ + \ \frac{1}{f'(x^*)} \right) \ .$$

Zusammenfassung: Das Newton-Verfahren für $f(x) = 0$ ist die Banach-Iteration mit $\Phi(x) = x - f(x)/f'(x)$.

- $+$ konvergiert lokal immer (selbst bei Entartung),
- $+$ quadratische Konvergenz (außer bei Entartung) mit nur 2 Funktionsauswertungen pro Schritt,
- $+$ sehr realistische a priori Abschätzung möglich,
- $\mp$ f' wird benötigt (durch Steffensen-Variante vermeidbar),
- $-$ globale Konvergenzprobleme.

3.4 Ein Newton–Verfahren für entartete Nullstellen

Nach Bemerkung 3.23 kann man bei entarteten Nullstellen die quadratische Konvergenz des Newton–Verfahrens retten, wenn die Vielfachheit bekannt ist. Bemerkenswerterweise kann man diesen Entartungsgrad mit einer weiteren Funktionsauswertung (von f'') automatisch ermitteln. Die Funktion

$$q(x) = \frac{(f'(x))^2}{(f'(x))^2 - f(x)\, f''(x)}$$

nimmt nämlich an einer p-fachen Nullstelle x^* von f den Wert

$$q(x^*) = \lim_{x \to x^*} q(x) = p$$

an. Um dies zu sehen, setze $f(x) = g(x)\,(x - x^*)^p$ mit einer Funktion g die $g(x^*) \neq 0$ erfüllt. Mit etwas Rechnung folgt:

$$q(x) \ - \ p \ =$$

$$(x - x^*) \quad \left(\frac{(x - x^*) \Big((1-p)\,(g'(x))^2 \ + \ p\, g(x)\, g''(x) \Big) \ + \ 2\, p\, g(x)\, g'(x)}{(x - x^*)^2 \Big((g'(x))^2 - g(x)\, g''(x) \Big) \ + \ p\, g(x)^2} \right) ,$$

also $\lim_{x \to x^*} (q(x) - p) = 0$. Ersetzt man nach Bemerkung 3.23 im quadratischen Newton–Verfahren $\Phi(x) = x - q\, f(x)/f'(x)$ für q-fache Nullstellen den konstanten Entartungsgrad q durch die obige Funktion $q(x)$, so erhält man das Verfahren

$$\Phi(x) \;=\; x \;-\; q(x)\,\frac{f(x)}{f'(x)} \;=\; x \;-\; \frac{f(x)\,f'(x)}{(f'(x))^2 - f(x)\,f''(x)}\;,$$

welches das Newton-Verfahren automatisch an den Entartungsgrad der gerade angefahrenen Nullstelle anpasst. Zwar ist dieses Verfahren wegen der zusätzlichen Auswertung von f'' um etwa 50 % teurer als das normale Newton-Verfahren, dafür wird aber jede Nullstelle –unabhängig von ihrem Entartungsgrad– mit quadratischer Konvergenz erreicht.
Ein weiterer Vorteil ergibt sich bei Polynomen f. Zunächst bemerkt man, daß die n-fache Nullstelle des Polynoms $f(x) = c\,(x - x^*)^n$ problemlos von jedem Startpunkt x aus mit nur einem einzigen Schritt erreicht wird:

$$\Phi(x) \;=\; x \;-\; \frac{\Big(c\,(x-x^*)^n\Big)\;\Big(c\,n\,(x-x^*)^{n-1}\Big)}{\Big(c\,n\,(x-x^*)^{n-1}\Big)^2 \;-\; \Big(c\,(x-x^*)^n\Big)\;\Big(c\,n\,(n-1)\,(x-x^*)^{n-2}\Big)}$$

$$=\; x \;-\; \frac{c^2\;n\;(x-x^*)^{2n-1}}{c^2\;(n^2 \;-\; n\,(n-1))\;(x-x^*)^{2n-2}} \;=\; x \;-\; (x - x^*) \;=\; x^*\;.$$

Dieses Phänomen ist ein entscheidender Vorteil auch bei allgemeinen Polynomen. Das normale Newton-Verfahren hat Singularitäten an den Extremstellen/Sattelpunkten $f'(x) = 0$, $f(x) \neq 0$, dieses Verfahren an den Punkten mit $(f'(x))^2 = f(x)\,f''(x)$, $f(x) \neq 0$. Gerät man in die Nähe einer solchen Singularität, so erhält man jeweils Iterationspunkte "weit draußen". Das normale Newton-Verfahren erholt sich nur sehr langsam von einem solchen Ausreißer (siehe auch Bemerkung 3.41). Da jedes Polynom für $|x| \gg 1$ näherungsweise wie ein Monom $f(x) \approx c\,x^n$ aussieht, bringt das obige Verfahren jeden Ausreißer in einem einzigen Schritt in die Nähe des Nullpunktes (die Nullstelle des Monoms $c\,x^n$) zurück.

Bemerkung 3.26: *Dieses Verfahren ist nichts anderes als das normale Newton–Verfahren, allerdings nicht für $f(x)$, sondern für $\tilde{f}(x) = f(x)/f'(x)$! Ist f in einer Umgebung einer Nullstelle x^* analytisch, so ist x^* stets eine <u>einfache</u> Nullstelle für $\tilde{f}$ (leicht nachzurechnen). Dies erklärt sofort alle oben aufgeführten angenehmen Eigenschaften des Verfahrens. Dies sind aber nur Aussagen über die Konvergenzgeschwindigkeit. Es verbleibt das prinzipielle Problem numerischer Rundungsfehler bei Entartung, das in Abschnit 3.8 genauer diskutiert wird. Dies kann durch diese Modifikation des Newton–Verfahrens natürlich nicht behoben werden. Im Gegenteil, die erreichbare Genauigkeit ist sogar geringfügig schlechter als beim üblichen Newton–Verfahren für f, da sich durch die zusätzliche Auswertung von $f''(x)$ die Auswertungs(un)genauigkeit $\Delta\Phi$ etwas verschlechtert.*

3.5 Höhere Verfahren

Betrachte

$$\begin{aligned}
\Phi_2(x) &= x - \frac{f(x)}{f'(x)} && \text{(Newton-Verfahren)}\\
\Phi_3(x) &= \Phi_2(x) - \frac{(f(x))^2 f''(x)}{2\,(f'(x))^3}\\
\Phi_4(x) &= \Phi_3(x) + (f(x))^3\left(\frac{f'''(x)}{6\,(f'(x))^4} - \frac{(f''(x))^2}{2\,(f'(x))^5}\right)\\
&\ \ \vdots \qquad\qquad \vdots
\end{aligned}$$

Die Iteration mit Φ_2 konvergiert quadratisch, mit Φ_3 kubisch, mit Φ_4 in 4.ter Ordnung gegen einfache Nullstellen von f (z.B. $\Phi_3'(x^*) = \Phi_3''(x^*) = 0$). Man kann leicht Verfahren beliebig hoher Ordnung angeben, *die für die Praxis jedoch nicht sinnvoll sind.*

Begründung (Beispiel): Φ_4 braucht 4 Funktionsauswertungen f, f', f'', f''' pro Schritt. Vergleiche mit Newton-Doppelschritt $\boxed{\downarrow 13.11.97}$

$$\tilde{\Phi}_4(x) = \Phi_2(\Phi_2(x)) = x - \frac{f(x)}{f'(x)} - \frac{f(x - \frac{f(x)}{f'(x)})}{f'(x - \frac{f(x)}{f'(x)})}\ ,$$

der auch nur 4 Auswertungen braucht und ebenfalls ein Verfahren 4.ter Ordnung darstellt:

$$|\Phi_2(\Phi_2(x)) - x^*| \le \text{const}\ |\Phi_2(x) - x^*|^2 \le \text{const}^3\ |x - x^*|^4\ .$$

Vergleich:

- $\tilde{\Phi}_4$ braucht nur f, f', Φ_4 braucht f, f', f'', f''',
- für $\tilde{\Phi}_4$ braucht nur Φ_2 implementiert werden,
- die Auswertungsgenauigkeit von $\tilde{\Phi}_4 = \Phi_2 \circ \Phi_2$ ist praktisch identisch mit der von Φ_2:

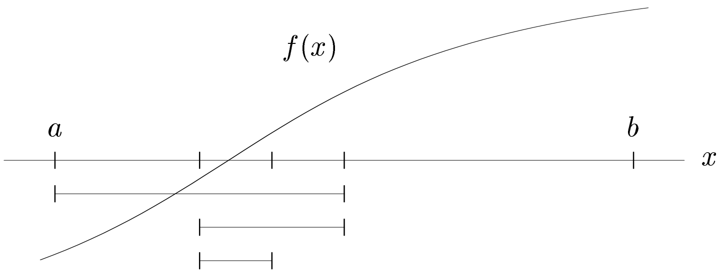
  $$\Delta\Phi_2(\Phi_2(x)) = \underbrace{\Phi_2'(\Phi_2(x))}_{\ll 1 \text{ für } x \approx \text{ Lösung}}\ \Delta\Phi_2 + \Delta\Phi_2$$

  mit $\Delta\Phi_2 =$ Auswertungsfehler eines Newton-Schrittes.

Damit ist $\tilde{\Phi}_4$ dem Φ_4 weit überlegen!

3.6 Bisektion (Intervallhalbierung)

Die Nullstellenfunktion f sei stetig, es seien $a < b$ bekannt mit $f(a)f(b) < 0$ (d.h., f hat Vorzeichenwechsel auf $[a, b]$). Betrachte Intervallmitte $c = (a+b)/2$, ermittle das Vorzeichen von $f(c)$. Es existiert entweder auf $[a, c]$ oder auf $[c, b]$ ein Vorzeichenwechsel. Fahre mit der entsprechenden Intervallhälfte fort.



In jedem Schritt wird die Nullstelle auf ein Intervall halber Länge festgelegt (Interpretation: Intervalllänge = erreichte Genauigkeit). Dies entspricht linearer Konvergenz mit Kontraktionskonstante 1/2.

+ **globale Konvergenz**: wenn Startintervall mit Vorzeichenwechsel bekannt ist, dann wird eine Nullstelle mit Sicherheit gefunden,
+ es wird eine Einschließung (obere und untere Schranke) der Nullstelle geliefert,
+ nur Stetigkeit von f nötig,
− nur lineare Konvergenz: mit $2^{-10} = \frac{1}{1024} \approx 10^{-3}$ gilt die Faustregel, daß durch je 10 Schritte etwa 3 Dezimalstellen absoluter Genauigkeit gewonnen werden.

3.7 Das Sekantenverfahren

Banach-Iterationen $x_{i+1} = \Phi(x_i)$ sind **Einschrittverfahren**. Betrachte nun **Mehrschrittverfahren** $x_{i+1} = \Phi(x_i, x_{i-1}, \ldots)$, speziell:

Definition 3.27:

Das **Sekantenverfahren** *zur Lösung von $f(x) = 0$ ist die Iteration*

$$x_{i+1} = \Phi(x_i, x_{i-1}) = \frac{x_{i-1}f(x_i) - x_i f(x_{i-1})}{f(x_i) - f(x_{i-1})} \qquad \left(= \; x_i \; - \; \frac{f(x_i)}{\frac{f(x_i)-f(x_{i-1})}{x_i - x_{i-1}}} \right).$$

Man braucht nun 2 Startwerte x_0, x_1 mit $f(x_0) \neq f(x_1)$.

Satz 3.28:

Es sei x^* einfache Nullstelle einer 2-fach stetig diff'baren Funktion f. Dann existiert eine Umgebung $U(x^*)$, so daß für die Sekanteniteration mit beliebigem $x_0 \neq x_1 \in U(x^*)$ gilt:

a) entweder es existiert ein Index i_0 mit $f(x_{i_0}) = 0$ (Abbruch: Nullstelle ist gefunden), oder $f(x_{i-1}) \neq f(x_i)$ $\forall i \in \mathbb{N}$ (d.h., alle Schritte $(x_{i-1}, x_i) \mapsto x_{i+1}$ sind durchführbar),

*b) die Iteration konvergiert mit der Konvergenzordnung $p = \frac{1+\sqrt{5}}{2} =$ 1.618... (**"goldener Schnitt"**) gegen x^*.*

Beweis: a) Auf einer hinreichend kleinen Umgebung einer einfachen Nullstelle ist f monoton. Unter der Voraussetzung, daß (x_i) diese Umgebung nicht verläßt (ist mit b) gerechtfertigt):
Induktionsannahme: es gelte $x_0 \neq x_1, \ldots, x_{i-2} \neq x_{i-1}$ (und damit $f(x_0) \neq f(x_1), \ldots, f(x_{i-2}) \neq f(x_{i-1})$), sodaß die Rekursion bis x_i wohldefiniert ist. Für $f(x_{i-1}) \neq 0$ folgt

$$x_i = x_{i-1} - \frac{(x_{i-1} - x_{i-2}) f(x_{i-1})}{f(x_{i-1}) - f(x_{i-2})} \neq x_{i-1} \quad \Rightarrow \quad f(x_i) \neq f(x_{i-1}) \ ,$$

womit x_{i+1} auch wohldefiniert ist.
b) Mit $e_i := x_i - x^*$ folgt

$$e_{i+1} = \frac{e_{i-1} f(x_i) - e_i f(x_{i-1})}{f(x_i) - f(x_{i-1})} = e_i \, e_{i-1} \, \frac{\text{Zähler}}{\text{Nenner}}$$

mit

$$\text{Nenner} = \frac{f(x_{i-1}) - f(x_i)}{x_{i-1} - x_i} = f'(\xi_i)$$

(Zwischenwert ξ_i) und

$$\text{Zähler} = \frac{\frac{f(x_{i-1}) - f(x_i)}{x_{i-1} - x_i} - \frac{f(x_i) - f(x^*)}{x_i - x^*}}{x_{i-1} - x^*} \ .$$

Dies ist eine sogenannte "zweite dividierte Differenz", für die in Satz 7.12.b) später bei der Polynominterpolation die Darstellung Zähler $= f''(\eta_i)/2$ mit einem Zwischenwert

$$\eta_i \in \Big(\min(x_{i-1}, x_i, x^*), \max(x_{i-1}, x_i, x^*) \Big)$$

gezeigt werden wird. Damit gilt die Fehlervererbung

$$e_{i+1} = e_i \, e_{i-1} \, \frac{1}{2} \, \frac{f''(\eta_i)}{f'(\xi_i)} \ .$$

Wähle ein $C > \frac{1}{2}\left|\frac{f''(x^*)}{f'(x^*)}\right|$. Wegen Stetigkeit von f', f'' existiert eine offene Umgebung $\tilde{U}(x^*)$ mit $C \geq \frac{1}{2}\left|\frac{f''(\eta)}{f'(\xi)}\right|$ $\forall \eta, \xi \in \tilde{U}(x^*)$. Sie enthält ein Intervall um x^* mit Radius r. Setze

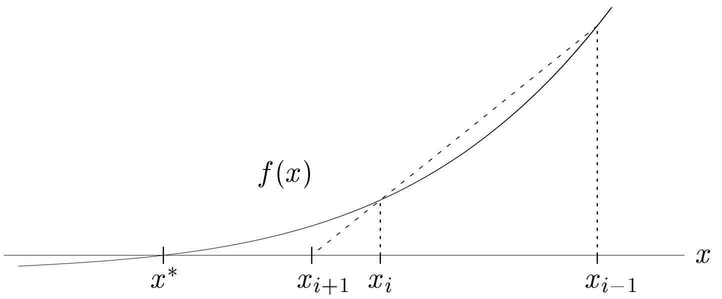
$$U(x^*) = \left\{ x;\, |x - x^*| \leq \min\left(r, \frac{1}{2C}\right) \right\} \subset \tilde{U}(x^*) \ .$$

Für $x_{i-1}, x_i \in U(x^*)$ folgt $x_{i+1} \in U(x^*)$, da $|e_{i+1}| \leq |e_i|\,|e_{i-1}|\,C \leq |e_i|/2$. Hiermit folgt auch Konvergenz $e_i \to 0$ für beliebige Startwerte $x_0 \neq x_1 \in U(x^*)$.

Zur Konvergenzrate: definiere $E_0 := |e_0|, E_1 := |e_1|$ und $E_{i+1} := C\,E_i E_{i-1}$, womit wegen $|e_{i+1}| \leq C|e_i|\,|e_{i-1}|$ induktiv $|e_i| \leq E_i$ für $i = 0, 1, 2, \ldots$ folgt. Nach Aufgabe 15 gilt $\lim\limits_{i\to\infty}(E_{i+1}/E_i^p) = C^{p-1}$, also $E_{i+1} \leq \text{const}\, E_i^p$.

Q.E.D.

Bemerkung 3.29: *Geometrische Konstruktion: x_{i+1} ist die Nullstelle der Sekante durch x_{i-1}, x_i.*



Strahlensatz: $\dfrac{f(x_i)}{x_i - x_{i+1}} = \dfrac{f(x_{i-1})}{x_{i-1} - x_{i+1}} \quad \Rightarrow \quad x_{i+1} = \dfrac{x_{i-1}f(x_i) - x_i f(x_{i-1})}{f(x_i) - f(x_{i-1})}$.

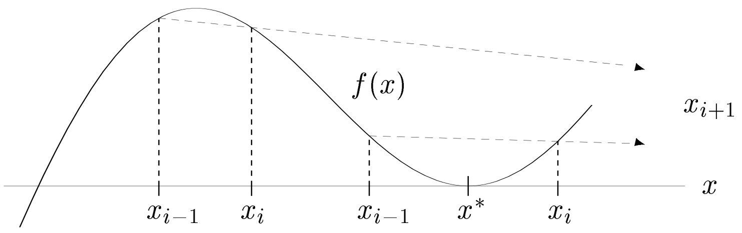
Bemerkung 3.30: *Zur Berechnung von x_i braucht man $f(x_{i-1})$ und $f(x_{i-2})$. Speichert man $f(x_{i-1})$, so braucht man für x_{i+1} nur noch* **eine Funktionsauswertung** *$f(x_i)$! Vergleich:*

- *1 Newton-Schritt: 2 Auswertungen f und f',*
- *1 Sekanten-Doppelschritt $x_i \to x_{i+1} \to x_{i+2}$: 2 Auswertungen $f(x_i)$, $f(x_{i+1})$.*

In diesem Vergleich ist das Sekantenverfahren schneller als das Newton-Verfahren:

$$|x_{i+2} - x^*| \leq C\,|x_{i+1} - x^*|^p \leq CC^p\,|x_i - x^*|^{p^2} \overset{(1+p=p^2)}{=} (C\,|x_i - x^*|)^{2.618\ldots} \ .$$

Bemerkung 3.31: *Es gibt globale Konvergenzprobleme bei lokalen Extrema und Sattelpunkten, auch bei mehrfachen Nullstellen:*



Bemerkung 3.32: *Eine Variante des Sekanten-Verfahrens ist die* **regula falsi***:*

$$x_{i+1} = \Phi(x_i) = x_i - \frac{(x_i - x_0)\, f(x_i)}{f(x_i) - f(x_0)}$$

mit fixiertem x_0 und Startpunkt $x_1 \neq x_0$. Dies ist eine Banach-Iteration mit der asymptotischen Kontraktionskonstante

$$\Phi'(x^*) = 1 - \frac{f'(x^*)}{\frac{f(x_0)-f(x^*)}{x_0-x^*}} \ ,$$

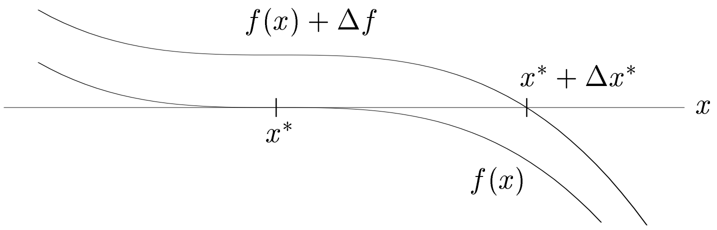
d.h., höchstens lineare Konvergenz. Ist nur historisch interessant.

Zusammenfassung des Sekantenverfahrens:

+ sehr schnell,
+ braucht nur f, nicht f',
+ realistische a priori Abschätzungen verfügbar (siehe z.B. [F.Stummel u. K.Hainer, Praktische Mathematik]),
– Probleme bei entarteten Nullstellen möglich,
– globale Konvergenzprobleme.

3.8 Numerische Genauigkeit bei mehrfachen Nullstellen

Mehrfache Nullstellen reagieren empfindlich auf Rundungsfehler: $\boxed{\downarrow 18.11.97}$



Behauptung 3.33:

Es sei Δf der Auswertungsfehler bei der numerischen Berechnung einer Funktion f, es sei x^* eine q-fache Nullstelle:

$$f(x^*) = f'(x^*) = \ldots = f^{(q-1)}(x^*) = 0 \ , \quad f^{(q)}(x^*) \neq 0 \ .$$

Dann läßt sich x^* **prinzipiell** *nur bis auf die absolute Genauigkeit*

$$|\Delta x^*| \approx \left(\frac{q!\ |\Delta f|}{|f^{(q)}(x^*)|} \right)^{1/q}$$

bestimmen.

Begründung: Es sei x^* die exakte Nullstelle von $f(x)$, es sei $\tilde{x}^*$ die exakte Nullstelle der numerischen Funktion $\tilde{f}(x) = f(x) - \Delta f$. Taylor-Entwicklung um x^* liefert

$$0 = \tilde{f}(\tilde{x}^*) = f(\tilde{x}^*) \ + \ (\tilde{f}(\tilde{x}^*) - f(\tilde{x}^*)) =$$

$$\underbrace{f(x^*) + \cdots + \frac{f^{(q-1)}(x^*)}{(q-1)!}(\tilde{x}^* - x^*)^{q-1}}_{=0} \ + \ \frac{f^{(q)}(\xi)}{q!}(\tilde{x}^* - x^*)^q \ - \ (f(\tilde{x}^*) - \tilde{f}(\tilde{x}^*))$$

$$\Longrightarrow \quad (\tilde{x}^* - x^*)^q = \frac{q!\ (f(\tilde{x}^*) - \tilde{f}(\tilde{x}^*))}{f^{(q)}(\xi)} \approx \frac{q!\ \Delta f}{f^{(q)}(x^*)}$$

mit einem Zwischenwert $\xi \approx x^* \approx \tilde{x}^*$. D.h., selbst mit einem (hypothetischen) Verfahren, das die Nullstelle $\tilde{x}^*$ von $\tilde{f}$ auf jede beliebige Genauigkeit bestimmen kann, wird x^* mittels $\tilde{x}^*$ nur bis auf die angegebene Genauigkeit approximiert.

Merke: Wenn $\frac{q!\ |\Delta f|}{|f^{(q)}(x^*)|}$ in der Größenordnung der Maschinengenauigkeit τ liegt, so ist eine q-fache Nullstelle <u>mit jedem Verfahren</u> nur bis auf $\tau^{1/q}$ bestimmbar. Z.B., eine dreifache Nullstelle ist mit 15stelliger Rechnung prinzipiell nur bis auf etwa 5 Nachkommastellen absoluter Genauigkeit zu berechnen!

3.9 Nullstellen von Polynomen

Hauptanwendung: bestimme Matrixeigenwerte, d.h., Nullstellen eines charakteristischen Polynoms. Aber: es gibt hierfür auch schnelle und stabile Methoden, die nicht das charakteristische Polynom benutzen, siehe Kapitel 6.

3.9.1 Das Horner-Schema

Aufgabe: werte $p(x) = a_0 x^n + \cdots + a_{n-1} x + a_n$ für $x = x_0$ aus.

Naiv:	berechne	$x_0^2, x_0^3, \dots, x_0^n$	($n-1$ Multiplikationen)
		$a_{n-1}\, x_0, \dots, a_0\, x_0^n$	(n Multiplikationen)
	und addiere		(n Additionen)

Besser:

$$p(x) \quad = \quad ((\ \dots\ (\ (\ a_0\ x + a_1\ )\ x + a_2\ )\ x + \cdots)\dots)\ x + a_{n-1}\ )\ x + a_n\ .$$

$$\begin{array}{l} \uparrow \\ b_0 = a_0 \\ \underbrace{\qquad\qquad}_{} \\ b_1(x) = a_0\, x + a_1 \\ \underbrace{\qquad\qquad\qquad\qquad}_{} \\ b_2(x) = a_0\, x^2 + a_1\, x + a_2 \\ \qquad\qquad \ddots \\ \underbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}_{} \\ b_n(x) = a_0\, x^n + a_1\, x^{n-1} + \cdots + a_{n-1}\, x + a_n \end{array}$$

Also: $\quad b_i(x) = x\, b_{i-1}(x) + a_i \;$ mit $b_0 = a_0$.

Horner-Schema 3.34:

Zu gegebenen $a_0, \dots, a_n$ und x_0 liefert die Rekursion

$$\boxed{\begin{array}{l} b_0 := a_0\ ; \\ \textit{for } i := 1 \textit{ to } n \textit{ do } b_i := b_{i-1} x_0 + a_i\ ; \end{array}}$$

die Polynomauswertung $b_n = a_0\, x_0^n + \cdots + a_{n-1}\, x_0 + a_n = p(x_0)$ mit n Multiplikationen und n Additionen.

Bemerkung 3.35: *Mit $p(x) = a_0\, x^n + \cdots + a_{n-1}\, x + a_n$ gilt für die b_i aus dem Horner-Schema:*

$$\frac{p(x) - p(x_0)}{x - x_0} = b_0\, x^{n-1} + b_1 x^{n-2} + \cdots + b_{n-2}\, x + b_{n-1}$$

(Beweis: Aufgabe 17). Also: das Horner-Schema erlaubt numerische Polynomdivision, die b_i sind die Koeffizienten des Faktorpolynoms.

Bemerkung 3.36: *Nach Konstruktion ist b_i ein Polynom i.ten Grades in x_0:*

$$b_0 = a_0\ , \quad b_1 = a_0\, x_0 + a_1\ , \quad b_2 = a_0\, x_0^2 + a_1 x_0 + a_2\ , \quad \dots\ .$$

Für $c_i := db_{i+1}/dx_0$ folgt durch Differenzieren von $b_{i+1}(x_0) = x_0\, b_i(x_0) + a_{i+1}$ die Rekursion

$$c_i = x_0\, c_{i-1} + b_i \quad \textit{mit dem Start} \quad c_0 = \frac{db_1}{dx_0} = b_0\ .$$

Damit wird die Ableitung $p'(x_0) = db_n/dx_0 = c_{n-1}$ numerisch berechenbar. Analog für $d_i := dc_{i+1}/dx_0 = d^2 b_{i+2}/dx_0^2$:

$$d_i = x_0\, d_{i-1} + 2\, c_i \quad \textit{mit dem Start} \quad d_0 = \frac{dc_1}{dx_0} = 2\, c_0 \ .$$

Höhere Ableitungen sind analog auswertbar:

Erweitertes Horner-Schema *für $p(x) = a_0 x^n + \cdots + a_n$:*

$$\boxed{\begin{array}{lll} b_0 := a_0 \ ; & \textit{for } i := 1 \textit{ to } n \textit{ do} & b_i := x_0\, b_{i-1} + a_i \ ; \\ c_0 := b_0 \ ; & \textit{for } i := 1 \textit{ to } n-1 \textit{ do} & c_i := x_0\, c_{i-1} + b_i \ ; \\ d_0 := 2\, c_0 \ ; & \textit{for } i := 1 \textit{ to } n-2 \textit{ do} & d_i := x_0\, d_{i-1} + 2\, c_i \ ; \\ e_0 := 3\, d_0 \ ; & \textit{for } i := 1 \textit{ to } n-3 \textit{ do} & e_i := x_0\, e_{i-1} + 3\, d_i \ ; \\ \quad\vdots & \qquad\quad\vdots & \qquad\quad\vdots \end{array}}$$

Liefert: $b_n = p(x_0)$, $c_{n-1} = p'(x_0)$, $d_{n-2} = p''(x_0)$, $e_{n-3} = p'''(x_0)$ *etc.*

3.9.2 Nullstellenabschätzung

Die Lage der Nullstellen eines Polynoms kann a priori aus den Koeffizienten abgeschätzt werden:

Satz 3.37:

Für die Nullstellen $z \in \mathbb{C}$ von $p(x) = a_n\, x^n + \cdots + a_0$ gilt

$$|z| \leq \max\left(\ \left|\frac{a_0}{a_n}\right| \ , \ 1 + \left|\frac{a_1}{a_n}\right| \ , \ \cdots \ , \ 1 + \left|\frac{a_{n-1}}{a_n}\right| \ \right) .$$

Achtung: andere Indizierung der a_i als in Sektion 3.9.1!

Beweis: Es sei m das angegebene Maximum.

Behauptung: Für $|z| > m$ gilt $\displaystyle\sum_{k=0}^{j-1} \left|\frac{a_k}{a_n}\right| \ |z|^{k-j} < 1$ für $j = 1, \ldots, n$.

Beweis: Induktion nach j:

$$j = 1: \quad \left|\frac{a_0}{a_n}\right| \ \frac{1}{|z|} < 1 \quad \Longleftrightarrow \quad |z| > \left|\frac{a_0}{a_n}\right| \qquad \text{(ok).}$$

$$j \mapsto j+1: \quad \sum_{k=0}^{j} \left|\frac{a_k}{a_n}\right| \ |z|^{k-j-1} = \frac{1}{|z|} \left(\sum_{k=0}^{j-1} \left|\frac{a_k}{a_n}\right| \ |z|^{k-j} + \left|\frac{a_j}{a_n}\right| \right)$$

$$< \frac{1}{|z|}\left(1+\left|\frac{a_j}{a_n}\right|\right) < \frac{1}{m}\left(1+\left|\frac{a_j}{a_n}\right|\right) \overset{(j<n)}{\leq} 1 \; .$$

□ (Behauptung)

Hiermit folgt

$$|p(z)| = \left|\sum_{k=0}^{n} a_k\, z^k\right| = |a_n|\,|z^n|\,\left|1+\sum_{k=0}^{n-1}\frac{a_k}{a_n}\, z^{k-n}\right|$$

$$\underset{\text{Dreiecksungl.}}{\overset{\text{umgekehrte}}{\geq}} \quad |a_n|\,|z^n|\,\Bigg(1-\underbrace{\sum_{k=0}^{n-1}\left|\frac{a_k}{a_n}\right|\;|z|^{k-n}}_{<1}\Bigg) \;>0\; .$$

Also: $|z|>m \Rightarrow z$ ist nicht Nullstelle.

Q.E.D.

Bemerkung 3.38: *Es existieren noch viele andere Abschätzungen (z.B. Satz 5.5.8 in [J. Stoer, Numerische Mathematik 1]).*

Bemerkung 3.39: *Satz 3.37 beschreibt den maximalen Abstand der Nullstellen vom Nullpunkt. Als einfaches Korollar ergeben sich Abschätzungen des minimalen Abstands der Nullstellen vom Nullpunkt. Dazu betrachte man zu* $p(x)=a_n\,x^n+\cdots+a_0$ *das Polynom* $\tilde{p}(\tilde{x})=a_0\,\tilde{x}^n+\cdots+a_n$, *in dem die Reihenfolge der Koeffizienten vertauscht wurde. Für* $\tilde{x}=1/x$ *gilt*

$$\tilde{p}\Big(\frac{1}{x}\Big) \;=\; \frac{a_0}{x^n}+\frac{a_1}{x^{n-1}}+\cdots+\frac{a_{n-1}}{x}+a_n \;=\; \frac{a_0+a_1\,x+\cdots+a_n\,x^n}{x^n} \;=\; \frac{p(x)}{x^n}\;,$$

d.h., die Kehrwerte der Wurzeln von $\tilde{p}$ *sind die Wurzeln von* p. *Wendet man Satz 3.37 auf* $\tilde{p}$ *an, so gilt (beachte* $a_i \leftrightarrow a_{n-i}$*)*

$$|\tilde{z}| \leq \max\Big(\;\left|\frac{a_n}{a_0}\right|\;,\;1+\left|\frac{a_{n-1}}{a_0}\right|\;,\;\cdots\;,\;1+\left|\frac{a_1}{a_0}\right|\;\Big)$$

für alle Wurzeln $\tilde{z}$ *von* $\tilde{p}$, *also gilt*

$$\boxed{|z| \geq \max\Big(\;\left|\frac{a_n}{a_0}\right|\;,\;1+\left|\frac{a_{n-1}}{a_0}\right|\;,\;\cdots\;,\;1+\left|\frac{a_1}{a_0}\right|\;\Big)^{-1}}$$

für alle Wurzeln $z=1/\tilde{z}$ *von* $p(x)=a_n\,x^n+\cdots+a_1\,x+a_0$.

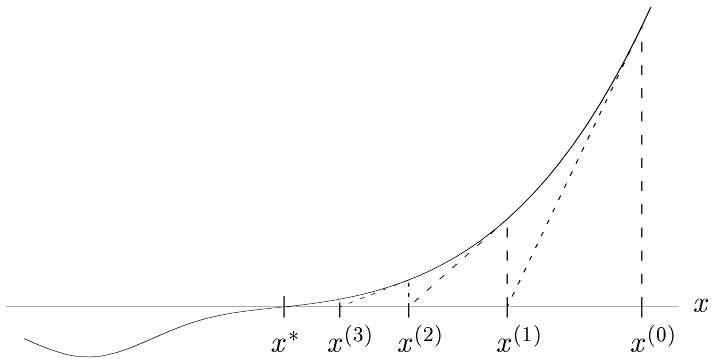
Hat man eine Approximation $\bar{x}$ *einer Wurzel eines Polynoms* $f(x)$, *so kann man die Taylor-Entwicklung* $p(\Delta x)=f(\bar{x}+\Delta x)=a_n\,(\Delta x)^n+\cdots+a_1\,\Delta x+a_0$ *von* f *um* $\bar{x}$ *betrachten und so abschätzen, wie groß der Abstand* Δx *zwischen* $\bar{x}$ *und der nächstgelegenen exakten Nullstelle von* f *mindestens ist.*

3.9.3 Das Newton-Verfahren für Polynome

Mit

$$p(x) = a_n\, x^n + \cdots + a_0 = a_n\, x^n \left(1 + \frac{a_{n-1}}{a_n\, x} + \cdots + \frac{a_0}{a_n\, x^n}\right) \overset{(|x| \gg 1)}{\approx} a_n\, x^n$$

ist die Newton-Folge für Startwerte $|x^{(0)}| \gg 1$ zunächst monoton:



Satz 3.40: (Globale Konvergenz des Newton-Verfahrens für Polynome)
Das reelle Polynom $p(x)$ habe die reellen Nullstellen $x_1 \geq \ldots \geq x_N$ und die komplexen Nullstellenpaare $\xi_1, \bar{\xi}_1, \ldots, \xi_M, \bar{\xi}_M$ (d.h., Polynomgrad$(p) = N + 2\,M$). Falls $x_1 \geq \max\{\Re e(\xi_1), \ldots, \Re e(\xi_M)\}$ gilt ($\Re e$ = Realteil), dann konvergiert die Newton-Folge $x^{(i+1)} = x^{(i)} - p(x^{(i)})/p'(x^{(i)})$ für jeden Startwert $x^{(0)} > x_1$ monoton fallend gegen x_1.

Beweis: Aufgabe 18.

20.11.97↓ **Bemerkung 3.41:** *Die Lage der Nullstellen kann mittels Satz 3.37 (Bemerkung 3.38) abgeschätzt werden, sodaß Startwerte $x^{(0)} > x_1$ leicht ermittelt werden können. Ist der Startwert jedoch zu weit von x_1 entfernt, so beobachtet man zunächst, daß sich die Newton-Punkte nur langsam der Nullstelle x_1 nähern, bis die (bei nicht-entartetem x_1) quadratische Konvergenz einsetzt: Für sehr großes x gilt $p(x) = a_n\, x^n + \cdots + a_0 \approx a_n\, x^n$, sodaß mit*

$$x^{(i+1)} = x^{(i)} - \frac{p(x^{(i)})}{p'(x^{(i)})} \;\approx\; x^{(i)} - \frac{a_n\,(x^{(i)})^n}{n\,a_n\,(x^{(i)})^{n-1}} \;=\; \left(1 - \frac{1}{n}\right) x^{(i)}$$

die Punkte zunächst um den konstanten Faktor $1 - 1/n$ verkleinert werden, der für $n \gg 1$ dicht bei 1 liegt. Bei hochgradigen Polynomen ist eine gute Abschätzung der größten Nullstelle wichtig!

Nach Auffinden der größten Nullstelle kann das Newton-Verfahren zur Suche nach den weiteren Nullstellen benutzt werden:

Bezeichnung 3.42:

Es sei x_1 eine schon gefundene Nullstelle von $p(x)$. Dann heißt $p_1(x) = \dfrac{p(x)}{x - x_1}$ **Deflationspolynom**. *Es enthält die restlichen Nullstellen.*

Über das Horner-Schema 3.34 sind nach Bemerkung 3.35 die Koeffizienten b_i von

$$p_1(x) = \frac{p(x)}{x - x_1} = \frac{p(x) - p(x_1)}{x - x_1} = b_0 x^{n-1} + \cdots + b_{n-1}$$

berechenbar, so daß mittels p_1 die weiteren Nullstellen ermittelt werden können. Aber (Problem!): die berechnete Nullstelle x_1 ist mit einem Fehler behaftet. Dieser pflanzt sich auf p_1 fort, so daß weitere Nullstellen immer ungenauer bestimmt werden.

Besser "formale Deflation": Es seien $x_1, \ldots, x_j$ schon gefundene Nullstellen von p. Das Newton-Verfahren für das Deflationspolynom

$$p_j(x) = \frac{p(x)}{\omega_j(x)}\ , \quad \omega_j(x) = (x - x_1) \cdots (x - x_j)$$

ist die Iteration

$$x \ \to \ \Phi_j(x) := x - \frac{p_j(x)}{p_j'(x)} = x - \frac{\frac{p(x)}{\omega_j(x)}}{\frac{p'(x)}{\omega_j(x)} - p(x)\frac{\omega_j'(x)}{\omega_j(x)^2}} = x - \frac{p(x)}{p'(x) - p(x)\frac{\omega_j'(x)}{\omega_j(x)}}$$

mit

$$\frac{\omega_j'(x)}{\omega_j(x)} = \frac{\sum\limits_{k=1}^{j} (x - x_1) \cdots \cancel{(x - x_k)} \cdots (x - x_j)}{(x - x_1) \cdots (x - x_j)} = \sum_{k=1}^{j} \frac{1}{x - x_k}\ .$$

Also: die Newton-Iteration für $p_j(x) = \dfrac{p(x)}{(x - x_1) \cdots (x - x_j)}$ ist die Iteration

$$x \ \to \ \Phi_j(x) = x - \frac{p(x)}{p'(x) - p(x) \sum\limits_{k=1}^{j} \frac{1}{x - x_k}}\ .$$

Dies ist unabhängig davon, ob $x_1, \ldots, x_j$ Nullstellen von p sind. Wenn sie Nullstellen sind, dann ist p_j wieder ein Polynom und die Iteration mit Φ_j konvergiert für große Startwerte monoton fallend gegen die nächste Nullstelle von p.

Dies führt zum **Newton-Maehly-Verfahren** zur Bestimmung <u>aller</u> reellen Nullstellen eines Polynoms:

Algorithmus 3.43: ("Newton-Maehly-Verfahren")

Gegeben reelles Polynom p mit rein reellen Nullstellen $x_1 \geq x_2 \geq \dots$.

a) *Finde mit Satz 3.37 einen Startpunkt $x^{(0)} > x_1$.*

b) *Finde x_1 durch das Newton-Verfahren mit Start $x^{(0)}$.*

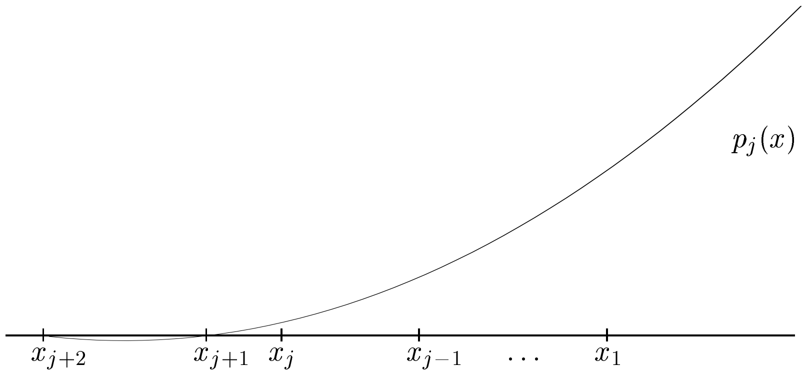
b) *Wenn $x_1 \geq x_2 \geq \cdots \geq x_j$ schon gefundene Nullstellen sind, dann finde x_{j+1} durch Iteration mit*

$$\Phi_j(x) = x - \frac{p(x)}{p'(x) - p(x) \displaystyle\sum_{k=1}^{j} \frac{1}{x - x_k}}$$

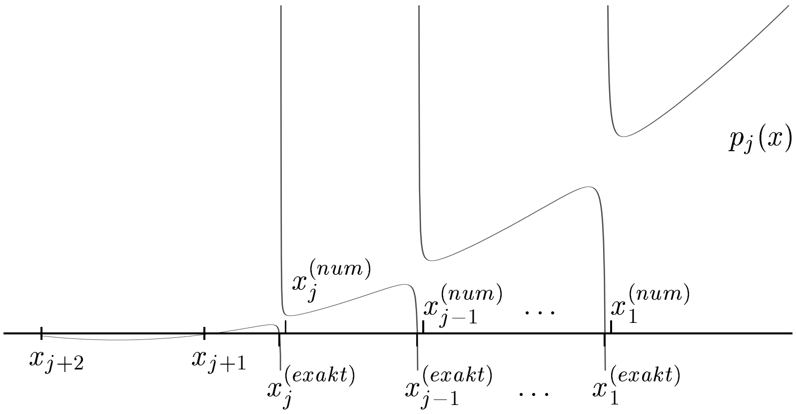
mit Startwert $x^{(0)}$ (die Konvergenz ist monoton fallend).

Bemerkung 3.44: *Die Nullstellen können durchaus entartet sein und werden dann mehrfach angefahren. Sie werden aber nur in linearer Konvergenz und mit großen Fehlern berechnet.*

Bemerkung 3.45: *Außer bei Entartung ist dies Verfahren numerisch sehr stabil, da es sich um eine Banach-Iteration mit der Ausgangsfunktion p handelt. Die explizite Deflation auf p_j wird vermieden. Der Zusatzterm mit x_k in Φ_j kommt nur für $x \approx x_k$ zum Tragen und sorgt dafür, daß die Newton-Iteration monoton fallend über x_k hinwegläuft und x_{j+1} ansteuert:*



Da die numerischen Nullstellen $x_i = x_i^{(num)}$ allerdings mit Fehlern behaftet sind, hat $p_j(x) = \dfrac{p(x)}{(x - x_1^{(num)}) \cdots (x - x_j^{(num)})}$ dicht beieinander Nullstellen $x_i^{(exakt)}$ und Pole $x_i^{(num)}$:



Bemerkung 3.46: *Auch bei vorhandenen komplexen Wurzelpaaren werden in der Regel alle reellen Wurzeln gefunden, allerdings nicht mehr notwendigerweise in der Reihenfolge* $x_1 \geq x_2 \geq \dots$.

Bemerkung 3.47: *Die komplexen Wurzelpaare reeller Polynome können ebenfalls mit reeller Arithmetik berechnet werden (***Bairstow-Verfahren***, siehe z.B. [Stoer, Numerische Mathematik 1] oder [Oevel, Einführung in die Numerische Mathematik]).*

Kapitel 4

Lineare Gleichungssysteme

Aufgabe: berechne die Lösung $\vec{x} = A^{-1}\vec{b}$ eines linearen Gleichungssystem $A\vec{x} = \vec{b}$ mit invertierbarer $n \times n$-Matrix A.

4.1 Vorbemerkungen

a) Der Rechenaufwand ist i.a. $\frac{n^3}{3}$ ... n^3 Operationen für beliebiges A. Weniger, falls A spezielle Strukturen hat ("dünnbesetzt" ist).

b) Es gibt 2 universelle Verfahren

- Gauß-Elimination = LR-Zerlegung,
- QR-Zerlegung,

die für jedes A funktionieren. Diese sind <u>direkt</u>, d.h., die Lösung ist nach $O(n^3)$ Rechenschritten exakt bis auf Rundungsfehler berechnet. Für spezielle A gibt es schnellere <u>iterative</u> Verfahren.

c) Man berechnet A^{-1} <u>nicht</u> (!) explizit.

d) Der Rechenaufwand ist aus einer (Pseudo-)Implementierung abzuleiten, z.B., Matrixmultiplikation $C = (c_{ij}) = AB = (a_{ij})(b_{ij})$ zweier $n \times n$-Matrizen: $\boxed{\downarrow 25.11.97}$

```
for i := 1 to n do
    for j := 1 to n do
        c_ij := sum_{k=1}^{n} a_ik b_kj ;
```

$$\Longrightarrow \begin{cases} \displaystyle\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} 1 = n^3 \text{ Multiplikationen} \\ \displaystyle\sum_{i=1}^{n}\sum_{j=1}^{n}(n-1) = n^3 - n^2 \text{ Additionen} \end{cases}$$

Nützliche Formeln: $\displaystyle\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$, $\displaystyle\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$, etc.

4.2 Blockzerlegungen

Man kann große Matrizen als kleinere Matrizen mit "Blockeinträgen" auffassen, die formal multipliziert werden können, z.B.:

$$\left(\begin{array}{cc} \boxed{\begin{array}{cc} a_{11} & a_{12} \\ a_{21} & a_{22} \end{array}} & \boxed{\begin{array}{cc} a_{13} & a_{14} \\ a_{23} & a_{24} \end{array}} \\ \boxed{\begin{array}{cc} a_{31} & a_{32} \\ a_{41} & a_{42} \end{array}} & \boxed{\begin{array}{cc} a_{33} & a_{34} \\ a_{43} & a_{44} \end{array}} \end{array}\right) \left(\begin{array}{c} \boxed{\begin{array}{cc} b_{11} & b_{12} \\ b_{21} & b_{22} \end{array}} \\ \boxed{\begin{array}{cc} b_{31} & b_{32} \\ b_{41} & b_{42} \end{array}} \end{array}\right)$$

$$=: \left(\begin{array}{cc} \boxed{A_{11}} & \boxed{A_{12}} \\ \boxed{A_{21}} & \boxed{A_{22}} \end{array}\right) \left(\begin{array}{c} \boxed{B_1} \\ \boxed{B_2} \end{array}\right) = \left(\begin{array}{c} \boxed{A_{11}B_1 + A_{12}B_2} \\ \boxed{A_{21}B_1 + A_{22}B_2} \end{array}\right) .$$

Anderes Beispiel:

$$AB = A \underbrace{(\vec{b}_1, \vec{b}_2, \ldots)}_{\text{Spalten von } B} = \underbrace{(A\vec{b}_1, A\vec{b}_2, \ldots)}_{\text{Spalten von } AB} .$$

Also: Spalten des Produktes $= A$ wirkend auf die Spalten von B.

4.3 Strassens schnelle Matrixmultiplikation

Die Multiplikation $C = AB$ zweier $n \times n$-Matrizen braucht weniger als $O(n^3)$ Operationen!

Standard:

$$C = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix} = \begin{pmatrix} A_{11}B_{11} + A_{12}B_{21} & A_{11}B_{12} + A_{12}B_{22} \\ A_{21}B_{11} + A_{22}B_{21} & A_{21}B_{12} + A_{22}B_{22} \end{pmatrix} .$$

Strassens Beobachtung: man kommt mit 7 statt 8 Multiplikationen aus:

$$C = \begin{pmatrix} m_1 + m_2 - m_3 + m_4 & m_3 + m_5 \\ m_2 + m_6 & m_1 + m_5 - m_6 + m_7 \end{pmatrix}$$

mit

$$\begin{aligned} m_1 &= (A_{11} + A_{22})\,(B_{11} + B_{22}) \ , & m_5 &= A_{11}\,(B_{12} - B_{22}) \ , \\ m_2 &= A_{22}\,(B_{21} - B_{11}) \ , & m_6 &= (A_{21} + A_{22})\,B_{11} \ , \\ m_3 &= (A_{11} + A_{12})\,B_{22} \ , & m_7 &= (A_{21} - A_{11})\,(B_{11} + B_{12}) \ , \\ m_4 &= (A_{12} - A_{22})\,(B_{21} + B_{22}) \ . & & \end{aligned}$$

Z.B.: $m_3 + m_5 = (A_{11} + A_{12})B_{22} + A_{11}(B_{12} - B_{22}) = A_{11}B_{12} + A_{12}B_{22}$ etc.

Also:	statt (Standard)	8 Multiplikationen	+	4 Additionen
	nun (Strassen)	7 Multiplikationen	+	18 Additionen.

Sei n gerade, es seien $A_{11}, \ldots, B_{22}$ Matrixblöcke der halben Dimension $\frac{n}{2} \times \frac{n}{2}$. Angenommen, $m_1, \ldots, m_7$ werden durch "naive" Matrixmultiplikation berechnet:

$$\textbf{Standard:} \quad \left\{ \begin{array}{ll} & n^3 \text{ skalare Multiplikationen} \\ + & (n^3 - n^2) \text{ skalare Additionen} \\ \hline = & 2n^3 - n^2 \text{ Operationen.} \end{array} \right.$$

$$\textbf{Strassen:} \quad \left\{ \begin{array}{ll} & 7\,(\frac{n}{2})^3 \text{ skalare Multiplikationen} \\ + & \Big(7\,(\,(\frac{n}{2})^3 - (\frac{n}{2})^2\,) + 18\,(\frac{n}{2})^2\Big) \text{ skalare Additionen} \\ \hline = & \frac{7}{4}\,n^3 + \frac{11}{4}\,n^2 \text{ Operationen.} \end{array} \right.$$

Für $n \gg 1$ gewinnt man etwa 12.5% an Rechenzeit ($\frac{7}{4}n^3$ gegenüber $2n^3$).

Satz 4.1: (Strassen 1969)

Die Multiplikation zweier $2^m \times 2^m$-Matrizen kann mit 7^m skalaren Multiplikationen und $6\,(7^m - 4^m)$ skalaren Additionen berechnet werden. Mit $n = 2^m$ und $7^m = 2^{m \log_2 7} = n^{\log_2 7} = n^{2.807\ldots}$ ist dies für $n \gg 1$ schneller als die Standardmultiplikation ($O(n^3)$ Operationen).

Beweis: Zurückführen auf Blöcke halber Größe, diese werden wiederum halbiert usw. (Aufgabe 21).

Bemerkung 4.2: *Für die Praxis ist dies nicht sehr relevant. Z.B., $m = 10$, $n = 2^m = 1024$:*

Standard:	$2n^3 - n^2$	$\approx 2.146... \times 10^9$	*Operationen,*
Strassen:	$7^m + 6\,(7^m - 4^m)$	$\approx 1.971... \times 10^9$	*Operationen,*

d.h., nur $\approx 8.2\%$ Gewinn bei viel höherem Implementierungsaufwand.

4.4 Dreieckssysteme

Löse $A\vec{x} = \vec{b}$ mit unterer Dreiecksmatrix $A = (a_{ij}) = \begin{pmatrix} a_{11} & & 0 \\ \vdots & \ddots & \\ a_{n1} & \ldots & a_{nn} \end{pmatrix}$:

$$\begin{array}{ccccccccc} a_{11}\,x_1 & & & & & & & = & b_1 \\ a_{21}\,x_1 & + & a_{22}\,x_2 & & & & & = & b_2 \\ \vdots & & \vdots & & \ddots & & & & \vdots \\ a_{n1}\,x_1 & + & a_{n2}\,x_2 & + & \cdots & + & a_{nn}\,x_n & = & b_n \end{array}$$

Beachte: A invertierbar $\Longleftrightarrow$ alle $a_{ii} \neq 0$. Lösung durch

Rücksubstitution 4.3:

$$\boxed{\begin{array}{l} x_1 \;:=\; b_1 \,/\, a_{11} \;; \\ \textbf{for } i := 2 \textbf{ to } n \textbf{ do } \;\; x_i \;:=\; \left(b_i - \sum_{j=1}^{i-1} a_{ij}\, x_j \right) / \, a_{ii} \;; \end{array}}$$

mit

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

Multiplikationen/Divisionen (und ähnlich vielen Additionen).

Analog bei oberen Dreieckssystemen:

$$\begin{array}{ccccccccc} a_{11}\, x_1 & + & \cdots & + & a_{1,n-1}\, x_{n-1} & + & a_{1n}\, x_n & = & b_1 \\ & & \ddots & & \vdots & & \vdots & & \vdots \\ & & & & a_{n-1,n-1}\, x_{n-1} & + & a_{n-1,n}\, x_n & = & b_{n-1} \\ & & & & & & a_{nn}\, x_n & = & b_n \end{array}$$

Lösung:

$$\boxed{\begin{array}{l} x_n := b_n \,/\, a_{nn} \;; \\ \textbf{for } i := n-1 \textbf{ downto } 1 \textbf{ do } \;\; x_i := \left(b_i - \sum_{j=i+1}^{n} a_{ij}\, x_j \right) / \, a_{ii} \;; \end{array}}$$

Bemerkung 4.4:

a) *Bezeichnung: lineare Systeme mit Dreiecksmatrizen heißen* **gestaffelt**.

b) *Der Lösungsaufwand ist für gestaffelte Systeme* $A\vec{x} = \vec{b}$ *nur* $O(n^2)$ *(statt* $O(n^3)$ *bei vollbesetzten Matrizen).*

c) *Bei zusätzlichen* **Bandstrukturen**, *z.B.*

$$A = \begin{array}{c} p\left\{ \begin{array}{c} \\ \\ \\ \end{array} \right. \\ \\ \\ \end{array} \!\!\begin{pmatrix} * & & & & 0 \\ \vdots & \ddots & & & \\ * & \ddots & \ddots & & \\ & \ddots & \ddots & \ddots & \\ 0 & & \underbrace{* \quad \cdots \quad *}_{p} \end{pmatrix}$$

reduziert sich der Aufwand auf $\approx n\,p$ Multiplikationen (und ähnlich viele Additionen). Allgemein reduziert sich der Aufwand bei dünnbesetzten ("sparse") Dreiecksmatrizen auf m Multiplikationen, wo m die Anzahl der nicht-trivialen Matrixkomponenten ist. Dies gilt auch bei komplizierteren Dünnbesetztheitsgeometrien ("sparsity pattern"), wo sich die nicht-trivialen Matrixelemente nicht notwendigerweise zu wenigen Bändern um die Diagonale formieren sondern irgendwie in der Matrix verteilt sind. Hierzu müssen lediglich die Summen in der Rücksubstitution so implementiert werden, daß sie nur über die nicht-trivialen Matrixelemente laufen.

d) Die explizite Berechnung von A^{-1} braucht auch für Dreiecksmatrizen $O(n^3)$ Operationen (Aufgabe 20.c).

4.5 Der Gauß-Algorithmus

ist ein Standardalgorithmus für Systeme ohne besondere Struktur. Idee: transformiere $A\vec{x} = \vec{b}$ auf ein äquivalentes oberes Dreieckssystem $R\vec{x} = \vec{c}$, das dann durch Rücksubstitution schnell lösbar ist. Umformung mittels **Elementaroperationen**: $\boxed{\downarrow 4.12.97}$

- vertausche Gleichungen,
- addiere Vielfaches einer Gleichung zu einer anderen.

Start:

$$\begin{array}{ccccccccc} a_{11}\,x_1 & + & a_{12}\,x_2 & + & \cdots & + & a_{1n}\,x_n & = & b_1 \\ a_{21}\,x_1 & + & a_{22}\,x_2 & + & \cdots & + & a_{2n}\,x_n & = & b_2 \\ \vdots & & \vdots & & \ddots & & \vdots & & \vdots \\ a_{n1}\,x_1 & + & a_{n2}\,x_2 & + & \cdots & + & a_{nn}\,x_n & = & b_n \end{array}$$

1.ter Eliminationsschritt: ziehe geeignete Vielfache der ersten Zeile von den anderen ab, um Nullen unter a_{11} zu erzeugen:

$$\begin{array}{c|cccccccc} a_{11}\,x_1 \quad + & a_{12}\,x_2 & + & \cdots & + & a_{1n}\,x_n & = & b_1 \\ \hline a_{21}/a_{11} & a'_{22}\,x_2 & + & \cdots & + & a'_{2n}\,x_n & = & b'_2 \\ \vdots & \vdots & & \ddots & & \vdots & & \vdots \\ a_{n1}/a_{11} & a'_{n2}\,x_2 & + & \cdots & + & a'_{nn}\,x_n & = & b'_n \end{array}$$

mit

$$a'_{ik} = a_{ik} - \frac{a_{i1}}{a_{11}}\,a_{1k}\ , \quad b'_i = b_i - \frac{a_{i1}}{a_{11}}\,b_1\ , \quad i,k = 2,\ldots,n\,.$$

Bezeichnung: a_{i1}/a_{11} = **"Eliminationsfaktoren"**.

2.ter Eliminationsschritt: erzeuge Nullen unter a'_{22}:

$$\begin{array}{ccccccccccc}
a_{11}\,x_1 & + & a_{12}\,x_2 & + & a_{13}\,x_3 & + & \cdots & + & a_{1n}\,x_n & = & b_1 \\
\hline
a_{21}/a_{11} & & a'_{22}\,x_2 & + & a'_{23}\,x_3 & + & \cdots & + & a'_{2n}\,x_n & = & b'_2 \\
\vdots & & a'_{32}/a'_{22} & & a''_{33}\,x_3 & + & \cdots & + & a''_{3n}\,x_n & = & b''_3 \\
\vdots & & \vdots & & \vdots & & \ddots & & \vdots & & \vdots \\
a_{n1}/a_{11} & & a'_{n2}/a'_{22} & & a''_{n3}\,x_3 & + & \cdots & + & a''_{nn}\,x_n & = & b''_n
\end{array}$$

mit

$$a''_{ik} = a'_{ik} - \frac{a'_{i2}}{a'_{22}}\,a'_{2k}\ , \quad b''_i = b'_i - \frac{a'_{i2}}{a'_{22}}\,b'_2\ , \quad i,k = 3,\ldots,n\,.$$

j**.ter Eliminationsschritt:** ziehe Vielfaches der j.ten Zeile von den restlichen Zeilen ab, um unter dem j.ten Diagonalelement Nullen zu erzeugen.

Einzig mögliches Problem: das j.te Diagonalelement verschwindet.

1. Fall: Darunter stehen nur Nullen. Es ist nichts mehr zu tun, gehe zum $(j+1)$.ten Schritt. Dieser Fall tritt genau dann irgendwann ein, wenn A nicht invertierbar ist.

2. Fall: Es gibt mindestens einen nichtverschwindenden Eintrag unter dem j.ten Diagonalelement. Wähle irgendeinen dieser Einträge (das **"Pivotelement"**), bringe es in die j.te Zeile, indem die gesamte Pivotzeile (inklusive der abzuspeichernden Eliminationsfaktoren!) mit der j.ten Zeile ausgetauscht wird. Dann erzeuge die gewünschten Nullen.

Implementierung:

Algorithmus 4.5: (Gauß-Elimination = LR-Zerlegung, siehe Satz 4.12)

Erweitere $A = (a_{ij})$ zum $n \times (n+1)$-Schema:

$$\left(\begin{array}{ccc|c}
a_{11} & \ldots & a_{1n} & a_{1,n+1} \\
\vdots & \ddots & \vdots & \vdots \\
a_{n1} & \ldots & a_{nn} & a_{n,n+1}
\end{array}\right) \quad \textit{mit } a_{i,n+1} = b_i, \quad i = 1,\ldots,n\ .$$

for $j := 1$ **to** $n-1$ **do begin**

  falls $a_{jj} = a_{j+1,j} = \ldots = a_{nj} = 0$**, so gehe zum** $(j+1)$**.ten Schritt** ;

  $p :=$ **Pivotindex(**j**)**$\in \{j, j+1, \ldots, n\}$ **;**

  Vertauschung der Zeilen j **und** p **im** $n \times (n+1)$**-Schema** ;

  for $i := j+1$ **to** n **do begin**

    $a_{ij} := a_{ij}/a_{jj}$;   (∗ **Abspeichern der Eliminationsfaktoren** ∗)

    for $k := j+1$ **to** $n+1$ **do** $a_{ik} := a_{ik} - a_{ij}\, a_{jk}$;

  end ;

end ;

Resultat: zuletzt ist die zu $A\vec{x} = \vec{b}$ äquivalente gestaffelte Gleichung $R\vec{x} = \vec{c}$

$$\left(\begin{array}{ccc|c} r_{11} & \cdots & r_{1n} & c_1 \\ & \ddots & \vdots & \vdots \\ * & & r_{nn} & c_n \end{array}\right)$$

im Schema gespeichert und kann schnell durch Rücksubstitution gelöst werden.

Rechenaufwand zur Erzeugung des gestaffelten Systems:

$$\sum_{j=1}^{n-1} \sum_{i=j+1}^{n} \Big(1 + \sum_{k=j+1}^{n+1} 1\Big) = \frac{n\,(n-1)\,(2n+5)}{6} = \frac{n^3}{3}\Big(1 + O\Big(\frac{1}{n}\Big)\Big)$$

Multiplikationen (und ähnlich viele Additionen).

Bemerkung 4.6: *Die Umformung $A\vec{x} = \vec{b} \Longleftrightarrow R\vec{x} = \vec{c}$ funktioniert immer, selbst wenn A nicht invertierbar ist (wobei dann auch R nicht invertierbar ist).*

4.6 LR-Faktorisierung

Bezeichnung 4.7:

Die Standardbasis des $\mathbb{R}^n$ wird mit $\vec{e}_i = (0, \ldots, 0, 1, 0, \ldots, 0)^T$ bezeichnet.

Bemerkung 4.8: *Die Elementaroperationen des Gauß-Algorithmus lassen sich als Multiplikation mit* **Elementarmatrizen** *interpretieren: mit*

$$P = \begin{array}{cl} \begin{pmatrix} \mathbb{1} & & & & \\ & 0 & & 1 & \\ & & \mathbb{1} & & \\ & 1 & & 0 & \\ & & & & \mathbb{1} \end{pmatrix} & \begin{array}{l} \\ \leftarrow i \\ \\ \leftarrow j \\ \\ \end{array} \\ \begin{array}{ccccc} \phantom{\mathbb{1}} & \uparrow & \phantom{\mathbb{1}} & \uparrow & \phantom{\mathbb{1}} \\ & i & & j & \end{array} & \end{array} = \mathbb{1} - \vec{e}_i\,\vec{e}_i^{\,T} - \vec{e}_j\,\vec{e}_j^{\,T} + \vec{e}_i\,\vec{e}_j^{\,T} + \vec{e}_j\,\vec{e}_i^{\,T}$$

$$F = \begin{array}{cl} \begin{pmatrix} 1 & & & \\ & \ddots & & \\ & \alpha & \ddots & \\ & & & 1 \end{pmatrix} & \begin{array}{l} \\ \\ \leftarrow i \\ \\ \end{array} \\ \begin{array}{cccc} & \uparrow & & \\ & j & & \end{array} & \end{array} = \mathbb{1} + \alpha\,\vec{e}_i\,\vec{e}_j^{\,T}$$

gilt:

$$\begin{array}{lcl} A \to PA & : & \textit{Vertauschung der Zeilen } i \textit{ und } j, \\ A \to AP & : & \textit{Vertauschung der Spalten } i \textit{ und } j, \\ A \to FA & : & \textit{addiere das } \alpha\textit{-fache der } j\textit{.ten Zeile zur } i\textit{.ten Zeile.} \end{array}$$

Definition 4.9:

Eine **Permutationsmatrix** *ist eine quadratische Matrix, die in jeder Zeile und jeder Spalte mit genau einer* 1 *und ansonsten mit* 0 *besetzt ist.*

Lemma 4.10:

a) *Produkte von Permutationsmatrizen sind wieder Permutationsmatrizen.*

b) *Permutationsmatrizen sind orthogonal:* $PP^T = P^TP = \mathbb{1}$.

c) *Linksmultiplikation mit einer Permutationsmatrix entspricht einer Zeilenvertauschung, Rechtsmultiplikation einer Spaltenvertauschung.*

Beweis: offensichtliche Übung.

Beispiel 4.11: Sei $P = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$, $A = \begin{pmatrix} \vec{z}_1^{\,T} \\ \vec{z}_2^{\,T} \\ \vec{z}_3^{\,T} \end{pmatrix} = (\vec{s}_1, \vec{s}_2, \vec{s}_3)$ (Zerlegung von A nach Zeilen bzw. Spalten). Es gilt $PA = \begin{pmatrix} \vec{z}_2^{\,T} \\ \vec{z}_3^{\,T} \\ \vec{z}_1^{\,T} \end{pmatrix}$, $AP = (\vec{s}_3, \vec{s}_1, \vec{s}_2)$.

Satz 4.12: (LR-Zerlegung)
Das System $A\vec{x} = \vec{b}$ wird im Gauß-Algorithmus 4.5 auf die Endform

$$\left(\begin{array}{cccc|c} r_{11} & \dots & \dots & r_{1n} & c_1 \\ l_{21} & \ddots & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & \vdots & \vdots \\ l_{n1} & \dots & l_{n,n-1} & r_{nn} & c_n \end{array}\right)$$

gebracht (der untere Dreiecksanteil (l_{ij}) entsteht aus den abgespeicherten Eliminationsfaktoren). Es seien $P_1, \dots, P_{n-1}$ (eventuell mit $P_j = \mathbb{1}$) die bei der Pivotierung durchgeführten Zeilenvertauschungen. Mit

$$L = \begin{pmatrix} 1 & & & 0 \\ l_{11} & \ddots & & \\ \vdots & \ddots & \ddots & \\ l_{n1} & \dots & l_{n,n-1} & 1 \end{pmatrix}, \quad R = \begin{pmatrix} r_{11} & \dots & \dots & r_{1n} \\ & \ddots & \ddots & \vdots \\ & & \ddots & \vdots \\ 0 & & & r_{nn} \end{pmatrix}$$

*und $P = P_{n-1} \cdots P_1$ gilt: $PA = LR$ (“LR-**Faktorisierung**”).*

Beweis: Zu Beginn des j.ten Schrittes liegt das Zwischenschema $\boxed{\downarrow 9.12.97}$

$$\left(\begin{array}{cccc|ccc} r_{11} & \dots & \dots & r_{1,j-1} & r_{1j} & \dots & r_{1n} \\ l_{21} & \ddots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots & \vdots & \ddots & \vdots \\ l_{j-1,j} & \dots & l_{j-1,j-2} & r_{j-1,j-1} & r_{j-1,j} & \dots & r_{j-1,n} \\ \hline l^{(j)}_{j1} & \dots & \dots & l^{(j)}_{j,j-1} & a^{(j)}_{jj} & \dots & a^{(j)}_{jn} \\ \vdots & \ddots & \ddots & \vdots & \vdots & \ddots & \vdots \\ l^{(j)}_{n1} & \dots & \dots & l^{(j)}_{n,j-1} & a^{(j)}_{nj} & \dots & a^{(j)}_{nn} \end{array}\right) = L^{(j)} + R^{(j)}$$

mit

$$L^{(j)} = \left(\begin{array}{ccc|c} 0 & & 0 & \\ l_{21} & \ddots & & 0 \\ \vdots & \ddots & 0 & \\ \hline l^{(j)}_{j1} & \ddots & l^{(j)}_{j,j-1} & \\ \vdots & \ddots & \vdots & 0 \\ l^{(j)}_{n1} & \dots & l^{(j)}_{n,j-1} & \end{array}\right)$$

und

$$R^{(j)} = \left(\begin{array}{ccc|ccc} r_{11} & \cdots & \cdots & \cdots & \cdots & r_{1n} \\ & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & & r_{j-1,j-1} & \cdots & \cdots & r_{j-1,n} \\ \hline & & & a_{jj}^{(j)} & \cdots & a_{jn}^{(j)} \\ & 0 & & \vdots & \ddots & \vdots \\ & & & a_{nj}^{(j)} & \cdots & a_{nn}^{(j)} \end{array}\right)$$

vor. Die nächste Zeilenvertauschung P_j liefert $\tilde{L}^{(j)} = P_j L^{(j)}$ und

$$\tilde{R}^{(j)} = P_j R^{(j)} = \left(\begin{array}{ccc|ccc} r_{11} & \cdots & \cdots & \cdots & \cdots & r_{1n} \\ & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & & r_{j-1,j-1} & \cdots & \cdots & r_{j-1,n} \\ \hline & & & r_{jj} & \cdots & r_{jn} \\ \cline{4-6} & & & \tilde{a}_{j+1,j}^{(j)} & \cdots & \tilde{a}_{j+1,n}^{(j)} \\ & 0 & & \vdots & \ddots & \vdots \\ & & & \tilde{a}_{nj}^{(j)} & \cdots & \tilde{a}_{nn}^{(j)} \end{array}\right) .$$

Die neuen Eliminationsfaktoren $\vec{l}_j = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \tilde{a}_{j+1,j}^{(j)}/r_{jj} \\ \vdots \\ \tilde{a}_{n,j}^{(j)}/r_{jj} \end{pmatrix}$ werden als j.te Spalte an $\tilde{L}^{(j)}$ angefügt:

$$L^{(j+1)} = \tilde{L}^{(j)} + \vec{l}_j \, \vec{e}_j^{\,T} .$$

Vielfache der j.ten Zeile $\vec{e}_j^{\,T} \tilde{R}^{(j)} = (0, \ldots, 0, r_{jj}, \ldots, r_{jn})$ von $\tilde{R}^{(j)}$ werden von den unteren Zeilen von $\tilde{R}^{(j)}$ abgezogen:

$$R^{(j+1)} = \tilde{R}^{(j)} - \vec{l}_j \, \vec{e}_j^{\,T} \tilde{R}^{(j)} .$$

Beachte hierzu

$$\begin{pmatrix} 0 \\ \vdots \\ 0 \\ l_{j+1} \\ \vdots \\ l_n \end{pmatrix} (0, \ldots, 0, r_{jj}, \ldots, r_{jn}) = \left(\begin{array}{c|ccc} 0 & & 0 & \\ \hline & l_{j+1} r_{jj} & \cdots & l_{j+1} r_{jn} \\ 0 & \vdots & \ddots & \vdots \\ & l_n r_{jj} & \cdots & l_n r_{jn} \end{array}\right) .$$

Ergebnis: der Gauß-Algorithmus wird beschrieben durch

$$L^{(j+1)} = P_j L^{(j)} + \vec{l}_j e_j^T \ , \quad R^{(j+1)} = P_j R^{(j)} - \vec{l}_j e_j^T \, P_j R^{(j)}$$

mit $L^{(1)} = 0$, $R^{(1)} = A$.

Behauptung: In jedem Schritt gilt $(\mathbb{1} + L^{(j)})R^{(j)} = P_{j-1} \cdots P_1 \, A$.

(Mit $L = \mathbb{1} + L^{(n)}$, $R = R^{(n)}$, $P = P_{n-1} \cdots P_1$ ist der Satz damit bewiesen).

Beweis: Induktion nach j:

$j = 1$: $\quad (\mathbb{1} + L^{(1)})R^{(1)} = A$ gilt offensichtlich.

Schritt $j \to j+1$:

$$\begin{aligned}
& (\mathbb{1} + L^{(j+1)}) \, R^{(j+1)} \\
= \ & (\mathbb{1} + P_j L^{(j)} + \vec{l}_j e_j^T) \, (\mathbb{1} - \vec{l}_j e_j^T) \, P_j R^{(j)} \\
= \ & (\mathbb{1} - \cancel{\vec{l}_j \vec{e}_j^T} + P_j L^{(j)} - P_j L^{(j)} \vec{l}_j \vec{e}_j^T + \cancel{\vec{l}_j \vec{e}_j^T} - \vec{l}_j \vec{e}_j^T \vec{l}_j \vec{e}_j^T) \, P_j R^{(j)} \\
= \ & (\mathbb{1} + P_j L^{(j)}) P_j R^{(j)} \qquad \qquad (\text{beachte } L^{(j)} \vec{l}_j = 0 \ , \ \vec{e}_j^T \vec{l}_j = 0) \\
= \ & P_j (\mathbb{1} + L^{(j)} P_j) \, R^{(j)} \overset{(*)}{=} P_j (\mathbb{1} + L^{(j)}) \, R^{(j)} \ = \ P_j \, (P_{j-1} \cdots P_1 \, A) \ .
\end{aligned}$$

Für $(*)$ beachte $L^{(j)} P_j = L^{(j)}$, da Rechtsmultiplikation mit P_j die *Spalten* $j, \ldots, n$ von $L^{(j)}$ vertauscht, die alle gleich (nämlich 0) sind.

Q.E.D.

Bemerkung 4.13: $L = \begin{pmatrix} 1 & & \\ * & \ddots & \\ * & * & 1 \end{pmatrix}$ *ist immer invertierbar. Damit ist* $R = L^{-1} P A$ *genau dann invertierbar, wenn* A *invertierbar ist.*

Bemerkung 4.14: *Zu jeder quadratischen Matrix existiert damit eine Zeilenvertauschung* P*, so daß* PA *in der angegebenen Form* LR *faktorisiert werden kann. Bei invertierbarem* A *und fixiertem* P *sind die LR-Faktoren eindeutig:*

$$LR = \tilde{L}\tilde{R} \quad \Longrightarrow \quad \underbrace{\tilde{L}^{-1} L}_{\substack{\text{untere} \\ \text{Dreiecksmatrix}}} \quad = \quad \underbrace{\tilde{R} R^{-1}}_{\substack{\text{obere} \\ \text{Dreiecksmatrix}}} \quad = \ \text{Diagonalmatrix } D \ .$$

Mit $L = \tilde{L} D$ *folgt* $D = \mathbb{1}$*, da die Diagonalen von* L *und* $\tilde{L}$ *mit 1 besetzt sind.*

Bemerkung 4.15: *Es gibt Matrizen, für die keine Faktorisierung $A = LR$ existiert, man braucht i.a. P. Faktorisierbarkeit ohne Zeilenvertauschungen ist z.B. für* **diagonaldominante** *Matrizen (Aufgabe 23) oder* **positiv definite** *symmetrische Matrizen (Bemerkung 4.32) garantiert.*

Bemerkung 4.16: *P ist die Gesamtheit aller im Gauß-Algorithmus durchgeführten Zeilenvertauschungen. Man speichert P nicht als Matrix, sondern als "***Buchhaltungsvektor***" $\vec{p}$. Dazu initialisiert man die Spalte $\vec{p} = (1, 2, \ldots, n)^T$ und wendet die Zeilenvertauschungen auch auf $\vec{p}$ an. Zuletzt hat man $\vec{p} = (p_1, \ldots, , p_n)^T$. Interpretation: die ursprüngliche Zeile p_i ist in die i.te Zeile gewandert. Hiermit ergibt sich für beliebiges $\vec{b}$ dann $\vec{c} = P\vec{b}$ durch*

$$\boxed{\textbf{for } i := 1 \textbf{ to } n \textbf{ do } \;\; c_i := b_{p_i} \; ;}$$

Bemerkung 4.17: *Mit* $\det(L) = 1$ *und leicht berechenbarem*

$$\det(R) = \text{ Produkt der Diagonalelemente}$$

kann $\det(A)$ *aus* $\det(P)\det(A) = \det(R)$ *berechnet werden. Mit* $\det(P) = \det(P_{n-1}) \cdots \det(P_1)$ *und*

$$\det(P_j) = \begin{cases} \;\;1 & \text{, falls } P_j = \mathbb{1} \\ -1 & \text{, falls } P_j = \text{ Austausch zweier Zeilen} \end{cases}$$

braucht nur die Anzahl der nichttrivialen Zeilenvertauschungen im Gauß-Algorithmus mitgezählt zu werden.

Bemerkung 4.18: Die Faktorisierungsdaten P, L, R sind genauso gut wie eine explizite Inverse A^{-1}! *Die Lösung von*

$$A\vec{x} = \vec{b} \quad \Longrightarrow \quad PA\vec{x} = P\vec{b} \; , \quad \text{d.h.,} \quad L\underbrace{R\,\vec{x}}_{\vec{y}} = P\vec{b}$$

ist durch 2 Rücksubstitutionen

$$\begin{array}{rl} & \text{bestimme } \vec{y} \text{ aus } L\vec{y} = P\vec{b}, \\ \text{dann} & \text{bestimme } \vec{x} \text{ aus } R\vec{x} = \vec{y} \end{array}$$

schnell zu berechnen. Der Aufwand (etwa n^2 Multiplikationen) entspricht dem der Matrix-Vektor-Multiplikation $A^{-1}\vec{b}$ bei explizit vorliegendem A^{-1}.

> **Merke:** Der Gauß-Algorithmus liefert mit $\approx n^3/3$ Multiplikationen die Zerlegung $PA = LR$ einer $n \times n$-Matrix. Liegen P, L, R vor, dann ist $A\vec{x} = \vec{b}$ durch doppelte Rücksubstitution mit etwa n^2 Multiplikationen schnell zu lösen.

4.7 Bandmatrizen

Definition 4.19:

Eine quadratische Matrix der Form

$$
\overbrace{\hphantom{*\;\; *\;\; \ldots\;\; *}}^{q}
$$
$$
p\left\{\vphantom{\begin{matrix}*\\ \vdots\\ *\end{matrix}}\right.
\begin{pmatrix}
* & * & \ldots & * & & \\
* & \ddots & \ddots & \ddots & \ddots & \\
\vdots & \ddots & \ddots & \ddots & \ddots & * \\
* & \ddots & \ddots & \ddots & \ddots & \vdots \\
 & \ddots & \ddots & \ddots & \ddots & * \\
 & & * & \ldots & * & *
\end{pmatrix}
\left.\vphantom{\begin{matrix}\vdots\\ *\\ *\end{matrix}}\right\} q
$$
$$
\underbrace{\hphantom{*\;\; \ldots\;\; *}}_{p}
$$

heißt **Bandmatrix** *mit p unteren (exklusive Diagonale) und q oberen (inklusive Diagonale) Bändern.*

Satz 4.20:

Falls für eine Bandmatrix A mit p unteren und q oberen Bändern eine <u>LR-Zerlegung $A = LR$ ohne Zeilenvertauschungen</u> existiert, dann hat

$$
L = p\left\{\vphantom{\begin{matrix}*\\ \vdots\\ *\end{matrix}}\right.
\begin{pmatrix}
1 & & & & & \\
* & \ddots & & & & \\
\vdots & \ddots & \ddots & & & \\
* & \ddots & \ddots & \ddots & & \\
 & \ddots & \ddots & \ddots & \ddots & \\
 & & * & \ldots & * & 1
\end{pmatrix}
$$
$$
\underbrace{\hphantom{*\;\; \ldots\;\; *}}_{p}
$$

bzw.

$$R \;=\; \begin{array}{l} \quad\overbrace{\hphantom{*\quad *\quad \dots\quad *}}^{q} \\ \left(\begin{array}{cccccc} * & * & \dots & * & & \\ & \ddots & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & \ddots & * \\ & & & \ddots & \ddots & \vdots \\ & & & & \ddots & * \\ & & & & & * \end{array}\right) \left.\begin{array}{c} \\ \\ \\ \\ \end{array}\right\} q \end{array}$$

p untere bzw. q obere Bänder. Der Gauß-Algorithmus 4.5 liefert die Zerlegung mit weniger als $np\,q$ Multiplikationen.

Beweis: Gauß-Algorithmus ohne Zeilenvertauschungen für

$$\left(\begin{array}{cccccccc|c} a_{11} & \dots & a_{1q} & & & & & & b_1 \\ a_{21} & \dots & \dots & a_{2,q+1} & & & & & b_2 \\ \vdots & \ddots & \ddots & \ddots & \ddots & & & & \vdots \\ a_{p+1,1} & \dots & \dots & \dots & \dots & a_{p+1,q+p} & & & b_{p+1} \\ & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & & \vdots \\ & & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \end{array}\right)$$

1.ter Schritt: Eliminationen sind nur in den p Zeilen $2, \dots, p+1$ nötig mit je q Multiplikationen pro Zeile.
weitere Schritte: analog.

Der Rechenaufwand und die Form von L und R sind hiermit klar.

Q.E.D.

Beispiel 4.21: Sei $A = \begin{pmatrix} d_1 & a_1 & & 0 \\ c_1 & \ddots & \ddots & \\ & \ddots & \ddots & a_{n-1} \\ 0 & & c_{n-1} & d_n \end{pmatrix}$. Angenommen, die Faktorisierung

$A = LR$ existiert. Ansatz:

$$\begin{array}{ccccc} A & = & L & & R \\ \begin{pmatrix} d_1 & a_1 & & \\ c_1 & d_2 & a_2 & \\ & \ddots & \ddots & \ddots \end{pmatrix} & = & \begin{pmatrix} 1 & & & \\ l_1 & 1 & & \\ & l_2 & 1 & \\ & & \ddots & \ddots \end{pmatrix} & & \begin{pmatrix} r_1 & a_1 & & \\ & r_2 & a_2 & \\ & & \ddots & \ddots \end{pmatrix} \end{array}$$

$$= \begin{pmatrix} r_1 & a_1 & & & \\ l_1 r_1 & l_1 a_1 + r_2 & a_2 & & \\ & l_2 r_2 & l_2 a_2 + r_3 & a_3 & \\ & & \ddots & \ddots & \ddots \\ & & & l_{n-1} r_{n-1} & l_{n-1} a_{n-1} + r_n \end{pmatrix} .$$

Vergleich mit A liefert:

$$d_1 = r_1 \ , \quad d_i = l_{i-1} a_{i-1} + r_i \quad (i = 2, \ldots, n) \ , \quad c_i = l_i r_i \quad (i = 1, \ldots, n-1) \ .$$

Ergebnis: Die LR-Daten sind durch die Rekursion

$$\boxed{r_1 := d_1 \ ; \ \textbf{for } i = 2 \textbf{ to } n \textbf{ do begin } \ l_{i-1} := c_{i-1}/r_{i-1} \ ; \ r_i := d_i - l_{i-1}\, a_{i-1} \ ; \ \textbf{end};}$$

mit $O(n)$ Operationen bestimmt. Es können sich Probleme durch $r_i = 0$ ergeben (wenn die Zerlegung $A = LR$ ohne Zeilenvertauschungen nicht existiert).

4.8 Pivotierung

Betrachte den Gauß-Algorithmus 4.5 zu Beginn des j.ten Schrittes: $\boxed{\downarrow 11.12.97}$

$$\left(\begin{array}{ccc|ccc|c} a_{11} & \cdots & \cdots & \cdots & \cdots & \cdots & a_{1,n+1} \\ & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ & & a_{j-1,j-1} & \cdots & \cdots & \cdots & a_{j-1,n+1} \\ \hline & & & a_{jj} & \cdots & a_{jn} & a_{j,n+1} \\ & & & \vdots & \ddots & \vdots & \vdots \\ & & & a_{nj} & \cdots & a_{nn} & a_{n,n+1} \end{array}\right)$$

Suche ein Pivotelement aus $\{a_{jj}, \ldots, a_{nj}\}$, um es auf die Diagonale zu bringen. Kriterium: minimiere den Einfluß von Rundungsfehlern. Mit a_{pj} als Pivotelement entstehen nach Elimination die neuen Zahlen

$$a'_{ik} = a_{ik} - \frac{a_{ij}}{a_{pj}}\, a_{pk} \ , \quad i = j, \ldots, n \ , \ i \neq p \ , \quad k = j+1, \ldots, n+1$$

im rechten unteren Block. Differentielle Fehleranalyse von

$$l := a_{ij}/a_{pj} \ , \quad t := l\, a_{pk} \ , \quad a'_{ik} := a_{ik} - t$$

liefert

$$\frac{\Delta a'_{ik}}{a'_{ik}} = \frac{\Delta a_{ik}}{a_{ik}} + \tau_{a'_{ik}} + \frac{1}{\dfrac{a_{pj}a_{ik}}{a_{pk}a_{ij}} - 1}\left(\frac{\Delta a_{ik}}{a_{ik}} - \frac{\Delta a_{ij}}{a_{ij}} + \frac{\Delta a_{pj}}{a_{pj}} - \frac{\Delta a_{pk}}{a_{pk}} - \tau_l - \tau_t\right)$$

mit

$$\begin{aligned} |\tau_{a'_{ik}}|,\ |\tau_l|,\ |\tau_t| &= \text{Darstellungsfehler von } a'_{ik},\ l \text{ und } t \\ &\leq \tau = \text{Maschinengenauigkeit.} \end{aligned}$$

Ergebnis: Wähle p so, daß die Konditionszahlen

$$\frac{1}{\dfrac{a_{pj}a_{ik}}{a_{pk}a_{ij}} - 1}, \quad i = j, \ldots, n, \quad i \neq p, \quad k = j+1, \ldots, n+1$$

möglichst klein werden. D.h., bei gegebenem j suche $p \in \{j, j+1, \ldots, n\}$ so, daß der kleinste Nenner

$$K(j,q) = \min_{k=j+1\ldots n+1} \min_{\substack{i=j\ldots n \\ i\neq q}} \left| \frac{a_{qj}a_{ik}}{a_{qk}a_{ij}} - 1 \right|, \quad q = j, \ldots, n$$

maximal wird: $K(j,p) = \max(K(j,j), \ldots, K(j,n))$.

Problem hierbei: für gegebenes j braucht man $O((n-j)^3)$ Operationen, um $K(j,j), \ldots, K(j,n)$ zu berechnen (und dann in der Suche nach p zu vergleichen). Mit $j = 1, \ldots, n-1$ ergibt dies $O(n^4)$ Operationen!

Heuristische Maximierung der $K(j,p)$: Es gilt

$$\begin{aligned} K(j,p) &= \min_k \min_i \left|\frac{a_{pj}}{a_{pk}}\right| \left|\frac{a_{ik}}{a_{ij}} - \frac{a_{pk}}{a_{pj}}\right| \\ &\geq \underbrace{\left(\min_{k=j+1\ldots n+1} \left|\frac{a_{pj}}{a_{pk}}\right|\right)}_{\text{maximiere}} \underbrace{\left(\min_{k=j+1\ldots n+1} \min_{\substack{i=j\ldots n \\ i\neq p}} \left|\frac{a_{ik}}{a_{ij}} - \frac{a_{pk}}{a_{pj}}\right|\right)}_{\text{ignoriere } p\text{-Abhängigkeit}} \\ &\geq \overbrace{\left(\frac{|a_{pj}|}{\max\limits_{k=j+1\ldots n+1} |a_{pk}|}\right)} \left(\min_{q=j\ldots n} \min_{k=j+1\ldots n+1} \min_{\substack{i=j\ldots n \\ i\neq q}} \left|\frac{a_{ik}}{a_{ij}} - \frac{a_{qk}}{a_{qj}}\right|\right). \end{aligned}$$

Definition 4.22:

Bei **“Spaltenpivotierung”** *wird der Pivotindex $p \in \{j, \ldots, n\}$ im j.ten Schritt der Gauß-Elimination 4.5 so bestimmt, daß*

$$\tilde{K}(j,q) = \frac{|a_{qj}|}{\max\limits_{k=j+1\ldots n+1} |a_{qk}|}$$

für $q = p$ maximal wird: $\tilde{K}(j,p) = \max\limits_{q=j..n} \tilde{K}(j,q)$.

$$\left(\begin{array}{cccc|c} * & * & * & * & * \\ & * & * & * & * \\ \cline{3-5} & \multicolumn{1}{c|}{} & a_{jj} & \ldots & a_{j,n+1} \\ & \multicolumn{1}{c|}{} & \vdots & \ddots & \vdots \\ & \multicolumn{1}{c|}{} & a_{nj} & \ldots & a_{n,n+1} \end{array}\right) \begin{array}{l} \\ \\ \to\ \tilde{K}(j,j) = |a_{jj}|/\max(|a_{j,j+1}|, \ldots, |a_{j,n+1}|) \\ \qquad\qquad\qquad\qquad \vdots \\ \to\ \tilde{K}(j,n) = |a_{nj}|/\max(|a_{n,j+1}|, \ldots, |a_{n,n+1}|) \end{array}$$

Berechne $\tilde{K}(j,j), \ldots, \tilde{K}(j,n)$ und finde das Maximum.

Bemerkung 4.23: *Die Berechnung von $\tilde{K}(j,j), \ldots, \tilde{K}(j,n)$ mit $j = 1, \ldots, n-1$ braucht insgesamt etwa $n^2/2$ Multiplikationen und $O(n^3)$ Vergleichsoperationen.*

Bemerkung 4.24: *Bei* **Totalpivotierung** *wird im Gauß-Algorithmus*

$$\left(\begin{array}{ccc|ccc|c} a_{11} & \cdots & \cdots & \cdots & \cdots & \cdots & a_{1,n+1} \\ & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ & & a_{j-1,j-1} & \cdots & \cdots & \cdots & a_{j-1,n+1} \\ \hline & & & a_{jj} & \cdots & a_{jn} & a_{j,n+1} \\ & & & \vdots & a_{pq} & \vdots & \vdots \\ & & & a_{nj} & \cdots & a_{nn} & a_{n,n+1} \end{array}\right)$$

ein Pivotelement a_{pq} im gesamten unteren Block gesucht und durch **Zeilen- und Spaltenvertauschung** *auf die Diagonale gebracht (eine Spaltenvertauschung entspricht einer Umnumerierung der Unbekannten $x_1, \ldots, x_n$ im Gleichungssystem). Es ist (heuristisch) so zu bestimmen, daß*

$$\tilde{K}(j,p,q) = \frac{|a_{pq}|}{\max\limits_{\substack{k=j\ldots n+1 \\ k \neq q}} |a_{pk}|}, \quad p, q \in \{j, \ldots, n\}$$

maximiert wird. Mit $j = 1, \ldots, n-1$ braucht dies $O(n^3)$ Multiplikationen und $O(n^4)$ Vergleichsoperationen und ist daher für die Praxis i.a. uninteressant.

Bemerkung 4.25: *Statt aufwendiger Pivotierungsstrategien ist die* **Nachiteration** *effektiver und schneller. Sei $\vec{x}_{num}$ eine numerische Lösung von $A\vec{x} = \vec{b}$. Die Rundungsfehler lassen sich durch folgenden Schritt erheblich reduzieren:*

mit gegebenem	$\vec{x}_{num}$
berechne das **"Residuum"**	$\vec{r} \, = \, A\vec{x}_{num} \, - \, \vec{b}$,
löse	$A\vec{y} \, = \, \vec{r}$
und korrigiere	$\vec{x}_{besser} \, = \, \vec{x}_{num} \, - \, \vec{y}$

Grob gesprochen gilt: liefert der Gleichungslöser die ersten q Dezimalstellen der Lösung, so werden beim Schritt $\vec{x}_{num} \to \vec{x}_{besser}$ weitere q Dezimalstellen relativer Genauigkeit hinzugewonnen. Nach mehreren Nachiterationsschritten sind die Rundungsfehler des Gleichungslösers praktisch eliminiert! Die Grenzgenauigkeit ist durch den relativen Fehler der Berechnung von $A\vec{x}_{num}$ in $\vec{r}$ bestimmt (Details: siehe Kapitel 4.13). Bei vorliegender LR-Faktorisierung kostet jede Nachiteration nur $O(n^2)$ Operationen!

Merke: Pivotierung dient zur Verbesserung der numerischen Stabilität der Gauß-Elimination. Spaltenpivotierung braucht einen ähnlichen Aufwand wie die eigentliche Elimination, Totalpivotierung erhöht den Rechenaufwand dramatisch (von $O(n^3)$ auf $O(n^4)$). In der Praxis kombiniert man Spaltenpivotierung mit der effektiven und schnellen Nachiteration.

4.9 Cholesky-Faktorisierung

16.12.97↓ ist eine spezielle Form der LR-Zerlegung für symmetrische positiv definite Matrizen.

Definition 4.26:

Eine symmetrische reelle $n \times n$-Matrix A heißt **(symmetrisch) positiv definit (s.p.d.)**, *wenn mit dem üblichen euklidischen Skalarprodukt $\langle \; , \; \rangle$ gilt:*

$$\langle \vec{x}, A\vec{x} \, \rangle > 0 \quad \forall \vec{x} \in \mathbb{R}^n \setminus \{0\} \, .$$

Satz 4.27: (Cholesky-Zerlegung)

Jede s.p.d. Matrix A ist in der Form $A = LL^T$ mit reellem invertierbarem Dreiecksfaktor $L = \begin{pmatrix} * & & 0 \\ \vdots & \ddots & \\ * & \dots & * \end{pmatrix}$ zerlegbar.

Beweis: Induktion nach der Dimension i.
$i = 1$: für $0 < A \in \mathbb{R}$,gilt trivialerweise $A = LL^T$ mit $L = L^T = \sqrt{A} \neq 0$.
Schritt $i-1 \to i$: zerlege $A = \left(\begin{array}{c|c} a & \vec{a} \\ \hline \vec{a}^T & a_{ii} \end{array}\right)$ mit $\vec{a}^T = (a_{i1}, \ldots, a_{i,i-1})$.
Der linke obere $(i-1)\times(i-1)$-Block a ist nach Aufgabe 26) s.p.d. und hat nach Induktionsvoraussetzung eine Cholesky-Zerlegung $a = l\,l^T$ mit invertierbarem

$$l = \begin{pmatrix} l_{11} & & 0 \\ \vdots & \ddots & \\ l_{i-1,1} & \ldots & l_{i-1,i-1} \end{pmatrix}.$$

Im Ansatz $L = \left(\begin{array}{c|c} l & 0 \\ \hline \vec{l}^T & l_{ii} \end{array}\right)$ mit $\vec{l}^T = (l_{i1}, \ldots, l_{i,i-1})$ ist nur die unterste Zeile zu bestimmen. Mit

$$A \;=\; LL^T \;=\; \left(\begin{array}{c|c} l & 0 \\ \hline \vec{l}^T & l_{ii} \end{array}\right) \left(\begin{array}{c|c} l^T & \vec{l} \\ \hline 0 & l_{ii} \end{array}\right) \;=\; \left(\begin{array}{c|c} l\,l^T & l\,\vec{l} \\ \hline (l\,\vec{l})^T & \vec{l}^T\vec{l} + l_{ii}^2 \end{array}\right)$$

gilt:

$$\vec{l} \text{ ist aus der Gleichung } \vec{a} = l\,\vec{l} \text{ eindeutig bestimmt,} \qquad (\#)$$

$$l_{ii} \text{ ist aus der Gleichung } a_{ii} = \vec{l}^T\vec{l} + l_{ii}^2 \text{ bestimmt.} \qquad (\#\#)$$

Nach Aufgabe 26 gilt

$$a_{ii} > \langle \vec{a}, a^{-1}\vec{a}\,\rangle = \langle l\,\vec{l}, (l\,l^T)^{-1} l\,\vec{l}\,\rangle = \langle \vec{l}, \vec{l}\,\rangle = \vec{l}^T\vec{l} \;\Longrightarrow\; l_{ii}^2 = a_{ii} - \vec{l}^T\vec{l} > 0\ ,$$

womit l_{ii} reell ist. Wegen $l_{ii} \neq 0$ verschwindet kein Diagonalelement, womit L invertierbar ist.

Q.E.D.

Der Cholesky-Faktor $L = (l_{ij})$ wird durch den folgenden Algorithmus geliefert:

Algorithmus 4.28: (Cholesky-Zerlegung)

for $i := 1$ **to** n **do begin**
for $j := 1$ **to** $i-1$ **do** $\displaystyle l_{ij} := \Big(a_{ij} - \sum_{k=1}^{j-1} l_{ik}\, l_{jk} \Big) \,/\, l_{jj}\ ;$ (∗1∗)
$\displaystyle l_{ii} := \sqrt{a_{ii} - \sum_{k=1}^{i-1} l_{ik}^2}\ ;$ (∗2∗)
for $j := i+1$ **to** n **do** $l_{ij} := 0\ ;$ (∗3∗)
end;

Erklärung: In Zeile (∗1∗) wird das Gleichungssystem (#), d.h.,

$$\begin{pmatrix} a_{i1} \\ \vdots \\ a_{i,i-1} \end{pmatrix} = \begin{pmatrix} l_{11} & & 0 \\ \vdots & \ddots & \\ l_{i-1,1} & \dots & l_{i-1,i-1} \end{pmatrix} \begin{pmatrix} l_{i1} \\ \vdots \\ l_{i,i-1} \end{pmatrix}$$

durch Rücksubstitution gelöst. Zeile (∗2∗) bestimmt l_{ii} aus (##):

$$a_{ii} = l_{i1}^2 + \dots + l_{i,i-1}^2 + l_{ii}^2\ .$$

Bemerkung 4.29: *Es wird nur auf den unteren Block a_{ij} mit $j \le i$ zugegriffen. Man kann $A = (a_{ij})$ mit L überschreiben:*

$$\textbf{for } j := 1 \textbf{ to } i-1 \textbf{ do } \ a_{ij} := \Big(a_{ij} - \sum_{k=1}^{j-1} a_{ik}\, a_{jk} \Big) \,/\, a_{jj}\ ; \quad (*1*) \quad (\textit{usw.})$$

Das explizite Setzen (∗3∗) des oberen Dreiecksanteils wird in der Praxis natürlich vermieden.

Bemerkung 4.30: *Der Rechenaufwand ist mit etwa $n^3/6$ Multiplikationen (und n Wurzeln) etwa halb so groß wie bei der Gauß-Elimination/LR-Zerlegung. Der Grund ist die Symmetrie der Matrix.*

Bemerkung 4.31: *Die Zerlegung $A = LL^T = \tilde{L}\tilde{L}^T$ ist bis auf Vorzeichen eindeutig:*

$$\underbrace{\tilde{L}^{-1}\tilde{L}^T}_{\text{untere Dreiecksmatrix}} = \underbrace{\tilde{L}^T (L^T)^{-1}}_{\text{obere Dreiecksmatrix}} = \text{Diagonalmatrix } D\ .$$

Mit $L = \tilde{L}D$ folgt $LL^T = \tilde{L}\tilde{L}^T = LD^2L^T$, also $D^2 = \mathbb{1}$:

$$L = \tilde{L} \begin{pmatrix} \pm 1 & & 0 \\ & \ddots & \\ 0 & & \pm 1 \end{pmatrix} .$$

Der Algorithmus 4.28 bestimmt L so, daß die Diagonalelemente l_{ii} positiv sind.

Bemerkung 4.32: *Die Cholesky-Zerlegung ist eine Variante der LR-Zerlegung. Mit D = Diagonale von L gilt*

$$A = LL^T = \underbrace{(LD^{-1})}_{\begin{pmatrix} 1 & & 0 \\ * & \ddots & \\ * & * & 1 \end{pmatrix}} \underbrace{(DL^T)}_{\begin{pmatrix} * & \dots & * \\ & \ddots & \vdots \\ 0 & & * \end{pmatrix}} = LR - \text{Zerlegung} .$$

Hiermit ist gezeigt, daß s.p.d. Matrizen eine LR-Zerlegung $A = LR$ ohne Zeilenvertauschungen besitzen. Bei vorliegender Cholesky-Zerlegung ist jedes Gleichungssystem $A\vec{x} = LL^T\vec{x} = \vec{b}$ durch 2 Rücksubstitutionen

$$\begin{aligned} & \text{bestimme } \vec{y} \text{ aus } L\vec{y} = \vec{b}, \\ \text{dann} \quad & \text{bestimme } \vec{x} \text{ aus } L^T\vec{x} = \vec{y} \end{aligned}$$

schnell gelöst.

Bemerkung 4.33: *Ist A eine s.p.d. Bandmatrix, so hat der Cholesky-Faktor L genauso viele untere Bänder wie A (siehe Satz 4.20 und Bemerkung 4.32). Der Rechenaufwand reduziert sich hierdurch erheblich.*

Bemerkung 4.34: *Die LR-Zerlegung kann ohne Pivotierung unter numerischer Instabilität leiden (vergleiche Aufgabe 25). Die Cholesky-Zerlegung ist stabil (Bemerkung 4.67).*

Bemerkung 4.35: *A ist genau dann s.p.d., wenn $A = B^TB$ mit irgendeiner $m \times n$-Matrix B mit $\mathrm{kern}(B) = \{0\}$:*

$$A = B^TB \quad \Longrightarrow \quad \langle \vec{x}, B^TB\vec{x} \rangle = \langle B\vec{x}, B\vec{x} \rangle > 0 \quad \forall \vec{x} \neq 0 .$$

Bemerkung 4.36: *Typische Anwendung in der Approximationstheorie: "löse" ein überbestimmtes Gleichungssystem $B\vec{x} = \vec{c}$ mit einer $m \times n$-Matrix B, $m \geq n$ (d.h., mehr Gleichungen als Unbekannte). Eine Näherungslösung, die den Fehler $\langle B\vec{x} - \vec{c}, B\vec{x} - \vec{c} \rangle$ minimiert, ist durch*

$$\underbrace{B^T B}_{A} \vec{x} = \underbrace{B^T \vec{c}}_{\vec{b}}$$

bestimmt, wobei $A = B^T B$ s.p.d. ist, falls $\mathrm{kern}(B) = \{0\}$ *gilt.*

4.10 Invertierung von Matrizen

Ziel: explizite Invertierung von A. Löse dazu

$$A\vec{x}_i = \vec{e}_i \ , \quad i = 1, \ldots, n \tag{\#}$$

mit der Standardbasis 4.7 des $\mathbb{R}^n$. Dann gilt

$$A(\vec{x}_1, \ldots, \vec{x}_n) = (A\vec{x}_1, \ldots, A\vec{x}_n) = (\vec{e}_1, \ldots, \vec{e}_n) = \mathbb{1} \ ,$$

d.h., $\vec{x}_i$ ist die i.te Spalte der gesuchten Inversen $A^{-1} = (\vec{x}_1, \ldots, \vec{x}_n)$. Jeder Gleichungslöser kann damit benutzt werden, um die Inverse spaltenweise zu bestimmen. Aufwand beim Gauß-Algorithmus:

- Bestimme eine LR-Zerlegung $PA = LR$ mit $\approx n^3/3$ Multiplikationen.
- Löse die n Gleichungen $A\vec{x}_i = \vec{e}_i$ durch Rücksubstitution aus $LR\vec{x}_i = P\vec{e}_i$ mit jeweils $\approx n^2$ Multiplikationen: insgesamt $\approx n^3$ Multiplikationen.

Damit ist die Inverse $A^{-1} = (\vec{x}_1, \ldots, \vec{x}_n)$ mit insgesamt $\approx 4n^3/3$ Multiplikationen (4-facher Aufwand einer LR-Zerlegung) bestimmt.

Bemerkung 4.37: *Bei Matrizen mit wenigen Bändern können LR-Daten in der Laufzeit $O(n)$ berechnet werden: siehe etwa Beispiel 4.21. Zur Bestimmung der n^2 Komponenten von A^{-1} (i.a. vollbesetzt) braucht man mindestens $O(n^2)$ Operationen. Daher sollte man –zumindest bei großen dünnbesetzten Matrizen– die explizite Berechnung der Inversen vermeiden.*

4.11 QR-Faktorisierung

Idee: finde eine Zerlegung $A = QR$ mit orthogonalem Q (d.h., $Q^{-1} = Q^T$) und oberer Dreiecksmatrix R. Damit ist dann $A\vec{x} = QR\vec{x} = \vec{b}$ äquivalent zu $R\vec{x} = Q^T\vec{b}$ und mit $O(n^2)$ Operationen lösbar.

Satz 4.38: (QR-Faktorisierung)

Für jede relle quadratische Matrix A existiert eine Zerlegung $A = QR$ mit orthogonalem Q und einer oberen Dreiecksmatrix R. Für invertierbares A sind die Faktoren bis auf Vorzeichen eindeutig:

$$QR = \tilde{Q}\tilde{R} \quad \Longrightarrow \quad \tilde{Q} = QD \ , \ \tilde{R} = DR \quad \textit{mit} \quad D = \mathrm{diag}(\pm 1, \pm 1, \ldots) \ .$$

Beweis: Zur Existenz: algorithmische Konstruktion 4.44 von Q und R folgt. Zur Eindeutigkeit: mit A ist auch R invertierbar, also

$$QR = \tilde{Q}\tilde{R} \quad \Longleftrightarrow \quad \underbrace{R\tilde{R}^{-1}}_{\substack{\text{obere Dreiecks-}\\ \text{matrix}}} \quad = \quad \underbrace{Q^{-1}\tilde{Q}}_{\text{orthogonal}} \quad =: D \ .$$

Orthogonale obere Dreiecksmatrizen sind diagonal:

$$\underbrace{D^{-1}}_{\substack{\text{obere Dreiecks-}\\ \text{matrix}}} \quad \stackrel{\text{(orthogonal)}}{=} \quad \underbrace{D^{T}}_{\substack{\text{untere Dreiecks-}\\ \text{matrix}}} \quad .$$

Es folgt $D^{-1} = D^T =$ Diagonalmatrix D mit $D^2 = \mathbb{1}$.

Q.E.D.

Definition 4.39:

Die einem $\vec{v} \in \mathbb{R}^n$ **zugeordnete Householder-Matrix** *ist*

$$H \ = \ \begin{cases} \mathbb{1} - \dfrac{2}{\vec{v}^T\vec{v}}\, \vec{v}\,\vec{v}^T & , \textit{falls } \vec{v} \neq 0 \ , \\ \mathbb{1} & , \textit{falls } \vec{v} = 0 \ . \end{cases}$$

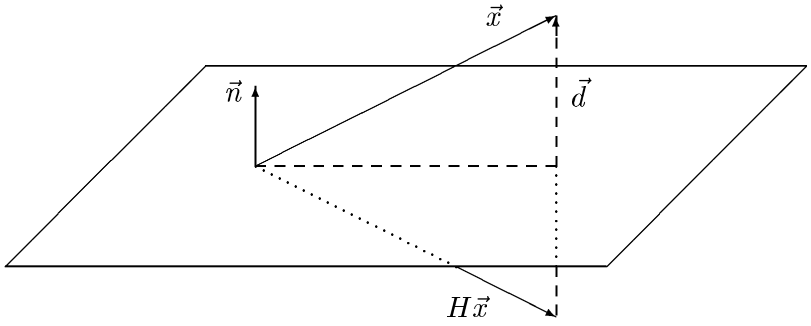
Bemerkung 4.40: *Für Householder-Matrizen gilt:*

a) *Symmetrie:* $H = H^T$,

b) $H^2 = \mathbb{1} - 4\, \frac{1}{\vec{v}^T\vec{v}}\, \vec{v}\,\vec{v}^T + 4\, \frac{1}{\vec{v}^T\vec{v}}\, \vec{v}\,\vec{v}^T\, \vec{v}\,\vec{v}^T \frac{1}{\vec{v}^T\vec{v}} = \mathbb{1}$,

c) *Orthogonalität:* $H^2 = HH^T = \mathbb{1}$,

d) *die geometrische Interpretation:* $\vec{x} \to H\vec{x}$ *ist die Spiegelung an der Ebene durch den Nullpunkt mit dem Normaleneinheitsvektor* $\vec{n} = \vec{v}/\sqrt{\vec{v}^T\vec{v}}$*:*



Der Spiegelpunkt ist $H\vec{x} = \vec{x} - 2\,\vec{d}\,,\quad \vec{d} = \vec{n}\,\langle\vec{n},\vec{x}\rangle = \vec{v}\,\dfrac{\vec{v}^{\,T}\vec{x}}{\vec{v}^{\,T}\vec{v}}\,.$

Definition 4.41:

18.12.97↓ *Zu* $j \in \{1,\ldots,n\}$ *und* $\vec{a} = (a_1,\ldots,a_n)^T \in \mathbb{R}^n$ *definiere die* **von** $\vec{a}$ **erzeugte Householder-Matrix** $H_{\vec{a}}^{(j)}$*:*

$$t_j := \operatorname{sign}(a_j)\,\sqrt{a_j^2 + a_{j+1}^2 + \cdots + a_n^2}\ ,$$

$$\vec{v}_j := (0,\ldots,0,t_j + a_j, a_{j+1},\ldots,a_n)^T\ ,$$

$$H_{\vec{a}}^{(j)} := \begin{cases} \mathbb{1} - \dfrac{2}{\vec{v}_j^{\,T}\vec{v}_j}\,\vec{v}_j\,\vec{v}_j^{\,T} & ,\ \textit{falls}\ \vec{v}_j \neq 0\ , \\ \mathbb{1} & ,\ \textit{falls}\ \vec{v}_j = 0\ . \end{cases}$$

Vereinbarung: $\operatorname{sign}(0) = 1$.

Lemma 4.42:

Für $H = H_{\vec{a}}^{(j)}$ *gilt:* $H\vec{a} = (a_1,\ldots,a_{j-1},-t_j,0,\ldots,0)^T$ *und* $H\vec{b} = \vec{b}$ *für alle Vektoren der Form* $\vec{b} = (b_1,\ldots,b_{j-1},0,\ldots,0)^T$.

Beweis: Mit $t_j^2 = a_j^2 + \cdots + a_n^2$ folgt

$$\frac{2\vec{v}_j^{\,T}\vec{a}}{\vec{v}_j^{\,T}\vec{v}_j} = \frac{2\,(\ (t_j + a_j)\,a_j + a_{j+1}^2 + \cdots + a_n^2\ )}{(t_j + a_j)^2 + a_{j+1}^2 + \cdots + a_n^2} = \frac{2\,(t_j a_j + t_j^2)}{t_j^2 + 2\,t_j\,a_j + t_j^2} = 1$$

und

$$H\vec{a} = \vec{a} - \vec{v}_j\,\frac{2\vec{v}_j^{\,T}\vec{a}}{\vec{v}_j^{\,T}\vec{v}_j} = \vec{a} - \vec{v}_j = \begin{pmatrix} a_1 \\ \vdots \\ a_{j-1} \\ a_j \\ a_{j+1} \\ \vdots \\ a_n \end{pmatrix} - \begin{pmatrix} 0 \\ \vdots \\ 0 \\ t_j + a_j \\ a_{j+1} \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} a_1 \\ \vdots \\ a_{j-1} \\ -t_j \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

Für $\vec{a} = (a_1, \ldots, a_{j-1}, 0, \ldots, 0)^T$ (der einzige Fall mit $H = \mathbb{1}$) gilt die Aussage ebenfalls (beachte $t_j = 0$). Für $\vec{b} = (b_1, \ldots, b_{j-1}, 0, \ldots, 0)^T$ gilt $\vec{v}_j^T \vec{b} = 0$, also $H\vec{b} = \vec{b}$.

Q.E.D.

Bemerkung 4.43: *Lemma 4.42 gibt die später benötigten Eigenschaften. Es wurde nur $t_j^2 = a_j^2 + \cdots + a_n^2$ benutzt, womit das Vorzeichen von t_j beliebig wählbar ist. Das Vorzeichen in Definition 4.41 dient zur Vermeidung von Auslöschung in $\vec{v}_j$. Für $a_j = 0$ setze entweder $\mathrm{sign}(0) = 1$ oder $\mathrm{sign}(0) = -1$ (die oft übliche Vereinbarung $\mathrm{sign}(0) = 0$ führt hier zu falschen Ergebnissen).*

Konstruktion 4.44:

QR-Zerlegung durch Householder-Transformationen:

Start: $A = (\vec{a}_1, \ldots, \vec{a}_n)$ *(Zerlegung nach Spalten).*

Schritt 1: *Sei $H^{(1)} = H^{(1)}_{\vec{a}_1}$ die von $\vec{a}_1$ erzeugte Householder-Matrix:*

$$H^{(1)} A = \left(\begin{pmatrix} -t_1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, H^{(1)}\vec{a}_2, \ldots, H^{(1)}\vec{a}_n \right) .$$

Schritt 2: *Sei $H^{(2)}$ die von $H^{(1)}\vec{a}_2$ erzeugte Householder-Matrix:*

$$H^{(2)} H^{(1)} A = \left(\begin{pmatrix} -t_1 \\ 0 \\ \vdots \\ \vdots \\ 0 \end{pmatrix}, \begin{pmatrix} * \\ -t_2 \\ 0 \\ \vdots \\ 0 \end{pmatrix}, H^{(2)}H^{(1)}\vec{a}_3, \ldots, H^{(2)}H^{(1)}\vec{a}_n \right) .$$

Schritt j: *Sei $H^{(j)}$ die von der j.ten Spalte $H^{(j-1)} \cdots H^{(1)}\vec{a}_j$ des Schemas erzeugte Householder-Matrix. Durch Multiplikation mit $H^{(j)}$ werden nach Lemma 4.42 in der j.ten Spalte die Elemente unter der Diagonalen eliminiert, währen die ersten $j-1$ Spalten nicht verändert werden:*

$$H^{(j)} \cdots H^{(1)} A =$$

$$\left(\begin{pmatrix} -t_1 \\ 0 \\ \vdots \\ \vdots \\ \vdots \\ 0 \end{pmatrix}, \ldots, \begin{pmatrix} * \\ \vdots \\ -t_j \\ 0 \\ \vdots \\ 0 \end{pmatrix}, H^{(j)} \cdots H^{(1)} \vec{a}_{j+1}, \ldots, H^{(j)} \cdots H^{(1)} \vec{a}_n\right).$$

Ergebnis: *Nach $n-1$ Schritten ist obere Dreiecksform erreicht:*

$$R = \underbrace{H^{(n-1)} \cdots H^{(1)}}_{\textit{orthogonal}} A = \begin{pmatrix} -t_1 & r_{12} & \ldots & \ldots \\ & -t_2 & r_{23} & \ddots \\ & & \ddots & \ddots \end{pmatrix}.$$

Der Q-Faktor ergibt sich durch $Q^T = H^{(n-1)} \cdots H^{(1)}$ bzw. $Q = H^{(1)} \cdots H^{(n-1)}$. Dieses Produkt läßt sich durch Ausführung an der Einheitsmatrix speichern.

Transformiere $A\vec{x} = \vec{b}$ auf $R\vec{x} = Q^T\vec{b}$, berechne Q^T:

Start:	A	$\vec{b}$	$\mathbb{1}$
Householder-	$H^{(1)} A$	$H^{(1)} \vec{b}$	$H^{(1)} \mathbb{1}$
Transformationen:	$H^{(2)} H^{(1)} A$	$H^{(2)} H^{(1)} \vec{b}$	$H^{(2)} H^{(1)} \mathbb{1}$
	$\vdots$	$\vdots$	$\vdots$
Ende:	$\underbrace{H^{(n-1)} \cdots H^{(1)} A}_{R}$	$\underbrace{H^{(n-1)} \cdots H^{(1)} \vec{b}}_{Q^T\vec{b}}$	$\underbrace{H^{(n-1)} \cdots H^{(1)} \mathbb{1}}_{Q^T}$

Damit sind die Daten des zu $A\vec{x} = \vec{b}$ äquivalenten oberen Dreieckssystems $R\vec{x} = Q^T\vec{b}$ berechnet, R und Q^T (falls benötigt), liegen explizit vor [1].

Bemerkung 4.45: *Die Matrix-Vektor-Multiplikation*

$$H^{(j)} \vec{s} = \vec{s} - 2\, \vec{v}_j \, \frac{\langle \vec{v}_j, \vec{s} \rangle}{\langle \vec{v}_j, \vec{v}_j \rangle}$$

auf die Spalten $\vec{s}$ des Schemas braucht nur $\approx 3\,(n-j)$ Multiplikationen, so daß jede einzelne H-Transformation des Gesamtschemas mit $O(n^2)$ Operationen ausführbar ist.

[1] Man verzichtet jedoch i.a. auf die explizite Berechnung von Q^T. Das Speichern der Daten $\vec{v}_j$ der $H^{(j)}$ im frei werdenden unteren Teil des Schemas ist praktisch umsonst. Mit diesen Daten lassen sich Multiplikationen mit Q^T schnell durchführen (siehe Bemerkung 4.47).

Algorithmus 4.46: (QR-Zerlegung)

Erklärung der in Tafel 4.1 angegebenen Implementierung: Zunächst wird A um $\vec{b}$ zum $n \times (n+1)$-Gesamtschema $[A \,|\, \vec{b}\,]$ erweitert (Schritt $(*1*)$). Der Householder-Konstruktion folgend wird die der j.ten Spalte des Schemas entsprechende Householder-Matrix $H^{(j)}$ konstruiert. Mit dem in $(*2*)$ berechneten Wert t_j ist der $H^{(j)} = \mathbb{1} - 2\,\vec{v}_j\,\vec{v}_j^{\,T} / \langle \vec{v}_j, \vec{v}_j \rangle$ definierende Vektor $\vec{v}_j$ durch

$$\vec{v}_j \; = \; (0, \ldots, 0, t_j + a_{jj}, a_{j+1,j}, \ldots, a_{nj})^T$$

gegeben. Der Fall $t_j = 0$ entspricht $a_{jj} = \ldots = a_{nj} = 0$, in dem mit $H^{(j)} = \mathbb{1}$ das Schema unverändert bleibt (Zeile $(*3*)$). Dieser Fall tritt dann und nur dann ein, wenn die Matrix nicht invertierbar ist[2]*.*

Die Transformation

$$H^{(j)} \vec{s} \; = \; \vec{s} \; - \; e\,\vec{v}_j \;, \quad e \; = \; d\,\langle \vec{v}_j, \vec{s}\,\rangle \;, \quad d \; = \; \frac{2}{\langle \vec{v}_j, \vec{v}_j \rangle} \qquad (\#)$$

ist für alle Spalten $\vec{s}$ des Schemas zu berechnen. Der gemeinsame Faktor d ergibt sich durch

$$\begin{aligned} d \; &= \; \frac{2}{\langle \vec{v}_j, \vec{v}_j \rangle} \; = \; \frac{2}{(t_j + a_{jj})^2 \; + \; a_{j+1,j}^2 \; + \; \cdots \; + \; a_{nj}^2} \\ &= \; \frac{2}{t_j^2 \; + \; 2\,a_{jj}\,t_j \; + \; a_{jj}^2 \; + \; \cdots \; + \; a_{nj}^2} \; = \; \frac{1}{(t_j + a_{jj})\,t_j} \end{aligned}$$

und die Skalarprodukte mit $\vec{s} = (s_1, \ldots, s_n)^T$ mittels

$$\langle \vec{v}_j, \vec{s}\,\rangle = (t_j + a_{jj})\,s_j + a_{j+1,j}s_{j+1} + \cdots + a_{nj}s_n .$$

Daher ist es sinnvoll, den Wert $t_j + a_{jj}$ als a_{jj} zu speichern (Zeile $(*4*)$), womit die j.te Spalte die Daten des Vektors

$$(0, \ldots, 0, a_{jj}, \ldots, a_{nj})^T \; = \; \vec{v}_j$$

enthält. Der Faktor d ergibt sich damit in der Form $(*5*)$. Man beachte, daß t_j dasselbe Vorzeichen wie das ursprüngliche a_{jj} hat, womit $t_j\,(t_j + a_{jj}) \neq 0$ für $t_j \neq 0$ gilt.

Die ersten $j-1$ Spalten des Schemas werden von der Householder-Matrix invariant gelassen, so daß nur die restlichen Spalten $k = j, \ldots, n+1$ des Gesamtschemas transformiert zu werden brauchen. Mit der Schleife $(*6*)$ werden die Spalten $k = j+1, \ldots, n+1$ mittels $(*7*)$ und $(*8*)$ gemäß $(\#)$ transformiert.

---

[2]Dies ist klar, da die gerade bearbeitete Spalte in den später folgenden Transformationen nicht mehr verändert wird und der Wert $-t_j$ somit das j.te Diagonalelement der zuletzt erzeugten oberen Dreiecksmatrix R ist. Die Invertierbarkeit von R (also das Nichtverschwinden der Diagonalelemente) ist aber äquivalent zur Invertierbarkeit von A.

(* **Erweiterung:** *)
for $i := 1$ **to** n **do** $a_{i,n+1} := b_i$; (*1*)

(* **Elimination:** *)
for $j := 1$ **to** $n-1$ **do begin**

$t_j := \operatorname{sign}(a_{jj}) \sqrt{\sum_{i=j}^{n} a_{ij}^2}$; (* **sign(0)=1 !!** *) (*2*)

Falls $t_j = 0$ **, so gehe zum** $(j+1)$**.ten Schritt** ; (*3*)

$a_{jj} := t_j + a_{jj}$; (*4*)

$d := 1/(t_j\, a_{jj})$; (*5*)

for $k := j+1$ **to** $n+1$ **do begin** (*6*)

$e := d \left(\sum_{i=j}^{n} a_{ij} a_{ik} \right)$; (*7*)

for $i := j$ **to** n **do** $a_{ik} := a_{ik} - e\, a_{ij}$; (*8*)

end ;

end ;

Tafel 4.1: Transformation von $A\vec{x} = \vec{b}$ auf $R\vec{x} = Q^T\vec{b}$.

Zuletzt müßte noch die j.te Spalte transformiert werden, wobei nach Lemma 4.42 nur das Diagonalelement auf den neuen Wert $-t_j$ abzuändern wäre. Da dieses Datum aber im Vektor $\vec{t} = (t_1, t_2, \ldots)$ gespeichert ist, wird hierauf verzichtet.

Nach Ablauf des Schrittes $j = n-1$ liegen neben den Werten $t_1, \ldots, t_{n-1}$ im Gesamtschema die neuen Daten

$$\left(\begin{array}{ccc|c} a_{11} & \ldots & a_{1n} & a_{1,n+1} \\ \vdots & \ddots & \vdots & \vdots \\ \vdots & \ddots & \vdots & \vdots \\ a_{n1} & \ldots & a_{nn} & a_{n,n+1} \end{array}\right) = \left(\begin{array}{cccc|c} v_{11} & r_{12} & \ldots & r_{1n} & c_1 \\ \vdots & \ddots & \ddots & \vdots & \vdots \\ \vdots & \ddots & v_{n-1,n-1} & r_{n-1,n} & \vdots \\ v_{n1} & \ldots & v_{n,n-1} & r_{nn} & c_n \end{array}\right)$$

vor. Die transformierte rechte Seite $\vec{c} = Q^T\vec{b}$ ist in der $(n+1)$.ten Spalte gespeichert, der R-Faktor ergibt sich zu

$$R = \begin{pmatrix} -t_1 & r_{12} & \dots & r_{1n} \\ & \ddots & \ddots & \vdots \\ & & -t_{n-1} & r_{n-1,n} \\ & & & r_{nn} \end{pmatrix}.$$

Die Daten $\vec{v}_j = (0,\dots,0,v_{jj},\dots,v_{nj})^T$, $j = 1,\dots,n-1$, der Householder-Matrizen $H^{(j)}$ sind im unteren Teil des Schemas gespeichert, aus denen sich Q^T bzw. Q bei Bedarf leicht konstruieren lassen (siehe Bemerkung 4.47).

Bemerkung 4.47: *Man wird i.a. darauf verzichten, Q^T bzw. Q explizit zu berechnen. Mit den Daten der Vektoren $\vec{v}_j = (0,\dots,0,a_{jj},\dots,a_{nj})^T$ im unteren Dreiecksanteil des Schemas kann nämlich eine Matrix-Vektor-Multiplikation mit Q^T bzw. Q schnell durchgeführt werden. Die Multiplikation $\vec{x} \to Q^T\vec{x} = H^{(n-1)} \cdots H^{(1)}\vec{x}$ eines Vektors $\vec{x} = (x_i)$ ist in Analogie zu den Schritten $(*3*)$–$(*8*)$ nach*

> (* **Berechne** $\vec{x} := Q^T\vec{x}$ **mittels der Daten** $t_j, \vec{v}_j$ **:** *)
> **for** $j := 1$ **to** $n-1$ **do begin**
>   **Falls** $t_j = 0$**, so gehe zum** $(j+1)$**.ten Schritt** ;
> $e := \dfrac{1}{t_j\, a_{jj}} \Big(\sum_{i=j}^{n} a_{ij}\, x_i \Big)$;
>   **for** $i := j$ **to** n **do** $x_i := x_i - e\, a_{ij}$;
> **end** ;

im transformierten Vektor $\vec{x}$ gespeichert. Mit $\approx n^2$ elementaren Multiplikationen läßt sich $Q^T\vec{x}$ so praktisch genauso schnell berechnen wie durch die Matrix-Vektor-Multiplikation $Q^T\vec{x}$ bei explizit vorliegendem Q^T. Bei Bandstrukturen ist dies Verfahren sogar erheblich günstiger, da der untere Dreiecksteil von Q^T voll besetzt ist, während die untere Dreiecksmatrix der Daten $(\vec{v}_1, \vec{v}_2, \dots)$ die Bandstruktur von A erbt.

Mit $Q\vec{x} = H^{(1)} \dots H^{(n-1)}\vec{x}$ ergibt sich eine Multiplikation mit Q aus den Daten $t_j, \vec{v}_j$, indem man einfach die j-Schleife rückwärts durchläuft:

(* **Berechne** $\vec{x} := Q\vec{x}$ **mittels der Daten** $t_j, \vec{v}_j$ **:** *)
for $j := n-1$ **downto** 1 **do begin**
  Falls $t_j = 0$ **, so gehe zum** $(j+1)$**.ten Schritt** ;
  $e := \dfrac{1}{t_j\, a_{jj}} \Big(\sum\limits_{i=j}^{n} a_{ij} b_i \Big)$;
  for $i := j$ **to** n **do** $x_i := x_i - e\, a_{ij}$;
end ;

Wendet man diese Transformationen auf die Standardbasis $\vec{e}_1, \ldots, \vec{e}_n$ des $\mathbb{R}^n$ an, so ergeben sich Q^T bzw. Q mit einem Aufwand von jeweils $\approx n^3$ elementaren Multiplikationen aus den Daten $t_j, \vec{v}_j$.

Bemerkung 4.48: *Die Transformation $A\vec{x} = \vec{b} \;\Rightarrow\; R\vec{x} = Q^T\vec{b}$ kostet $\approx 2\,n^3/3$ Multiplikationen (entspricht dem LR-Aufwand mit Spaltenpivotierung 4.22), wobei die Q^T bzw. Q definierenden Daten $\vec{v}_j$ ohne Zusatzkosten geliefert werden. Die explizite Berechnung von Q^T bzw. Q (falls benötigt) kostet <u>zusätzlich</u> $\approx n^3$ Multiplikationen.*

Bemerkung 4.49: *Der Aufwand reduziert sich bei Bandmatrizen. Für eine Matrix mit p unteren und q oberen Bändern hat der Q-Faktor p untere und der R-Faktor $p+q$ obere Bänder:*

$$
\begin{array}{cc}
 & \overbrace{\hphantom{*\quad *\quad \cdots\quad *}}^{q} \\
p\left\{\begin{array}{c} \\ \\ \\ \end{array}\right. &
\begin{pmatrix}
* & * & \cdots & * & & \\
* & \ddots & \ddots & \ddots & \ddots & \\
\vdots & \ddots & \ddots & \ddots & \ddots & * \\
* & \ddots & \ddots & \ddots & \ddots & \vdots \\
 & \ddots & \ddots & \ddots & \ddots & * \\
 & & * & \cdots & * & *
\end{pmatrix}
\end{array}
\; = \; QR \; =
$$

$$
\begin{array}{cc}
p\left\{\begin{array}{c} \\ \\ \\ \end{array}\right. &
\begin{pmatrix}
* & \cdots & \cdots & \cdots & \cdots & * \\
* & \ddots & \ddots & \ddots & \ddots & \vdots \\
\vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\
* & \ddots & \ddots & \ddots & \ddots & \vdots \\
 & \ddots & \ddots & \ddots & \ddots & \vdots \\
 & & * & \cdots & * & *
\end{pmatrix}
\end{array}
\begin{array}{c}
\overbrace{\hphantom{*\quad *\quad \cdots\quad *}}^{p+q} \\
\begin{pmatrix}
* & * & \cdots & * & & \\
 & \ddots & \ddots & \ddots & \ddots & \\
 & & \ddots & \ddots & \ddots & * \\
 & & & \ddots & \ddots & \vdots \\
 & & & & \ddots & * \\
 & & & & & *
\end{pmatrix}
\end{array}
\; .
$$

Dies kann man durch Induktion nach der Schleifenvariable j der Householder-Konstruktion 4.44 beweisen. Die Zwischenschemata $H^{(j-1)} \cdots H^{(1)} A$ zur Berechnung von R sind von der Form

$$
\begin{array}{ccccccccccccccccc}
1 & & j-1 & j & & & j+p & j+p+1 & & & p+q & & & j+p+q-1 & & & n \\
\downarrow & & \downarrow & \downarrow & & & \downarrow & \downarrow & & & \downarrow & & & \downarrow & & & \downarrow
\end{array}
$$

$$
\begin{array}{|ccc|cccc|ccccccc|ccc|l}
* & \cdots & * & * & \cdots & \cdots & * & * & \cdots & \cdots & * & & & & & & & \leftarrow 1 \\
 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & & & & & & \\
 & & * & * & \cdots & \cdots & * & * & \cdots & \cdots & \cdots & \cdots & * & & & & & \leftarrow j-1 \\
\hline
 & & & * & \cdots & \cdots & * & * & \cdots & \cdots & \cdots & \cdots & * & 0 & & & & \leftarrow j \\
 & & & \vdots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \vdots & \vdots & & & & \\
 & & & \vdots & \ddots & \ddots & \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \vdots & 0 & & & & \\
 & & & * & \cdots & \cdots & * & * & \cdots & \cdots & \cdots & \cdots & * & * & & & & \leftarrow j+p \\
\hline
 & & & & * & \cdots & * & * & \cdots & \cdots & \cdots & \cdots & \cdots & * & * & & & \leftarrow j+p+1 \\
 & & & & & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & & \\
 & & & & & & * & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & * & \leftarrow n-q+1 \\
 & & & & & & & * & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots & \\
 & & & & & & & & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots & \\
 & & & & & & & & & * & \cdots & \cdots & \cdots & * & * & \cdots & * & \leftarrow n
\end{array}
$$

Damit besitzt die nächste Householder-Matrix $H^{(j)}$ die Gestalt

$$
\begin{array}{ll}
H^{(j)} \; = \left(
\begin{array}{ccc}
\boxed{\mathbb{1}} & & \\
 & \boxed{\begin{array}{ccc} * & \cdots & * \\ \vdots & \ddots & \vdots \\ * & \cdots & * \end{array}} & \\
 & & \boxed{\mathbb{1}}
\end{array}
\right)
&
\begin{array}{l} \leftarrow j \\ \\ \leftarrow j+p \end{array}
\end{array}
$$

woraus der Induktionsschritt $j-1 \to j$ zum Beweis der Form von $H^{(j-1)} \cdots H^{(1)} A$ folgt. Für die Zwischenschemata zur Berechnung von Q^T gilt:

$$
\begin{array}{c}
\\[1ex]
H^{(j-1)}\cdots H^{(1)} \;=
\end{array}
\begin{array}{ccccc|cccc|ccc|l}
\overset{1}{\downarrow} & & \overset{p+1}{\downarrow} & & \overset{j-1}{\downarrow} & \overset{j}{\downarrow} & & & \overset{j+p}{\downarrow} & & & & \\
* & \ldots & * & & & & & & & & & & \leftarrow 1 \\
\vdots & \ddots & \ddots & \ddots & & & & & & & & & \\
\vdots & \ddots & \ddots & \ddots & * & & & & & & & & \\
\vdots & \ddots & \ddots & \ddots & \ddots & * & & & & & & & \\
\vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & & & & & & \\
* & \ldots & \ldots & \ldots & * & * & \ldots & * & 0 & & & & \leftarrow j-1 \\
\hline
* & \ldots & \ldots & \ldots & * & * & \ldots & * & 0 & & & & \leftarrow j \\
\vdots & \ddots & \ddots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & & & & \\
* & \ldots & \ldots & \ldots & * & * & \ldots & * & 0 & & & & \leftarrow j+p-1 \\
0 & \ldots & \ldots & \ldots & 0 & 0 & \ldots & 0 & 1 & & & & \leftarrow j+p \\
\hline
 & & & & & & & & & 1 & & & \leftarrow j+p+1 \\
 & & & & & & & & & & \ddots & & \\
 & & & & & & & & & & & 1 & \leftarrow n
\end{array}
$$

Bei Matrizen mit wenigen Bändern reduziert sich die Laufzeit der Transformation $A\vec{x} = \vec{b} \Rightarrow R\vec{x} = Q^T\vec{b}$ bei geeigneter Implementierung auf $O(n)$.

Bemerkung 4.50: *Die QR-Zerlegung mittels Householder-Transformationen ist numerisch sehr stabil (siehe Bemerkung 4.67).*

Zusammenfassung: Durch Multiplikation mit orthogonalem Q^T kann jede quadratische Matrix A auf obere Dreiecksform R transformiert werden. Der Aufwand zur Umformung $A\vec{x} = \vec{b} \;\Rightarrow\; R\vec{x} = Q^T\vec{b}$ entspricht dem einer LR-Zerlegung mit Spaltenpivotierung.

4.12 Normen und Fehlerkontrolle

6.1.98↓ Erinnerung an die Analysis:

Definition 4.51:

Eine Abbildung $\|\,.\,\|$: Vektorraum $\to [0,\infty)$ mit

a) $\|x\| \;=\; 0 \iff x = 0$,

b) $\|\lambda x\| \;=\; |\lambda|\,\|x\|$, λ *skalar,*

c) $\|x+y\| \;\le\; \|x\| + \|y\|$ *(Dreiecksungleichung)*

heißt **Norm**.

Beispiel 4.52: Die Standardnormen für $\vec{x} = (x_1, \ldots, x_n)^T \in \mathbb{R}^n$ bzw. $\mathbb{C}^n$ sind

$$\begin{array}{rcll} \|\vec{x}\|_1 &=& |x_1| + |x_2| + \cdots + |x_n| & \text{(Summennorm)} \\ \|\vec{x}\|_2 &=& \sqrt{|x_1|^2 + |x_2|^2 + \cdots + |x_n|^2} & \text{(euklidische Norm)} \\ \|\vec{x}\|_p &=& \sqrt[p]{|x_1|^p + |x_2|^p + \cdots + |x_n|^p} & \text{(p-Norm)} \\ \|\vec{x}\|_\infty &=& \max\{\,|x_1|, |x_2|, \ldots, |x_n|\,\} & \text{(Maximumsnorm)} . \end{array}$$

Satz 4.53:

Auf endlich dimensionalen Vektorräumen V sind alle Normen äquivalent: zu $\|\,.\,\|$ und $\|\!|\,.\,\|\!|$ existieren Konstanten c und C, so daß

$$\|x\| \leq c\,\|\!|x\|\!| \ , \quad \|\!|x\|\!| \leq C\,\|x\| \quad \forall\ x \in V \ .$$

Beweis: siehe Analysis.

Definition 4.54:

Die einer Vektornorm $\|\,.\,\|$ auf dem $\mathbb{R}^n$ bzw. $\mathbb{C}^n$ **zugeordnete** *Matrixnorm (***Operatornorm***) ist*

$$\|A\| \;=\; \max_{\vec{x}\neq 0} \frac{\|A\vec{x}\|}{\|\vec{x}\|} \;=\; \max_{\vec{x}\neq 0} \|A\frac{\vec{x}}{\|\vec{x}\|}\| \stackrel{(\vec{y}=\vec{x}/\|\vec{x}\|)}{=} \max_{\|\vec{y}\|=1} \|A\vec{y}\| \ .$$

Dies ist in der Tat eine Norm. Das Maximum existiert, da $\{\vec{y}\,;\,\|\vec{y}\| = 1\}$ kompakt und $\|A\,.\,\|$ stetig ist.

Bemerkung 4.55: *Für Operatornormen gilt*

a) **Verträglichkeit:** $\|A\vec{x}\| \leq \|A\|\,\|\vec{x}\|$,

b) **Submultiplikativität:** $\|AB\| \leq \|A\|\,\|B\|$

$$\Bigl(\|AB\| = \max_{\vec{x}\neq 0} \frac{\|AB\vec{x}\|}{\|\vec{x}\|} \stackrel{a)}{\leq} \|A\| \max_{\vec{x}\neq 0} \frac{\|B\vec{x}\|}{\|\vec{x}\|} \;=\; \|A\|\,\|B\|\Bigr) \ .$$

Im folgenden werden nur noch Operatornormen betrachtet.

Satz 4.56: (Identifizierung einiger Matrixnormen)

Die $\|\,.\,\|_1$, $\|\,.\,\|_2$ und $\|\,.\,\|_\infty$ auf dem $\mathbb{R}^n$ bzw. $\mathbb{C}^n$ zugeordneten Matrixnormen für $A = (a_{ij})$ sind

$$\|A\|_1 = \max_{\vec{x}\neq 0} \frac{\|A\vec{x}\|_1}{\|\vec{x}\|_1} = \max_{j=1\ldots n} \sum_{i=1}^{n} |a_{ij}| \quad \textbf{(Spaltensummennorm)}$$

$$\|A\|_2 = \max_{\vec{x}\neq 0} \frac{\|A\vec{x}\|_2}{\|\vec{x}\|_2} = \sqrt{\textit{größter Eigenwert von } A^T A} \quad \textbf{(Spektralnorm)}$$

$$\|A\|_\infty = \max_{\vec{x}\neq 0} \frac{\|A\vec{x}\|_\infty}{\|\vec{x}\|_\infty} = \max_{i=1\ldots n} \sum_{j=1}^{n} |a_{ij}| \quad \textbf{(Zeilensummennorm)}\ .$$

Anmerkung: es gilt $\|A^T\|_1 = \|A\|_\infty$, $\|A\|_2 = \|A^T\|_2$.

Beweis: siehe Analysis (Aufgabe 29.a) für $\|A\|_1$, Satz 6.3.a) für $\|A\|_2$, beachte $\|A\vec{x}\|_2^2/\|\vec{x}\|_2^2 = \langle A\vec{x}, A\vec{x}\rangle/\langle\vec{x},\vec{x}\rangle = \langle\vec{x}, A^T A\vec{x}\rangle/\langle\vec{x},\vec{x}\rangle =$ Rayleigh-Quotient von $A^T A$).

Definition 4.57:

$$\rho(A) = \max\ \{\ |\lambda|\ ;\ \lambda \in \mathbb{C} \textit{ ist Eigenwert von } A\ \}\ .$$

heißt **Spektralradius** *von A.*

Satz 4.58:

a) *Für jede Operatornorm gilt $\|A\| \geq \rho(A)$.*

b) *Für eine diagonalisierbare $n \times n$-Matrix A existiert eine Vektornorm $\|\,.\,\|_A$ auf dem $\mathbb{C}^n$ mit*

$$\|A\|_A := \max_{\vec{x}\neq 0} \frac{\|A\vec{x}\|_A}{\|\vec{x}\|_A} = \rho(A)\ .$$

c) *Für eine beliebige $n \times n$-Matrix A existiert für jedes $\epsilon > 0$ eine Vektornorm $\|\,.\,\|_{A,\epsilon}$ auf dem $\mathbb{C}^n$ mit*

$$\rho(A) \leq \|A\|_{A,\epsilon} := \max_{\vec{x}\neq 0} \frac{\|A\vec{x}\|_{A,\epsilon}}{\|\vec{x}\|_{A,\epsilon}} \leq \rho(A) + \epsilon\ .$$

Beweis: a) Mit $A\vec{x} = \lambda\vec{x}$ folgt $\|A\| \geq \dfrac{\|A\vec{x}\|}{\|\vec{x}\|} = |\lambda|$.
b) Sei $\vec{x}_1, \ldots, \vec{x}_n$ mit $A\vec{x}_i = \lambda_i\vec{x}_i$ eine Eigenvektorbasis. Zerlege $\vec{x} = \sum x_i\vec{x}_i$ und definiere $\|\vec{x}\|_A := \max_i |x_i|$. Es folgt

$$\max_{\vec{x}\neq 0} \frac{\|A\vec{x}\|_A}{\|\vec{x}\|_A} = \max_{\vec{x}\neq 0} \frac{\max_i |\lambda_i|\,|x_i|}{\max_i |x_i|} \leq \rho(A)$$

und mit a) auch $\|A\|_A \geq \rho(A)$.
c) wird analog zu b) per Jordan-Normalform bewiesen (siehe z.B. Satz 6.8.2 in [StB90]).

Q.E.D.

Merke: Der Spektralradius ist die größte untere Schranke aller Operatornormen.

Bemerkung 4.59: *Für symmetrische Matrizen ist*

$$\|A\|_2 = \sqrt{\rho(A^TA)} = \sqrt{\rho(A^2)} = \rho(A)$$

die beste (kleinste) aller Operatornormen.

Hilfssatz 4.60:

Gilt in einer Operatornorm $\|1\!\!1 - A\| < 1$, dann ist A invertierbar und

$$\frac{1}{\|A\|} \;\leq\; \|A^{-1}\| \;\leq\; \frac{1}{1 - \|1\!\!1 - A\|} \;.$$

Beweis: Für singuläres A ist $\lambda = 0$ Eigenwert. Damit hat $1\!\!1 - A$ den Eigenwert $\mu = 1$ im Widerspruch zu $|\mu| \leq \|1\!\!1 - A\| < 1$. Mit

$$1 = \|1\!\!1\| = \|A^{-1}A\| \leq \|A^{-1}\|\,\|A\|$$

und

$$\begin{aligned}\|A^{-1}\| = \|A^{-1}(1\!\!1 - A) + 1\!\!1\| &\leq \|A^{-1}\|\,\|1\!\!1 - A\| + \|1\!\!1\| \\ \implies \quad (1 - \|1\!\!1 - A\|)\,\|A^{-1}\| &\leq 1\end{aligned}$$

folgt die Behauptung.

Q.E.D.

Definition 4.61:

Einer invertierbaren Matrix wird die **Konditionszahl**

$$\mathrm{cond}(A) \;=\; \|A\|\,\|A^{-1}\|$$

zugeordnet. Schreibweise: $\mathrm{cond}_p(A)$ bei Verwendung von p-Normen.

Satz 4.62:

Mit den Eigenwerten λ_i von A gilt

$$\mathrm{cond}(A) \;=\; \frac{\max\limits_{\|\vec{x}\|=1} \|A\vec{x}\|}{\min\limits_{\|\vec{x}\|=1} \|A\vec{x}\|} \;\geq\; \frac{\max_i |\lambda_i|}{\min_i |\lambda_i|} \;\geq\; 1 \;.$$

Für symmetrisches A gilt $\mathrm{cond}_2(A) = \max_i |\lambda_i| / \min_i |\lambda_i|$.

Beweis: Es gilt $\|A\| = \max\limits_{\|\vec{x}\|=1} \|A\vec{x}\| \geq \max_i |\lambda_i|$ und

$$\|A^{-1}\| \;=\; \max_{\vec{x}\neq 0} \frac{\|A^{-1}\vec{x}\|}{\|\vec{x}\|} \overset{(\vec{x}=A\vec{y})}{=} \max_{\vec{y}\neq 0} \frac{\|\vec{y}\|}{\|A\vec{y}\|} \;=\; \frac{1}{\min\limits_{\vec{y}\neq 0} \frac{\|A\vec{y}\|}{\|\vec{y}\|}}$$

$$\overset{(\vec{x}=\vec{y}/\|\vec{y}\|)}{=} \frac{1}{\min\limits_{\|\vec{x}\|=1} \|A\vec{x}\|} \;\geq\; \frac{1}{\min\limits_i |\lambda_i|} \;.$$

Für symmetrisches A gilt $\mathrm{cond}_2(A) \overset{(4.59)}{=} \rho(A)\,\rho(A^{-1}) = (\max_i |\lambda_i|)(\max_i |\lambda_i^{-1}|)$.

Q.E.D.

Der Einfluß von Rundungsfehlern in der Lösung von $A\vec{x} = \vec{b}$ ist a priori nur schwierig abzuschätzen. Man kann jedoch a posteriori die Qualität einer vorliegenden numerischen Approximation von $A^{-1}\vec{b}$ leicht und kostengünstig kontrollieren:

Definition 4.63:

8.1.98↓ *Einem Gleichungssystem $A\vec{x} = \vec{b}$ und einem beliebigen Vektor $\vec{x}$ wird das* **Residuum** $\vec{r}(\vec{x}) = A\vec{x} - \vec{b}$ *zugeordnet (intuitiv: man setzt $\vec{x}$ in die Gleichung ein).*

Das Residuum ist ein leicht berechenbares Maß, wie gut $\vec{x}$ die Lösung $A^{-1}\vec{b}$ der Gleichung approximiert:

Satz 4.64: (a posteriori Fehlerkontrolle über das Residuum)

Für jedes $\vec{x}$ gilt

$$\frac{\|\vec{r}(\vec{x})\|}{\|A\|} \;\leq\; \overbrace{\|\vec{x} - A^{-1}\vec{b}\|}^{\textit{absoluter Fehler}} \;\leq\; \|A^{-1}\|\,\|\vec{r}(\vec{x})\| \;,$$

$$\frac{1}{\mathrm{cond}(A)}\,\frac{\|\vec{r}(\vec{x})\|}{\|\vec{b}\|} \;\leq\; \underbrace{\frac{\|\vec{x} - A^{-1}\vec{b}\|}{\|A^{-1}\vec{b}\|}}_{\textit{relativer Fehler}} \;\leq\; \mathrm{cond}(A)\,\frac{\|\vec{r}(\vec{x})\|}{\|\vec{b}\|} \;.$$

Beweis: Mit

$$\begin{aligned} \vec r = A(\vec x - A^{-1}\vec b) \quad &\Rightarrow \quad \|\vec r\| \le \|A\|\,\|\vec x - A^{-1}\vec b\| \\ \vec x - A^{-1}\vec b = A^{-1}\vec r \quad &\Rightarrow \quad \|\vec x - A^{-1}\vec b\| \le \|A^{-1}\|\,\|\vec r\| \end{aligned}$$

folgt die Abschätzung des absoluten Fehlers. Mit $\|A^{-1}\vec b\| \le \|A^{-1}\|\,\|\vec b\|$ und $\|\vec b\| = \|AA^{-1}\vec b\| \le \|A\|\,\|A^{-1}\vec b\|$ folgt

$$\frac{1}{\|A^{-1}\|\,\|\vec b\|} \le \frac{1}{\|A^{-1}\vec b\|} \le \frac{\|A\|}{\|\vec b\|} \,.$$

Multiplikation mit den Abschätzungen des absoluten Fehlers liefert die Abschätzungen des relativen Fehlers.

Q.E.D.

Der folgende Satz liefert eine Abschätzung, wie stark sich die Lösung $\vec x = A^{-1}\vec b$ eines Gleichungssystems ändert, wenn die Daten $A, \vec b$ verändert werden (z.B. durch Rundungsfehler $\Delta A, \Delta\vec b$). Er beschreibt die prinzipielle mathematische Konditionierung der Abbildung $(A, \vec b\,) \to A^{-1}\vec b$ (unabhängig vom benutzten numerischen Gleichungslöser):

Satz 4.65: (Einfluß der Daten auf die Lösung, "exakte Rechnung, gestörte Daten")

Es seien $\vec x$ bzw. $\vec x + \Delta\vec x$ die exakten Lösungen von

$$A\vec x = \vec b \qquad \text{bzw.} \qquad (A + \Delta A)\,(\vec x + \Delta\vec x) = \vec b + \Delta\vec b \,.$$

Für $\|A^{-1}\|\,\|\Delta A\| < 1$ gilt

$$\frac{\|\Delta\vec x\|}{\|\vec x\|} \le \frac{\mathrm{cond}(A)}{1 - \mathrm{cond}(A)\,\dfrac{\|\Delta A\|}{\|A\|}} \left(\frac{\|\Delta A\|}{\|A\|} + \frac{\|\Delta\vec b\|}{\|\vec b\|} \right) .$$

Beweis: $\Delta\vec x$ löst die Gleichung

$$(A + \Delta A)\,\Delta\vec x = \Delta\vec b - \Delta A\,\vec x \,.$$

Hierbei ist $A + \Delta A = A\,(\mathbb{1} + A^{-1}\Delta A)$ invertierbar, wenn A invertierbar ist und (Hilfssatz 4.60) $\|A^{-1}\Delta A\| < 1$ gilt. Mit $\|A^{-1}\Delta A\| \le \|A^{-1}\|\,\|\Delta A\| < 1$ folgt

$$\Delta\vec x = (A + \Delta A)^{-1}\,(\Delta\vec b - \Delta A\,\vec x) = (\mathbb{1} + A^{-1}\,\Delta A)^{-1}\,A^{-1}\,(\Delta\vec b - \Delta A\,\vec x)$$

$$\Rightarrow \quad \|\Delta\vec x\| \overset{(4.60)}{\le} \frac{\|A^{-1}\|}{1 - \|A^{-1}\Delta A\|}\,(\|\Delta\vec b\| + \|\Delta A\|\,\|\vec x\|)$$

$$\Rightarrow \quad \frac{\|\Delta\vec{x}\|}{\|\vec{x}\|} \leq \frac{\|A^{-1}\|\|A\|}{1-\|A^{-1}\|\,\|\Delta A\|}\Big(\frac{\|\Delta\vec{b}\|}{\|A\|\,\|\vec{x}\|}+\frac{\|\Delta A\|}{\|A\|}\Big) .$$

Aus $\|\vec{b}\| = \|A\vec{x}\| \leq \|A\|\,\|\vec{x}\|$ folgt

$$\frac{\|\Delta\vec{x}\|}{\|\vec{x}\|} \leq \frac{\mathrm{cond}(A)}{1-\|A^{-1}\|\,\|\Delta A\|}\Big(\frac{\|\Delta\vec{b}\|}{\|\vec{b}\|}+\frac{\|\Delta A\|}{\|A\|}\Big) .$$

Q.E.D.

Merke: $\mathrm{cond}(A)$ ist der Verstärkungsfaktor von Daten-(Eingabe-)fehlern. Ein Gleichungssystem mit $\mathrm{cond}(A) \gg 1$ ist **schlecht konditioniert** (hängt empfindlich von den Daten ab).

Beispiel 4.66: Die $n \times n$ **Hilbert-Matrizen** $A = (a_{ij}) = \Big(\dfrac{1}{i+j-1}\Big)$ sind extrem schlecht konditioniert:

n	$\|A\|_2$	$\|A^{-1}\|_2$	$\mathrm{cond}_2(A)$
5	1.56...	$3.04... \times 10^{5}$	$4.76... \times 10^{5}$
10	1.75...	$9.14... \times 10^{12}$	$1.60... \times 10^{13}$
20	1.90...	$1.28... \times 10^{28}$	$2.45... \times 10^{28}$
100	2.18...	$1.73... \times 10^{150}$	$3.77... \times 10^{150}$
200	2.27...	$1.57... \times 10^{303}$	$3.57... \times 10^{303}$

Ein solches System mit 20 Unbekannten kann numerisch kaum noch gelöst werden!

Bemerkung 4.67: *Die bisherigen Gleichungslöser beruhen auf Transformationen*

$$A\vec{x} = \vec{b} \;\Rightarrow\; A'\vec{x} = \vec{b}' \;\Rightarrow\; A''\vec{x} = \vec{b}'' \;\Rightarrow\; \ldots \;\Rightarrow\; R\vec{x} = \vec{c} ,$$

wobei die Daten R,$\vec{c}$ des resultierenden gestaffelten Gleichungssystems mit Rundungsfehlern behaftet sind. Ein solcher Algorithmus ist numerisch nicht stabil, wenn die Konditionszahlen

$$\mathrm{cond}(A) \;,\; \mathrm{cond}(A') \;,\; \ldots \;,\; \mathrm{cond}(R)$$

ansteigen können.

Dies ist z.B. beim Gauß-Algorithmus ohne Pivotierung der Fall (Aufgabe 31.a).

Bei der QR-Zerlegung $A = QR$ ist das erzeugte gestaffelte System genauso konditioniert wie das Ausgangssystem (Aufgabe 31.c):

$$\mathrm{cond}_2(R) = \mathrm{cond}_2(Q^T A) = \mathrm{cond}_2(A) .$$

Bei der Cholesky-Zerlegung $A = LL^T$ gilt (Aufgabe 31.b)

$$\mathrm{cond}_2(L) = \mathrm{cond}_2(L^T) = \sqrt{\mathrm{cond}_2(A)} \leq \mathrm{cond}_2(A) ,$$

so daß die resultierenden gestaffelten Systeme sogar besser konditioniert sind.

4.13 Nachiteration

Betrachte die Nachiteration aus Bemerkung 4.25 zur Verbesserung einer Approximation $\vec{x}_{num}$ von $A^{-1}\vec{b}$:

mit gegebenem	$\vec{x}_{num}$
berechne das Residuum	$\vec{r} \;=\; A\vec{x}_{num} \;-\; \vec{b}\;,$
löse	$A\vec{y} \;=\; \vec{r}$
und korrigiere	$\vec{x}_{besser} \;=\; \vec{x}_{num} \;-\; \vec{y}\;.$

Ohne weitere Rundungsfehler wäre $\vec{x}_{besser}$ exakt:

$$\vec{x}_{besser} \;=\; \vec{x}_{num} - A^{-1}(A\vec{x}_{num} - \vec{b}) \;=\; A^{-1}\vec{b}\;,$$

womit der verbleibende Fehler nur von den Rundungsfehlern in der Berechnung vom $\vec{r}$ und $\vec{y}$ erzeugt wird.

Diese Beobachtung liefert sofort eine (etwas pauschale) Erklärung, warum $\vec{x}_{besser}$ eine bessere Approximation der Lösung als $\vec{x}_{num}$ liefert:
Der Gleichungslöser sei in der Lage, unabhängig von der Größe der rechten Seite die erste Dezimalstelle der Lösung (relative Genauigkeit!) exakt zu liefern. Die Mantissen/Exponenten-Darstellungen der i.ten Vektorkomponenten seien

$$\begin{array}{ccl} (A^{-1}\vec{b})_i & = & 0.\,a_1\,a_2\,a_3\,a_4\;\ldots\;\times\;10^{p_i} \\ (\vec{x}_{num})_i & = & 0.\,a_1\,b_2\,b_3\,b_4\;\ldots\;\times\;10^{p_i} \\ (\vec{y})_i & = & 0.\;0\;\;c_2\,c_3\,c_4\;\ldots\;\times\;10^{p_i} \\ (\vec{y}_{num})_i & = & 0.\;0\;\;c_2\,d_3\,d_4\;\ldots\;\times\;10^{p_i} \end{array}$$

wobei $\vec{y}_{num}$ die numerische Lösung von $A\vec{y} = \vec{r}$ sei, deren erste Mantissenziffern mit denen in $\vec{y} = A^{-1}\vec{r}$ übereinstimmen (die Rundungsfehler in $\vec{r}$ seien vernachlässigt). Wegen $A^{-1}\vec{b} = \vec{x}_{num} - \vec{y}$ gilt $a_2 = b_2 - c_2$, also

$$\begin{array}{rcl} (\vec{x}_{besser})_i \;=\; (\vec{x}_{num} - y_{num})_i & = & 0.\;a_1\;(b_2 - c_2)\;(b_3 - d_3)\ldots\;\times\;10^{p_i} \\ & = & 0.\;a_1\;a_2\;(b_3 - d_3)\ldots\;\times\;10^{p_i}\;. \end{array}$$

Durch den Nachiterationsschritt wurde damit eine weitere Dezimalstelle Genauigkeit hinzugewonnen.

Allgemein gilt: liefert der Gleichungslöser die ersten q Dezimalstellen der Lösung, so werden beim Schritt $\vec{x}_{num} \to \vec{x}_{besser}$ weitere q Dezimalstellen relativer Genauigkeit hinzugewonnen. Nach mehreren Nachiterationsschritten sind die Rundungsfehler des Gleichungslösers praktisch eliminiert! Die Grenzgenauigkeit ist damit nur durch den (bisher ignorierten) relativen Fehler der Berechnung von $A\vec{x}_{num}$ in $\vec{r}$ bestimmt.

Diese grobe Betrachtung soll nun genauer geprüft werden. Die fehlerbehaftete Nachiteration ist

zu	$\vec{x}_{num}$
berechne	$\vec{r}_{num} = A\vec{x}_{num} - \vec{b} + \Delta\vec{r}$,
berechne	$\vec{y}_{num} = A^{-1}\vec{r}_{num} + \Delta\vec{y}$
und korrigiere	$\vec{x}_{besser} = \vec{x}_{num} - \vec{y}_{num}$,

also

$$\vec{x}_{besser} = x_{num} - A^{-1}(A\vec{x}_{num} - \vec{b} + \Delta\vec{r}) - \Delta\vec{y} = A^{-1}\vec{b} - A^{-1}\Delta\vec{r} - \Delta\vec{y} ,$$

d.h.,

$$\|\vec{x}_{besser} - A^{-1}\vec{b}\| \leq \|A^{-1}\|\,\|\Delta\vec{r}\| + \|\Delta\vec{y}\| . \qquad (\#)$$

Realistische Annahmen:

a) Unter Vernachlässigung von Darstellungsfehlern sei die <u>absolute</u> Genauigkeit der Residuenberechnung

$$\|\Delta\vec{r}\| = \|\Delta(A\vec{x}_{num})\| \leq \tau_{res}\,\|A\vec{x}_{num}\| \leq \tau_{res}\,\|A\|\,\|\vec{x}_{num}\| ,$$

wobei

$$\tau_{res} \geq \frac{\|\Delta(A\vec{x}_{num})\|}{\|A\vec{x}_{num}\|}$$

der <u>relative</u> Fehler der Matrix-Vektor-Multiplikation sei.

b) Der verwendete Gleichungslöser liefert die Lösung von $A\vec{y} = \vec{r}$ bis auf eine <u>relative</u> Genauigkeit τ_A:

$$\frac{\|\Delta\vec{y}\|}{\|A^{-1}\vec{r}\|} \leq \tau_A \quad (\text{unabhängig von } \vec{r}) .$$

c) Zu erwarten ist $1 \gg \tau_A \gg \tau_{res} \gg \tau =$ Maschinengenauigkeit.

Mit

$$\begin{aligned} \|\Delta\vec{y}\| \leq \tau_A\,\|A^{-1}\vec{r}_{num}\| &= \tau_A\,\|A^{-1}(A\vec{x}_{num} - \vec{b} + \Delta\vec{r})\| \\ &\leq \tau_A\,\|\vec{x}_{num} - A^{-1}\vec{b}\| + \tau_A\,\|A^{-1}\|\,\|\Delta\vec{r}\| \end{aligned}$$

und

$$\|A^{-1}\|\,\|\Delta\vec{r}\| \leq \tau_{res}\,\mathrm{cond}(A)\,\|\vec{x}_{num}\|$$

liefert (#)

$$\boxed{\frac{\|\vec{x}_{besser} - A^{-1}\vec{b}\|}{\|A^{-1}\vec{b}\|} \leq \tau_A \frac{\|\vec{x}_{num} - A^{-1}\vec{b}\|}{\|A^{-1}\vec{b}\|} + \tau_{res}\,(1 + \tau_A)\,\mathrm{cond}(A)\,\frac{\|\vec{x}_{num}\|}{\|A^{-1}\vec{b}\|}} .$$

Interpretation: bis auf den durch τ_{res} gegebenen Fehler wurde die Anzahl korrekter Dezimalstellen durch $\vec{x}_{num} \to \vec{x}_{besser}$ um $\log_{10}(1/\tau_A)$ Stellen erhöht. Also: die führenden Stellen, die der verwendete Gleichungslöser exakt liefern kann, werden im Nachiterationsschritt als zusätzliche Stellen gewonnen. Wurde $\vec{x}_{num}$ selbst durch den Gleichungslöser bestimmt, so gilt

$$\frac{\|\vec{x}_{num} - A^{-1}\vec{b}\|}{\|A^{-1}\vec{b}\|} \leq \tau_A , \quad \frac{\|\vec{x}_{num}\|}{\|A^{-1}\vec{b}\|} \leq 1 + \tau_A$$

und es folgt

$$\frac{\|\vec{x}_{besser} - A^{-1}\vec{b}\|}{\|A^{-1}\vec{b}\|} \leq \tau_A^2 + \tau_{res}\,(1 + \tau_A)^2\,\mathrm{cond}(A) .$$

Nach mehreren Nachiterationen ist der Fehler τ_A des Gleichungslösers praktisch verschwunden, bei $\tau_A \ll 1$ ist die erreichbare Grenzgenauigkeit $\tau_{res}\,\mathrm{cond}(A)$. Selbst ein ungenauer Gleichungslöser, der nur die erste Dezimalstelle exakt liefert (d.h., $\tau_A \approx 1/10$), kann durch Nachiteration eingesetzt werden, um die Lösung bis auf die Grenzgenauigkeit zu bestimmen.

Zur Implementierung der Nachiteration:

Stellt der verwendete Compiler mehrere Genauigkeiten

“einfach” (SINGLEPRECISION) und “erhöht” (DOUBLEPRECISION)

zur Verfügung, so sollte der wesentliche Rechenaufwand der Gleichungslösung, also die Berechnung von Zerlegungsdaten der Matrix, in einfacher Genauigkeit durchgeführt werden, da die Rechenoperationen schneller sind als bei erhöhter Genauigkeit. In der Residuenberechnung der Nachiteration sollte erhöhte Genauigkeit verwendet werden.

Benutze zur Lösung von $A\vec{x} = \vec{b}$ die Variablen

$$A_D = A \ , \ \vec{b}_D = \vec{b} \ , \ \vec{x}_D \ , \ \vec{r}_D \quad \text{(DOUBLEPRECISION)}$$

und

$$A_S = A \ , \ \vec{b}_S = \vec{b} \ , \ \vec{y}_S \ , \ \vec{r}_S \quad \text{(SINGLEPRECISION)} .$$

Start: löse zunächst $A_S\vec{y}_S = \vec{b}_S$, wobei eine Zerlegung (z.B. $PA_S = L_S R_S$ in einfacher Genauigkeit) erzeugt wird. Man erhält so eine erste Approximation $\vec{x}_D := \vec{y}_S$ der Lösung.

Nachiteration: berechne das Residuum $\vec{r}_D := A_D\vec{x}_D - \vec{b}_D$ in erhöhter Genauigkeit und speichere es in einfacher Genauigkeit ab: $\vec{r}_S := \vec{r}_D$. Hierbei gehen in der Mantisse von $\vec{r}$ einige Stellen verloren, die im Exponenten gespeicherte Größe und die ersten Mantissenziffern bleiben aber erhalten. Berechne unter Verwendung der vorhandenen Zerlegungsdaten in einfacher Genauigkeit die Korrektur $\vec{y}_S$ aus $A_S\vec{y}_S = \vec{r}_S$ und verbessere $\vec{x}_D$:

$$\vec{x}_D := \vec{x}_D - \vec{y}_S \ .$$

Da $\vec{y}_S$ klein ist im Vergleich zu $\vec{x}_D$, machen sich die Ungenauigkeiten in $\vec{y}_S$ nur in den hinteren Mantissenstellen von $\vec{x}_D$ bemerkbar.
Führe die Nachiteration so lange durch, bis $\|\vec{r}_D\|$ nicht mehr kleiner wird.

Ergebnis: bei nicht zu schlecht konditionierten Marizen wird sich mit den Maschinengenauigkeiten τ_S (einfache Genauigkeit) und τ_D (erhöhte Genauigkeit) in der Regel die Situation

$$\tau_D \ll \tau_{res} \ll \tau_S \ll \tau_A \ll 1$$

einstellen, so daß sich trotz schneller Gleichungslösung in einfacher Genauigkeit für die Lösung $\vec{x}_D$ eine Grenzgenauigkeit $\mathrm{cond}(A)\,\tau_{res}$ erreichen läßt, die unterhalb τ_S liegen kann.

4.14 Iterative Verfahren

13.1.97↓ Der Banachsche Fixpunktsatz 3.3 gilt auf dem $\mathbb{R}^n$ bzw. $\mathbb{C}^n$, wenn man Beträge durch (beliebige) Normen ersetzt:

Satz 4.68: (Banachscher Fixpunktsatz (BFS))

Sei Ω eine abgeschlossene Teilmenge eines vollständigen normierten Raumes. Für $\Phi : \Omega \to \Omega$ existiere $k \in [0,1)$ mit $\|\Phi(x) - \Phi(y)\| \le k\,\|x - y\|$ $\forall x, y \in \Omega$. Dann konvergiert $x^{(i+1)} = \Phi(x^{(i)})$ für jedes $x^{(0)} \in \Omega$ gegen den eindeutigen Fixpunkt $x^* = \Phi(x^*) \in \Omega$, es gilt

$$\|x^{(i)} - x^*\| \ \le \ \frac{k}{1-k}\,\|x^{(i)} - x^{(i-1)}\| \ \le \ \frac{k^i}{1-k}\,\|x^{(1)} - x^{(0)}\| \ .$$

Beweis: wortwörtlich wie in Satz 3.3, wenn man $|\,.\,|$ durch $\|\,.\,\|$ ersetzt.

Eine affin-lineare Abbildung $\vec{\Phi}(\vec{x}) = B\vec{x} + \vec{c}$ auf dem $\mathbb{R}^n$ ist genau dann eine Kontraktion bzgl. $\|\,.\,\|$, wenn $\|B\| < 1$ gilt:

$$\|\vec{\Phi}(\vec{x}) - \vec{\Phi}(\vec{y})\| \ = \ \|B(\vec{x} - \vec{y})\| \ \le \ \|B\|\,\|\vec{x} - \vec{y}\| \ .$$

Es besteht die Freiheit, irgendeine Norm zu wählen (Konvergenz in einer Norm bedeutet Konvergenz in allen Normen, da auf dem $\mathbb{R}^n$ alle Normen äquivalent sind). Damit entscheidet der Spektralradius $\rho(B)$ über die Konvergenz einer Banach-Iteration:

Satz 4.69:

Eine Iteration der Form $\vec{x}^{(i+1)} = B\vec{x}^{(i)} + \vec{c}$ auf dem $\mathbb{R}^n$ mit $\rho(B) < 1$ konvergiert für alle Startvektoren $\vec{x}^{(0)} \in \mathbb{R}^n$. Der Grenzwert ist unabhängig vom Startvektor.

Beweis: Betrachte eine Vektornorm aus Satz 4.58.c) mit $0 < \epsilon < 1 - \rho(B)$, so daß

$$k \;:=\; \|B\|_{B,\epsilon} \;\leq\; \rho(B) + \epsilon \;<\; 1$$

gilt. Wende den BFS mit $\Omega = \mathbb{R}^n$ an.

Q.E.D.

Bemerkung 4.70: *Für $\rho(B) > 1$ wird die Folge i.a. divergieren. Sei $B\vec{e} = \lambda\vec{e}$ mit $|\lambda| > 1$. Angenommen, es existiert ein Grenzwert $\vec{x}^* = B\vec{x}^* + \vec{c}$ für einen Startvektor. Der neue Startvektor $\vec{x}^{(0)} = \vec{x}^* + \vec{e}$ führt auf die divergierende Folge $\vec{x}^{(i)} = \vec{x}^* + \lambda^{i-1}\,\vec{e}$.*

Damit ergibt sich die folgende Strategie zur Lösung von $A\vec{x} = \vec{b}$:

finde ein äquivalentes Fixpunktproblem $\vec{x} = \vec{\Phi}(\vec{x}) = B\vec{x} + \vec{c}$ mit möglichst kleinem $\|B\|$ (genauer: mit möglichst kleinem $\rho(B)$).

Ansatz: betrachte additive Zerlegungen

$$A \;=\; W \;+\; R \;=\; \text{“wesentlicher Teil”} + \text{(kleiner) “Rest”}\;.$$

Dann gilt

$$A\vec{x} = \vec{b} \quad\Longleftrightarrow\quad W\vec{x} = -R\vec{x} + \vec{b} \quad\Longleftrightarrow\quad \vec{x} = \underbrace{-W^{-1}R\,\vec{x} + W^{-1}\vec{b}}_{\vec{\Phi}(\vec{x})} =: B\,\vec{x} + \vec{c}\;.$$

Hierbei sollte

- W schnell invertierbar (typisch: W = Dreiecksmatrix)
- und R im Sinne von $\|W^{-1}R\| < 1$ hinreichend klein sein.

Satz 4.71:

Sei $A = W + R$ mit invertierbarem W und $k := \|W^{-1}R\| < 1$. Dann konvergiert die durch

$$W\vec{x}^{(i+1)} = -R\,\vec{x}^{(i)} + \vec{b}$$

definierte Folge für jedes $\vec{x}^{(0)}$ gegen $A^{-1}\vec{b}$. Für $\vec{x}^{(0)} = 0$ und $\vec{b} \neq 0$ gilt

$$\frac{\|\vec{x}^{(i)} - A^{-1}\vec{b}\|}{\|A^{-1}\vec{b}\|} \;\leq\; k^i \quad \textit{(Kontrolle des \underline{relativen} Fehlers).}$$

Beweis: Mit Hilfssatz 4.60 ist $A = W\,(\mathbb{1} + W^{-1}R)$ automatisch invertierbar. Die Banach-Iteration

$$\vec{x}^{(i+1)} = \vec{\Phi}(\vec{x}^{(i)}) = -W^{-1}R\,\vec{x}^{(i)} + W^{-1}\vec{b}$$

konvergiert mit der Kontraktionskonstanten $k < 1$ gegen einen Fixpunkt $\vec{x}^* = -W^{-1}R\,\vec{x}^* + W^{-1}\vec{b}$, also $(W+R)\,\vec{x}^* = \vec{b}$, d.h., $\vec{x}^* = A^{-1}\vec{b}$. Mit

$$\vec{x}^{(i)} - \vec{x}^* \;=\; (-W^{-1}R)\,(\vec{x}^{(i-1)} - \vec{x}^*) \;=\; \ldots \;=\; (-W^{-1}R)^i\,(\vec{x}^{(0)} - \vec{x}^*)$$

folgt $\|\vec{x}^{(i)} - \vec{x}^*\| \;\leq\; \|(W^{-1}R)^i\|\,\|\vec{x}^{(0)} - \vec{x}^*\| \;\overset{(4.55.b)}{\leq}\; \|W^{-1}R\|^i\,\|\vec{x}^{(0)} - \vec{x}^*\|\,.$

Q.E.D.

Beispiel 4.72: Für

$$\begin{array}{lclclcl} 2\;\;x_1 & + & 0.001\,x_2 & + & 0.002\,x_3 & = & b_1 \\ 0.002\,x_1 & + & 3\;\;x_2 & + & 0.001\,x_3 & = & b_2 \\ 0.001\,x_1 & + & 0.002\,x_2 & + & 4\;\;x_3 & = & b_3 \end{array}$$

bietet sich eine Fixpunktformulierung $W\vec{x} = \vec{b} - R\,\vec{x}$ mit "kleinem Rest" R an:

$$\begin{array}{rclclcl} 2\,x_1 & = & b_1 & - & 0.001\,x_2 & - & 0.002\,x_3 \\ 3\,x_2 & = & b_2 & - & 0.002\,x_1 & - & 0.001\,x_3 \\ 4\,x_3 & = & b_3 & - & 0.001\,x_1 & - & 0.002\,x_2\;. \end{array}$$

Definition 4.73:

Eine $n \times n$-Matrix $A = (a_{ij})$ mit $|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|$, $i = 1, \ldots, n$, heißt (stark) **diagonaldominant**.

Bemerkung 4.74: *Zerlege $A = (a_{ij}) = l + D + r$ in die Diagonale*

$$D = \mathrm{diag}(a_{11}, \ldots, a_{nn})$$

und die streng unteren bzw. oberen Dreiecksanteile l bzw. r:

$$l = \begin{pmatrix} 0 & & & 0 \\ a_{21} & \ddots & & \\ \vdots & \ddots & \ddots & \\ a_{n1} & \ldots & a_{n,n-1} & 0 \end{pmatrix}, \quad r = \begin{pmatrix} 0 & a_{12} & \ldots & a_{1n} \\ & \ddots & \ddots & \vdots \\ & & \ddots & a_{n-1,n} \\ 0 & & & 0 \end{pmatrix}.$$

Mit

$$D^{-1}(l+r) \;=\; \begin{pmatrix} 0 & \frac{a_{12}}{a_{11}} & \cdots & \frac{a_{1n}}{a_{11}} \\ \frac{a_{21}}{a_{22}} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \frac{a_{n-1,n}}{a_{n-1,n-1}} \\ \frac{a_{n1}}{a_{nn}} & \cdots & \frac{a_{n,n-1}}{a_{nn}} & 0 \end{pmatrix}$$

gilt

$$A \text{ ist diagonaldominant} \quad \Leftrightarrow \quad \|D^{-1}(l+r)\|_\infty = \max_{i=1..n} \frac{1}{|a_{ii}|} \sum_{\substack{j=1\\ j\neq i}}^{n} |a_{ij}| \;<\; 1 \;.$$

Satz 4.75: (Gesamt- und Einzelschrittverfahren)
Die diagonaldominante Matrix $A = l+D+r$ sei gemäß 4.74 zerlegt. Dabei sind A, D und $D+l$ invertierbar, und es gilt

$$\|(D+l)^{-1}r\|_\infty \;\leq\; \|D^{-1}(l+r)\|_\infty \;<\; 1 \;.$$

Nach Satz 4.71 mit $W = D$, $R = l+r$ bzw. $W = D+l$, $R = r$ konvergieren die durch

$$D\,x^{(i+1)} \;=\; -\;(l+r)\,\vec{x}^{(i)} \;+\; \vec{b} \qquad \textit{(}\textbf{Jacobi-}\textit{ oder } \textbf{Gesamtschrittverfahren}\textit{)}$$

bzw.

$$(D+l)\,x^{(i+1)} \;=\; -\;r\,\vec{x}^{(i)} \;+\; \vec{b} \qquad \textit{(}\textbf{Gauß-Seidel-}\textit{ oder } \textbf{Einzelschrittverfahren}\textit{)}\;,$$

gegebenen Banach-Iterationen mit den Kontraktionskonstanten

$$k_{Einzel} = \|(D+l)^{-1}r\|_\infty \;\leq\; k_{Gesamt} = \|D^{-1}(l+r)\|_\infty \;<\; 1$$

gegen $A^{-1}\vec{b}$.

Beweis: Da kein Diagonalelement verschwindet, sind D und $D+l$ invertierbar, A^{-1} existiert nach Aufgabe 23. Nach 4.74 gilt $k := \|D^{-1}(l+r)\|_\infty < 1$, womit nur noch $\|(D+l)^{-1}r\|_\infty \leq k$ zu zeigen ist. Nehme dazu ein beliebiges $\vec{x}$ mit $\|\vec{x}\|_\infty = \max_i |x_i| = 1$ und setze $\vec{y} = \vec{y}(\vec{x}) = (D+l)^{-1} r\,\vec{x}$, also $D\vec{y} = -l\vec{y} + r\,\vec{x}$. Rücksubstitution liefert

$$\begin{aligned} y_1 &= \frac{1}{a_{11}} \sum_{j=2}^{n} a_{1j}\,x_j \;, \\ y_i &= \frac{1}{a_{ii}} \Big(-\sum_{j=1}^{i-1} a_{ij}\,y_j + \sum_{j=i+1}^{n} a_{ij}\,x_j \Big)\;, \quad i = 2,\ldots,n \;. \end{aligned}$$

Induktiv folgt $|y_1|, \ldots, |y_i| \leq k$:

$$\underline{i=1}: \qquad |y_1| \;\leq\; \frac{1}{|a_{11}|} \sum_{j=2}^{n} |a_{1j}| \;=\; 1.\ \text{Zeilensumme von } D^{-1}(l+r) \;\leq\; k\ ,$$

$$\underline{i-1 \to i}: \; |y_i| \;\leq\; \frac{1}{|a_{ii}|} \Big(\sum_{j=1}^{i-1} |a_{ij}|\, k \;+\; \sum_{j=i+1}^{n} |a_{ij}| \Big)$$

$$\stackrel{(k<1)}{\leq} \; \frac{1}{|a_{ii}|} \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}| \;=\; \text{j. Zeilensumme von } D^{-1}(l+r) \;\leq\; k\ .$$

Es folgt $\|\vec{y}\|_\infty = \max_i |y_i| \leq k$ und damit

$$\|(D+l)^{-1} r\|_\infty \;=\; \max_{\|\vec{x}\|_\infty = 1} \|(D+l)^{-1} r\, \vec{x}\|_\infty \;=\; \max_{\|\vec{x}\|_\infty = 1} \|\vec{y}(\vec{x})\|_\infty \;\leq\; k\ .$$

Q.E.D.

Bemerkung 4.76: *Der Aufwand des Schrittes* $\vec{x}^{(i)} = (x_1^{(i)}, \ldots, x_n^{(i)})^T \to$
$\boxed{15.1.98\downarrow}$ $\vec{x}^{(i+1)} = (x_1^{(i+1)}, \ldots, x_n^{(i+1)})^T$ *(Matrix-Vektor-Multiplikation und Rücksubstitution) ist in beiden Verfahren gleich, nämlich* n^2 *Multiplikationen bei vollbesetzten Matrizen:*

for $k := 1$ **to** n **do** $\quad$ (∗ Gesamtschritt: $D\,\vec{x}^{(i+1)} = \vec{b} - l\,\vec{x}^{(i)} - r\,\vec{x}^{(i)}$ ∗)

$$x_k^{(i+1)} \;:=\; \frac{1}{a_{kk}} \Big(b_k \;-\; \sum_{j=1}^{i-1} a_{kj}\, x_j^{(i)} - \sum_{j=i+1}^{n} a_{kj}\, x_j^{(i)} \Big)\ ;$$

bzw.

for $k := 1$ **to** n **do** $\quad$ (∗ Einzelschritt: $D\,\vec{x}^{(i+1)} = \vec{b} - l\,\vec{x}^{(i+1)} - r\,\vec{x}^{(i)}$ ∗)

$$x_k^{(i+1)} \;:=\; \frac{1}{a_{kk}} \Big(b_k \;-\; \sum_{j=1}^{i-1} a_{kj}\, x_j^{(i+1)} - \sum_{j=i+1}^{n} a_{kj}\, x_j^{(i)} \Big)\ ;$$

Also:

Gesamtaufwand (Multiplikatonen) $=$ $n^2 \times$ *Anzahl der benötigten Schritte.*

Falls die Kontraktionskonstanten nicht zu dicht bei 1 liegen, reichen weniger als n *Schritte, um die Lösung auf eine gewünschte Genauigkeit zu bestimmen. Hiermit können iterative Verfahren (abhängig von der Kontraktionskonstanten) schneller als direkte Verfahren sein, wobei das Einzelschrittverfahren schneller als das Gesamtschrittverfahren ist (letzteres ist dafür leichter parallelisierbar).*

Bei dünnbesetzten Matrizen ist zusätzlich auch jeder Iterationsschritt schnell durchführbar: die Multiplikation bzw. Rücksubstitution mit den dünnbesetzten Dreiecksanteilen braucht nur die nichttrivialen Einträge zu berücksichtigen. Hierbei ist egal, wie diese in der Matrix verteilt sind. Die direkten Verfahren (LR-, Cholesky-, QR-Zerlegung) sind nur dann schnell, wenn die nichttrivialen Matrixkomponenten sich zu wenigen Bändern um die Diagonale herum formieren.

Bemerkung 4.77: *Die Konvergenz der Verfahren ist auch dann garantiert, wenn die (starke) Diagonaldominanz* $|a_{ii}| > \sum_{j\neq i} |a_{ij}|$ *durch "schwache" Diagonaldominanz* $|a_{ii}| \geq \sum_{j\neq i} |a_{ij}|$ *(plus Zusatzbedingungen) ersetzt wird. Zu Details siehe z.B. [Scw93].*

Bemerkung 4.78: *Als Banach-Iterationen sind die Verfahren selbstkorrigierend. Die prinzipiell erreichbare Genauigkeit ist i.a. besser als die der direkten Verfahren (LR- bzw. QR-Zerlegung), eine Nachiteration erübrigt sich in der Regel.*

Bemerkung 4.79: *Leider sind in der Praxis schwach diagonaldominante Matrizen typisch, für die das Einzelschrittverfahren* $(D + l)\,\vec{x}^{(i+1)} = -r\,\vec{x}^{(i)} + \vec{b}$ *zu dicht bei 1 liegenden Kontraktionskonstanten führt. Man muß das Verfahren daher beschleunigen, um zu praktikablen Laufzeiten zu gelangen. Mit der Zerlegung*

$$A = \underbrace{l + \frac{1}{\omega} D}_{W} \; + \; \underbrace{\Big(1 - \frac{1}{\omega}\Big) D + r}_{R}$$

führt man z.B.

$$\Big(\frac{1}{\omega}\, D + l\Big)\; \vec{x}^{(i+1)} \;=\; \Big(\Big(\frac{1}{\omega} - 1\Big)\, D - r\Big)\; \vec{x}^{(i)} \;+\; \vec{b}$$

durch **(SOR-Verfahren)**. *Hierbei wird der* **Relaxationsparameter** $\omega \in (0, 2)$ *so gewählt, daß der Spektralradius der Iterationsmatrix*

$$\rho\left(\Big(\frac{1}{\omega}\, D + l\Big)^{-1} \Big(\Big(1 - \frac{1}{\omega}\Big) D + r\Big)\right)$$

minimal wird ($\omega = 1$ *ist das Einzelschrittverfahren). Für gewisse Klassen von Matrizen (Stichwort: "konsistente Ordnung") kann der optimale Parameter* ω, *für den der obige Spektralradius minimal wird, analytisch bestimmt und formelmäßig angegeben werden. Zur Theorie siehe z.B. [Scw93] oder [Oev96]. Hiermit können große dünnbesetzte (schwach) diagonaldominante Systeme (typisch bei der Numerik von Differentialgleichungen) effizient gelöst werden.*

Bemerkung 4.80: *Weitere (noch schnellere) iterative Verfahren für große dünnbesetzte (s.p.d.) Systeme sind:*

ADI	*siehe z.B. [StB90, Kapitel 8.6],*
cg	*(Verfahren konjugierter Gradienten), siehe z.B. [Scw93],*
multigrid	*(Mehrgitter-Verfahren), siehe z.B. [StB90, Kapitel 8.9].*

Siehe auch: W. HACKBUSCH, *Iterative Lösung großer schwachbesetzter Gleichungssysteme, Teubner 1991.*

Kapitel 5

Nichtlineare Gleichungssysteme

Aufgabe: löse ein nichtlineares Gleichungssystem $\vec{f}(\vec{x}) = 0$ (mit $\vec{f} : \mathbb{R}^n \to \mathbb{R}^n$) in n Unbekannten. Die grundsätzliche Strategie ist wieder die Umformung in ein Fixpunktproblem

$$\vec{f}(\vec{x}) = 0 \quad \Longleftrightarrow \quad \vec{x} = \vec{\Phi}(\vec{x})$$

und Anwendung des BFS 4.68.

Bezeichnung 5.1:

Für $\vec{\Phi}(\vec{x}) = \begin{pmatrix} \Phi_1(x_1, \ldots, x_n) \\ \vdots \\ \Phi_n(x_1, \ldots, x_n) \end{pmatrix}$ *sei* $\vec{\Phi}'(\vec{x}) = \begin{pmatrix} \frac{\partial \Phi_1}{\partial x_1} & \cdots & \frac{\partial \Phi_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial \Phi_n}{\partial x_1} & \cdots & \frac{\partial \Phi_n}{\partial x_n} \end{pmatrix}$

die ***Ableitungsmatrix***.

Lemma 5.2:

Für stetig diff'bares $\vec{\Phi} : \mathbb{R}^n \to \mathbb{R}^n$ *gilt*

$$\|\vec{\Phi}(\vec{x}) - \vec{\Phi}(\vec{y})\| \ \le \ \|\vec{\Phi}'(\vec{\xi})\| \ \|\vec{x} - \vec{y}\|$$

mit einem Zwischenpunkt $\vec{\xi} = \vec{x} + \epsilon_0 \, (\vec{y} - \vec{x})$, $\epsilon_0 \in (0, 1)$, *auf dem Geradenstück zwischen* $\vec{x}$ *und* $\vec{y}$.

Beweis: Mit $\vec{\Psi}(\epsilon) := \vec{\Phi}(\vec{x} + \epsilon(\vec{y} - \vec{x}))$ gilt

$$\|\vec{\Phi}(\vec{y}) - \vec{\Phi}(\vec{x})\| \ = \ \|\vec{\Psi}(1) - \vec{\Psi}(0)\| \ = \ \left\| \int_0^1 \frac{d}{d\epsilon} \vec{\Psi}(\epsilon) \ d\epsilon \right\|$$

$$= \left\| \int_0^1 \vec{\Phi}'(\vec{x} + \epsilon(\vec{y} - \vec{x})) \ (\vec{y} - \vec{x}) \ d\epsilon \right\| \ \le \int_0^1 \|\vec{\Phi}'(\vec{x} + \epsilon(\vec{y} - \vec{x}))\| \, d\epsilon \ \ \|\vec{y} - \vec{x}\| \ ,$$

wobei nach dem Mittelwertsatz der Integralrechnung ein $\epsilon_0 \in (0,1)$ mit

$$\int_0^1 \|\vec{\Phi}'(\vec{x}+\epsilon(\vec{y}-\vec{x}))\|\, d\epsilon \;=\; \|\vec{\Phi}'(\vec{x}+\epsilon_0(\vec{y}-\vec{x}))\|$$

existiert.

Q.E.D.

Damit sind Kontraktionskonstanten als Maxima von Ableitungsnormen berechenbar.

Satz 5.3: (Zur Berechnung von Kontraktionskonstanten)
Für konvexes $A \subset \mathbb{R}^n$ gilt mit $k := \sup_{\vec{\xi}\in A} \|\vec{\Phi}'(\xi)\|$ *:*

$$\|\vec{\Phi}(\vec{x}) - \vec{\Phi}(\vec{y})\| \le k\,\|\vec{x}-\vec{y}\| \quad \forall\, \vec{x},\vec{y} \in A\,.$$

Beweis: Der Zwischenpunkt $\vec{\xi}$ in Lemma 5.2 liegt für $\vec{x},\vec{y} \in A$ wieder in A.

Q.E.D.

Analog zu Satz 3.7 ist der BFS lokal anwendbar, wenn in irgendeiner Operatornorm $\|\vec{\Phi}'(\vec{x}^*)\| < 1$ gilt. Die Existenz einer geeigneten Norm ist nach Satz 4.58.c) garantiert, wenn $\rho(\vec{\Phi}'(\vec{x}^*)) < 1$ gilt.

Satz 5.4:
Sei $\vec{\Phi}: \mathbb{R}^n \to \mathbb{R}^n$ stetig diff'bar mit Fixpunkt $\vec{x}^* = \vec{\Phi}(\vec{x}^*)$.

a) Wenn $\rho(\vec{\Phi}'(\vec{x}^*)) < 1$ gilt, dann existieren eine abgeschlossene Umgebung Ω von $\vec{x}^*$ und eine Norm auf dem $\mathbb{R}^n$, bezüglich der $\vec{\Phi}$ eine Kontraktion mit $\vec{\Phi}(\Omega) \subset \Omega$ ist.

b) Gilt $\vec{\Phi}'(\vec{x}^*) = 0$ für zweifach stetig diff'bares $\vec{\Phi}$, dann gibt es eine Umgebung von $\vec{x}^*$, auf der

$$\|\vec{\Phi}(\vec{x}) - \vec{\Phi}(\vec{x}^*)\| \;\le\; c\,\|\vec{x}-\vec{x}^*\|^2$$

mit einer geeigneten Konstanten c gilt ("quadratische Konvergenz").

Beweis: a) analog zum Beweis von Satz 3.7 mit einer Norm gemäß Satz 4.58, für die $\|\vec{\Phi}'(\vec{x}^*)\| < 1$ gilt.

b) Sei Φ_i die i.te Komponente von $\vec{\Phi}$. Taylor-Entwicklung liefert

$$\Phi_i(\vec{x}) = \Phi_i(\vec{x}^*) + \sum_j \frac{\partial \Phi_i}{\partial x_j}(\vec{x}^*)\,(x_j - x_j^*) + \frac{1}{2}\sum_{j,k} \frac{\partial^2 \Phi_i}{\partial x_j \partial x_k}(\vec{\xi}_i)\,(x_j - x_j^*)(x_k - x_k^*)$$

mit Zwischenpunkten $\vec{\xi}_i$ auf dem Geradenstück zwischen $\vec{x}$ und $\vec{x}^*$. Es folgt

$$|\Phi_i(\vec{x}) - \Phi_i(\vec{x}^*)| \;\leq\; \frac{1}{2} \sum_{j,k} \left| \frac{\partial^2 \Phi_i}{\partial x_j \partial x_k}(\vec{\xi}_i) \right| \; \Big(\max_j |x_j - x_j^*| \Big)^2 .$$

Die zweiten partiellen Ableitungen sind auf einer Umgebung von $\vec{x}^*$ beschränkt, so daß

$$\|\vec{\Phi}(\vec{x}) - \vec{\Phi}(\vec{x}^*)\|_\infty \;\leq\; c\, \|\vec{x} - \vec{x}^*\|_\infty^2$$

mit geeignetem c gilt. Jede beliebige Norm ist äquivalent zu $\|\,.\,\|_\infty$.

Q.E.D.

Das vektorwertige Analogon des skalaren Newton-Verfahrens 3.18 ist

Satz 5.5: (Das Newton-Verfahren)

Es sei $\vec{x}^*$ Nullstelle der dreifach stetig diff'baren Funktion $\vec{f} : \mathbb{R}^n \to \mathbb{R}^n$ mit invertierbarem $\vec{f}'(\vec{x}^*)$. Dann existiert eine Umgebung $U(\vec{x}^*)$, so daß das **Newton-Verfahren** $\vec{x}^{(i+1)} = \vec{\Phi}(\vec{x}^{(i)})$ *mit*

$$\vec{\Phi}(\vec{x}) \;=\; \vec{x} - (\vec{f}'(\vec{x}))^{-1} \vec{f}(\vec{x})$$

für alle Startwerte aus $U(\vec{x}^*)$ quadratisch gegen $\vec{x}^*$ konvergiert.

Beweis: Mit Satz 5.4 ist nur $\vec{\Phi}'(\vec{x}^*) = 0$ zu zeigen. Aus

$$\vec{f}'(\vec{x})\, \vec{\Phi}(\vec{x}) \;=\; \vec{f}'(\vec{x})\, \vec{x} - \vec{f}(\vec{x}) \;, \quad \text{d.h.,} \quad \sum_{j=1}^n \frac{\partial f_i}{\partial x_j}\, \Phi_j \;=\; \sum_{j=1}^n \frac{\partial f_i}{\partial x_j}\, x_j - f_i$$

folgt

$$\sum_{j=1}^n \frac{\partial^2 f_i}{\partial x_j \partial x_k}\, \Phi_j \;+\; \sum_{j=1}^n \frac{\partial f_i}{\partial x_j}\, \frac{\partial \Phi_j}{\partial x_k} \;=\; \sum_{j=1}^n \frac{\partial^2 f_i}{\partial x_j \partial x_k}\, x_j \;,$$

d.h.,

$$\sum_{j=1}^n \frac{\partial f_i}{\partial x_j}\, \frac{\partial \Phi_j}{\partial x_k} \;=\; \sum_{j=1}^n \frac{\partial^2 f_i}{\partial x_j \partial x_k}\, (x_j - \Phi_j) \;.$$

Mit $\vec{x}^* = \vec{\Phi}(\vec{x}^*)$ ergibt sich hieraus $\vec{f}'(\vec{x}^*)\, \vec{\Phi}'(\vec{x}^*) = 0$ und damit $\vec{\Phi}'(\vec{x}^*) = 0$.

Q.E.D.

Bemerkung 5.6: *Für nicht invertierbares $\vec{f}'(\vec{x}^*)$ konvergiert die Newton-Folge i.a. nicht (Gegenbeispiele sind leicht konstruierbar).*

Bemerkung 5.7: *Der wesentliche Aufwand ist die Auswertung und Invertierung von $\vec{f}'(\vec{x})$. Für große Systeme bietet es sich an, mit einer festen Matrix $C \approx (\vec{f}'(\vec{x}^*))^{-1}$ zu arbeiten:*

$$\vec{\Phi}(\vec{x}) \;=\; \vec{x} - C\vec{f}(\vec{x}) \;.$$

Dies liefert zwar nur lineare Konvergenz mit der (lokalen) Kontraktionskonstanten $\|\vec{\Phi}'(\vec{x}^*)\| = \|\mathbb{1} - C\vec{f}'(\vec{x}^*)\|$, dafür ist jeder einzelne Schritt wesentlich schneller durchführbar.

$\boxed{20.1.98\downarrow}$ **Bemerkung 5.8:** *Da mit Satz 5.5 nur lokale Konvergenz garantiert ist, braucht man gute Startwerte. Dies ist in höheren Dimensionen ein großes Problem! Eine mögliche Strategie zur "Globalisierung" ist das* **schrittweitengesteuerte Newton-Verfahren***:*

$$\vec{x}^{(i+1)} \;:=\; \vec{x}^{(i)} \;-\; t^{(i)}\;(\vec{f}'(\vec{x}^{(i)}))^{-1}\;\vec{f}(\vec{x}^{(i)}) \;.$$

Hierbei wird $t^{(i)} = 1, \frac{1}{2}, \frac{1}{4}, \ldots$ ausprobiert, bis (für $\vec{f}(\vec{x}^{(i)}) \neq 0$) zum ersten Mal

$$\|\vec{f}(\vec{x}^{(i+1)})\|_2 < \|\vec{f}(\vec{x}^{(i)})\|_2$$

gilt. Dieses $\vec{x}^{(i+1)}$ wird akzeptiert.

Erklärung:

1) Mit $F(\vec{x}) = \langle \vec{f}(\vec{x}), \vec{f}(\vec{x})\rangle$ und $\vec{x}(t) := \vec{x}^{(i)} - t\;(\vec{f}'(\vec{x}^{(i)}))^{-1}\vec{f}(\vec{x}^{(i)})$ gilt

$$\frac{d}{dt}F(\vec{x}(t)) \;=\; -\;2\;\langle \vec{f}(\vec{x}(t))\;,\;\vec{f}'(\vec{x}(t))\;(\vec{f}'(\vec{x}^{(i)}))^{-1}\vec{f}(\vec{x}^{(i)})\;\rangle \;.$$

Damit ist $F(\vec{x}(t))$ für stetig diff'bares $\vec{f}$ in einer Umgebung von $t = 0$ streng monoton fallend, falls $\vec{x}^{(i)}$ nicht schon Nullstelle ist:

$$\frac{d}{dt}F(\vec{x}(t))_{|t=0} \;=\; -\;2\;\langle \vec{f}(\vec{x}^{(i)}), \vec{f}(\vec{x}^{(i)})\rangle \;\leq\; 0 \;.$$

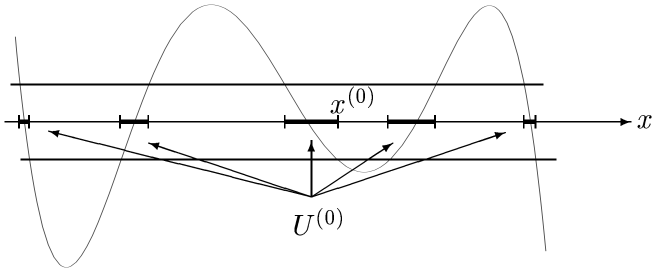
Die Suche nach $t^{(i)} \in \{1, \frac{1}{2}, \frac{1}{4}, \ldots\}$ führt damit garantiert nach endlich vielen Versuchen zum Erfolg!

2) Der Aufwand der Suche nach der Schrittweite $t^{(i)}$ ist akzeptabel: der Hauptaufwand des Schrittes $\vec{x}^{(i)} \to \vec{x}^{(i+1)}$ ist die Berechnung der "Suchrichtung" $(\vec{f}'(\vec{x}^{(i)}))^{-1}\;\vec{f}(\vec{x}^{(i)})$. Für jeden Versuch $t^{(i)} = 1, 1/2, \ldots$ fällt dann nur die Auswertung von $\|\vec{f}(\vec{x}^{(i+1)})\|_2$ zum Vergleich mit $\|\vec{f}(\vec{x}^{(i)})\|_2$ an.

3) Nach Konstruktion gilt $U^{(0)} \underset{\neq}{\supset} U^{(1)} \underset{\neq}{\supset} U^{(2)} \underset{\neq}{\supset} \;\ldots$ mit

$$U^{(i)} \;=\; \{\;\vec{x} \in \mathbb{R}^n\;;\;\|\vec{f}(\vec{x})\|_2 \leq \|\vec{f}(\vec{x}^{(i)})\|_2\;\}\;.$$

Mit $\vec{x}^{(i)} \in U^{(i)}$ vollzieht sich die Iteration damit auf immer kleineren Gebieten, die sämtliche Nullstellen von $\vec{f}$ enthalten:



Falls $U^{(0)}$ beschränkt –und damit kompakt– ist, so hat die Newton-Folge einen Häufungspunkt $\vec{x}^*$, für den man (unter technischen Zusatzvoraussetzungen, siehe z.B. [Sto93, Satz 5.4.2.5]) $\vec{f}'(\vec{x}^*)^T \vec{f}(\vec{x}^*) = 0$ zeigen kann. Die Bedingung

"$\vec{x}^{(0)}$ hinreichend nahe an einer Nullstelle"

beim üblichen Newton-Verfahren wird damit durch die wesentlich schwächere Bedingung

"$U^{(0)}$ beschränkt (und damit kompakt)"

ersetzt.

4) In der Nähe einer Nullstelle werden wegen

$$\lim_{\vec{x} \to \vec{x}^*} \frac{\| \vec{f}(\vec{x} - (\vec{f}'(\vec{x}))^{-1} \vec{f}(\vec{x}))\|_2}{\| \vec{f}(\vec{x})\|_2} = 0$$

die Schrittweiten $t^{(i)} = 1$ gefunden. Damit erhält man in der Endphase das normale Newton-Verfahren mit quadratischer Konvergenz.

Bemerkung 5.9: *Eine Anwendung des Newton-Verfahrens auf dem $\mathbb{R}^2$ ist das* **Bairstow-Verfahren** *zur Suche nach (eventuell komplexen) Nullstellenpaaren von Polynomen (siehe z.B. [Scw93] oder [Oev96]).*

Bemerkung 5.10: *Weiteres zu nichtlinearen Systemen:*

J.M. Orthega und W.C. Rheinboldt: *Iterative solution of nonlinear equations in several variables, Academic Press, New York 1970.*

Viele Strategien versuchen, die (globalen) Minima von $F(\vec{x}) = \langle \vec{f}(\vec{x}), \vec{f}(\vec{x})\rangle$ zu suchen: **nichtlineare Optimierung**.

Kapitel 6

Eigenwertprobleme

Zu einer $n \times n$-Matrix A sind **Eigenwerte** $\lambda \in \mathbb{C}$ und **Eigenvektoren** $\vec{x} \in \mathbb{C}^n$, $\vec{x} \neq 0$, mit $A\vec{x} = \lambda\vec{x}$ gesucht.

Zusammenfassung der wichtigsten Fakten aus der Linearen Algebra: Die Eigenwerte sind die Nullstellen des **charakteristischen Polynoms**

$$p(\lambda) \;=\; \det(A - \lambda\mathbb{1}) \;=\; (-1)^n \; (\lambda - \lambda_1)^{a_1} \cdots (\lambda - \lambda_k)^{a_k} \; .$$

Die **algebraische Vielfachheit** von λ_i ist a_i. Der **Eigenraum** zu λ_i ist

$$E_{\lambda_i} \;=\; \{ \; \vec{x} \in \mathbb{C}^n \; ; \; A\vec{x} = \lambda_i\vec{x} \; \} \; .$$

Die **geometrische Vielfachheit** von λ_i ist $\dim E_{\lambda_i}$. Es gilt stets

$$a_i \;\geq\; \dim E_{\lambda_i} \;\geq\; 1 \; .$$

Wenn für alle Eigenwerte $a_i = \dim E_{\lambda_i}$ gilt, dann existiert eine Basis $\vec{x}_1, \ldots, \vec{x}_n$ des $\mathbb{C}^n$ aus Eigenvektoren. Mit $T = (\vec{x}_1, \ldots, \vec{x}_n)$ gilt

$$A \;=\; T \,\mathrm{diag}(\lambda_1, \ldots, \lambda_n)\, T^{-1} \quad \textbf{(Diagonalisierung)}.$$

Jedes symmetrische reelle A ist diagonalisierbar, es existiert eine Basis des $\mathbb{R}^n$ aus **orthonormalen** Eigenvektoren (d.h., T ist orthogonal), die Eigenwerte und -vektoren sind reell.
Wichtige Tatsache: die Eigenwerte sind stetige Funktionen der Matrixkomponenten! Einfache Eigenwerte hängen sogar analytisch von den Matrixdaten ab.

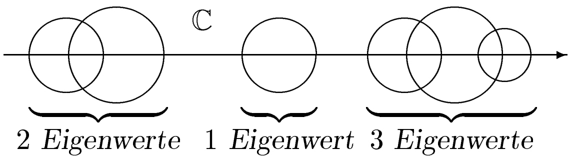
6.1 Allgemeine Abschätzungen

Satz 6.1: (Gerschgorin-Kriterium)

a) Alle Eigenwerte der $n \times n$-Matrix $A = (a_{ij})$ liegen in der Vereinigung der Kreisscheiben

$$K_i \;=\; \Big\{ z \in \mathbb{C} \;;\; |z - a_{ii}| \;\le\; \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}| \Big\} \,, \quad i = 1, \ldots, n \,.$$

b) Es sei $\{i_1, \ldots, i_k\} \cup \{i_{k+1}, \ldots, i_n\} =: I_1 \cup I_2 = \{1, \ldots, n\}$. Sind $\bigcup_{i \in I_1} K_i$ und $\bigcup_{i \in I_2} K_i$ disjunkt, dann liegen in $\bigcup_{i \in I_1} K_i$ genau k und in $\bigcup_{i \in I_2} K_i$ genau $n - k$ Eigenwerte (mit ihrer algebraischen Vielfachheit gezählt).



Beweis: a) Sei $\vec{x} = (x_1, \ldots, x_n)^T$ Eigenvektor, sei $|x_k| = \max_j |x_j|$. Aus $\lambda x_k = \sum_j a_{kj} x_j$ folgt

$$|\lambda - a_{kk}|\, |x_k| \;=\; \Big| \sum_{\substack{j=1 \\ j \neq k}}^{n} a_{kj}\, x_j \Big| \;\le\; \Big(\sum_{\substack{j=1 \\ j \neq k}}^{n} |a_{kj}| \Big)\, |x_k| \,, \quad \text{d.h.,} \quad \lambda \in K_k \,.$$

b) Sei D die Diagonale von A und $N = A - D$, sei $A(\epsilon) = D + \epsilon N$ mit den Eigenwerten $\lambda(\epsilon)$. Für $\epsilon = 0$ bestehen die Kreise $K_i(\epsilon)$ aus den durch die Diagonalelemente gegebenen Punkten, die beim stetigen Vergrößern von $\epsilon = 0$ auf $\epsilon = 1$ zu den Gerschgorin-Kreisen $K_i = K_i(1)$ von A anwachsen (die Radien sind proportional zu ϵ). Die Eigenwerte hängen stetig von ϵ ab und können nach a) nicht zwischen $\bigcup_{i \in I_1} K_i(\epsilon)$ und $\bigcup_{i \in I_2} K_i(\epsilon)$ wechseln.

Q.E.D.

Ein anderer Weg zu Eigenwertabschätzungen basiert auf "Testvektoren".

Definition 6.2:

$\boxed{22.1.98\downarrow}$ *Der* ***Rayleigh-Quotient*** *von $\vec{x} \neq 0$ bezüglich der Matrix A ist*

$$r_A(\vec{x}) \;=\; \frac{\langle \vec{x}, A\vec{x} \rangle}{\langle \vec{x}, \vec{x} \rangle} \,.$$

Für komplexes A oder komplexes $\vec{x}$ ist hierbei das komplexe Skalarprodukt $\langle \vec{x}, \vec{y} \rangle = \sum_i \bar{x}_i\, y_i$ gemeint, das für reelle Daten zum üblichen Euklidischen Skalarprodukt auf dem $\mathbb{R}^n$ wird.

Satz 6.3:

Sei A reell und symmetrisch, $\vec{x} \in \mathbb{R}^n \setminus \{0\}$ beliebig.

a) *Mit dem kleinsten bzw. größten Eigenwert λ_{min} bzw. λ_{max} von A gilt*

$$\lambda_{min} \;\leq\; r_A(\vec{x}) \;\leq\; \lambda_{max} \; .$$

Die Extremwerte werden dabei offensichtlich für die entsprechenden Eigenvektoren angenommen.

b) *Eigenwertabschätzung durch den Rayleigh-Quotienten eines Testvektors: es existiert ein Eigenwert λ von A mit*

$$|\lambda - r_A(\vec{x})|^2 \;\leq\; \underbrace{r_{A^2}(\vec{x}) - (r_A(\vec{x}))^2}_{\text{Vorsicht: Auslöschung!}} \;=\; \underbrace{\frac{\|(A - r_A(\vec{x})\mathbb{1})\,\vec{x}\|_2^2}{\langle \vec{x}, \vec{x} \rangle}}_{\text{numerisch stabile Darstellung}} \; .$$

Beweis: a) Sei $\vec{x}_1, \ldots, \vec{x}_n$ eine Orthonormalbasis von Eigenvektoren: $A\vec{x}_i = \lambda_i \vec{x}_i$. Mit $\vec{x} = \sum_i x_i \, \vec{x}_i$ folgt

$$r_A(\vec{x}) \;=\; \frac{\langle \sum_i x_i \, \vec{x}_i, \sum_i \lambda_i \, x_i \, \vec{x}_i \rangle}{\langle \sum_i x_i \, \vec{x}_i, \sum_i x_i \, \vec{x}_i \rangle} \;=\; \frac{\sum_i \lambda_i \, x_i^2}{\sum_i x_i^2} \;\; \begin{cases} \leq \; \lambda_{max} \; , \\ \geq \; \lambda_{min} \; . \end{cases}$$

b) Sei μ nicht Eigenwert von A:

$$1 \;=\; \frac{\|(A - \mu\,\mathbb{1})^{-1}(A - \mu\,\mathbb{1})\,\vec{x}\|_2^2}{\|\vec{x}\|_2^2} \;\leq\; \|(A - \mu\,\mathbb{1})^{-1}\|_2^2 \; \frac{\|(A - \mu\,\mathbb{1})\,\vec{x}\|_2^2}{\|\vec{x}\|_2^2} \; .$$

Mit

$$\begin{aligned} \frac{\|(A - \mu\mathbb{1})\,\vec{x}\|_2^2}{\langle \vec{x}, \vec{x} \rangle} \;&\geq\; \frac{1}{\|(A - \mu\,\mathbb{1})^{-1}\|_2^2} \;=\; \frac{1}{\rho((A - \mu\,\mathbb{1})^{-1})^2} \\ &=\; \frac{1}{\max\limits_{i=1\ldots n} |\lambda_i - \mu|^{-2}} \;=\; \min_{i=1\ldots n} |\lambda_i - \mu|^2 \end{aligned}$$

folgt (nun auch für μ = Eigenwert):

$$\frac{\langle (A - \mu\,\mathbb{1})\,\vec{x}, (A - \mu\,\mathbb{1})\,\vec{x} \,\rangle}{\langle \vec{x}, \vec{x} \,\rangle} = \frac{\|(A - \mu\mathbb{1})\,\vec{x}\|_2^2}{\langle \vec{x}, \vec{x} \rangle} \;\geq\; \min_{i=1\ldots n} |\lambda_i - \mu|^2 \; .$$

Mit

$$\frac{\langle (A - \mu\,\mathbb{1})\,\vec{x}, (A - \mu\,\mathbb{1})\,\vec{x} \,\rangle}{\langle \vec{x}, \vec{x} \,\rangle} \;=\; \underbrace{\frac{\langle A\vec{x}, A\vec{x} \,\rangle}{\langle \vec{x}, \vec{x} \,\rangle}}_{r_{A^2}(\vec{x})} - \Big(\underbrace{\frac{\langle \vec{x}, A\vec{x} \,\rangle}{\langle \vec{x}, \vec{x} \,\rangle}}_{r_A(\vec{x})} \Big)^2 + \left(\mu - \frac{\langle \vec{x}, A\vec{x} \,\rangle}{\langle \vec{x}, \vec{x} \,\rangle} \right)^2$$

wird die Abschätzung für $\mu = r_A(\vec{x})$ optimal.

Q.E.D.

Bemerkung 6.4: *Wenn $\vec{x}$ ein Eigenvektor ist, dann liefert $r_A(\vec{x})$ den entsprechenden Eigenwert. Der Rayleigh-Quotient ist ein Hilfsmittel, aus einer Approximation eines Eigenvektors eine Approximation des Eigenwerts abzuleiten.*

Bemerkung 6.5: *Wann ist $\vec{x} \in \mathbb{R}^n$ (bzw. $\mathbb{C}^n$) Approximation eines Eigenvektors? Es sei E der Eigenraum zum Eigenwert λ und F der von den restlichen Eigen- (bzw. Haupt-)vektoren aufgespannte Raum, so daß $\mathbb{R}^n = E \oplus F$ (bzw. $\mathbb{C}^n = E \oplus F$). Zerlege $\vec{x} = \vec{x}_E + \vec{x}_F$ mit $\vec{x}_E \in E$, $\vec{x}_F \in F$. Dann bedeute:*

$$\vec{x} \text{ ist approximativer Eigenvektor zum Eigenwert } \lambda$$
$$\Longleftrightarrow \quad \vec{x} \approx \vec{x}_E \quad \Longleftrightarrow \quad \|\vec{x}_F\| \ll \|\vec{x}_E\| \ .$$

Für symmetrische Matrizen ist F das orthogonale Komplement von E: $\langle \vec{x}_E, \vec{x}_F \rangle = 0$. Der Winkel ϕ zwischen $\vec{x}$ und seiner orthogonalen Projektion $\vec{x}_E$ auf den Eigenraum E, definiert durch

$$\cos^2(\phi) \ = \ \frac{\langle \vec{x}_E, \vec{x}_E \rangle}{\langle \vec{x}, \vec{x} \rangle} \quad \text{bzw.} \quad \sin^2(\phi) \ = \ \frac{\langle \vec{x}_F, \vec{x}_F \rangle}{\langle \vec{x}, \vec{x} \rangle} \ ,$$

ist ein Maß für den "Abstand von $\vec{x}$ zum Eigenraum E". Für symmetrisches A mit den Eigenwerten λ_i gilt

$$\Big(\min_{\lambda_i \neq \lambda} |\lambda - \lambda_i| \Big) \ \sin^2(\phi) \ \leq \ |\lambda - r_A(\vec{x})| \ \leq \ \Big(\max_{\lambda_i} |\lambda - \lambda_i| \Big) \ \sin^2(\phi) \ . \qquad (\#)$$

Damit hängt die Approximation eines Eigenwertes durch den Rayleigh-Quotienten quadratisch vom durch den Winkel ϕ gemessenen "Abstand des Testvektors $\vec{x}$ zum Eigenraum" ab.

Beweis von (#):

$$\begin{aligned}
\lambda \ - \ r_A(\vec{x}) \ &= \ \lambda \ - \ \frac{\langle \vec{x}_E + \vec{x}_F, A(\vec{x}_E + \vec{x}_F) \rangle}{\langle \vec{x}, \vec{x} \rangle} \\
&= \ \lambda \ - \ \frac{\langle \vec{x}_E, A\vec{x}_E \rangle}{\langle \vec{x}, \vec{x} \rangle} \ - \ 2 \ \frac{\langle \vec{x}_F, A\vec{x}_E \rangle}{\langle \vec{x}, \vec{x} \rangle} \ - \ \frac{\langle \vec{x}_F, A\vec{x}_F \rangle}{\langle \vec{x}, \vec{x} \rangle} \\
&= \ \lambda \left(1 \ - \ \frac{\langle \vec{x}_E, \vec{x}_E \rangle}{\langle \vec{x}, \vec{x} \rangle} \right) \ - \ \frac{\langle \vec{x}_F, \vec{x}_F \rangle}{\langle \vec{x}, \vec{x} \rangle} \ \frac{\langle \vec{x}_F, A\vec{x}_F \rangle}{\langle \vec{x}_F, \vec{x}_F \rangle} \\
&= \ \sin^2(\phi) \ \left(\lambda \ - \ \frac{\langle \vec{x}_F, A\vec{x}_F \rangle}{\langle \vec{x}_F, \vec{x}_F \rangle} \right) \ .
\end{aligned}$$

Der auf F eingeschränkte Rayleigh-Quotient nimmt in Analogie zu Satz 6.3.a) als Extremwerte einen der von λ verschiedenen Eigenwerte von A an:

$$\min_{\lambda_i \neq \lambda} \lambda_i \leq \frac{\langle \vec{x}_F, A\vec{x}_F \rangle}{\langle \vec{x}_F, \vec{x}_F \rangle} \leq \max_{\lambda_i \neq \lambda} \lambda_i \ .$$

Q.E.D.

Merke: Bei symmetrischen Matrizen können auch mäßige Eigenvektorapproximationen durch den Rayleigh-Quotienten gute Eigenwertapproximationen liefern.

Satz 6.6: (Abhängigkeit der Eigenwerte von den Matrixdaten)
Sei $A = T\,\mathrm{diag}(\lambda_1, \ldots, \lambda_n)\,T^{-1}$ eine diagonalisierbare $n \times n$-Matrix mit Eigenwerten $\lambda_1, \ldots, \lambda_n$ und diagonalisierender Transformationsmatrix T. Für einen beliebigen Eigenwert $\tilde{\lambda}$ einer beliebigen anderen Matrix $\tilde{A}$ gilt bezüglich der p-Normen

$$\min_{i=1..n} |\lambda_i - \tilde{\lambda}| \;\leq\; \mathrm{cond}_p(T)\; \|A - \tilde{A}\|_p \;.$$

Beweis: Sei $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$, $\tilde{\lambda}$ sei nicht Eigenwert von A. Es folgt

$$\|(A - \tilde{\lambda}\mathbb{1})^{-1}\|_p \;=\; \|T(\Lambda - \tilde{\lambda}\mathbb{1})^{-1}T^{-1}\|_p \;\leq\; \mathrm{cond}_p(T)\; \|(\Lambda - \tilde{\lambda}\mathbb{1})^{-1}\|_p \;.$$

Die p-Norm einer Diagonalmatrix ist der maximale Betrag der Diagonalelemente:

$$\|(\Lambda - \tilde{\lambda}\mathbb{1})^{-1}\|_p \;=\; \max_{i=1..n} \frac{1}{|\lambda_i - \tilde{\lambda}|} \;=\; \frac{1}{\min\limits_{i=1..n} |\lambda_i - \tilde{\lambda}|} \;.$$

Es folgt

$$\min_{i=1..n} |\lambda_i - \tilde{\lambda}| \;\leq\; \frac{\mathrm{cond}_p(T)}{\|(A - \tilde{\lambda}\mathbb{1})^{-1}\|_p} \;. \qquad (\#)$$

Mit einem Eigenvektor $\vec{y}$ von $\tilde{A}$ zu $\tilde{\lambda}$ gilt

$$\tilde{A}\vec{y} = \tilde{\lambda}\vec{y} \quad\Longrightarrow\quad (A - \tilde{A})\vec{y} = (A - \tilde{\lambda}\mathbb{1})\vec{y} \quad\Longrightarrow\quad (A - \tilde{\lambda}\mathbb{1})^{-1}(A - \tilde{A})\,\vec{y} = \vec{y}$$

und damit

$$1 \;\leq\; \|(A - \tilde{\lambda}\mathbb{1})^{-1}(A - \tilde{A})\|_p \;\leq\; \|(A - \tilde{\lambda}\mathbb{1})^{-1}\|_p\, \|A - \tilde{A}\|_p \;.$$

Mit $\|(A - \tilde{\lambda}\mathbb{1})^{-1}\|_p^{-1} \leq \|A - \tilde{A}\|_p$ und (#) folgt die Behauptung.

Q.E.D.

Bemerkung 6.7: *Für diagonalisierbare Matrizen ist die Störung der Eigenwerte linear in der Störung der Matrixdaten. Das Eigenwertproblem ist gutmütig, solange nicht Eigenräume "fast parallel" sind (d.h., solange $\mathrm{cond}_p(T)$ nicht zu groß ist, vergleiche Aufgabe 30).*

Bemerkung 6.8: *Symmetrische Matrizen können mit orthogonalem T diagonalisiert werden. Mit*

$$\mathrm{cond}_2(T) \overset{(4.62)}{=} \frac{\max\limits_{\|\vec{x}\|_2=1} \|T\vec{x}\|_2}{\min\limits_{\|\vec{x}\|_2=1} \|T\vec{x}\|_2} = \frac{\max\limits_{\|\vec{x}\|_2=1} \|\vec{x}\|_2}{\min\limits_{\|\vec{x}\|_2=1} \|\vec{x}\|_2} = 1$$

folgt $\min\limits_{i=1..n} |\lambda_i - \tilde{\lambda}| \le \|A - \tilde{A}\|_2$ *für symmetrisches A und beliebiges $\tilde{A}$.*

> **Merke:** *Das Eigenwertproblem symmetrischer Matrizen ist <u>stets</u> (auch bei Entartung) gutmütig!*

Satz 6.9: (Vergleichssatz für die Eigenwerte symmetrischer Matrizen)

Für symmetrische reelle $n \times n$-Matrizen A und $\tilde{A}$ mit den Eigenwerten

$$\lambda_1 \le \ldots \le \lambda_n \quad \text{von } A \quad \text{bzw.} \quad \tilde{\lambda}_1 \le \ldots \le \tilde{\lambda}_n \quad \text{von } \tilde{A}$$

gilt $\quad |\lambda_i - \tilde{\lambda}_i| \le \rho(A - \tilde{A}) \le \|A - \tilde{A}\|$ *(für jede Operatornorm).*

Beweis: Es seien $\vec{x}_1, \ldots, \vec{x}_n$ bzw. $\vec{y}_1, \ldots, \vec{y}_n$ orthonormale Eigenvektorbasen des $\mathbb{R}^n$ von A bzw. $\tilde{A}$. Es sei $E_{i..n}$ der von $\vec{x}_i, \ldots, \vec{x}_n$ aufgespannte Teilraum des $\mathbb{R}^n$. In Analogie zu Satz 6.3.a) gilt für den auf $E_{i..n}$ eingeschränkten Rayleigh-Quotienten

$$\min_{\substack{\vec{x} \in E_{i..n} \\ \vec{x} \neq 0}} \frac{\langle \vec{x}, A\vec{x} \rangle}{\langle \vec{x}, \vec{x} \rangle} = \min_{(x_i, \ldots, x_n) \neq 0} \frac{\sum\limits_{k=i}^{n} \lambda_k x_k^2}{\sum\limits_{k=i}^{n} x_k^2} = \lambda_i \ ,$$

wobei ein beliebiges $\vec{x} \in E_{i..n} \setminus \{0\}$ nach der Basis $\vec{x}_i, \ldots, \vec{x}_n$ zerlegt wurde: $\vec{x} = \sum\limits_{k=i}^{n} x_k \vec{x}_k$. Es sei $F_{1..i}$ der von $\vec{y}_1, \ldots, \vec{y}_i$ aufgespannte Teilraum des $\mathbb{R}^n$. Ein beliebiges $\vec{x} \in (E_{i..n} \cap F_{1..i}) \setminus \{0\}$ wird nach der Basis $\vec{y}_1, \ldots, \vec{y}_i$ von $F_{1..i}$ zerlegt[1]: $\vec{x} = \sum\limits_{k=1}^{i} y_k \vec{y}_k$. Es folgt

[1]Die Koeffizienten y_k sind dabei durch die Bedingung $\vec{x} \in E_{i..n}$ eingeschränkt. Da $E_{i..n}$ das orthogonale Komplement des von $\vec{x}_1, \ldots, \vec{x}_{i-1}$ aufgespannten Teilraums ist, haben die y_k das Gleichungssystem $\sum_{k=1}^{i} y_k \langle \vec{x}_j, \vec{y}_k \rangle = 0$ mit $j = 1, \ldots, i-1$ zu lösen. Es gibt sicherlich nichttriviale Lösungen dieser $i-1$ homogenen linearen Gleichungen für die i Unbestimmten $y_1, \ldots, y_i$, so daß der Teilraum $E_{i..n} \cap F_{1..i}$ nicht nur den Nullvektor enthält. Damit existiert ein nichttrivialer Vektor $\vec{x} \in E_{i..n} \cap F_{1..i}$.

$$\min_{\substack{\vec{x}\in E_{i..n}\\ \vec{x}\neq 0}} \frac{\langle \vec{x}, \tilde{A}\vec{x}\rangle}{\langle \vec{x}, \vec{x}\rangle} \le \min_{\substack{\vec{x}\in E_{i..n}\cap F_{1..i}\\ \vec{x}\neq 0}} \frac{\langle \vec{x}, \tilde{A}\vec{x}\rangle}{\langle \vec{x}, \vec{x}\rangle} \le \max_{(y_1,\ldots,y_i)\neq 0} \frac{\sum\limits_{k=1}^{i} \tilde{\lambda}_k y_k^2}{\sum\limits_{k=1}^{i} y_k^2} = \tilde{\lambda}_i$$

und damit

$$\begin{aligned} \tilde{\lambda}_i &\ge \min_{\substack{\vec{x}\in E_{i..n}\\ \vec{x}\neq 0}} \frac{\langle \vec{x}, A\vec{x}\rangle - \langle \vec{x}, (A-\tilde{A})\vec{x}\rangle}{\langle \vec{x}, \vec{x}\rangle} \\ &\ge \min_{\substack{\vec{x}\in E_{i..n}\\ \vec{x}\neq 0}} \frac{\langle \vec{x}, A\vec{x}\rangle}{\langle \vec{x}, \vec{x}\rangle} - \max_{\substack{\vec{x}\in E_{i..n}\\ \vec{x}\neq 0}} \frac{\langle \vec{x}, (A-\tilde{A})\vec{x}\rangle}{\langle \vec{x}, \vec{x}\rangle} \\ &\ge \lambda_i - \max_{\substack{\vec{x}\in \mathbb{R}^n\\ \vec{x}\neq 0}} \frac{\langle \vec{x}, (A-\tilde{A})\vec{x}\rangle}{\langle \vec{x}, \vec{x}\rangle} \ge \lambda_i - \rho(A-\tilde{A}) \ , \end{aligned}$$

also $\rho(A-\tilde{A}) \ge \lambda_i - \tilde{\lambda}_i$. Mit vertauschten Rollen von A und $\tilde{A}$ ergibt sich analog $\rho(\tilde{A}-A) \ge \tilde{\lambda}_i - \lambda_i$ und damit insgesamt $\rho(A-\tilde{A}) = \rho(\tilde{A}-A) \ge |\lambda_i - \tilde{\lambda}_i|$.

Q.E.D.

Bemerkung 6.10: *Wählt man $\tilde{A}$ als Diagonale von A, und schätzt man den Spektralradius durch die Zeilensummennorm ab, so ergibt sich*

$$\begin{aligned} |\lambda_i - d_i| \ \le\ \|A-\tilde{A}\|_\infty \ &=\ \max_{k=1..n} \sum_{\substack{j=1\\ j\neq k}} |a_{kj}| \ =:\ r \\ &=\ \textit{Maximum der Gerschgorin-Radien}, \end{aligned}$$

wobei $d_1 \le d_2 \le \ldots$ eine Anordnung der Diagonale von A ist. Das Gerschgorin-Kriterium liefert für separierende Gerschgorin-Kreise bessere Abschätzungen, da jedem Diagonalelement sein individueller Radius zugeordnet ist. Dafür liefert Satz 6.9 bei überlappenden Kreisen bessere Ergebnisse, da durch die Anordnung jedem Diagonalelement genau ein Eigenwert mit Abstand $\le r$ zugeordnet werden kann.

6.2 Die von-Mises-Iteration

Ziel: berechne den Spektralradius einer Matrix durch Konstruktion einer Vektorfolge, die gegen den entsprechenden Eigenraum konvergiert.
Konzeptionelles Problem: es gibt keinen eindeutigen Eigenvektor als Kandidat für den Grenzwert. Statt "Konvergenz gegen einen Eigenvektor" betrachte "Konvergenz gegen einen Eigenraum".

Definition 6.11:

Es sei $\mathbb{C}^n = E \oplus F$ *durch 2 Unterräume aufgespannt. Eine Vektorfolge* $\vec{x}^{(i)} \in \mathbb{C}^n$ *heißt* **konvergent gegen** E, *wenn mit der eindeutigen Zerlegung*

$$\vec{x}^{(i)} = \vec{x}_E^{(i)} + \vec{x}_F^{(i)} \ , \quad \vec{x}_E^{(i)} \in E \ , \ \vec{x}_F^{(i)} \in F$$

in einer beliebigen Norm $\lim_{i\to\infty} \|\vec{x}_F^{(i)}\|/\|\vec{x}^{(i)}\| = 0$ *gilt.*

Bemerkung 6.12: *"Konvergenz gegen einen Unterraum" hat nichts mit dem üblichen Konvergenzbegriff für Vektoren zu tun. Die Folgen* $\vec{x}^{(i)}$, $\vec{x}_E^{(i)}$, $\vec{x}_F^{(i)}$ *brauchen nicht im üblichen Sinne zu konvergieren!*

Bemerkung 6.13: *Die Konvergenz gegen* E *ist unabhängig von der Wahl des Komplementärraums* F *(Aufgabe 36).*

Satz 6.14: (von-Mises-Iteration zur Bestimmung des Spektralradius)

$\boxed{27.1.98\downarrow}$ *Sei* A *eine diagonalisierbare* $n\times n$*-Matrix mit den paarweise verschiedenen (evtl. entarteten) Eigenwerten* $\lambda_1, \ldots, \lambda_k$, *die gemäß*

$$|\lambda_1| \ \le \ \ldots \ \le \ |\lambda_{k-1}| \ < \ |\lambda_k| \ = \ \rho(A)$$

angeordnet seien (d.h., der betragsgrößte Eigenwert λ_k *darf entartet, aber nicht betragsmäßig entartet sein). Die* **von-Mises-Folge**

$$\vec{x}^{(i+1)} \ = \ \frac{1}{\rho^{(i)}} \, A\vec{x}^{(i)} \ , \quad \rho^{(i)} \ = \ \|A\vec{x}^{(i)}\|$$

konvergiert für fast jedes $\vec{x}^{(0)} \in \mathbb{C}^n$ *gegen den zu* λ_k *gehörenden Eigenraum und*

$$\lim_{i\to\infty} \rho^{(i)} \ = \ \rho(A) \ .$$

Die Konvergenz ist linear mit dem Konvergenzfaktor $|\lambda_{k-1}|/|\lambda_k|$.

Beweis: Offensichtlich gilt $\vec{x}^{(i)} = A^i\vec{x}^{(0)}/\|A^i\vec{x}^{(0)}\|$ (beachte $\|\vec{x}^{(i)}\| = 1$). Sei $\mathbb{C}^n = E_1 \oplus \cdots \oplus E_k$ die Zerlegung nach den Eigenräumen E_j zu λ_j, seien $\vec{x}_j = \vec{x}_{E_j}^{(0)}$ die Projektionen des Startvektors auf diese Eigenräume. Mit

$$\vec{x}^{(0)} \ = \ \sum_{j=1}^{k-1} \vec{x}_j \ + \ \vec{x}_k \ , \quad A\vec{x}_j = \lambda_j \vec{x}_j \ , \quad j = 1, \ldots, k$$

folgt

$$\vec{x}^{(i)} = \frac{\sum_{j=1}^{k-1} \lambda_j^i \vec{x}_j + \lambda_k^i \vec{x}_k}{\left\| \sum_{j=1}^{k-1} \lambda_j^i \vec{x}_j + \lambda_k^i \vec{x}_k \right\|} = \left(\frac{\lambda_k}{|\lambda_k|} \right)^i \frac{\vec{x}_k + \sum_{j=1}^{k-1} \left(\frac{\lambda_j}{\lambda_k} \right)^i \vec{x}_j}{\left\| \vec{x}_k + \sum_{j=1}^{k-1} \left(\frac{\lambda_j}{\lambda_k} \right)^i \vec{x}_j \right\|} . \qquad (\#)$$

Es sei $F = E_1 \oplus \cdots \oplus E_{k-1}$. Mit $\vec{y}_F^{(i)} := \sum_{j=1}^{k-1} \left(\frac{\lambda_j}{\lambda_k} \right)^i \vec{x}_j$ folgt für den in F liegenden Anteil von $\vec{x}^{(i)}$:

$$\vec{x}_F^{(i)} = \left(\frac{\lambda_k}{|\lambda_k|} \right)^i \frac{\vec{y}_F^{(i)}}{\|\vec{x}_k + \vec{y}_F^{(i)}\|}$$

und

$$\frac{\|\vec{x}_F^{(i)}\|}{\|\vec{x}^{(i)}\|} = \|\vec{x}_F^{(i)}\| \leq \left| \frac{\lambda_{k-1}}{\lambda_k} \right|^i \frac{1}{\|\vec{x}_k + \vec{y}_F^{(i)}\|} \sum_{j=1}^{k-1} \|\vec{x}_j\| .$$

Außer für $\vec{x}_k = 0$ konvergiert damit $\vec{x}^{(i)}$ wegen $\vec{y}_F^{(i)} \to 0$ gegen den zu λ_k gehörenden Eigenraum und

$$\rho^{(i)} = \|A\vec{x}^{(i)}\| \stackrel{(\#)}{=} \frac{\|\lambda_k \vec{x}_k + A\vec{y}_F^{(i)}\|}{\|\vec{x}_k + \vec{y}_F^{(i)}\|} \stackrel{(\vec{y}_F^{(i)} \to 0)}{\longrightarrow} |\lambda_k| = \rho(A) .$$

Q.E.D.

Bemerkung 6.15: *Falls in der Zerlegung des Startvektors $\vec{x}^{(0)}$ der Eigenraum zu λ_k nicht beteiligt ist (der Ausnahmefall $\vec{x}_k = 0$ im Beweis von 6.14), so konvergiert (im Prinzip) $\vec{x}^{(i)}$ gegen den zum betragsmäßig größten Eigenwert gehörenden Eigenraum, der in der Zerlegung vorkommt. Diese Situation ist jedoch instabil: durch Rundungsfehler werden kleine Komponenten im zu λ_k gehörigen Eigenraum eingeschleppt, die sich nach längerer Rechnung durchsetzen, so daß praktisch immer $\rho^{(i)} \to |\lambda_k| = \rho(A)$ eintritt (Rundungsfehler können auch angenehm sein!).*

Bemerkung 6.16: *Man erhält $|\lambda_k|$. Durch $r_A(\vec{x}^{(i)})$ oder durch Betrachtung einzelner Komponenten von $\vec{x}^{(i)}$ läßt sich λ_k ermitteln:*

$$r_A(\vec{x}^{(i)}) \stackrel{i \to \infty}{\longrightarrow} \lambda_k , \quad \frac{(A\vec{x}^{(i)})_j}{(\vec{x}^{(i)})_j} \stackrel{i \to \infty}{\longrightarrow} \lambda_k .$$

Bemerkung 6.17: *Die Konvergenz ist langsam, wenn $|\lambda_{k-1}|/|\lambda_k| \approx 1$ gilt.*

Bemerkung 6.18: *Wenn ein Eigenwert und -vektor λ_k,$\vec{x}_k$ einer Matrix A bekannt sind, dann kann man diese Daten "löschen" (**Deflation**):*

$$\tilde{A} \;=\; \Big(\mathbb{1} - \frac{\vec{x}_k \vec{x}_k^T}{\langle \vec{x}_k, \vec{x}_k \rangle}\Big)\, A$$

hat dieselben Eigenwerte wie A, außer daß λ_k auf $\tilde{\lambda}_k = 0$ transformiert wird (Aufgabe ???). Hiermit können sukzessiv alle Spektraldaten von A ermittelt werden. Aber: Fehler addieren sich auf, die numerischen Daten werden immer ungenauer.

6.3 Die Wielandt-Rayleigh-Iteration

Idee: um die Spektraldaten von A zu ermitteln, benutze die von-Mises-Iteration mit einer Matrix $\tilde{A}$, die dieselben Eigenvektoren und verwandte Eigenwerte hat.

Beispiel 6.19: A^{-1} hat dieselben Eigenvektoren wie A zu den Eigenwerten $1/\lambda$. Damit konvergiert

$$\vec{x}^{(i+1)} \;=\; \frac{A^{-1}\vec{x}^{(i)}}{\|A^{-1}\vec{x}^{(i)}\|}$$

gegen den Eigenraum des betragskleinsten Eigenwertes von A.

Der Spektralradius von $(A-\mu\mathbb{1})^{-1}$ ist durch den Eigenwert von A bestimmt, der dem Wert μ am nächsten liegt. Durch Wahl von μ können damit Eigenwerte gezielt angesteuert werden. Satz 6.14 mit A ersetzt durch $(A-\mu\mathbb{1})^{-1}$ liefert unmittelbar:

Satz 6.20: (Inverse Wielandt-Iteration mit Spektralverschiebung)

Es sei λ derjenige Eigenwert der diagonalisierbaren Matrix A, der $\mu \in \mathbb{C}$ am nächsten liegt. Dann konvergiert für fast alle Startvektoren

$$\vec{x}^{(i+1)} \;=\; \frac{(A-\mu\mathbb{1})^{-1}\vec{x}^{(i)}}{\|(A-\mu\mathbb{1})^{-1}\vec{x}^{(i)}\|}$$

gegen den Eigenraum zu λ und

$$r_A(\vec{x}^{(i)}) \;=\; \frac{\langle \vec{x}^{(i)}, A\vec{x}^{(i)}\rangle}{\langle \vec{x}^{(i)}, \vec{x}^{(i)}\rangle} \quad \stackrel{i\to\infty}{\longrightarrow} \quad \lambda\ .$$

Die Konvergenz ist linear mit dem Konvergenzfaktor

$$\frac{\textit{Abstand von } \mu \textit{ zum nächsten Eigenwert von } A}{\textit{Abstand von } \mu \textit{ zum übernächsten Eigenwert von } A}\ .$$

Bemerkung 6.21: *Bei vorliegender LR- oder QR-Faktorisierung von $A - \mu 1\!\!1$ ist jeder Schritt*

$$\text{löse} \quad (A - \mu\, 1\!\!1)\, \vec{y} \;=\; \vec{x}^{(i)}\,, \quad \text{dann} \quad \vec{x}^{(i+1)} \;:=\; \frac{\vec{y}}{\|\vec{y}\|}$$

schnell durchführbar.

Bemerkung 6.22: *Die Konvergenz kann beliebig schnell gemacht werden, wenn die "Spektralverschiebung" μ nahe genug am gewünschten Eigenwert gewählt wird. Allerdings darf μ nicht exakt mit einem Eigenwert übereinstimmen, da sonst $A - \mu\, 1\!\!1$ singulär ist.*

Idee: mit $r_A(\vec{x}^{(i)})$ stehen immer besser werdende Approximationen des Eigenwertes zur Verfügung. Setzt man diese in die Iteration ein, so ergibt sich quadratische Konvergenz. Bei symmetrischen Matrizen liefert der Rayleigh-Quotient nach Bemerkung 6.5 besonders gute Eigenwertapproximationen und führt sogar zu kubischer Konvergenz.

Satz 6.23: (Das Wielandt-Rayleigh-Verfahren: Konvergenzbeschleunigung mit Rayleigh-Quotienten)

Für eine symmetrische Matrix A konvergiert die Iteration

$$\vec{x}^{(i+1)} \;=\; \frac{(A - \mu^{(i)}\, 1\!\!1)^{-1}\, \vec{x}^{(i)}}{\|(A - \mu^{(i)}\, 1\!\!1)^{-1}\, \vec{x}^{(i)}\|}\,, \quad \mu^{(i)} = r_A(\vec{x}^{(i)}) \;=\; \frac{\langle \vec{x}^{(i)}, A\vec{x}^{(i)} \rangle}{\langle \vec{x}^{(i)}, \vec{x}^{(i)} \rangle}$$

lokal kubisch gegen einen Eigenraum und den dazugehörigen (eventuell entarteten) Eigenwert $\lim\limits_{i \to \infty} \mu^{(i)}$. Für fast jeden Startvektor $\vec{x}^{(0)}$ ist die Konvergenz global, die Ausnahmefälle sind instabil und treten daher numerisch nicht auf.

Beweis: Wir beweisen hier nur die <u>lokale</u> Konvergenz: zu jedem Eigenraum gibt es eine "Umgebung" ("Konvergenzkegel"), innerhalb der der Iterationsvektor in superlinearer Weise angezogen wird. Der Konvergenzkegel zu einem Eigenwert λ soll nun charakterisiert werden. Sei E der Eigenraum zu λ, F sei das von den restlichen Eigenvektoren aufgespannte orthogonale Komplement. Zerlege $\vec{x}^{(i)} = \vec{x}_E^{(i)} + \vec{x}_F^{(i)}$. Der Winkel $\phi^{(i)} \in [0, \pi/2]$ zwischen $\vec{x}^{(i)}$ und $\vec{x}_E^{(i)}$ ist durch

$$\cos(\phi^{(i)}) \;=\; \frac{\|\vec{x}_E^{(i)}\|_2}{\|\vec{x}^{(i)}\|_2} \qquad \text{bzw.} \quad \sin(\phi^{(i)}) \;=\; \frac{\|\vec{x}_F^{(i)}\|_2}{\|\vec{x}^{(i)}\|_2}$$

definiert. Mit

$$\vec{y} \;=\; (A - \mu^{(i)}\, 1\!\!1)^{-1} \vec{x}^{(i)} \;=\; \frac{\vec{x}_E^{(i)}}{\lambda - \mu^{(i)}} \;+\; (A - \mu^{(i)}\, 1\!\!1)^{-1} \vec{x}_F^{(i)} \;=:\; \vec{y}_E \;+\; \vec{y}_F$$

und $\vec{x}^{(i+1)} = \vec{y}/\|\vec{y}\|_2$ folgt

$$\begin{aligned}\tan(\phi^{(i+1)}) &= \frac{\|\vec{y}_F\|_2}{\|\vec{y}_E\|_2} = \frac{\|(A-\mu^{(i)}\,\mathbb{I})^{-1}\vec{x}_F^{(i)}\|_2}{\|\vec{x}_E^{(i)}\|_2}\quad|\lambda-\mu^{(i)}|\\ &= \frac{\|(A-\mu^{(i)}\,\mathbb{I})^{-1}\vec{x}_F^{(i)}\|_2}{\|\vec{x}_F^{(i)}\|_2}\quad|\lambda-\mu^{(i)}|\;\tan(\phi^{(i)})\ .\end{aligned}$$

Mit den Eigenwerten λ_j von A und

$$c_1 = \min_{\lambda_j\neq\lambda}|\lambda-\lambda_j|\ ,\quad c_2 = \max_{\lambda_j}|\lambda-\lambda_j|$$

gilt nach Bemerkung 6.5 für den zweiten Faktor

$$|\lambda-\mu^{(i)}| \leq c_2\ \sin^2(\phi^{(i)})\ .$$

Für den ersten Faktor gilt

$$\frac{\|(A-\mu^{(i)}\,\mathbb{I})^{-1}\vec{x}_F^{(i)}\|_2}{\|\vec{x}_F^{(i)}\|_2} \leq \|(A-\mu^{(i)}\,\mathbb{I})^{-1}\|_{2F} = \frac{1}{\min\limits_{\lambda_j\neq\lambda}|\lambda_j-\mu^{(i)}|}\ ,$$

wobei $\|B\|_{2F} = \max\limits_{\vec{x}\in F}\|B\vec{x}\|_2/\|\vec{x}\|_2$ die Spektralnorm darstellt, die durch Einschränkung der Vektoren auf den Unterraum F entsteht. Für die symmetrische Matrix $(A-\mu^{(i)}\,\mathbb{I})^{-1}$ ist dies der durch Betrachtung aller möglichen Eigenvektoren in F gegebene Spektralradius, so daß nur die entsprechenden Eigenwerte $\lambda_j\neq\lambda$ zu berücksichtigen sind. Mit

$$\min_{\lambda_j\neq\lambda}|\lambda_j-\mu^{(i)}| \geq \min_{\lambda_j\neq\lambda}\Big(\ |\lambda_j-\lambda|-|\lambda-\mu^{(i)}|\ \Big) \geq c_1 - c_2\ \sin^2(\phi^{(i)})$$

ergibt sich

$$\tan(\phi^{(i+1)}) \leq \frac{c_2\ \sin^2(\phi^{(i)})}{c_1 - c_2\ \sin^2(\phi^{(i)})}\ \tan(\phi^{(i)})$$

für hinreichend kleine $\phi^{(i)}$, für die der Nenner positiv ist. Für

$$\sin^2(\phi^{(i)}) < \frac{1}{2}\,\frac{c_1}{c_2} = \frac{1}{2}\,\frac{\min\limits_{\lambda_j\neq\lambda}|\lambda_j-\lambda|}{\max\limits_{\lambda_j}|\lambda_j-\lambda|} \qquad (\#)$$

ist der Faktor zwischen $\tan(\phi^{(i+1)})$ und $\tan(\phi^{(i)})$ kleiner als 1, so daß der Winkel verkleinert wird. Der Faktor hängt selbst in monotoner Weise von dem Winkel ab, womit rekursiv die Bedingung (#) in den folgenden Schritten erhalten bleibt und sich die Konvergenz $\phi^{(i)} \to 0$ ergibt. Damit liefert (#) eine Charakterisierung des Konvergenzkegels, innerhalb dessen die bezüglich des Winkels monoton fallende Konvergenz gegen den zu λ gehörenden Eigenraum garantiert ist. Bei kleinen Winkeln ($\tan(\phi^{(i)}) \approx \sin(\phi^{(i)}) \approx \phi^{(i)}$) folgt mit

$$\phi^{(i+1)} \overset{(\approx)}{\leq} \frac{c_2}{c_1}\,(\phi^{(i)})^3$$

kubische Konvergenz. Mit Bemerkung 6.5 folgt dieselbe Konvergenzordnung für die Approximation des Eigenwertes durch die Rayleigh-Quotienten.

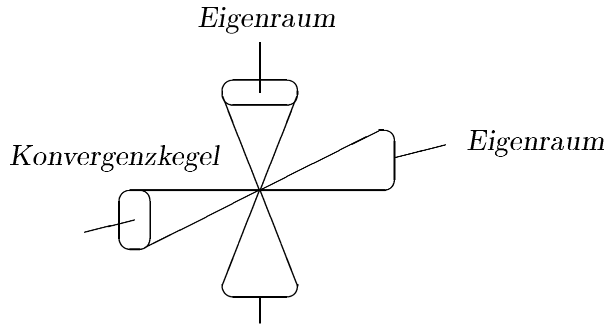
Q.E.D.

Bemerkung 6.24: *Plausibilitätsargument zur globalen Konvergenz: das "Residuum" $\vec{r}^{(i)} = (A - \mu^{(i)}\mathbb{1})\,\vec{x}^{(i)}$ (vergleiche Satz 6.3.b)) fällt für symmetrisches A in der Iteration monoton: $\|\vec{r}^{(i+1)}\|_2 \leq \|\vec{r}^{(i)}\|_2$ (Aufgabe ???).*

Bemerkung 6.25: *Nach dem Beweis von Satz 6.23 ist Konvergenz gegen den Eigenraum zu λ garantiert, sobald (#), also*

$$\sin^2(\phi^{(i)}) \;<\; \frac{1}{2}\,\frac{\min\limits_{\lambda_j \neq \lambda} |\lambda_j - \lambda|}{\max\limits_{\lambda_j} |\lambda_j - \lambda|}$$

gilt:



Um einen Eigenwert λ gezielt anzusteuern, müßte man den Startvektor genügend nahe am Eigenraum ("im Konvergenzkegel") wählen, was in der Praxis schwierig ist. Daher wird man zunächst einige Schritte mit konstantem μ ($\approx$ gewünschter Eigenwert) durchführen, wodurch die Iterationsvektoren $\vec{x}^{(i)}$ in den entsprechenden Konvergenzkegel hineingetrieben werden. Dann beschleunigt man mit Rayleigh-Quotienten.

Bemerkung 6.26: *Vorsicht in Implementierungen: da $A - \mu^{(i)}\mathbb{1}$ fast singulär wird, muß rechtzeitig abgebrochen werden. Mit den Eigenwerten λ_j von A und*

$$\mathrm{cond}(A - \mu^{(i)}\,\mathbb{1}) \overset{(4.62)}{\geq} \frac{\max_j |\lambda_j - \mu^{(i)}|}{\min_j |\lambda_j - \mu^{(i)}|} \overset{i\to\infty}{\longrightarrow} \infty$$

wird die Berechnung des Schrittes $\vec{x}^{(i)} \to \vec{x}^{(i+1)}$ immer schlechter konditioniert. In der Praxis ist dies jedoch i.a. kein Problem, denn das Verfahren ist <u>*selbstkorrigierend*</u>*: ein ungenau berechneter Nachfolger $\vec{x}^{(i+1)}$ wird im weiteren Verlauf wieder vom Eigenraum angezogen. Erst wenn $\vec{x}^{(i+1)}$ von Rundungsfehlern dominiert wird, d.h., wenn $\mathrm{cond}(A - \mu^{(i)}\,\mathbb{1})$ auf die Größenordnung von $1/\tau$ (τ = Maschinengenauigkeit) angewachsen ist*[2]*, so ist durch den Schritt $\vec{x}^{(i)} \to \vec{x}^{(i+1)}$ keine Verbesserung zu erwarten.* <u>*Für symmetrisches A*</u> *ergibt sich für den Schritt i, indem diese Situation eintritt*

$$\frac{1}{\tau} \approx \mathrm{cond}_2(A - \mu^{(i)}\,\mathbb{1}) = \frac{\max_j |\lambda_j - \mu^{(i)}|}{\min_j |\lambda_j - \mu^{(i)}|}\ ,$$

womit sich für den durch $\mu^{(i)}$ approximierten Eigenwert λ als (theoretisch) erreichbare relative Grenzgenauigkeit

$$\frac{|\Delta\lambda|}{|\lambda|} := \frac{\min_j |\lambda_j - \mu^{(i)}|}{|\lambda|} \overset{(\mu^{(i)}\approx\lambda)}{\approx} \frac{\max_j |\lambda_j - \lambda|}{|\lambda|}\,\tau \leq \Big(\frac{\max_j |\lambda_j|}{|\lambda|} + 1\Big)\,\tau\ .$$

ergibt. Der betragsgrößte Eigenwert ist damit (theoretisch) bis auf die Größenordnung der Maschinengenauigkeit bestimmbar, die betragskleineren werden ungenauer bestimmt. Für den betragskleinsten Eigenwert ergibt sich als (theoretisch) erreichbare relative Grenzgenauigkeit

$$\frac{|\Delta\lambda|}{|\lambda|} \leq \Big(\frac{\max_j |\lambda_j|}{\min_j |\lambda_j|} + 1\Big)\,\tau = (\mathrm{cond}_2(A) + 1)\,\tau\ .$$

27.2.98↓ **Bemerkung 6.27:** *Da $\mu^{(i)}$ sich in der Iteration ändert, kann man nicht wie in der unbeschleunigten Iteration nach Satz 6.20/Bemerkung 6.21 auf einmal berechnete Zerlegungsdaten zurückgreifen, sondern muß in jedem Schritt erneut eine vollständige Gleichungslösung $(A - \mu^{(i)}\,\mathbb{1})^{-1}\vec{x}^{(i)}$ durchführen. Dieses Aufwandsproblem läßt sich aber umgehen: man kann eine beliebige $n \times n$-Matrix A durch eine orthogonale Ähnlichkeitstransformation mit $O(n^3)$ Operationen zunächst auf* **Hessenberg-Form** *(ein unteres Band)*

$$\begin{pmatrix} * & \cdots & \cdots & * \\ * & \ddots & \ddots & \vdots \\ & \ddots & \ddots & \vdots \\ 0 & & * & * \end{pmatrix}$$

[2]Mit Satz 4.65 ist dann $\|\Delta\vec{x}^{(i+1)}\|/\|\vec{x}^{(i+1)}\|$ auf die Größenordnung 1 angewachsen.

bringen, bei symmetrischen Matrizen ergibt sich 3-Band-Form. Zu Details siehe z.B. [Scw93, Kapitel 6.3.1] oder [Oev96, Kapitel 6.6]. Dann ist jeder Schritt $(A - \mu^{(i)} \mathbb{1})^{-1} \vec{x}^{(i)}$ mit $O(n^2)$ bzw. $O(n)$ Operationen schnell berechenbar.
Da die Ähnlichkeitstransformation $A \to H = QAQ^T$ auf Hessenberg-Form H mit orthogonalem Q durchgeführt wird, verändern sich die Winkel zwischen den Eigenräumen nicht: die spaltenweise aus den Eigenvektoren aufgebauten Matrizen T_H zu H bzw. T_A zu A sind durch $T_H = QT_A$ verknüpft. Mit $\mathrm{cond}_2(T_H) = \mathrm{cond}_2(T_A)$ *(vergleiche Aufgabe 31.c) und Satz 6.6 folgt, daß die Konditionierungen der Eigenwertprobleme für A und H übereinstimmen, womit die Transformation keine zusätzlichen Stabilitätsprobleme in die numerische Eigenwertberechnung einbringt.*

Bemerkung 6.28: *Eine typische Anwendung: aus anderen Verfahren sind Eigenwertapproximationen bekannt. Mit der Wielandt-Rayleigh-Iteration lassen sich diese schnell und stabil verbessern und gleichzeitig Eigenvektoren bestimmen.*

6.4 Weitere Verfahren

6.4.1 Das Jacobi-Verfahren für symmetrische Matrizen

Sei A symmetrisch. Durch eine orthogonale Ähnlichkeitstransformation $A \to A' = Q^T A Q$ mit

$$Q = Q(p, q, \phi) = \begin{array}{cl} \begin{pmatrix} \mathbb{1} & & & & \\ & \cos(\phi) & & \sin(\phi) & \\ & & \mathbb{1} & & \\ & -\sin(\phi) & & \cos(\phi) & \\ & & & & \mathbb{1} \end{pmatrix} & \begin{array}{l} \\ \leftarrow p \\ \\ \leftarrow q \\ \\ \end{array} \\ \begin{array}{ccccc} \phantom{\mathbb{1}} & \uparrow & \phantom{\mathbb{1}} & \uparrow & \phantom{\mathbb{1}} \\ & p & & q & \end{array} & \end{array}$$

wird die "Diagonaldominanz" verstärkt. Bei gegebenem $p < q$ wird der Drehwinkel ϕ gemäß

$$\cos(\phi) = \frac{1}{\sqrt{1+t^2}}, \quad \sin(\phi) = \frac{t}{\sqrt{1+t^2}}$$

mit

$$t = \frac{1}{\kappa + \mathrm{sign}(\kappa)\,\sqrt{1+\kappa^2}}, \quad \kappa = \frac{a_{pp} - a_{qq}}{2\,a_{pq}}$$

(bzw. $t = 0$ für $a_{pq} = 0$ bzw. $t = 1$ für $a_{pp} = a_{qq}$) gewählt. Mit diesen Parametern verschwinden die Matrixeinträge $a'_{pq} = a'_{qp} = 0$ der transformierten Matrix

$A' = Q^T A Q = (a'_{ij})$. Man kann leicht zeigen, daß für $t \neq 0$ die Quadratsumme der nicht-diagonalen Elemente verkleinert wird:

$$\sum_{\substack{i,j \\ i \neq j}} a'^2_{ij} \; < \; \sum_{\substack{i,j \\ i \neq j}} a^2_{ij} \; .$$

Durch Hintereinanderschaltung solcher orthogonalen **Jacobi-Transformationen**, in denen (p,q) zyklisch den oberen Dreiecksanteil durchläuft, ergibt sich eine Folge ähnlicher Matrizen $A \to A' \to A'' \to \ldots$, die gegen Diagonalgestalt konvergiert, so daß die Eigenwerte letztlich von der Diagonalen abzulesen sind. Zu Details siehe z.B. [ScW92, Kapitel 15.7], [Scw93, Kapitel 6.2] oder [Oev96, Kapitel 6.7].

6.4.2 Das QR-Verfahren

Für eine beliebige Matrix A wird die Ähnlichkeitstransformation $A \to A' = Q^T A Q$ durchgeführt, wobei Q einer QR-Zerlegung $A = QR$ entstammt (d.h., $A' = RQ$ entsteht durch Vertauschung der Faktoren). Durch Hintereinanderschaltung solcher Transformationen entsteht eine Folge ähnlicher Matrizen, die unter gewissen technischen Voraussetzungen gegen obere Dreiecksform (bei symmetrischen Matrizen gegen Diagonalform) konvergieren.
Dies ist das schnellste und stabilste Verfahren zur Eigenwertberechnung! Man transformiert dabei zunächst A mit $O(n^3)$ Operationen auf Hessenberg- bzw. 3-Band-Form, so daß jeder QR-Schritt nur $O(n^2)$ bzw. $O(n)$ Operationen braucht. Mittels spektraler Verschiebungen kann die Konvergenz beschleunigt werden. Zu Details siehe z.B. [Scw93, Kapitel 6.4] oder [Oev96, Kapitel 6.5]. Praktisch brauchbare Implementierungen sind recht komplex. Statt eigener Implementierung sollte man auf professionelle Programmbibliotheken zurückgreifen. Letzere Bemerkung gilt allgemein: eigene Implementierungen werden selten die Qualität ausgetesteter professioneller Software erreichen.

Kapitel 7

Interpolation

Aufgabe: zu einer Wertetabelle $(x_0, f_0), \ldots, (x_n, f_n)$ finde eine Funktion $F(x)$, so daß

- $F(x_i) = f_i$ und
- $\max_x |F(x) - f(x)|$ möglichst klein ist, falls $f_i = f(x_i)$ einer glatten Funktion f entstammt.

Die allgemeine Idee ist, einen Ansatz $F(x; c_0, \ldots, c_n)$ mit Parametern c_i zu verwenden, die aus den Gleichungen $F(x_i; c_0, \ldots, c_n) = f_i$ bestimmt werden. Wähle dazu einen $n+1$-dimensionalen Funktionenraum mit "Basisfunktionen" $\phi_i(x)$. Der Ansatz

$$F(x; c_0, \ldots, c_n) \;=\; \sum_{i=0}^{n} c_i\, \phi_i(x)$$

führt dann auf ein <u>lineares</u> Gleichungssystem

$$\begin{pmatrix} \phi_0(x_0) & \ldots & \phi_n(x_0) \\ \vdots & \ddots & \vdots \\ \phi_0(x_n) & \ldots & \phi_n(x_n) \end{pmatrix} \begin{pmatrix} c_0 \\ \vdots \\ c_n \end{pmatrix} \;=\; \begin{pmatrix} f_0 \\ \vdots \\ f_n \end{pmatrix}$$

zur Bestimmung der c_i.

7.1 Polynominterpolation

Die Ansatzfunktion $F(x; c_0, \ldots, c_n)$ sei ein Polynom n.ten Grades in x.

Notation 7.1:

Das Interpolationspolynom durch die Stützstellen $x_0, \ldots, x_n$ wird mit

$$P_n(x) \;\equiv\; P_{[x_0,\ldots,x_n]}(x)$$

bezeichnet.

Mögliche Darstellungen:

$$\begin{aligned}
P_n(x) &= \sum_{i=0}^{n} c_i\, x^i \qquad\qquad \text{(Monomentwicklung)},\\
P_n(x) &= \sum_{i=0}^{n} c_i\, \frac{(x-x_0)\cdots(x-x_{i-1})\,(x-x_{i+1})\cdots(x-x_n)}{(x_i-x_0)\cdots(x_i-x_{i-1})(x_i-x_{i+1})\cdots(x_i-x_n)}\\
&\qquad\qquad \text{(Lagrange-Darstellung)},\\
P_n(x) &= c_0 + c_1\,(x-x_0) + c_2\,(x-x_0)\,(x-x_1) + \cdots\\
&\qquad + c_n\,(x-x_0)\cdots(x-x_{n-1}) \qquad \text{(Newton-Darstellung)}.
\end{aligned}$$

7.1.1 Entwicklung nach Monomen

Satz 7.2:

Zu $(x_0, f_0), \ldots, (x_n, f_n)$ mit paarweise verschiedenen x_i existiert ein eindeutiges Interpolationspolynom $P_n(x)$ vom Grad $\leq n$ mit $P_n(x_i) = f_i$, $i = 0, \ldots, n$. Es hat die Darstellung $P_n(x) = \sum_{j=0}^{n} c_j\, x^j$, wobei $c_0, \ldots, c_n$ die Lösungen von

$$\begin{pmatrix} 1 & x_0 & \ldots & x_0^n \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \ldots & x_n^n \end{pmatrix} \begin{pmatrix} c_0 \\ \vdots \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} f_0 \\ \vdots \\ \vdots \\ f_n \end{pmatrix}$$

sind.

Beweis: Das Vandermonde-System $P_n(x_i) = \sum_{j=0}^{n} c_j\, x_i^j = f_i$, $i = 0, \ldots, n$ ist eindeutig lösbar.

Q.E.D.

Diese Darstellung ist numerisch ungünstig, da ein Gleichungssystem zu lösen ist.

7.1.2 Lagrange-Darstellung

Definition 7.3:

$$L_j(x) = \prod_{\substack{k=0\\k\neq j}}^{n} \frac{x-x_k}{x_j-x_k}\,, \quad j = 0, \ldots, n$$

heißen die **Lagrange-Polynome** *zu $x_0, \ldots, x_n$.*

Satz 7.4: (Lagrange-Interpolation)

Das Interpolationspolynom aus Satz 7.2 hat die Darstellung

$$P_n(x) \;=\; \sum_{j=0}^{n} f_j\, L_j(x) \quad \textit{(Lagrangesche Interpolationsformel).}$$

Beweis: Offensichtlich gilt für die Lagrange-Polynome

$$L_j(x_i) \;=\; \begin{cases} 1\,,\; i=j\,, \\ 0\,,\; i\neq j\,. \end{cases}$$

Damit sind die Interpolationsforderungen $\sum_j f_j\, L_j(x_i) \;=\; f_i$ erfüllt. Das Polynom ist eindeutig.

Q.E.D.

Bemerkung 7.5: *Die Definition* $L_j(x) = \prod_{k\neq j} \frac{x-x_k}{x_j-x_k}$ *ist numerisch ungünstig, wenn an mehreren Stellen interpoliert werden soll: man kann nicht auf schon berechnete Werte zurückgreifen. Eine bessere Darstellung mittels* **Stützkoeffizienten** κ_j *ist (Aufgabe 38):*

$$L_j(x) \;=\; \frac{\dfrac{\kappa_j}{x-x_j}}{\displaystyle\sum_{k=0}^{n} \frac{\kappa_k}{x-x_k}}\,, \quad \kappa_j \;=\; \Big(\prod_{\substack{k=0\\k\neq j}}^{n} (x_j-x_k)\Big)^{-1}.$$

Für das Interpolationspolynom folgt

$$P_n(x) \;=\; \frac{\displaystyle\sum_{j=0}^{n} \frac{f_j\,\kappa_j}{x-x_j}}{\displaystyle\sum_{j=0}^{n} \frac{\kappa_j}{x-x_j}}\,. \qquad (\#)$$

Der wesentliche Aufwand von $O(n^2)$ Multiplikationen steckt in der Berechnung der κ_j. Liegen diese Daten vor, so ist dann für jedes x das Interpolationspolyom in der Darstellung (#) mit $O(n)$ Multiplikationen berechenbar. Bei äquidistanten x_i braucht auch die Berechnung der Stützkoeffizienten nur $O(n)$ Operationen (Aufgabe 38.f).

7.1.3 Newton-Darstellung

Definition 7.6:

Zu $(x_0, f_0), \ldots, (x_n, f_n)$ *mit paarweise verschiedenen* x_i *definiere die von* $x_i, x_{i+1}, \ldots, x_j$ *erzeugten* $(j-i)$**.ten dividierten Differenzen** *rekursiv durch*

$$f_{[x_i,\ldots,x_j]} := \frac{f_{[x_{i+1},\ldots,x_j]} - f_{[x_i,\ldots,x_{j-1}]}}{x_j - x_i} , \quad i < j$$

mit $f_{[x_i]} = f_i$*:*

$$\begin{array}{c|lcccccc}
x_0 & f_{[x_0]} & & & & & & \\
 & & \searrow & f_{[x_0,x_1]} & & & & \\
x_1 & f_{[x_1]} & \nearrow & & \searrow & f_{[x_0,x_1,x_2]} & & \\
 & & \searrow & f_{[x_1,x_2]} & \nearrow & & \searrow & f_{[x_0,x_1,x_2,x_3]} \\
x_2 & f_{[x_2]} & \nearrow & & \searrow & f_{[x_1,x_2,x_3]} & \nearrow & \\
 & & \searrow & f_{[x_2,x_3]} & \nearrow & & & \vdots \\
x_3 & f_{[x_3]} & \nearrow & & & \vdots & & \\
 & & & \vdots & & & & \\
\vdots & \vdots & & & & & &
\end{array}$$

Hilfssatz 7.7: (Syntheseprinzip)

Das Interpolationspolynom zu $x_i, \ldots, x_j$ *läßt sich aus denen zu* $x_{i+1}, \ldots, x_j$ *und* $x_i, \ldots, x_{j-1}$ *aufbauen:*

$$P_{[x_i,\ldots,x_j]}(x) = \frac{(x - x_i)\, P_{[x_{i+1},\ldots,x_j]}(x) + (x_j - x)\, P_{[x_i,\ldots,x_{j-1}]}(x)}{x_j - x_i} .$$

Beweis: Induktion nach der Anzahl der verwendeten Stützstellen. Induktionsschritt: setze $x = x_i, \ldots, x_j$ ein, wobei sich nach Induktionsvoraussetzung jeweils $f_i, \ldots, f_j$ ergibt.

Q.E.D.

Hilfssatz 7.8:

3.2.98↓ *Die dividierten Differenzen sind die Leitkoeffizienten der Interpolationspolynome:* $P_{[x_i,\ldots,x_j]}(x) = f_{[x_i,\ldots,x_j]}\, x^{j-i} + (\ldots)\, x^{j-i-1} + \cdots$.

Beweis: Bezeichnet man den Leitkoeffizienten mit $c_{[x_i,\ldots,x_j]}$ und betrachtet man in Hilfssatz 7.7 die höchsten x-Potenzen, so folgt die Rekursion

$$c_{[x_i,\ldots,x_j]} = \frac{c_{[x_{i+1},\ldots,x_j]} - c_{[x_i,\ldots,x_{j-1}]}}{x_j - x_i}$$

der dividierten Differenzen.

Q.E.D.

Folgerung 7.9:

Da das Interpolationspolynom durch $x_i, \ldots, x_j$ offensichtlich unabhängig von der gewählten Anordnung der Stützstellen ist, ist $f_{[x_i,\ldots,x_j]}$ symmetrisch in den Stützstellen.

Satz 7.10: (Newton-Interpolation)

Das Interpolationspolynom zu $(x_0, f_0), \ldots, (x_n, f_n)$ hat die Darstellung

$$P_{[x_0,\ldots,x_n]}(x) = f_{[x_0]} + f_{[x_0,x_1]}\,(x - x_0) + f_{[x_0,x_1,x_2]}\,(x - x_0)(x - x_1) + \cdots$$
$$\cdots + f_{[x_0,\ldots,x_n]}\,(x - x_0)\cdots(x - x_{n-1}) \ .$$

Beweis: Es gilt

$$P_{[x_0,\ldots,x_n]}(x) \ - \ f_{[x_0,\ldots,x_n]}\,(x - x_0)\cdots(x - x_{n-1}) \ = \ P_{[x_0,\ldots,x_{n-1}]}(x) \ ,$$

da dies nach Hilfssatz 7.8 ein Polynom vom Grand $n-1$ ist und Einsetzen von $x = x_0, \ldots, x_{n-1}$ jeweils $f_0, \ldots, f_{n-1}$ liefert. Aus

$$P_{[x_0,\ldots,x_n]}(x) \ = \ P_{[x_0,\ldots,x_{n-1}]}(x) \ + \ f_{[x_0,\ldots,x_n]}\,(x - x_0)\cdots(x - x_{n-1})$$

folgt induktiv die Behauptung.

Q.E.D.

Beispiel 7.11: Schätze $f(1.8)$ durch Interpolation der Wertetabelle

x	0	1	2	3
$f(x)$	2	4	2	14

.

Berechnung der dividierten Differenzen:

$$\begin{array}{c|ccccccc}
 & x_0 = 0 & & x_1 = 1 & & x_2 = 2 & & x_3 = 3 \\
\hline
f_{[x_i]} & \boxed{2} & & 4 & & 6 & & 14 \\
 & & \searrow \ \swarrow & & \searrow \ \swarrow & & \searrow \ \swarrow & \\
f_{[x_i,x_{i+1}]} & & \boxed{2} & & 2 & & 8 & \\
 & & & \searrow \ \swarrow & & \searrow \ \swarrow & & \\
f_{[x_i,x_{i+1},x_{i+3}]} & & & \boxed{0} & & 3 & & \\
 & & & & \searrow \ \swarrow & & & \\
f_{[x_0,x_1,x_2,x_3]} & & & & \boxed{1} & & &
\end{array}$$

Über die Newton-Darstellung des Interpolationspolynoms erhält man

$$\begin{aligned}
& P_{[x_0,x_1,x_2,x_3]}(x) \\
= \ & f_{[x_0]} + f_{[x_0,x_1]}\,(x - x_0) + f_{[x_0,x_1,x_2]}\,(x - x_0)\,(x - x_1) \\
& \qquad + f_{[x_0,x_1,x_2,x_3]}\,(x - x_0)\,(x - x_1)\,(x - x_2) \\
= \ & 2 + 2 \times (x - x_0) + 0 \times (x - x_0)\,(x - x_1) + 1 \times (x - x_0)\,(x - x_1)\,(x - x_2) \\
= \ & 2 + 2\,x + (x^3 - 3\,x^2 + 2\,x) = x^3 - 3\,x^2 + 4\,x + 2
\end{aligned}$$

und damit $P_{[x_0,x_1,x_2,x_3]}(1.8) = 5.312$.

Satz 7.12: (Restglieddarstellung, Interpolationsfehler)
Die Wertetabelle $(x_0, f_0), \ldots, (x_n, f_n)$ *entstamme einer glatten Funktion* f*, also* $f_i = f(x_i)$.

a) Für $(n+1)$*-fach stetig differenzierbares* f *gilt*

$$f(x) - P_{[x_0,\ldots,x_n]}(x) \;=\; \frac{f^{(n+1)}(\xi)}{(n+1)!}\,(x-x_0)\,\cdots\,(x-x_n)$$

mit einem Zwischenwert $\xi \in \Big(\min(x_0,\ldots,x_n,x), \max(x_0,\ldots,x_n,x)\Big)$.

b) Für n*-fach stetig differenzierbares* f *gilt* $f_{[x_0,\ldots,x_n]} = \dfrac{f^{(n)}(\xi)}{n!}$ *mit einem Zwischenwert* $\xi \in \Big(\min\limits_{i=0..n} x_i\ ,\ \max\limits_{i=0..n} x_i\Big)$.

Beweis: a) Für $x \in \{x_0,\ldots,x_n\}$ ist nichts zu zeigen. Für andere x betrachte

$$\Delta(y) \;:=\; f(y) - P_{[x_0,\ldots,x_n]}(y) - c(x)\,(y-x_0)\,\cdots\,(y-x_n)$$

mit

$$c(x) \;=\; \frac{f(x) - P_{[x_0,\ldots,x_n]}(x)}{(x-x_0)\,\cdots\,(x-x_n)}\ .$$

Mit $\Delta(x_0) = \ldots = \Delta(x_n) = \Delta(x) = 0$ gilt

$$\begin{array}{ccl}
 & \Delta & \text{hat (mindestens) } n+2 \text{ Nullstellen} \\
\overset{\text{(Rolle)}}{\Longrightarrow} & \Delta' & \text{hat dazwischen mindestens } n+1 \text{ Nullstellen} \\
\vdots & \vdots & \qquad\vdots \\
\overset{\text{(Rolle)}}{\Longrightarrow} & \Delta^{(n+1)} & \text{hat dazwischen mindestens eine Nullstelle } \xi.
\end{array}$$

Mit $0 = \Delta^{(n+1)}(\xi) = f^{(n+1)}(\xi) - c(x)\,(n+1)!$ folgt

$$c(x) \;=\; \frac{f(x) - P_{[x_0,\ldots,x_n]}(x)}{(x-x_0)\,\cdots\,(x-x_n)} \;=\; \frac{f^{(n+1)}(\xi)}{(n+1)!}\ .$$

b) Interpolation durch $(x_0, f_0), \ldots, (x_{n-1}, f_{n-1})$ liefert nach a)

$$f(x) \;=\; P_{[x_0,\ldots,x_{n-1}]}(x) \;+\; \frac{f^{(n)}(\xi)}{n!}\,(x-x_0)\,\cdots\,(x-x_{n-1})\ .$$

Im Beweis von Satz 7.10 wurde

$$P_{[x_0,\ldots,x_n]}(x) \;=\; P_{[x_0,\ldots,x_{n-1}]}(x) \;+\; f_{[x_0,\ldots,x_n]}\,(x-x_0)\cdots(x-x_{n-1})$$

gezeigt. Aus

$$f(x) - P_{[x_0,\ldots,x_n]}(x) \;=\; \left(\frac{f^{(n)}(\xi)}{n!} - f_{[x_0,\ldots,x_n]}\right)(x-x_0)\cdots(x-x_{n-1})$$

folgt mit $x = x_n$ die Behauptung.

Q.E.D.

Bemerkung 7.13: *Die Newton-Darstellung*

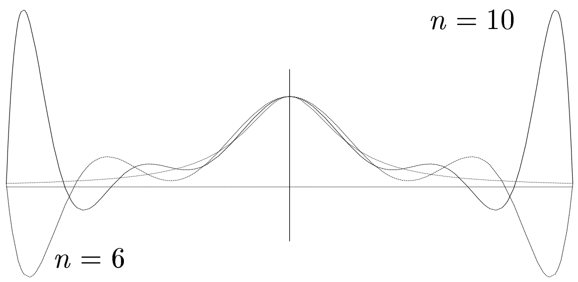
$$f(x) \;=\; \overbrace{f_{[x_0]} + f_{[x_0,x_1]}\,(x-x_0) + \cdots + f_{[x_0,\ldots,x_n]}\,(x-x_0)\,\cdots\,(x-x_{n-1})}^{P_{[x_0,\ldots,x_n]}(x)}$$
$$+\;\frac{f^{(n+1)}(\xi)}{(n+1)!}\,(x-x_0)\,\cdots\,(x-x_n)$$

mit $f_{[x_0,\ldots,x_k]} = \dfrac{f^{(k)}(\xi_k)}{k!}$ *ist eine Verallgemeinerung der Taylor-Entwicklung, die sich im Grenzübergang* $x_1,\ldots,x_n \to x_0$ *ergibt.*

Bemerkung 7.14: *Betrachte ein Intervall* $[a,b]$ *und äquidistante Stützstellen* $x_i = a + \frac{i}{n}(b-a)$, $i = 0,\ldots,n$. *Es gibt einfache Beispiele glatter Funktionen* f, *für die*

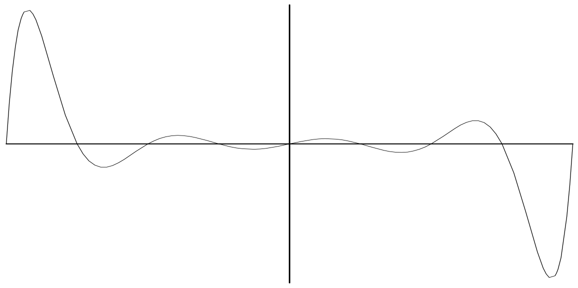
$$\max_{x\in[a,b]} |\, f(x) - P_{[x_0,\ldots,x_n]}(x)\,| \quad \stackrel{n\to\infty}{\longrightarrow} \quad \infty$$

gilt! Beispiel (nach Runge): $f(x) = 1/(1+x^2)$ *auf* $[a,b] = [-5,5]$:



Merke: Bei wachsender Zahl äquidistanter Stützstellen ist keine gleichmäßige Konvergenz der Polynominterpolierenden zu erwarten.

Bemerkung 7.15: *Der Grund hierfür ist das für $n \gg 1$ stark oszillierende Verhalten von $\omega(x) = (x - x_0) \cdots (x - x_n)$ an den Rändern:*



Außerhalb des Stützstellenintervalls $(\min_i x_i, \max_i x_i)$ *wächst* $|\omega(x)|$ *schnell an!*

Merke: Polynominterpolation mit äquidistanten Stützstellen nicht zur Extrapolation weit außerhalb des Stützstellenbereichs verwenden! Für $n \gg 1$ ist äquidistante Polynominterpolation nur im Zentrum des Stützstellenbereiches brauchbar!

Bemerkung 7.16: *(Optimale Stützstellenwahl)*
Die Stützstellen x_i seien wählbar. Aufgabe: finde zu einem gegebenen Intervall $[a, b]$ Stützstellen $x_0, \ldots, x_n$, mit denen

$$\max_{x \in [a,b]} |(x - x_0) \cdots (x - x_n)|$$

minimal wird. Antwort: wähle

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

Dann gilt

$$(x - x_0) \cdots (x - x_n) = \frac{1}{2^n} \left(\frac{b-a}{2} \right)^{n+1} T_{n+1}\left(\frac{2x - a - b}{b - a} \right)$$

mit den Tschebyscheff-Polynomen

$$T_0(y) = 1 , \quad T_1(y) = y , \quad T_2(y) = 2y^2 - 1 , \quad T_3(y) = 4y^3 - 3y , \quad \ldots ,$$

die für $|y| \le 1$ durch[1]

$$T_{n+1}(y) := \cos(\,(n+1) \arccos(y)\,) , \quad n + 1 = 0, 1, 2, \ldots$$

[1] Es gilt, daß $\cos((n+1)\,\alpha)$ für jedes $n+1 \in \mathbb{N}$ als Summe von Potenzen in $\cos(\alpha)$ geschrieben werden kann:

$\cos(2\,\alpha) = 2\,\cos^2(\alpha) - 1 = T_2(\cos(\alpha))$, $\cos(3\,\alpha) = 4\,\cos^3(\alpha) - 3\,\cos(\alpha) = T_3(\cos(\alpha))$, $\ldots$.

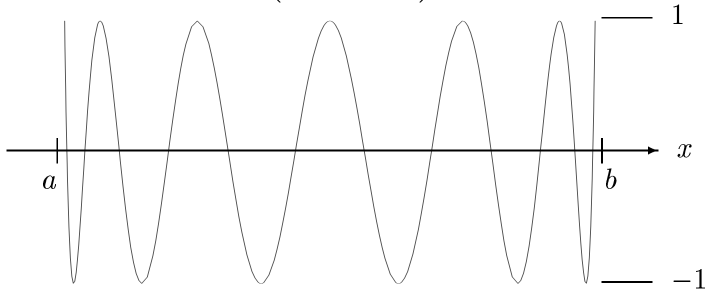
Damit liefert diese Definition wirklich ein Polynom in y. Aus dieser Darstellung folgt die entscheidende Eigenschaft $|T_{n+1}(y)| \le 1 \;\forall\; y \in [-1, 1]$.

mit den offensichtlichen Nullstellen

$$y_i \;=\; -\cos\left(\frac{(2i+1)\,\pi}{2\,(n+1)}\right) \;\in\; (-1,1)\;, \quad i=0,\ldots,n$$

definiert sind:

$$T_{n+1}\left(\frac{2x-a-b}{b-a}\right)$$



Intuitiv: die Tschebyscheff-Nullstellen (#) sind zu den Intervallenden hin dichter verteilt, was die starken Randoszillationen unterdrückt. Man kann leicht zeigen (z.B. [Oev96, Satz 8.5]), daß dann

$$\max_{x\in[a,b]} |x-\tilde{x}_0|\cdots|x-\tilde{x}_n| \;\geq\; \max_{x\in[a,b]} |x-x_0|\,\cdots\,|x-x_n| \;=\; \frac{1}{2^n}\left(\frac{b-a}{2}\right)^{n+1}$$

für jedes alternative Stützstellensystem $\tilde{x}_0,\ldots,\tilde{x}_n$ gilt.

7.1.4 Praktische Durchführung der Interpolation

Bemerkung 7.17: *(Berechnung dividierter Differenzen) Zur Berechnung von $P_{[x_0,\ldots,x_n]}(x)$ fordert das Newton-Schema zunächst die Berechnung von $f_{[x_0]},\ldots,f_{[x_0,\ldots,x_n]}$ mit $\approx n^2/2$ Multiplikationen. Das definierende Schema*

$$\begin{array}{ccccccccc}
 & & j=1 & & j=2 & & \ldots & & j=n \\
f_0 = \boxed{f_{[x_0]}} & & & & & & & & \\
 & \searrow & & & & & & & \\
f_1 = f_{[x_1]} & \to & \boxed{f_{[x_0,x_1]}} & & & & & & \\
 & \searrow & & \searrow & & & & & \\
f_2 = f_{[x_2]} & \to & f_{[x_1,x_2]} & \to & \boxed{f_{[x_0,x_1,x_2]}} & & & & \\
 & \searrow & & \searrow & & \searrow & & & \\
\vdots & \to & \ddots & \to & \ddots & \to & \ddots & & \\
 & \searrow & & \searrow & & \searrow & & \searrow & \\
f_n = f_{[x_n]} & \to & f_{[x_{n-1},x_n]} & \to & \ldots & \to & \ldots & \to & \boxed{f_{[x_0,\ldots,x_n]}}
\end{array}$$

der dividierten Differenzen kann spaltenweise durchlaufen werden, wobei im j.ten Schritt die ursprünglichen Daten $f_j, \ldots, f_n$ mit den j.ten dividierten Differenzen überschrieben werden:

```
(∗ gegeben (x_0, f_0), ..., (x_n, f_n) ∗)
for j := 1 to n do
   for i := n downto j do   f_i := (f_i − f_{i−1})/(x_i − x_{i−j});
(∗ Ergebnis: nun gilt  f_i = f_[x_0,...,x_i] , i = 0, ..., n. ∗)
```

Bemerkung 7.18: *(Auswertung des Interpolationspolynoms) Bei vorliegenden dividierten Differenzen kann die Auswertung von*

$$P_{[x_0,\ldots,x_n]}(x) = f_{[x_0]} + f_{[x_0,x_1]}\,(x-x_0) + \cdots + f_{[x_0,\ldots,x_n]}\,(x-x_0)\,\cdots\,(x-x_{n-1})$$

über die Horner-artige Darstellung

$$\Big(\ldots\Big(f_{[x_0,\ldots,x_n]}\,(x-x_{n-1}) + f_{[x_0,\ldots,x_{n-1}]}\Big)(x-x_{n-2}) + \ldots\Big)\,(x-x_0) + f_{[x_0]}$$

des Polynoms für jedes x mit n Multiplikationen geschehen:

```
p := f_[x_0,...,x_n] ;  for i := n − 1 downto 0 do  p := (x − x_i) p + f_[x_0,...,x_i] ;
(∗ Ergebnis:  p = P_[x_0,...,x_n](x) . ∗)
```

Bemerkung 7.19: *Die dividierten Differenzen $f_{[x_0]}, f_{[x_0,x_1]}, \ldots, f_{[x_0,\ldots,x_n]}$ seien berechnet und in $f_i := f_{[x_0,\ldots,x_i]}$ gespeichert. Nimmt man nachträglich ein weiteres Wertepaar (x_{n+1}, f_{n+1}) hinzu, so muß zur Interpolation durch die erweiterte Stützstellenmenge die nächste dividierte Differenz $f_{[x_0,\ldots,x_{n+1}]}$ aus den vorliegenden Daten berechnet werden. Mit Definition 7.6 und Folgerung 7.9 gilt die Rekursion*

$$f_{[x_0,\ldots,x_i,x_{n+1}]} = \frac{f_{[x_0,\ldots,x_{i-1},x_{n+1}]} - f_{[x_0,\ldots,x_i]}}{x_{n+1} - x_i}\ , \quad i = 1, \ldots, n$$

mit dem Start $f_{[x_0,x_{n+1}]} = (f_{[x_{n+1}]} - f_{[x_0]})/(x_{n+1} - x_0)$ für $i = 0$. Überschreibt man f_{n+1} nacheinander mit den Werten $f_{[x_0,\ldots,x_i,x_{n+1}]}$, so erhält man das benötigte Datum mit $n+1$ Multiplikationen[2]:

---

[2] Selbstverständlich können alternativ zu Bemerkung 7.17 auch $f_{[x_0]}, f_{[x_0,x_1]}, \ldots, f_{[x_0,\ldots,x_n]}$ auf diese Weise berechnet werden:

```
(∗ gegeben (x_0, f_0), ..., (x_n, f_n) ∗)
for j := 1 to n do
   for i := 0 to j − 1 do   f_j := (f_j − f_i)/(x_j − x_i) ;
(∗ Ergebnis: nun gilt f_j = f_[x_0,...,x_j] , j = 0, ..., n. ∗)
```

for $i := 0$ **to** n **do** $f_{n+1} := \dfrac{f_{n+1} - f_i}{x_{n+1} - x_i}$;
(∗ *Ergebnis: nun gilt* $f_{n+1} = f_{[x_0,\ldots,x_n,x_{n+1}]}$ ∗).

Mit weiteren $n+1$ Multiplikationen ist dann wie oben $P_{[x_0,\ldots,x_n,x_{n+1}]}(x)$ auszuwerten. Ein nachträgliches Hinzufügen einer weiteren Stützstelle ist damit in der Newton-Interpolation schnell mit $\approx 2n$ Multiplikationen durchführbar.

Bemerkung 7.20: *Die numerische Berechnung dividierter Differenzen im Newton-Schema ist durch Auslöschung gefährdet, wenn die Stützwerte $f_0, \ldots, f_n$ nur schwach variieren.*

Dieses Stablitätsproblem kann vermindert werden, in dem man die Berechnung der dividierten Differenzen nicht explizit durchführt, sondern geschickt in die Polynomauswertung einbaut:

Bemerkung 7.21: *Ist $P_{[x_0,\ldots,x_n]}(x)$ nur <u>an einer einzelnen Stelle</u> x auszuwerten, so bietet sich das* **Neville-Schema** *an. Nach Hilfssatz 7.7 gilt* ↓5.2.98

$$P_{[x_j,\ldots,x_i]}(x) \;=\; \frac{x - x_j}{x_i - x_j}\, P_{[x_{j+1},\ldots,x_i]}(x) \;+\; \frac{x_i - x}{x_i - x_j}\, P_{[x_j,\ldots,x_{i-1}]}(x) \;,$$

d.h., mit den Abkürzungen $t_{ij} = P_{[x_j,\ldots,x_i]}(x)$, $0 \leq j \leq i \leq n$:

$$t_{ij} \;=\; \frac{x - x_j}{x_i - x_j}\, t_{i,j+1} \;+\; \frac{x_i - x}{x_i - x_j}\, t_{i-1,j} = t_{i,j+1} + \frac{x_i - x}{x_i - x_j}\,(t_{i-1,j} - t_{i,j+1}) \;.$$

Diese Werte bilden das zeilenweise auszuwertende Schema

$$\begin{array}{ccccccc}
t_{00} = f_0 & & & & & & \\
\downarrow & & & & & & \\
t_{10} & \leftarrow & t_{11} = f_1 & & & & \\
\downarrow & & \downarrow & & & & \\
t_{20} & \leftarrow & t_{21} & \leftarrow & t_{22} = f_2 & & \\
\downarrow & & \downarrow & & \downarrow & & \\
\vdots & & \vdots & & \vdots & \ddots & \\
\downarrow & & \downarrow & & \downarrow & & \\
\boxed{t_{n0}} & \leftarrow & t_{n1} & \leftarrow & t_{n2} & \leftarrow \ldots \leftarrow & t_{nn} = f_n
\end{array}$$

wobei $t_{n0} = P_{[x_0,\ldots,x_n]}(x)$ den gewünschten Interpolationswert liefert. Speichert man die i.te Zeile als Array $(T_0, \ldots, T_i)$ mit $T_j = t_{ij}$, so ergibt sich mit Überschreiben des Arrays die Implementierung:

for $i := 0$ **to** n **do begin**

  $T_i := f_i$;

  for $j := i-1$ **downto** 0 **do** $T_j := T_{j+1} + \dfrac{x_i - x}{x_i - x_j}\,(T_j - T_{j+1})$;

end;

(* *Ergebnis:* $T_0 = P_{[x_0,\ldots,x_n]}(x)$. *)

Um einen weiteren Interpolationspunkt (x_{n+1}, f_{n+1}) hinzuzufügen, braucht nur die i-Schleife bis $n+1$ erweitert werden (zusätzliche $2n+2$ Multiplikationen). Die nächste Zeile $(T_0, \ldots, T_{n+1})$ des Neville-Schemas berechnet sich aus der schon vorliegenden letzten Zeile $(T_0, \ldots, T_n)$. Damit kann die `for` $i := 0$ `to` n *-Schleife (die eine feste Laufgrenze n benötigt) durch eine flexiblere* `while`*-Schleife ersetzt werden, die i heraufzählt und solange zusätzliche Interpolationspunkte hinzunimmt, bis ein geeignetes Abbruchkriterium erfüllt ist.*

Übersicht 7.22: *Vergleich der praktisch einsetzbaren Varianten zur Berechnung von $P_{[x_0,\ldots,x_n]}(x)$:*

Lagrange-Interpolation *gemäß Bemerkung 7.5:*

- • *Berechnung der Stützkoeffizienten κ_j mit $\approx n^2/2$ Multiplikationen (vergleiche [Oev96,Aufgabe 8.1.b)]).*
- ± *Im Spezialfall äquidistanter Stützstellen lassen sich die Stützkoeffizienten in der Laufzeit $O(n)$ berechnen. Dieser Spezialfall mit $n \gg 1$ ist für die Praxis jedoch irrelevant, da bei äquidistanten Stützstellen die Polynominterpolation höchstens auf einem kleinen Bereich "im Zentrum des Stützstellenbereichs" zu akzeptablen Approximationen führt.*
- \+ *Mehrfachinterpolationen sind schnell durchführbar: bei vorliegenden Stützkoeffizienten ist $P_{[x_0,\ldots,x_n]}(x)$ für jedes x mit $\approx 2n$ Multiplikationen schnell auswertbar.*

Newton-Interpolation:

- • *Berechnung der dividierten Differenzen mit $\approx n^2/2$ Multiplikationen.*
- − *Auslöschungsgefahr bei schwach variierenden Stützwerten $f_0, \ldots, f_n$.*
- \+ *Mehrfachinterpolationen sind schnell durchführbar: bei vorliegenden dividierten Differenzen ist $P_{[x_0,\ldots,x_n]}(x)$ für jedes x mit n Multiplikationen schnell auswertbar.*
- \+ *Zusätzliche Stützstellen können mit $\approx 2n$ Multiplikationen schnell nachträglich eingefügt werden.*

Neville-Schema:

- *Berechnung von* $P_{[x_0,\ldots,x_n]}(x)$ *mit* $\approx n^2$ *Multiplikationen.*
- \+ *Da die dividierten Differenzen nicht explizit berechnet werden, wird die Gefahr der Auslöschung vermindert.*
- − *Mehrfachinterpolationen sind nicht schnell durchführbar: für jeden Interpolationspunkt* x *werden* $\approx n^2$ *Multiplikationen benötigt.*
- \+ *Zusätzliche Stützstellen können mit* $\approx 2n$ *Multiplikationen schnell nachträglich eingefügt werden.*

In der Praxis nimmt man den doppelten Aufwand des Neville-Schemas zugunsten der höheren Stabilität in Kauf.

Kapitel 8

Splines

8.1 Spline-Räume

Sei $C^j([a,b])$ der Raum der j-fach stetig diff'baren reellen Funktionen über dem Intervall $[a,b]$ (einseitige Stetigkeit/Diff'barkeit an den Endpunkten).

Definition 8.1: (Die Spline-Räume)

Der Raum der **(polynomialen) Spline-Funktionen vom Grad** $k \in \mathbb{N}$ *zu reellen Stützstellen* $x_0 < \ldots < x_n$ *ist*

$$\mathcal{S}^{[k]}_{[x_0,\ldots,x_n]} \;=\; \{\; S \in C^{k-1}([x_0,x_n])\;;\;\; S \textit{ ist auf } [x_0,x_1],\ldots,[x_{n-1},x_n] \\ \textit{jeweils ein Polynom vom Grad} \leq k \;\}\;.$$

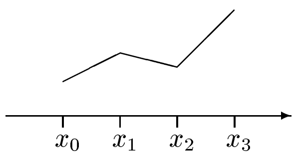
Auf $[x_i, x_{i+1}]$ hat ein Spline $S \in \mathcal{S}^{[k]}_{[x_0,\ldots,x_n]}$ die Darstellung

$$S(x) = \beta_{i0} + \beta_{i1}\,x + \cdots + \beta_{ik}\,x^k\;.$$

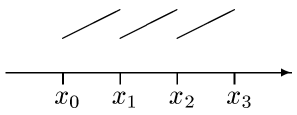
Die Forderung $S \in C^{k-1}([x_0,x_n])$ bedeutet, daß die ersten $k-1$ Ableitungen <u>an den Nahtstellen $x_1,\ldots,x_{n-1}$</u> stetig ineinander übergehen müssen.

Beispiel 8.2: Für $k=1$ muß nur die Funktion selbst stetig sein:

linearer Spline:



dies ist kein linearer Spline:



Satz 8.3:
Die Dimension des Funktionenraums $\mathcal{S}^{[k]}_{[x_0,\ldots,x_n]}$ ist $n+k$.

Beweis: Definiere die $n\,(k+1)$ stückweisen Polynome

$$\sigma_{ij}(x) := \begin{cases} x^j & \text{für } x \in [x_i, x_{i+1}]\,, \\ 0 & \text{für } x \notin [x_i, x_{i+1}]\,, \end{cases} \qquad i = 0, \ldots, n-1\ , \quad j = 0, \ldots, k$$

welche offensichtlich linear unabhängige Funktionen sind. Jedes $S \in \mathcal{S}^{[k]}_{[x_0,\ldots,x_n]}$ hat eine Darstellung der Form $S(x) = \sum_{i=0}^{n-1}\sum_{j=0}^{k} \beta_{ij}\,\sigma_{ij}(x)$. Mit

$$S(x_i{+}0) := \lim_{\substack{h\to 0\\ h>0}} S(x_i{+}h) = \sum_{j=0}^{k} \beta_{ij}\,x^j\ ,\ S(x_i{-}0) := \lim_{\substack{h\to 0\\ h>0}} S(x_i{-}h) = \sum_{j=0}^{k} \beta_{i-1,j}\,x^j$$

liefern die k Stetigkeitsforderungen

$$\begin{aligned}
S(x_i+0) - S(x_i-0) &= \sum_{j=0}^{k} (\,\beta_{ij} - \beta_{i-1,j}\,)\,x_i^j = 0\ , \\
S'(x_i+0) - S'(x_i-0) &= \sum_{j=1}^{k} (\,\beta_{ij} - \beta_{i-1,j}\,)\,j\,x_i^{j-1} = 0\ , \\
&\ \ \vdots \qquad\qquad\qquad \vdots \\
S^{(k-1)}(x_i+0) - S^{(k-1)}(x_i-0) &= \sum_{j=k-1}^{k} (\,\beta_{ij} - \beta_{i-1,j}\,)\,j!\,x_i^{j-k} = 0\ ,
\end{aligned}$$

d.h.,

$$\begin{pmatrix} 1 & x_i & x_i^2 & x_i^3 & \ldots & x_i^k \\ & 1 & 2\,x_i & 3\,x_i^2 & \ldots & k\,x_i^{k-1} \\ & & 2 & 6\,x_i & \ldots & k(k-1)\,x_i^{k-2} \\ & & & \ddots & \ddots & \vdots \\ & & & & (k-1)! & k!\,x_i \end{pmatrix} \begin{pmatrix} \beta_{i0} - \beta_{i-1,0} \\ \beta_{i1} - \beta_{i-1,1} \\ \vdots \\ \vdots \\ \beta_{i,k-1} - \beta_{i-1,k-1} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ \vdots \\ 0 \end{pmatrix}$$

an jeder der $n-1$ Nahtstellen $x_1, \ldots, x_{n-1}$ einen Satz von k unabhängigen linearen Gleichungen für die insgesamt $n\,(k+1)$ Koeffizienten $\beta_{00}, \ldots, \beta_{n-1,k}$. Der Rang des gesamten Gleichungssystems ist $k\,(n-1)$, womit nach Erfüllen der Stetigkeitsforderungen nur noch $n\,(k+1) - k\,(n-1) = n+k$ freie Parameter verbleiben.

Q.E.D.

8.2 B-Splines

Ziel: algorithmische Konstruktion von Basis-Funktionen ("B-Splines") in $\mathcal{S}^{[k]}_{[x_0,\ldots,x_n]}$. Diese lassen sich rekursiv aus jeweils 2 B-Splines in $\mathcal{S}^{[k-1]}_{[x_0,\ldots,x_n]}$ zusammensetzen:

Definition 8.4: (B-Splines)

Zu Stützstellen $\ldots < x_i < x_{i+1} < \ldots$ *definiere rekursiv*

$$\begin{aligned} B^{[k]}_{[x_i,\ldots,x_{i+k+1}]}(x) &= \frac{x - x_i}{x_{i+k} - x_i}\, B^{[k-1]}_{[x_i,\ldots,x_{i+k}]}(x) \\ &\quad + \frac{x_{i+k+1} - x}{x_{i+k+1} - x_{i+1}}\, B^{[k-1]}_{[x_{i+1},\ldots,x_{i+k+1}]}(x) \end{aligned}$$

mit den Treppenfunktionen $B^{[0]}_{[x_i,x_{i+1}]}(x) = \begin{cases} 1 & \text{für } x \in [x_i, x_{i+1}) \\ 0 & \text{sonst }. \end{cases}$

Typische Plots für $k = 1, \ldots, 4$: siehe Bild 8.1. Zur Notation: $[k]$ ist der Spline-Grad, $[x_i, \ldots, x_{i+k+1}]$ deutet die Stützstellen an, die rekursiv in die Definition des B-Splines eingehen. Punkt b) des nächsten Satzes besagt, daß sie auch den Träger dieser Funktion angeben:

Satz 8.5: (Einige Eigenschaften der B-Splines)

Es gilt für alle Grade $k \geq 1$:

a) $B^{[k]}_{[x_i,\ldots,x_{i+k+1}]} \in \mathcal{S}^{[k]}_{[\ldots,x_i,\ldots,x_{i+k+1},\ldots]}$.

b) $B^{[k]}_{[x_i,\ldots,x_{i+k+1}]}(x) = 0$ *für* $x \notin [x_i, x_{i+k+1}]$.

c) $B^{[k]}_{[x_i,\ldots,x_{i+k+1}]}(x) \in (0,1]$ *für* $x \in (x_i, x_{i+k+1})$.

d) $\displaystyle\sum_{i=-\infty}^{\infty} B^{[k]}_{[x_i,\ldots,x_{i+k+1}]}(x) = \sum_{i=j-k}^{j} B^{[k]}_{[x_i,\ldots,x_{i+k+1}]}(x) = 1$ *für alle* $x \in [x_j, x_{j+1}]$.

e) $\displaystyle\frac{d}{dx} B^{[k]}_{[x_i,\ldots,x_{i+k+1}]}(x) = k \left(\frac{B^{[k-1]}_{[x_i,\ldots,x_{i+k}]}(x)}{x_{i+k} - x_i} - \frac{B^{[k-1]}_{[x_{i+1},\ldots,x_{i+k+1}]}(x)}{x_{i+k+1} - x_{i+1}} \right)$.

Beweis: b) Mit Def. 8.4 zerlegt sich $B^{[k]}_{[x_i,\ldots,x_{i+k+1}]}(x)$ rekursiv in eine Summe über die Treppen $B^{[0]}_{[x_i,x_{i+1}]}(x)$, $B^{[0]}_{[x_{i+1},x_{i+2}]}(x)$, $\ldots$, $B^{[0]}_{[x_{i+k},x_{i+k+1}]}(x)$, die für $x \notin [x_i, x_{i+k+1})$ alle verschwinden. Der Fall $x = x_i$ folgt sofort aus Def. 8.4.

d) Für $x \in [x_j, x_{j+1}]$ gilt

$$\sum_{i=-\infty}^{\infty} B^{[k]}_{[x_i,\ldots,x_{i+k+1}]}(x) = \sum_{i=j-k}^{j} B^{[k]}_{[x_i,\ldots,x_{i+k+1}]}(x) ,$$

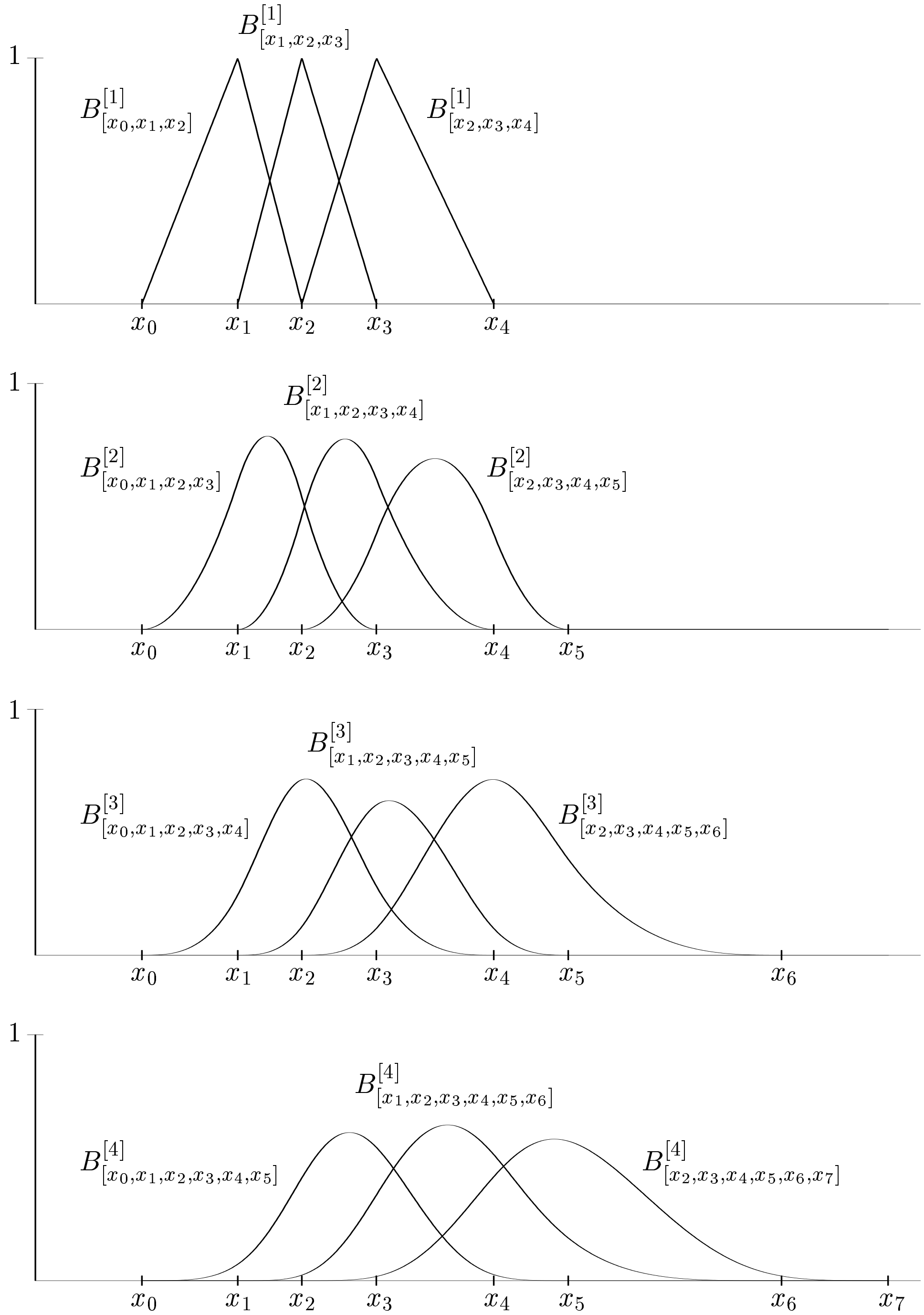


Bild 8.1: B-Splines.

da mit b) alle anderen Summanden verschwinden. Mit Def. 8.4 und Umsummation $i+1 \to i$ des zweiten Terms folgt die Rückführung $k \to k-1$:

$$\begin{aligned}
&\sum_{i=-\infty}^{\infty} B^{[k]}_{[x_i,\ldots,x_{i+k+1}]}(x)\\
&= \sum_{i=-\infty}^{\infty} \frac{x-x_i}{x_{i+k}-x_i}\, B^{[k-1]}_{[x_i,\ldots,x_{i+k}]}(x) + \sum_{i=-\infty}^{\infty} \frac{x_{i+k+1}-x}{x_{i+k+1}-x_{i+1}}\, B^{[k-1]}_{[x_{i+1},\ldots,x_{i+k+1}]}(x)\\
&= \sum_{i=-\infty}^{\infty} \left(\frac{x-x_i}{x_{i+k}-x_i} + \frac{x_{i+k}-x}{x_{i+k}-x_i} \right) B^{[k-1]}_{[x_i,\ldots,x_{i+k}]}(x) = \sum_{i=-\infty}^{\infty} B^{[k-1]}_{[x_i,\ldots,x_{i+k}]}(x)\,.
\end{aligned}$$

Für die Treppen $k=0$ ergibt die Summe offensichtlich 1.

c) Ist für $k=0$ klar. Induktionsschritt $k-1 \to k$: in Def. 8.4 werden nichtnegative Terme addiert, von denen höchstens einer verschwindet. Wegen d) können dabei keine Werte >1 angenommen werden.

e) Mit Def. 8.4 folgt für

$$\begin{aligned}
\Delta^{[k]}_{[x_i,\ldots,x_{i+k+1}]}(x) \quad := \quad & \frac{d}{dx} B^{[k]}_{[x_i,\ldots,x_{i+k+1}]}(x)\\
& - k \left(\frac{B^{[k-1]}_{[x_i,\ldots,x_{i+k}]}(x)}{x_{i+k}-x_i} - \frac{B^{[k-1]}_{[x_{i+1},\ldots,x_{i+k+1}]}(x)}{x_{i+k+1}-x_{i+1}} \right)
\end{aligned}$$

die Rekursion

$$\begin{aligned}
\Delta^{[k]}_{[x_i,\ldots,x_{i+k+1}]}(x) \quad = \quad & \frac{x-x_i}{x_{i+k}-x_i}\, \Delta^{[k-1]}_{[x_i,\ldots,x_{i+k}]}(x)\\
& + \frac{x_{i+k+1}-x}{x_{i+k+1}-x_{i+1}}\, \Delta^{[k-1]}_{[x_{i+1},\ldots,x_{i+k+1}]}(x)
\end{aligned}$$

mit dem leicht zu verifizierenden Start $\Delta^{[1]}_{[x_i,x_{i+1},x_{i+2}]}(x) \equiv 0$.

a) Aus Def. 8.4 folgt induktiv, daß $B^{[k]}_{[x_i,\ldots,x_{i+k+1}]}$ stückweise ein Polynom vom Grad $\leq k$ ist. Mit e) folgt die Stetigkeit der ersten $k-1$ Ableitungen von $B^{[k]}_{[x_1,\ldots,x_{i+k-1}]}$ durch Rückführung auf die offensichtlich stetigen "Dachfunktionen" $B^{[1]}_{[x_i,x_{i+1},x_{i+2}]}$.

Q.E.D.

Bemerkung 8.6: *B-Splines vom Grad k sind lokalisierte Funktionen (Eigenschaft b): sie werden nur von $k+1$ Teilintervallen getragen und gehen außerhalb dieses Bereiches $(k-1)$-fach stetig diff'bar in die Nullfunktion über* *(d.h., $B^{[k]}_{[\ldots]} \in C^{k-1}(\mathbb{R})$). Betrachtet man z.B. eine Linearkombination $S(x) = \sum_j \beta_j\, B^{[k]}_{[x_j,\ldots,x_{j+k+1}]}(x)$, so reduziert sich diese Summe an jeder Stelle x auf $k+1$ Terme. Für $x \in [x_i, x_{i+1}]$ gilt speziell*

$$S(x) = \sum_{j=i-k}^{i} \beta_j\, B^{[k]}_{[x_j,\ldots,x_{j+k+1}]}(x) \ ,$$

da alle anderen B-Splines an der Stelle x verschwinden. Diese Lokalisierung hat in Anwendungen (z.B., Interpolation) den angenehmen Effekt, daß man oft auf dünnbesetzte Gleichungssysteme stößt, die schnell lösbar sind.

Die Auswertung einer Linearkombination von B-Splines kann im Prinzip durch die rekursive Definition 8.4 geschehen, wird aber effizienter folgendermaßen durchgeführt:

Satz 8.7: (de Boor-Algorithmus zur Auswertung von Splines)

Für $x \in [x_i, x_{i+1})$ wird der Wert einer Linearkombination von B-Splines

$$S(x) = \sum_{j=i-k}^{i} \beta_j\, B^{[k]}_{[x_j,\ldots,x_{j+k+1}]}(x)$$

durch die Rekursion

for $j := i-k$ **to** i **do** $\beta_j^{[0]} := \beta_j$;

for $r := 0$ **to** $k-1$ **do**
  for $j = i$ **downto** $i-k+r+1$ **do**

$$\beta_j^{[r+1]} := \frac{(x_{j+k-r} - x)\, \beta_{j-1}^{[r]} \ + \ (x - x_j)\, \beta_j^{[r]}}{x_{j+k-r} - x_j} \ ;$$

geliefert: $S(x) = \beta_i^{[k]}$.

Beweis: Es wird gezeigt, daß für $r = 0, \ldots, k$

$$S(x) \ = \sum_{j=i-k+r}^{i} \beta_j^{[r]}\ B^{[k-r]}_{[x_j,\ldots,x_{j+k+1-r}]}(x) \qquad (\#)$$

gilt, was für $r = k$ die Behauptung beweist: $S(x) = \beta_i^{[k]}\, B^{[0]}_{[x_i,x_{i+1}]}(x) = \beta_i^{[k]}$. Für $r = 0$ ist (#) richtig. Mit Def. 8.4 und der Umsummierung $j \to j-1$ des zweiten Terms ergibt sich der Induktionsschritt von r nach $r+1$:

$$S(x) \ = \sum_{j=i-k+r}^{i} \beta_j^{[r]}\ B^{[k-r]}_{[x_j,\ldots,x_{j+k+1-r}]}(x)$$

$$
\begin{aligned}
&= \sum_{j=i-k+r+1}^{i} \beta_j^{[r]} \, \frac{x - x_j}{x_{j+k-r} - x_j} \, B_{[x_j,\ldots,x_{j+k-r}]}^{[k-r-1]}(x) \\
&\qquad\qquad + \sum_{j=i-k+r}^{i-1} \beta_j^{[r]} \, \frac{x_{j+k+1-r} - x}{x_{j+k+1-r} - x_{j+1}} \, B_{[x_{j+1},\ldots,x_{j+k+1-r}]}^{[k-r-1]}(x) \\
&= \sum_{j=i-k+r+1}^{i} \frac{(x - x_j)\,\beta_j^{[r]} \; + \; (x_{j+k-r} - x)\,\beta_{j-1}^{[r]}}{x_{j+k-r} - x_j} \, B_{[x_j,\ldots,x_{j+k-r}]}^{[k-r-1]}(x) \\
&= \sum_{j=i-k+r+1}^{i} \beta_j^{[r+1]} \, B_{[x_j,\ldots,x_{j+k-r}]}^{[k-r-1]}(x) \; .
\end{aligned}
$$

Q.E.D.

Bemerkung 8.8: *Damit ist zur Spline-Berechnung das Dreiecksschema*

$$
\begin{array}{c|ccccc}
r = 0 & \beta_{i-k} & \beta_{i-k+1} & & \cdots & \beta_i \\
 & & \searrow \; \downarrow & & & \searrow \; \downarrow \\
r = 1 & & \beta_{i-k+1}^{[1]} & & \cdots & \beta_i^{[1]} \\
 & & & \searrow \; \downarrow & & \searrow \; \downarrow \\
\vdots & & & & \ddots & \vdots \\
 & & & & & \searrow \; \downarrow \\
r = k & & & & & \boxed{\beta_i^{[k]}}
\end{array}
$$

auszuwerten, in dem rechts unten der gesuchte Spline-Wert zu finden ist. Überschreibt man die Koeffizienten β_j *mit den neuen Werten* $\beta_j^{[r]}$ *und formuliert die Rekursionsformel etwas um, so reduziert sich der de Boor-Algorithmus 8.7 zu*

```
for r := 0 to k − 1 do
   for j := i downto i − k + r + 1 do
      β_j := β_j + (x_{j+k−r} − x)/(x_{j+k−r} − x_j) (β_{j−1} − β_j) ;
(* Ergebnis: S(x) = β_i für x ∈ [x_i, x_{i+1}). *)
```

$$\beta_j := \beta_j + \frac{x_{j+k-r} - x}{x_{j+k-r} - x_j} \, (\beta_{j-1} - \beta_j) \; ;$$

Eine Spline-Auswertung braucht damit lediglich $k\,(k+1)$ *Multiplikationen.*

Die in Def. 8.4 definierten B-Splines bilden ein System unabhängiger Spline-Funktionen. Ziel: erzeuge hiermit eine Basis von $\mathcal{S}^{[k]}_{[x_0,\ldots,x_n]}$. Problem: für eine <u>endliche</u> Stützstellenmenge $x_0 < \ldots < x_n$ lassen sich die B-Splines

$$B^{[k]}_{[x_0,\ldots,x_{k+1}]}, B^{[k]}_{[x_1,\ldots,x_{k+2}]}, \ \ldots \ , B^{[k]}_{[x_{n-k-1},\ldots,x_n]}$$

bilden. Dies sind aber nur $n-k$ Funktionen, während die Dimension von $\mathcal{S}^{[k]}_{[x_0,\ldots,x_n]}$ nach Satz 8.3 $n+k$ ist. Es fehlen noch $2k$ Basisfunktionen! Eine Möglichkeit ist, die Stützstellenmengen willkürlich um $2k$ beliebige Punkte

$$x_{-k} < \ldots < x_{-1} \Big(< x_0 \Big) \ , \qquad \Big(x_n < \Big) \ x_{n+1} < \ldots < x_{n+k}$$

zu erweitern und die zugehörigen $n+k$ B-Splines auf das Intervall $[x_0, x_n]$ einzuschränken. Diese Willkür ist etwas ärgerlich. Man kann (im folgenden Sinne) die speziellen Zusatzpunkte $x_{-k} = \ldots = x_{-1} = x_0$, $x_n = x_{n+1} = \ldots = x_{n+k}$ wählen. Betrachte dazu die Grenzfunktion der B-Splines, wenn alle Zusatzpunkte gegen

$$x_{-k}, \ldots, x_{-1} \to x_0 \ , \qquad x_n \leftarrow x_{n+1}, \ldots, x_{n+k}$$

streben. Dies definiert sogenannte **Rand-Splines**, die dadurch ausgezeichnet sind, daß in ihrer Definition Stützpunkte mehrfach auftreten:

Definition 8.9:

Zu $x_0 < \ldots < x_n$ definiere die linken

$$B^{[k]}_{[\underbrace{x_0,\ldots,x_0}_{p+1},x_1,\ldots,x_{k+1-p}]}(x) \ := \lim_{x_{-p}\to x_0-0} \cdots \lim_{x_{-1}\to x_0-0} B^{[k]}_{[x_{-p},\ldots,x_{-1},x_0,\ldots,x_{k+1-p}]}(x)$$

bzw. rechten **Rand-Splines**

$$B^{[k]}_{[x_{n-k-1+p},\ldots,x_{n-1},\underbrace{x_n,\ldots,x_n}_{p+1}]}(x) \ :=$$

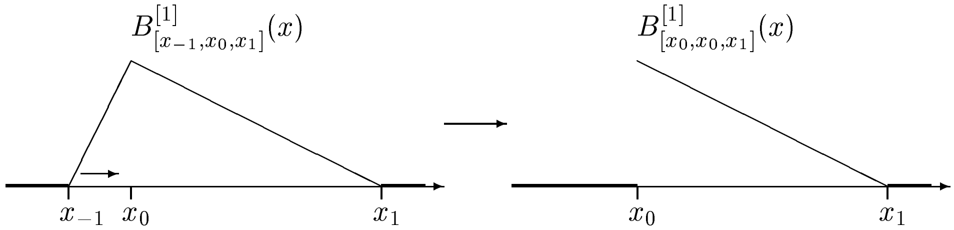
$$\lim_{x_{n+p}\to x_n+0} \cdots \lim_{x_{n+1}\to x_n+0} B^{[k]}_{[x_{n-k-1+p},\ldots,x_{n-1},x_n,\ldots,x_{n+p}]}(x)$$

mit $p = 1, \ldots, k$

Bemerkung 8.10: *Durch den Grenzübergang verlieren die Rand-Splines etwas Glattheit: $B^{[k]}_{[\underbrace{x_0,\ldots,x_0}_{p+1},x_1,\ldots,x_{k+1-p}]}$ ist an der Stelle x_0 nur noch $(k-1-p)$-fach stetig diff'bar (dieser B-Spline ist zwar rechtsseitig glatt, geht aber am linken Randpunkt nur noch $(k-1-p)$-fach stetig diff'bar in die Nullfunktion über). Für $p = k$: $B^{[k]}_{[x_0,\ldots,x_0,x_1]}$ ist dort nicht einmal mehr (linksseitig)* *stetig. Die Rand-Splines bleiben in ihrer Einschränkung auf* $[x_0, x_n]$, *speziell* <u>*an den inneren Nahtstellen*</u> $x_1, \ldots, x_{k+1-p}$, *immer noch* $(k-1)$*-fach stetig diff'bar und liegen damit in* $\mathcal{S}^{[k]}_{[x_0,\ldots,x_n]}$ *(siehe z.B.* [L.L. Schumaker: *Spline functions: basic theory.* Chichester: Wiley 1981]*). Die reduzierte Glattheit am Rand des Intervalls* $[x_0, x_n]$ *ist irrelevant.*

Beispiel:

$$B^{[1]}_{[x_0,x_0,x_1]}(x) = \lim_{x_{-1} \to x_0 - 0} B^{[1]}_{[x_{-1},x_0,x_1]}(x) = \begin{cases} \dfrac{x_1 - x}{x_1 - x_0} & \text{für } x \in [x_0, x_1] \ , \\ 0 & \text{für } x \notin [x_0, x_1] \ . \end{cases}$$



Für lineare, quadratische und kubische B-Splines sind die linken und rechten Rand-Splines zusammen mit einem inneren B-Spline in Bild 8.2 dargestellt.

Bemerkung 8.11: *Setzt man*

$$x_{-k} := \ \ldots \ := x_0 < x_1 < \ \ldots \ < x_n =: x_{n+1} =: \ \ldots \ =: x_{n+k} \ ,$$

so lassen sich beliebige Linearkombination von "inneren" und Rand-Splines

$$S(x) = \sum_{j=i-k}^{i} \beta_j \, B^{[k]}_{[x_j,\ldots,x_{j+k+1}]}(x) \ , \quad i \in \{0, \ldots, n-1\}$$

problemlos mit dem de Boor-Algorithmus 8.7 an beliebigen Stellen $x \in [x_i, x_{i+1}] \subset [x_0, x_n]$ *auswerten. Beachte, daß keiner der Nenner* $x_{j+k-r} - x_j$ *in der Rekursion* $r = 0, \ldots, k-1$ *verschwindet:*

$$j \in \{i - k + r + 1, \ldots, i\} \ , \ j + k - r \in \{i + 1, \ldots, i + k - r\} \, .$$

Nach Bemerkung 8.10 liegen auch die Rand-Splines in $\mathcal{S}^{[k]}_{[x_0,\ldots,x_n]}$, und es ist eine Basis konstruiert:

Satz 8.12: (B-Spline-Basis von $\mathcal{S}^{[k]}_{[x_0,\ldots,x_n]}$)
Zu den Stützstellen $x_0 < \ldots < x_n$ *bilden mit der Setzung*

$$x_{-k} := \ \ldots \ := x_{-1} := x_0 \ , \quad x_n =: x_{n+1} =: \ \ldots \ =: x_{n+k}$$

die B-Splines $B^{[k]}_{[x_i,\ldots,x_{i+k+1}]}$ *mit* $i = -k, \ldots, n-1$ *eine Basis des* $(n+k)$*-dimensionalen Spline-Raums* $\mathcal{S}^{[k]}_{[x_0,\ldots,x_n]}$.

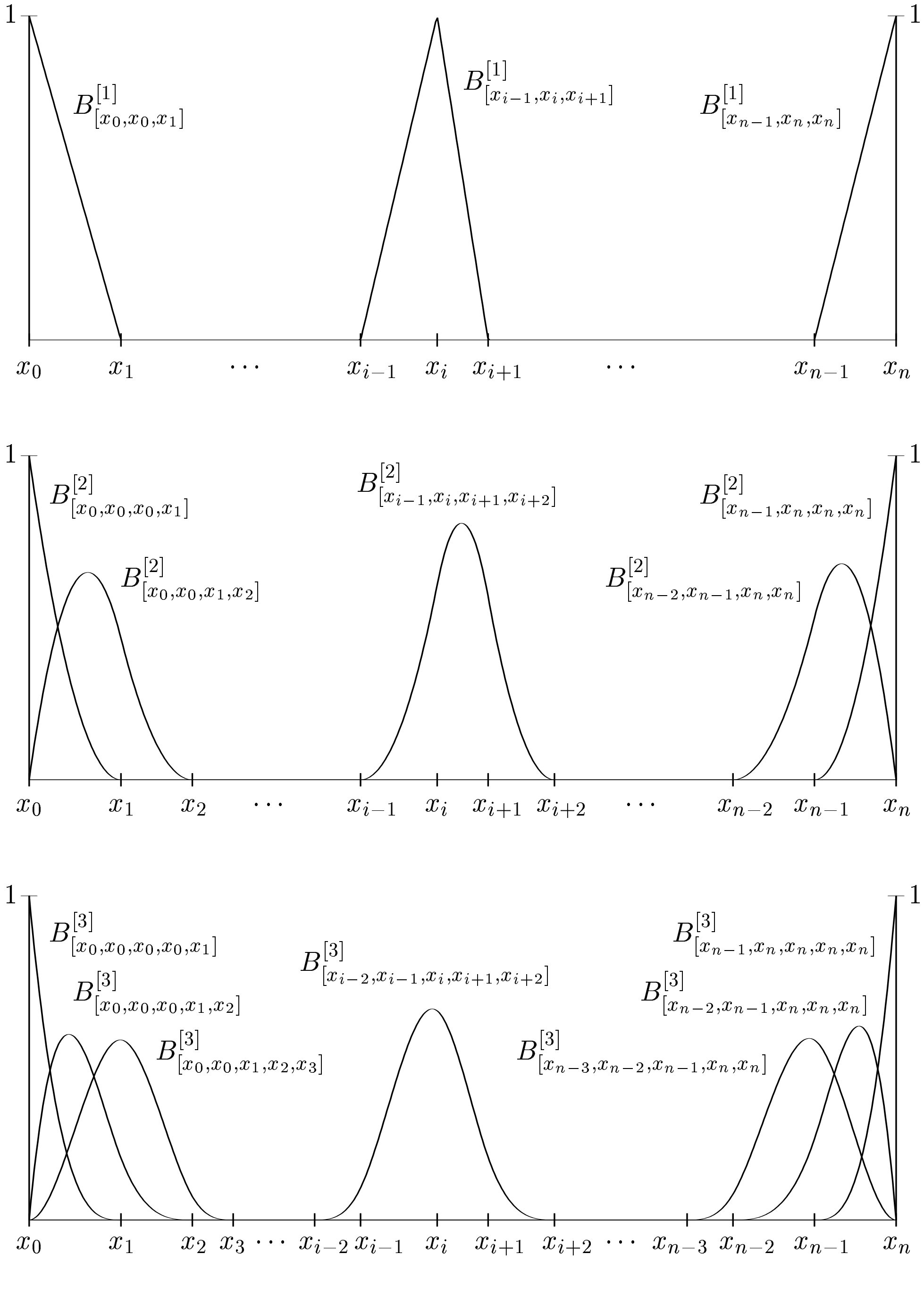


Bild 8.2: Innere und Rand-B-Splines.

Beweisskizze: Es ist nur die Unabhängigkeit aller B-Splines zu zeigen. Betrachte dazu eine Linearkombination $S(x) = \sum_j \beta_j B_j(x)$, wo B_j sowohl über die inneren als auch über die Rand-Splines läuft. Zeige, daß aus $S(x) \equiv 0$ folgt, daß alle Koeffizienten β_j verschwinden. Zunächst wird dies für alle k linken Rand-Splines gezeigt. Da $B^{[k]}_{[x_0,\ldots,x_0,x_1]}$ der einzige B-Spline ist, der am Punkt x_0 nicht verschwindet, ist der entsprechende Koeffizient wegen $S(x_0) = 0$ als 0 identifiziert. Da $B^{[k]}_{[x_0,\ldots,x_0,x_1]}$ und $B^{[k]}_{[x_0,\ldots,x_0,x_1,x_2]}$ die einzigen B-Splines sind, deren Ableitung bei x_0 (rechtsseitiger Grenzwert) nicht verschwindet, folgt wegen $S'(x) \equiv 0$ durch Auswertung bei x_0, daß auch der Koeffizient von $B^{[k]}_{[x_0,\ldots,x_0,x_1,x_2]}$ trivial sein muß. Analog folgt durch Betrachtung der ersten $k-1$ Ableitungen am linken Rand, daß keiner der linken Rand-Splines an der Linearkombination beteiligt sein kann. Durch Auswertung auf dem ersten Teilintervall $[x_0, x_1]$, welches von den inneren B-Splines nur $B^{[k]}_{[x_0,x_1,\ldots,x_{k+1}]}$ trägt, folgt dann das Verschwinden des Koeffizienten dieser Basisfunktion. Geht man auf die nächsten Intervalle über, so folgt nacheinander das Verschwinden der Koeffizienten aller inneren Splines und dann auch der rechten Rand-Splines. Mit der so bewiesenen Unabhängigkeit folgt, daß die angegebenen $n + k$ B-Splines eine Basis des $(n + k)$-dimensionalen Raums $\mathcal{S}^{[k]}_{[x_0,\ldots,x_n]}$ bilden.

Q.E.D.

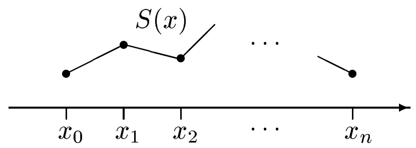
Zusammenfassung: Zu gegebenen Stützstellen $x_0 < \ldots < x_n$ kann rekursiv ein Satz von Basisfunktionen (B-Splines) der $(n + k)$-dimensionalen Spline-Räume $\mathcal{S}^{[k]}_{[x_0,\ldots,x_n]}$ definiert werden. Hiervon sind $n - k$ "innere" Splines, die auf jeweils $k + 1$ Teilintervallen getragen werden und außerhalb dieses Bereichs identisch verschwinden. Hinzu kommen jeweils k Splines am linken und rechten Rand, die durch mehrfache Verwendung der Endpunkte x_0 bzw. x_n als Stützstellen entstehen.

8.3 Interpolation mit Splines

Aufgabe: zu vorgegebenen Stützstellen $x_0 < \ldots < x_n$ und -werten $f_0, \ldots, f_n$ finde (für gegebenes k) einen Spline $S \in \mathcal{S}^{[k]}_{[x_0,\ldots,x_n]}$ mit $S(x_i) = f_i$.

Beispiel 8.13: Die lineare Spline-Interpolierende ($k = 1$) hat auf den Teilintervallen $x \in [x_i, x_{i+1}]$, $i \in \{0, \ldots, n-1\}$, die Darstellung

$$S(x) = \frac{(x_{i+1} - x)\, f_i + (x - x_i)\, f_{i+1}}{x_{i+1} - x_i} \,.$$



Begründung: offensichtlich ist S stückweise linear. Mit

$$\lim_{x\to x_i+0} S(x) = \lim_{x\to x_i+0} \frac{(x_{i+1}-x)\,f_i \;+\; (x-x_i)\,f_{i+1}}{x_{i+1}-x_i} = f_i\ ,$$

$$\lim_{x\to x_i-0} S(x) = \lim_{x\to x_i-0} \frac{(x_i-x)\,f_{i-1} \;+\; (x-x_{i-1})\,f_i}{x_i-x_{i-1}} = f_i\ ,$$

ist S an den Nahtstellen $x_1,\ldots,x_{n-1}$ stetig und erfüllt die Interpolationsbedingungen. Die Darstellung mittels B-Splines lautet

$$S(x) = \sum_{i=0}^{n} f_i\, B^{[1]}_{[x_{i-1},x_i,x_{i+1}]}(x)\ ,$$

wobei $x_{-1} := x_0$, $x_{n+1} := x_n$. Beachte hierzu, daß für die "Dachfunktionen" $B^{[1]}_{[x_{i-1},x_i,x_{i+1}]}(x_j) = \delta_{ij}$ gilt.

Interpolationsfehler: es gelte $f_i = f(x_i)$ mit $f \in C^2([x_0,x_n])$. Da S auf den Teilintervallen mit der linearen Polynominterpolierenden übereinstimmt, folgt für $x \in [x_i,x_{i+1}]$ aus Satz 7.12.b) die Fehlerdarstellung

$$f(x) - S(x) = \frac{f''(\xi)}{2!}\,(x-x_i)\,(x-x_{i+1})$$

mit einem Zwischenwert $\xi \in [x_i,x_{i+1}]$. Mit $|x-x_i||x-x_{i+1}| \le (x_{i+1}-x_i)^2/4$ folgt

$$\max_{x\in[x_0,x_n]} |f(x) \;-\; S(x)| \;\le\; \frac{h^2}{8} \max_{x\in[x_0,x_n]} |f''(x)|$$

mit $h := \max(x_1-x_0,\ldots,x_n-x_{n-1})$.

Läßt man durch Erhöhung der Stützstellenzahl die "Schrittweite" $h \to 0$ konvergieren, so konvergiert die lineare Spline-Interpolierende *gleichmäßig* gegen die Funktion f. Dies demonstriert das entschieden bessere Konvergenzverhalten der Splines im Vergleich zur Polynominterpolation.

8.3.1 Kubische Splines

Ein für die Praxis relevanter Fall ist die Interpolation mit kubischen Splines $S \in \mathcal{S}^{[3]}_{[x_0,\ldots,x_n]}$. Wegen $\dim(\mathcal{S}^{[3]}_{[x_0,\ldots,x_n]}) = n+3$ sind neben den $n+1$ Interpolationswerten zwei zusätzliche Randbedingungen vorzugeben, um die Interpolierende eindeutig festzulegen. Interessant sind die folgenden Randbedingungen:

a) ("natürliche" Randbedingungen) Neben $S(x_0) = f_0,\ldots,S(x_n) = f_n$ wird $S''(x_0) = S''(x_n) = 0$ vorgeschrieben.

b) ("vollständige" Randbedingungen) Neben $S(x_0) = f_0,\ldots,S(x_n) = f_n$ werden Ableitungswerte $S'(x_0) = f'_0$, $S'(x_n) = f'_n$ am Rand vorgeschrieben.

c) ("periodische" Randbedingungen) Neben $S(x_0) = f_0, \ldots, S(x_{n-1}) = f_{n-1}$ und $S(x_n) = S(x_0) = f_0$ wird $S'(x_0) = S'(x_n)$ und $S''(x_n) = S''(x_0)$ gefordert.

d) ("not-a-knot" Randbedingungen) Neben $S(x_0) = f_0, \ldots, S(x_n) = f_n$ wird die Stetigkeit von S''' an den Stellen x_1 und x_{n-1} gefordert. Dies bedeutet, daß S auf den Doppelintervallen $[x_0, x_2]$ und $[x_{n-2}, x_n]$ ein Polynom dritten Grades darstellt, sodaß x_1 und x_{n-1} nicht mehr im Sinne eingeschränkter Differenzierbarkeit zu den Stützstellen gehören ("not a knot").

In allen Fällen besitzt das Problem eine eindeutige Lösung. Ein Algorithmus zur Berechnung der Spline-Daten für vollständige Randbedingungen wird jetzt vorgestellt:

Herleitung des Algorithmus: Verwende die Darstellung

$$S(x) \;=\; a_i \;+\; b_i\,(x-x_i) \;+\; c_i\,(x-x_i)^2 \;+\; d_i\,(x-x_i)^3 \tag{8.1}$$

auf dem Intervall $x \in [x_i, x_{i+1}]$, $i \in \{0, \ldots, n-1\}$, also

$$S'(x) = b_i + 2\,c_i\,(x-x_i) + 3\,d_i\,(x-x_i)^2 \;, \qquad S''(x) = 2\,c_i + 6\,d_i\,(x-x_i) \;.$$

Die zu suchenden Koeffizienten $a_i, \ldots, d_i$ sind durch die Werte $f_i = S(x_i)$ und $f_i' := S'(x_i)$ an den Stellen $x_0, \ldots, x_n$ festgelegt:

$$\begin{aligned} a_i &= f_i \;, \quad c_i \;=\; 3\,\frac{f_{i+1}-f_i}{h_i^2} - \frac{2\,f_i' + f_{i+1}'}{h_i} \;, \\ b_i &= f_i' \;, \quad d_i \;=\; -\,2\,\frac{f_{i+1}-f_i}{h_i^3} + \frac{f_i' + f_{i+1}'}{h_{i+1}^2} \;, \end{aligned} \tag{8.2}$$

wobei

$$h_i := x_{i+1} - x_i \;, \quad i = 0, \ldots, n-1 \;. \tag{8.3}$$

Man findet diese Darstellung dadurch, daß man die 4 Gleichungen

$$\begin{aligned} f_i = S(x_i+0) = a_i \quad &, \quad f_{i+1} = S(x_{i+1}-0) = a_i + b_i\,h_i + c_i\,h_i^2 + d_i\,h_i^3 \;, \\ f_i' = S'(x_i+0) = b_i \quad &, \quad f_{i+1}' = S'(x_{i+1}-0) = b_i + 2\,c_i\,h_i + 3\,d_i\,h_i^2 \end{aligned}$$

nach den 4 Koeffizienten a_i, b_i, c_i und d_i auflöst. Die durch (8.1),(8.2),(8.3) definierte Funktion mit (noch) beliebigen Parametern f_i' ist zusammen mit ihrer ersten Ableitung an den Knoten $x_1, \ldots, x_{n-1}$ stetig, denn nach Konstruktion gilt

$$S(x_i-0) = S(x_i+0) = f_i \;, \quad S'(x_i-0) = S'(x_i+0) = f_i' \;.$$

Die noch zu erfüllende Stetigkeitsforderung

$$2\,c_{i-1} + 6\,d_{i-1}\,h_{i-1} = S''(x_i-0) = S''(x_i+0) = 2\,c_i$$

für S'' an diesen Stellen liefert nach Einsetzen von (8.2) das System

$$\frac{f'_{i-1}}{h_{i-1}} \;+\; 2\left(\frac{1}{h_{i-1}}+\frac{1}{h_i}\right)f'_i \;+\; \frac{f'_{i+1}}{h_i} \;=\; \delta_i\;,$$

$$\delta_i := 3\,\frac{f_i - f_{i-1}}{h_{i-1}^2} \;+\; 3\,\frac{f_{i+1}-f_i}{h_i^2} \tag{8.4}$$

von $n-1$ Gleichungen ($i=1,\ldots,n-1$) für die $n-1$ Unbekannten $f'_1,\ldots,f'_{n-1}$ (die Werte f'_0 und f'_n sind im Falle vollständiger Randbedingungen vorgegeben). Die Lösung dieses symmetrischen positiv definiten Systems

$$\begin{pmatrix} 2\left(\frac{1}{h_0}+\frac{1}{h_1}\right) & \frac{1}{h_1} & & & \\ \frac{1}{h_1} & 2\left(\frac{1}{h_1}+\frac{1}{h_2}\right) & \frac{1}{h_2} & & \\ & \frac{1}{h_2} & \ddots & \ddots & \\ & & \ddots & \ddots & \frac{1}{h_{n-2}} \\ & & & \frac{1}{h_{n-2}} & 2\left(\frac{1}{h_{n-2}}+\frac{1}{h_{n-1}}\right) \end{pmatrix} \begin{pmatrix} f'_1 \\ f'_2 \\ \vdots \\ f'_{n-2} \\ f'_{n-1} \end{pmatrix} = \begin{pmatrix} \delta_1 - \frac{f'_0}{h_0} \\ \delta_2 \\ \vdots \\ \delta_{n-2} \\ \delta_{n-1} - \frac{f'_n}{h_{n-1}} \end{pmatrix} \tag{8.5}$$

kann schnell in der Laufzeit $O(n)$ durch LR- bzw. (schneller und stabiler) durch Cholesky-Faktorisierung ermittelt werden. Zusammenfassung:

Algorithmus 8.14: (vollständige kubische Spline-Interpolierende)

Bestimme zu $x_0 < \ldots < x_n$ und $f_0,\ldots,f_n$ sowie f'_0, f'_n die Spline-Interpolierende $S \in \mathcal{S}^{[3]}_{[x_0,\ldots,x_n]}$ mit $S(x_0)=f_0,\ldots,S(x_n)=f_n$ und vollständigen Randbedingungen $S'(x_0)=f'_0$, $S'(x_n)=f'_n$.

a) *Setze $h_0,\ldots,h_{n-1}$ und $\delta_1,\ldots,\delta_{n-1}$ gemäß (8.3) und (8.4).*

b) *Löse (8.5) für $f'_1,\ldots,f'_{n-1}$.*

c) *Bilde aus diesen Daten mittels (8.2) die Werte $a_0,\ldots,d_{n-1}$.*

d) *Ergebnis: für $x \in [x_i, x_{i+1}]$ hat der gesuchte Spline die Darstellung (8.1).*

Die Laufzeit der Schritte a),b),c), mit denen die Spline-Daten $(a_i), (b_i), (c_i), (d_i)$ berechnet werden, ist $O(n)$. Liegen diese Daten vor, so liefert d) für jedes x dann den Interpolationswert mittels eines auf (8.1) angewandten Horner-Schemas

$$s := S(x) = \underbrace{\Big(\underbrace{\Big(\underbrace{d_i}_{} (x - x_i) + c_i \Big)}_{} (x - x_i) + b_i \Big)}_{} (x - x_i) + a_i$$

mit nur 3 Multiplikationen und 4 Additionen:

$$\boxed{\texttt{h:=x-x}_i\texttt{;}\quad \texttt{s:=d}_i\texttt{*h+c}_i\texttt{;}\quad \texttt{s:=s*h+b}_i\texttt{;}\quad \texttt{s:=s*h+a}_i\texttt{;}}$$

Vergleiche dazu den Aufwand der Newton-Interpolation: Berechnung dividierter Differenzen $O(n^2)$, Auswertung des Interpolationspolynoms $O(n)$.

Bemerkung 8.15: *Die Eigenwerte λ der Matrix A des Gleichungssystems (8.5) sind über das Gerschgorin-Kriterim 6.1 durch*

$$\underbrace{\min\Big(\frac{2}{h_0} + \frac{1}{h_1}\,,\ \frac{1}{h_1} + \frac{1}{h_2}\,,\ \ldots \Big)}_{\geq 2/\max_i h_i} \leq \lambda \leq \underbrace{\max\Big(\frac{2}{h_0} + \frac{3}{h_1}\,,\ \frac{3}{h_1} + \frac{3}{h_2}\,,\ \ldots \Big)}_{\leq 6/\min_i h_i}$$

abzuschätzen. Damit ist Konditionierung von (8.5) durch das Verhältnis

$$\mathrm{cond}_2(A) \;=\; \frac{\max\lambda}{\min\lambda} \;\leq\; 3\,\frac{\max(h_0,\ldots,h_{n-1})}{\min(h_0,\ldots,h_{n-1})}$$

des größten und kleinsten Stützstellenabstandes bestimmt, welches in praktischen Anwendungen keine extrem großen Werte annehmen sollte (dies ist z.B. bei äquidistanten Stützstellen garantiert). Die numerische Berechnung kubischer Spline-Daten ist damit in der Regel unproblematisch.

Bemerkung 8.16: *Eine Darstellung der vollständigen kubischen Spline-Interpolierenden mittels B-Splines ist z.B. in* [Oev96, Beispiel 8.12] *zu finden.*

Analoge Ergebnisse ergeben sich für das Interpolationsproblem mit natürlichen, periodischen oder not-a-knot Randbedingungen (siehe z.B. [Oev96]).

Zusammenfassung: Die Daten der kubischen Spline-Interpolierenden $S(x_0) = f_0, \ldots, S(x_n) = f_n$ können durch Lösung eines einfachen (tridiagonalen) Gleichungssystems mit $O(n)$ Operationen numerisch stabil berechnet werden. Die folgende Auswertung von Interpolationspunkten $S(x)$ ist sehr schnell.

8.3.2 Die Minimaleigenschaft kubischer Splines

Die kubischen Splines sind "physikalisch" ausgezeichnet:

Satz 8.17: (Minimaleigenschaft der Spline-Interpolierenden)
Unter allen $f \in C^2([x_0, x_n])$, die das Interpolationsproblem $f(x_0) = f_0, \ldots, f(x_n) = f_n$ sowie die natürlichen bzw. vollständigen bzw. periodischen Randbedingungen erfüllen, ist die kubische Spline-Interpolierende diejenige Funktion, welche die durch

$$E(f) \;=\; \int_{x_0}^{x_n} (f''(x))^2 \; dx$$

definierte "mittlere Krümmung" minimiert.

Beweis: Sei S die eindeutige Spline-Interpolierende, sei f eine beliebige andere zweifach stetig diff'bare Funktion, welche die Interpolationsbedingungen sowie die entsprechenden natürlichen/vollständigen/periodischen Randbedingungen erfüllt. Mit $\delta := f - S \in C^2([x_0, x_n])$ gilt

$$E(f) \;=\; E(S+\delta) \;=\; E(S) \;+\; \underbrace{E(\delta)}_{\geq 0} \;+\; 2 \int_{x_0}^{x_n} S''(x)\delta''(x) \; dx \;.$$

Ist gezeigt, daß das letzte Integral verschwindet, so folgt unmittelbar die Behauptung: $E(f) = E(S) + E(\delta) \geq E(S)$. Zweimalige partielle Integration liefert

$$\int_{x_0}^{x_n} S''(x)\delta''(x) \; dx \;=\; \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} S''(x)\delta''(x) \; dx$$

$$= \; \sum_{i=0}^{n-1} \left(\; \Big[S''(x)\delta'(x) - S'''(x)\delta(x) \Big]_{x=x_i+0}^{x=x_{i+1}-0} \; + \; \int_{x_i}^{x_{i+1}} S''''(x)\delta(x) \; dx \right) \;,$$

wobei das letzte Integral wegen $S''''(x) = 0$ verschwindet, da die Spline-Interpolierende auf den Teilintervallen ein Polynom maximal dritten Grades ist. Es gilt $\delta(x_0) = \ldots = \delta(x_n) = 0$, da sowohl S als auch f die Interpolationsbedingungen erfüllen, sodaß sich mit der Stetigkeit von S'' und δ'

$$\int_{x_0}^{x_n} S''(x)\delta''(x) \; dx = \sum_{i=0}^{n-1} \Big[S''(x)\delta'(x) \Big]_{x=x_i}^{x=x_{i+1}} = S''(x_n)\delta'(x_n) - S''(x_0)\delta'(x_0)$$

ergibt. Bei natürlichen Randbedingungen gilt $S''(x_0) = S''(x_n) = 0$. Bei vollständigen Randbedingungen gilt $\delta'(x_0) = \delta'(x_n) = 0$, da S und f dieselben vorgeschriebenen Ableitungswerte an den Rändern annehmen. Im periodischen Fall gilt $S''(x_n) = S''(x_0)$ und $\delta'(x_n) = \delta'(x_0)$, da die Periodizität der ersten Ableitung sowohl für S als auch für f angenommen wird. In allen betrachteten Fällen verschwindet somit das obige Integral.

Q.E.D.

Diese Argumente zeigen auch, daß die Spline-Interpolierende durch ihre Minimaleigenschaft bereits eindeutig festgelegt ist. Aus $E(f) = E(S)$ folgt nämlich notwendigerweise $E(\delta) = \int_{x_0}^{x_n} (\delta''(x))^2 dx = 0$, sodaß $\delta''(x) = f''(x) - S''(x) \equiv 0$ gelten muß. Damit stimmen f und S bis auf einen linearen Ausdruck überein, sodaß f selbst eine kubische Spline-Funktion sein muß. Wegen der Eindeutigkeit der Spline-Interpolierenden folgt $f \equiv S$.

Physikalische Interpretation des Satzes: Zu einer Funktion f betrachte man das vom Graphen definierte Kurvenstück $\{ (x, f(x)) \in \mathbb{R}^2 \; ; \; x \in [x_0, x_n] \}$ mit dem durch Integration gemittelten Krümmungsquadrat

$$K(f) \;=\; \int_{x_0}^{x_n} \frac{(f''(x))^2}{1 + (f'(x))^2} \; dx \, .$$

Betrachtet man die durch Aufbringen äußerer Kräfte erzwungene Deformation eines dünnen Stabes (englisch “spline”), so nimmt dieser nach fundamentalen physikalischen Prinzipien diejenige Form an, welche die Deformationsenergie minimiert. Diese ist entsprechenden Modellen der Elastizitätstheorie gemäß durch das Quadrat der Kurvenkrümmung gegeben, sodaß $K(f)$ bis auf einen physikalischen Faktor die im Stab gespeicherte Energie darstellt. Wenn man an den Stellen $x_0, \ldots, x_n$ durch Aufbringen geeigneter Punktkräfte die Position $f_0, \ldots, f_n$ jeweils vorschreibt (also mathematisch ein Interpolationsproblem vorgibt), so ergibt sich die vom Stab zwischen den Stützstellen angenommene Kurve als Lösung des Minimierungsproblems für $K(f)$.

Ersetzt man unter der Voraussetzung $|f'(x)| \ll 1$ das physikalische Integral $K(f)$ durch den mathematisch einfacher zu behandelnden Ausdruck $E(f)$, so liefern die Standardmethoden der Variationsrechnung unmittelbar das Resultat, daß für die minimierende Funktion zwischen den Stützstellen $f''''(x) = 0$ gelten muß, d.h., daß die Interpolierende stückweise ein Polynom maximal dritten Grades ist. In dieser Näherung ergibt sich damit folgende **physikalische Interpretation der kubischen Spline-Interpolierenden: sie beschreibt den Verlauf eines dünnen Stabes, der an den Stellen $(x_0, f_0), \ldots, (x_n, f_n)$ fixiert wurde.** In der Tat wurde dies früher in der Praxis verwendet, um eine Anzahl auf dem Zeichenpapier vorgegebener Punkte durch eine Kurve zu verbinden, indem ein dünner biegsamer Stab an diesen Stellen festgeklemmt wurde. Da jenseits der äußersten Klemmpunkte (x_0, f_0) und (x_n, f_n) keine weiteren Kräfte auf den Stab wirken, weist dieser außerhalb des Bereichs $[x_0, x_n]$ keinerlei Krümmung auf, sodaß speziell in den Randpunkten mit der Krümmung die zweite Ableitung der Kurve verschwindet. Dies entspricht den “natürlichen” Randbedingungen.

8.3.3 Fehlerabschätzungen

Sei $f_i = f(x_i)$ die Wertetabelle einer glatten Funktion f. Im Gegensatz zur Polynominterpolation ergeben sich *gleichmäßige* Fehlerabschätzungen, die bei kubischen Splines von der vierten Potenz des Stützstellenabstandes abhängen. Dieses Ergebnis ist dabei für die unterschiedlichen Randvorgaben "natürlich", "vollständig" etc. ähnlich und soll hier nur für den vollständigen Fall genauer formuliert werden:

Satz 8.18: (Interpolationsfehler kubischer Splines)

Sei $f \in C^4([x_0, x_n])$, sei $S \in \mathcal{S}^{[3]}_{[x_0,\ldots,x_n]}$ die kubische Spline-Interpolierende zu $x_0 < \ldots < x_n$ mit "vollständigen" Randbedingungen:

$$S(x_0) = f(x_0)\ ,\ \ldots\ ,\ S(x_n) = f(x_n)\ ,\ S'(x_0) = f'(x_0)\ ,\ S'(x_n) = f'(x_n)\ .$$

Mit

$$h := \max_{i=1\ldots n}(x_i - x_{i-1})\ ,\quad \kappa := \frac{\max\limits_{i=1\ldots n}(x_i - x_{i-1})}{\min\limits_{i=1\ldots n}(x_i - x_{i-1})}\ \max_{\xi\in[x_0,x_n]}|f^{(4)}(\xi)|$$

gilt für alle[1] *$x \in [x_0, x_n]$:*

$$\begin{aligned} |\,S(x) - f(x)\,| &\le h^4\,\kappa\ , & |\,S'(x) - f'(x)\,| &\le 2\,h^3\,\kappa\ , \\ |\,S''(x) - f''(x)\,| &\le 2\,h^2\,\kappa\ , & |\,S'''(x) - f'''(x)\,| &\le 2\,h\,\kappa\ . \end{aligned}$$

Beweis: sehr technisch, siehe z.B. [Oev96].

Damit werden die ersten Ableitungen einer Funktion durch die entsprechenden Ableitungen der Spline-Interpolierenden für $h \to 0$ *gleichmäßig* angenähert!

Bemerkung 8.19: *Auch im Fall natürlicher Randbedingungen ergibt sich eine solche Abschätzung,* **falls die die Daten erzeugende Funktion ebenfalls die natürliche Randbedingung $f''(x_0) = f''(x_n) = 0$ erfüllt**. *Wenn nicht, so erhält man nur $S(x) - f(x) = O(h^2)$. Analoges gilt für Interpolation mit periodischen Splines, wobei der Fehler in der Regel quadratisch ist und nur für "periodische" Ausgangsfunktionen mit $f(x_0) = f(x_n)$, $f'(x_0) = f'(x_n)$, $f''(x_0) = f''(x_n)$ von vierter Ordnung wird.*

[1] S''' ist stückweise konstant und an den Sprung(=Stütz)stellen nicht definiert, sodaß für die dritte Ableitung $x \neq x_i$ vorauszusetzen ist.

Dementsprechend wird man sinnvollerweise nur Funktionen mit natürlichen/periodischen Randbedingungen durch natürliche/periodische Splines interpolieren. In anderen Fällen wird man vollständige Randdaten benutzen oder – falls die Ableitungswerte $f'(x_0)$ und $f'(x_n)$ nicht bekannt sind – zu not-a-knot Vorgaben übergehen. Letztere ermöglicht eine Interpolation vierter Ordnung, auch wenn die die Daten erzeugende Funktion unbekannt ist und keinerlei Informationen am Rand zur Verfügung stehen.

Bemerkung 8.20: *Mit Splines, die auf den Teilintervallen durch Polynome höheren Grades gegeben sind, lassen sich Interpolationen erzeugen, die die Ausgangsfunktion bis auf entsprechend höhere Potenzen im Stützstellenabstand h annähern. Da höhere Polynome jedoch dazu tendieren, zwischen den Stützstellen stärker zu oszillieren, werden die entsprechenden Koeffizienten in den Fehlerabschätzungen größer. Man begnügt sich in der Praxis daher oft mit kubischen Splines, wobei die Stützstellen im Rahmen der gewünschten Interpolationsgenauigkeit hinreichend dicht zu wählen sind.*

Bemerkung 8.21: *Eine Anwendung: der menschliche Körper kann sich auf sich stetig verändernde Kräfte (in einem gewissen Rahmen) einstellen. Sich unstetig verändernde Kräfte werden als unangenehme "Schläge" empfunden. Daher wird der Kurvenverlauf von Achterbahnen oft mit Hilfe kubischer Splines entworfen. Die auf den Passagier wirkenden Kräfte sind durch die Beschleunigung (die zweite Ableitung der Kurve) bestimmt. Diese sind bei kubischen Splines stetig!*

Kapitel 9

Quadratur (Integration)

Aufgabe: zu einem endlichen Integrationsintervall $[a, a+L]$ finde **Knoten** (**Stützstellen**) $x_0, \ldots, x_n$ und **Gewichte** $b_0, \ldots, b_n$, so daß die **Quadraturformel**

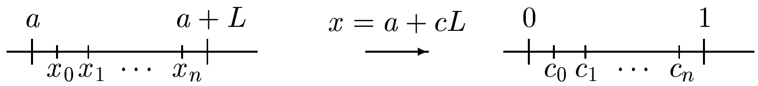
$$Q_n[f] \;=\; L \sum_{j=0}^{n} b_j\, f(x_j) \;\approx\; \int_a^{a+L} f(x)\, dx$$

mit den **"Daten"** $x_0, \ldots, x_n, b_0, \ldots, b_n$ für hinreichend glatte Integranden f gute Approximationen des Integrals liefert.

Bemerkung 9.1: *Bilde das Integrationsintervall $x \in [a, a+L]$ mit $x(c) = a + cL$ auf das Standardintervall $c \in [0,1]$ ab:*

$$\int_a^{a+L} f(x)\, dx \;=\; L \int_0^1 f(a+cL)\, dc\ .$$

Den Stützstellen $x_j \in [a, a+L]$ ensprechen dann $c_j \in [0,1]$ mit $x_j = a + c_j L$:



Die c_j beschreiben die "Verteilung der Stützstellen" unabhängig von der Lage a und der Länge L des Integrationsintervalls. Also: alternativ zu $x_0, \ldots, x_n, b_0, \ldots, b_n$ suche nach $c_0, \ldots, c_n, b_0, \ldots, b_n$, so daß

$$\begin{aligned} Q_n[f] \;&=\; L \sum_{j=0}^{n} b_j\, f(x_j) \;=\; L \sum_{j=0}^{n} b_j\, f(a + c_j L) \\ &\approx\; \int_a^{a+L} f(x)\, dx \;=\; L \int_0^1 f(a+cL)\, dc\ . \end{aligned}$$

Idee zur Bestimmung der Daten: fordere, daß die Quadraturformel exakt ist für alle Polynome bis zu möglichst hohem Grad.

Bemerkung 9.2: *Der maximal erreichbare Exaktheitsgrad einer Quadraturformel mit $n+1$ Knoten $x_0, \ldots, x_n$ ist $2n+1$, denn für das Polynom*

$$P_{2n+2}(x) \;=\; (x-x_0)^2 \cdots (x-x_n)^2$$

vom Grad $2n+2$ gilt $Q_n[P_{2n+2}] = 0 \neq \int_a^{a+L} P_{2n+2}(x)\,dx > 0$. Der Exaktheitsgrad $2n+1$ ist durch geschickte Knotenwahl erreichbar (Gauß-Quadratur).

Bemerkung 9.3: *Für die Monome $f_i(x) = (x-a)^i$ gilt*

$$\int_a^{a+L} (x-a)^i\,dx \;\stackrel{(x=a+cL)}{=}\; L \int_0^1 (cL)^{i-1} dc \;=\; \frac{L^{i+1}}{i+1}\,,$$

$$Q_n[f_i] \;=\; L \sum_{j=0}^{n} b_j\,(c_j L)^i \;=\; L^{i+1} \sum_{j=0}^{n} b_j\,c_j^i\,.$$

Die Quadraturformel ist damit genau dann exakt für alle Polynome vom Grad $\leq q$, wenn die Daten $c_0, \ldots, c_n, b_0, \ldots, b_n$ das Gleichungssystem

$$\sum_{j=0}^{n} b_j\,c_j^i \;=\; \frac{1}{i+1}\,, \quad i = 0, \ldots, q$$

lösen.

9.1 Newton-Cotes-Formeln

Die Stützstellen $x_0, \ldots, x_n$ bzw. $c_0, \ldots, c_n$ (mit $x_j = a + c_j h$) sind paarweise verschieden vorgegeben. Aufgabe: finde $b_0, \ldots, b_n$.

Satz 9.4: (Newton-Cotes-Quadratur)

Wähle

$$b_j \;=\; \frac{1}{L}\int_a^{a+L} L_j(x)\,dx \;\stackrel{(x=a+cL)}{=}\; \int_0^1 L_j^*(c)\,dc\,, \quad j = 0, \ldots, n$$

mit den Lagrange-Polynomen

$$L_j(x) \;=\; \prod_{\substack{k=0\\k\neq j}}^{n} \frac{x-x_k}{x_j-x_k} \;\underset{(x_i=a+c_iL)}{\stackrel{(x=a+cL)}{=}}\; L_j^*(c) \;=\; \prod_{\substack{k=0\\k\neq j}}^{n} \frac{c-c_k}{c_j-c_k}\,.$$

Dann ist die **Newton-Cotes-Formel**

$$\mathrm{NC}_n[f] \;:=\; L\sum_{j=0}^{n} b_j\, f(x_j) \;=\; L\sum_{j=0}^{n} b_j\, f(a+c_j h)$$

zu den Knoten $x_0,\ldots,x_n$ für alle Polynome P bis zum Grad n exakt: $\mathrm{NC}_n[P] = \int_a^{a+L} P(x)\,dx$.

Beweis: Nach Satz 7.4 gilt $P(x) = \sum_{j=0}^{n} P(x_j)\, L_j(x)$ für Polynome P vom Grad $\le n$, denn P ist seine eigene Polynominterpolierende. Es folgt

$$\int_a^{a+L} P(x)\,dx \;=\; \sum_{j=0}^{n} P(x_j)\,\underbrace{\int_a^{a+L} L_j(x)\,dx}_{L\,b_j} \;=\; \mathrm{NC}_n[P]\ .$$

Q.E.D.

Bemerkung 9.5: *Die so gewählten Gewichte b_j hängen damit nicht vom Intervall $[a, a+L]$ ab, sondern nur von der Knotenverteilung $c_0,\ldots,c_n$.*

Beispiel 9.6: Betrachte äquidistante Stützstellen $c_j = j/n$, $j = 0,\ldots,n$. Mit den Gewichten

$$b_j \;=\; \int_0^1 \prod_{\substack{k=0\\k\neq j}}^{n} \frac{c-c_k}{c_j-c_k}\,dc \;=\; \int_0^1 \prod_{\substack{k=0\\k\neq j}}^{n} \frac{n\,c-k}{j-k}\,dc$$

erhält man die **einfachen Newton-Cotes-Formeln** zum Interpolationsgrad n:

$n=1$ (Trapezformel):

$$\int_a^{a+L} f(x)\,dx \;=\; \frac{L}{2}\,\Big(f(a)+f(a+L)\Big) \;+\; \mathrm{Fehler}_1[f]\ .$$

$n=2$ (Simpsonformel):

$$\int_a^{a+L} f(x)\,dx \;=\; \frac{L}{6}\,\Big(f(a)+4f(a+\tfrac{L}{2})+f(a+L)\Big) \;+\; \mathrm{Fehler}_2[f]\ .$$

$n=3$ (Newton's 3/8-Regel):

$$\int_a^{a+L} f(x)\,dx = \frac{L}{8}\,\Big(f(a)+3f(a+\tfrac{L}{3})+3f(a+\tfrac{2L}{3})+f(a+L)\Big) \;+\; \mathrm{Fehler}_3\,[f]\ .$$

$n=4$ (Milne-Formel):

$$\int_a^{a+L} f(x)\,dx \;=\; \frac{L}{90}\,\Big(7f(a)+32\,f(a+\tfrac{L}{4})+12\,f(a+\tfrac{L}{2}) +32\,f(a+\tfrac{3L}{4})+7\,f(a+L)\Big) \;+\; \mathrm{Fehler}_4\,[f]\ .$$

Interpretation 9.7: *Sei* $P_n(x) = \sum_{j=0}^{n} f(x_j)\, L_j(x)$ *das Interpolationspolynom zur Wertetabelle* $(x_0, f(x_0)), \ldots, (x_n, f(x_n))$. *Es folgt*

$$\int_a^{a+L} P_n(x)\, dx \;=\; \sum_{j=0}^{n} f(x_j) \int_a^{a+L} L_j(x)\, dx \;=\; L \sum_{j=0}^{n} b_j\, f(x_j) \;=\; \mathrm{NC}_n[f] \;,$$

also $\boxed{\text{Quadraturformel = exaktes Integral über das Interpolationspolynom}}$.

Mit

$$\text{Quadraturfehler} \;=\; \int_a^{a+L} \Big(f(x) \;-\; P_n(x) \Big)\, dx$$

liefert Satz 7.12.a) Fehlerabschätzungen:

Satz 9.8: (Fehler der Newton-Cotes-Quadratur)

Sei $\mathrm{NC}_n[f]$ *die Newton-Cotes-Formel aus Satz 9.4 zu den Stützstellen* $x_j = a + c_j L \in [a, a+L]$ *mit der Verteilung* $c_0 < \ldots < c_n$ *in* $[0,1]$. *Für* $(n+1)$*-fach stetige Integranden gilt*

$$\left| \int_a^{a+L} f(x)\, dx \;-\; \mathrm{NC}_n[f] \right| \;\le\; d_n\, L^{n+2} \max_{x \in [a, a+L]} |f^{(n+1)}(x)|$$

mit

$$d_n \;:=\; \frac{1}{(n+1)!} \int_0^1 |c - c_0| \cdots |c - c_n|\, dc \;.$$

Für gerades n *und* $(n+2)$*-fach stetige Integranden gilt sogar*

$$\left| \int_a^{a+L} f(x)\, dx \;-\; \mathrm{NC}_n[f] \right| \;\le\; e_n\, L^{n+3} \max_{x \in [a, a+L]} |f^{(n+2)}(x)|$$

mit

$$e_n := \frac{1}{(n+2)!} \int_0^1 |c - \tfrac{1}{2}|\, |c - c_0| \cdots |c - c_n|\, dc \;,$$

falls die Stützstellen symmetrisch im Intervall liegen:

$$x_{n-i} = 2\,a + L - x_i \;, \quad \text{d.h.,} \quad c_{n-i} = 1 - c_i \;, \quad i = 0, \ldots, n \;.$$

Beweis: Nach Satz 7.12.a) gilt für das Interpolationspolynom $P_n(x)$ durch $(x_0, f(x_0)), \ldots, (x_n, f(x_n))$:

$$
\begin{aligned}
f(x) - P_n(x) \;&=\; \frac{f^{(n+1)}(\xi)}{(n+1)!}\,(x-x_0)\cdots(x-x_n) \quad \text{mit} \quad \xi \in [a, a+L] \\
\Longrightarrow \quad \left| \int_a^{a+L} f(x)\,dx - \mathrm{NC}_n[f] \right| \;&=\; \left| \int_a^{a+L} \frac{f^{(n+1)}(\xi)}{(n+1)!}\,(x-x_n)\cdots(x-x_n)\;dx \right| \\
&\leq\; \frac{1}{(n+1)!} \int_a^{a+L} |x-x_0|\cdots|x-x_n|\,dx \quad \Big(\max_{\xi\in[a,a+L]} |f^{(n+1)}(\xi)| \Big) \\
&\overset{(*)}{=}\; L^{n+2} \underbrace{\frac{1}{(n+1)!} \int_0^1 |c-c_0|\cdots|c-c_n|\,dc}_{d_n} \quad \Big(\max_{\xi\in[a,a+L]} |f^{(n+1)}(\xi)| \Big)
\end{aligned}
$$

mit der Substitution $x = a + cL$ in $(*)$. Für gerades n nehme einen beliebigen weiteren Punkt $x_{n+1} = a + c_{n+1}L \in [a, a+L]$ mit $x_{n+1} \notin \{x_0, \ldots, x_n\}$ hinzu. Mit dem Interpolationspolynom P_{n+1} durch $(x_0, f(x_0)), \ldots, (x_{n+1}, f(x_{n+1}))$ folgt wie oben mit $n \mapsto n+1$:

$$
\begin{aligned}
&\left| \int_a^{a+L} f(x)\,dx \;-\; \int_a^{a+L} P_{n+1}(x)\,dx \right| \;\leq \\
&\frac{L^{n+3}}{(n+2)!} \int_0^1 |c-c_0|\cdots|c-c_n|\,|c-c_{n+1}|\,dc \quad \Big(\max_{\xi\in[a,a+L]} |f^{(n+2)}(\xi)| \Big) \;.
\end{aligned}
$$

Es wird gezeigt, daß bei symmetrischen Stützstellen

$$
\int_a^{a+L} P_{n+1}(x)\,dx \;=\; \int_a^{a+L} P_n(x)\,dx \;=\; \mathrm{NC}_n[f]
$$

gilt. Aus Satz 7.10 folgt $P_{n+1}(x) = P_n(x) + f_{[x_0,\ldots,x_{n+1}]}\,(x-x_0)\cdots(x-x_n)$, d.h.,

$$
\int_a^{a+L} P_{n+1}(x)\,dx \;=\; \mathrm{NC}_n[f] \;+\; f_{[x_0,\ldots,x_{n+1}]} \int_a^{a+L} (x-x_0)\cdots(x-x_n)\,dx \;.
$$

Für eine symmetrische Verteilung $x_{n-i} = 2a + L - x_i$ folgt mit $x = 2a + L - \xi$:

$$
\begin{aligned}
I \;&:=\; \int_a^{a+L} (x-x_0)\cdots(x-x_n)\,dx \\
&=\; -\int_{a+L}^{a} (2a+L-\xi-x_0)\cdots(2a+L-\xi-x_n)\,d\xi \\
&=\; \int_a^{a+L} (x_n-\xi)\cdots(x_0-\xi)\,d\xi \;=\; (-1)^{n+1}\,I \;.
\end{aligned}
$$

Für gerades n folgt $I = 0$, also

$$\left| \int_a^{a+L} f(x)\,dx \;-\; \mathrm{NC}_n[f] \right| \;\leq$$

$$\frac{L^{n+3}}{(n+2)!} \int_0^1 |c-c_0| \cdots |c-c_n|\,|c-c_{n+1}|\,dc \quad \Big(\max_{\xi \in [a,a+L]} |f^{(n+2)}(\xi)| \Big)$$

mit beliebigem c_{n+1}, z.B. $c_{n+1} = 1/2$ (aus Stetigkeitsgründen darf nun c_{n+1} auch mit einem der $c_0, \ldots, c_n$ übereinstimmen).

Q.E.D.

Bemerkung 9.9: *Für äquidistante Stützstellen $c_j = j/n$ kann man (mit wesentlich größerem Aufwand) sogar zeigen, daß*

$$\int_a^{a+L} f(x)\,dx \;-\; \mathrm{NC}_n[f] \;=\; \begin{cases} d_n^*\, L^{n+2}\, f^{(n+1)}(\xi)\ , & n \text{ ungerade} \\ e_n^*\, L^{n+3}\, f^{(n+2)}(\xi)\ , & n \text{ gerade} \end{cases}$$

mit den (positiven) Konstanten

$$d_n^* \;=\; -\,\frac{1}{(n+1)!} \int_0^1 c\,(c-\tfrac{1}{n})(c-\tfrac{2}{n}) \cdots (c-1)\,dc\ , \quad n \text{ ungerade}$$

bzw.

$$e_n^* \;=\; -\,\frac{1}{(n+2)!} \int_0^1 c\,(c-\tfrac{1}{n})(c-\tfrac{2}{n}) \cdots (c-\tfrac{1}{2})^2 \cdots (c-1)\,dc\ , \quad n \text{ gerade}$$

und geeigneten Zwischenwerten $\xi \in (a, a+L)$ gilt (keine Betragszeichen!).

Bemerkung 9.10: *Der Exaktheitsgrad von NC_n bei äquidistanten (allgemeiner: bei symmetrischen) Knoten ist damit für gerades n nicht nur n, sondern sogar $n+1$.*

Bemerkung 9.11: *Der Interpolationsgrad n liefert einerseits den polynomialen Exaktheitsgrad n bzw. $n+1$ für ungerades bzw. gerades n. Andererseits beschreibt er die Abhängigkeit des Fehlers von der Intervalllänge ("Ordnung"):*

$$\int_a^{a+L} f(x)\,dx \;-\; \mathrm{NC}_n[f] \;=\; \begin{cases} O(L^{n+2})\ , & n \text{ ungerade,} \\ O(L^{n+3})\ , & n \text{ gerade.} \end{cases}$$

Die höheren Formeln sind damit i.a. <u>für kleine Intervalle</u> die genaueren.

Um bei großen Intervallen $[a,b]$ bzw. stark variierendem Integranden hohe Genauigkeiten erreichen zu können, sollte man wegen Bemerkung 7.14 bei den äquidistanten Newton-Cotes-Formeln nicht einfach den Interpolationsgrad n beliebig erhöhen. Stattdessen wendet man die einfachen Newton-Cotes-Formeln nicht auf das gesamte Intervall an, sondern zerlegt $[a,b]$ äquidistant in N Gruppen von je n Punkten

$$x_j \;=\; a \;+\; j\,h\;, \quad j \;=\; 0,1,\ldots,nN\;, \quad h \;=\; \frac{b-a}{nN}\;.$$

Das Gesamtintegral

$$\int_a^b f(x)\;dx \;=\; \int_{x_0}^{x_n} f(x)\;dx \;+\; \int_{x_n}^{x_{2n}} f(x)\;dx \;+\; \cdots \;+\; \int_{x_{(N-1)n}}^{x_{Nn}} f(x)\;dx$$

ergibt sich dann mit $x_{(i+1)n} = x_{in} + L$, $L = nh$, als die Summe der Approximationen

$$\int_{x_{in}}^{x_{(i+1)n}} f(x)\;dx \;\approx\; n\,h\,\sum_{j=0}^{n} b_j\,f(x_{in+j})\;, \quad i = 0,1,\ldots,N-1$$

auf den Teilintervallen. Die Fehler der einfachen Newton-Cotes-Formeln addieren sich auf, und es ergeben sich die

Aufsummierten Newton-Cotes-Formeln 9.12:

$$\mathrm{NC}_{N,n}[f] \;:=\; n\,h\,\sum_{i=0}^{N-1}\sum_{j=0}^{n} b_j\,f(x_{in+j}) \;=\; \int_a^b f(x)\;dx \;+\; F_{N,n}[f]$$

zum Interpolationsgrad n mit N Teilintervallen und den äquidistanten Stützstellen

$$x_{in+j} \;=\; a \;+\; (in+j)\,h\;, \quad h \;=\; \frac{b-a}{nN}\;.$$

Ihr Quadraturfehler ist mit Bemerkung 9.9

$$\begin{aligned} |\,F_{N,n}[f]\,| \;&\leq\; \sum_{i=0}^{N-1} d_n^*\,L^{n+2} \max_{\xi_i\in[x_{in},x_{(i+1)n}]} |f^{(n+1)}(\xi_i)| \\ &\leq\; d_n^*\,n^{n+1}\,(b-a)\;h^{n+1} \max_{\xi\in[a,b]} |f^{(n+1)}(\xi)| \end{aligned}$$

für ungerades n bzw.

$$|\,F_{N,n}[f]\,| \;\leq\; e_n^*\,n^{n+2}\,(b-a)\;h^{n+2} \max_{\xi\in[a,b]} |f^{(n+2)}(\xi)|$$

für gerades n. Hierbei ist der Abstand benachbarter Stützstellen $h = \frac{b-a}{n\,N}$ bei festem Interpolationsgrad n durch Wahl von N beliebig klein zu machen.

Speziell ergibt sich für $n = 1$ die **Trapezregel**

$$\boxed{\begin{aligned} \int_a^b f(x)\,dx \;&=\; \frac{h}{2}\left(f(a) \;+\; 2\sum_{i=1}^{N-1} f(x_i) \;+\; f(b)\right) \;+\; F_{N,1}[f] \\ |\,F_{N,1}[f]\,| \;&\le\; \frac{b-a}{12}\,h^2 \max_{\xi\in[a,b]} |f^{(2)}(\xi)|\;, \quad x_i = a + i\,h\;, \quad h \;=\; \frac{b-a}{N}\;, \end{aligned}}$$

für $n = 2$ die **Simpson-Regel**

$$\boxed{\begin{aligned} \int_a^b f(x)\,dx \;&=\; \frac{h}{3}\left(f(a) \;+\; 2\sum_{i=1}^{N-1} f(x_{2i}) \;+\; 4\sum_{i=0}^{N-1} f(x_{2i+1}) \;+\; f(b)\right) \\ &\qquad + \; F_{N,2}[f] \\ |\,F_{N,2}[f]\,| \;&\le\; \frac{b-a}{180}\,h^4 \max_{\xi\in[a,b]} |f^{(4)}(\xi)|\;, \quad x_i = a + i\,h\;, \quad h \;=\; \frac{b-a}{2\,N}\;, \end{aligned}}$$

für $n = 4$ die **Milne-Regel**

$$\boxed{\begin{aligned} \int_a^b f(x)\,dx \;&=\; \frac{2\,h}{45}\Bigg(7\Big\{f(a)+f(b)\Big\} \;+\; 14\sum_{i=1}^{N-1} f(x_{4i}) \\ &\qquad + \; 32\sum_{i=0}^{N-1}\Big\{f(x_{4i+1})+f(x_{4i+3})\Big\} \;+\; 12\sum_{i=0}^{N-1} f(x_{4i+2})\Bigg) \;+\; F_{N,4}[f] \\ |\,F_{N,4}[f]\,| \;&\le\; \frac{2\,(b-a)}{945}\,h^6 \max_{\xi\in[a,b]} |f^{(6)}(\xi)|\;, \quad x_i = a + i\,h\;, \quad h \;=\; \frac{b-a}{4\,N}\;, \end{aligned}}$$

und für $n = 6$ die **Weddle-Regel**

$$\boxed{\begin{aligned} \int_a^b f(x)\,dx \;&=\; \frac{h}{140}\Bigg(41\Big\{f(a)+f(b)\Big\} \;+\; 82\sum_{i=1}^{N-1} f(x_{6i}) \\ &\qquad + \; 216\sum_{i=0}^{N-1}\Big\{f(x_{6i+1})+f(x_{6i+5})\Big\} \;+\; 27\sum_{i=0}^{N-1}\Big\{f(x_{6i+2})+f(x_{6i+4})\Big\} \\ &\qquad + \; 272\sum_{i=0}^{N-1} f(x_{6i+3})\Bigg) \;+\; F_{N,6}[f] \\ |\,F_{N,6}[f]\,| \;&\le\; \frac{3\,(b-a)}{2800}\,h^8 \max_{\xi\in[a,b]} |f^{(8)}(\xi)|\;, \quad x_i = a + i\,h\;, \quad h \;=\; \frac{b-a}{6\,N}\;. \end{aligned}}$$

Beispiel 9.13: Der Effekt der unterschiedlichen Fehlerordnungen der Newton-Cotes-Formeln soll für das einfache Beispiel $\int_0^1 e^{-x^2/2}\,dx$ demonstriert werden. Mit

$$f(x) = e^{-x^2/2}\,, \quad f^{(2)}(x) = (x^2-1)\,f(x)\,, \quad f^{(4)}(x) = (3-6\,x^2+x^4)\,f(x) \quad \text{etc.}$$

folgt wegen der Monotonie der Ableitungen

$$\max_{\xi\in[0,1]} |f^{(2)}\xi)| = |f^{(2)}(0)| = 1\,, \quad \max_{\xi\in[0,1]} |f^{(4)}(\xi)| = |f^{(4)}(0)| = 3\,,$$

$$\max_{\xi\in[0,1]} |f^{(6)}(\xi)| = |f^{(6)}(0)| = 15\,, \quad \max_{\xi\in[0,1]} |f^{(8)}(\xi)| = |f^{(8)}(0)| = 105\,.$$

Zum Erreichen einer garantierten Genauigkeit von $\varepsilon = 10^{-10}$ wird mit den angegebenen Fehlerformeln ein Stützstellenabstand

$$h \le \sqrt{\frac{12\,\varepsilon}{(b-a)\,\max|f^{(2)}|}} \approx \frac{1}{28\,867.51...} \qquad (n=1,\text{ Trapez})$$

$$h \le \sqrt[4]{\frac{180\,\varepsilon}{(b-a)\,\max|f^{(4)}|}} \approx \frac{1}{113.62...} \qquad (n=2,\text{ Simpson})$$

$$h \le \sqrt[6]{\frac{945\,\varepsilon}{2\,(b-a)\,\max|f^{(6)}|}} \approx \frac{1}{10.59...} \qquad (n=4,\text{ Milne})$$

$$h \le \sqrt[8]{\frac{2800\,\varepsilon}{3\,(b-a)\,\max|f^{(8)}|}} \approx \frac{1}{4.22...} \qquad (n=6,\text{ Weddle})$$

benötigt, d.h., man braucht 28 868 bzw. 115 bzw. 13 bzw. 7 äquidistante Stützstellen (die Anzahl ist von der Form $nN+1$, $N\in\mathbb{N}$). Da in diesem Beispiel die Ableitungsmaxima des Integranden nur langsam wachsen, ergibt sich ein dramatischer Unterschied im benötigten Rechenaufwand.

Bemerkung 9.14: *Die zusammengesetzte Trapezregel ist trotz ihrer niedrigen Ordnung $n=1$ für* periodische *Integranden ideal, wenn über eine Periode integriert wird (vergleich z.B. [Oev96, Kapitel 9.2]).*

Bemerkung 9.15: *Für $n \gg 1$ sind die Newton-Cotes-Formeln numerisch instabil und praktisch unbrauchbar. Dies liegt daran, daß für $n \ge 8$ negative Gewichte b_j auftauchen und für die meisten Gewichte $|b_j| \gg 1$ gilt, wodurch es zu starker Auslöschung kommt*[1]*. In der Praxis beschränkt man sich auf $n \le 6$ und wählt N in den aufsummierten NC-Formeln 9.12 hinreichend groß, um höhere Genauigkeiten zu erreichen.*

[1] Wegen $\sum_j b_j = 1$ folgt $b_j \in (0,1)$, wenn alle Gewichte positiv sind. Die Positivität der Gewichte ist ein wesentliches Kriterium für die Stabilität einer Quadratur-Formel. Bei der Newton-Cotes-Formel mit $n = 100$ gilt z.B. $b_{50} \approx -1.2 \times 10^{24}, b_{51} \approx 1.2 \times 10^{24}$. Durch Auslöschung wird bei 16-stelliger Rechnung das Quadraturergebnis ein nur durch Rundung erzeugtes Phantasieresultat.

Zusammenfassung: In den Quadraturformeln nach Newton-Cotes werden bei $n+1$ vorgegebenen Stützstellen die Gewichte als die Integrale der zugeordneten Lagrange-Polynome gewählt, sodaß alle Polynome bis zum Grad n exakt integriert werden. Bei geradem n erhöht sich die Fehlerordnung, da die Formel auch noch für Polynome vom Grad $n+1$ exakt ist. Ein allgemeiner Integrand wird stückweise durch ein Interpolationspolynom niedrigen Grades angenähert, der Integralwert wird numerisch als das Integral über das Polynom berechnet. Abhängig vom Interpolationsgrad n ergeben sich Quadraturformeln, deren Fehler durch die höheren Ableitungen des Integranden bestimmt werden. Bei äquidistanten Stützstellen mit Abstand h ist der Fehler proportional zu h^{n+1} bzw. h^{n+2} für ungerades bzw. gerades n.

9.2 Gauß-Quadratur

Idee: die Knoten x_j (bzw. ihre Verteilung c_j) sind nicht vorgegeben, sondern sollen "optimal" gewählt werden. Schreibtechnisch ist es hier schöner, n Knoten/Gewichte $c_1, \ldots, c_n / b_1, \ldots, b_n$ (statt $c_0, \ldots, c_n / b_0, \ldots, b_n$) zu betrachten. In der Gauß-Quadratur werden die Knoten so konstruiert, daß sich der nach Bemerkung 9.2 maximale Exaktheitsgrad $2n-1$ für n Knoten ergibt. Die entsprechende Quadraturformel

$$G_n[f] \;=\; L \sum_{j=1}^{n} b_j\, f(x_j) \;=\; L \sum_{j=1}^{n} b_j\, f(a + c_j L) \;\approx\; \int_a^{a+L} f(x)\, dx$$

heißt **Gauß-Formel der Knotenzahl** n.

Fakten:

+ das die Daten bestimmende Gleichungssystem in Bemerkung 9.3 mit $q = 2n-1$ hat eine bis auf Permutation eindeutige reelle Lösung,
+ es gilt $c_1, \ldots, c_n \in (0,1)$, d.h., $x_j = a + c_j L \in (a, a+L)$,
+ es gilt $b_j > 0$ (wichtig für numerische Stabilität),
− für $n > 5$ haben die Daten $c_1, \ldots, c_n, b_1, \ldots, b_n$ keine geschlossene Darstellung,
+ man kann $c_1, \ldots, b_n$ aber <u>numerisch</u> schnell und stabil berechnen.

Zugang über Orthogonalpolynome:

Definition 9.16:

$f, g : [a, a+L] \to \mathbb{R}$ seien (quadratisch) integrierbar.

a) $\ll f, g \gg := \displaystyle\int_a^{a+L} f(x)\, g(x)\, dx$ *heißt* **Skalarprodukt**,

b) f **orthogonal** g *bedeutet* $\ll f, g \gg = 0$.

Definition und Satz 9.17: Definiere rekursiv

$$P_k(x) \;=\; x^k \;+\; \sum_{j=0}^{k-1} \frac{\ll x^k, P_j \gg}{\ll P_j, P_j \gg}\, P_j(x)\;, \quad k = 1, 2, \ldots$$

mit dem Start $P_0(x) = 1$.

a) Es gilt $\ll P_n, P \gg = 0$ *für alle Polynome* P *vom Grad* $< n$.

b) P_n *hat genau* n *einfache Nullstellen* $x_j = a + c_j L \in (a, a+L)$, $j = 1, \ldots, n$. *Achtung:* x_j, c_j *hängen von* n *ab!*

c) Wählt man $b_j = \int_0^1 L_j^*(c)\, dc$ *mit den Lagrange-Polynomen*

$$L_j^*(c) \;=\; \prod_{\substack{k=1\\k\neq j}}^{n} \frac{c - c_k}{c_j - c_k}\;, \quad j = 1, \ldots, n\;,$$

so sind $c_1, \ldots, c_n, b_1, \ldots, b_n$ *die Daten der Gauß-Formel* $G_n[f]$. *Es gilt* $b_j > 0$.

Beweis: a) Induktion nach n mit der Induktionsbehauptung

$$\ll P_k, P_j \gg = 0 \quad \forall\, k, j \in \{0, \ldots, n\}\;,\; k \neq j\;.$$

<u>Schritt $n \to n+1$:</u> zu zeigen ist nur $\ll P_{n+1}, P_j \gg = 0\ \forall j = 0, \ldots, n$.

$$\begin{aligned}
\ll P_{n+1}, P_j \gg \;&=\; \ll x^{n+1} - \sum_{k=0}^{n} \frac{\ll x^{n+1}, P_k \gg}{\ll P_k, P_k \gg}\, P_k\;,\; P_j \gg \\
&=\; \ll x^{n+1}, P_j \gg \;-\; \sum_{k=0}^{n} \frac{\ll x^{n+1}, P_k \gg}{\ll P_k, P_k \gg}\, \underbrace{\ll P_k, P_j \gg}_{0 \text{ für } k \neq j} \\
&=\; \ll x^{n+1}, P_j \gg \;-\; \ll x^{n+1}, P_j \gg \;=\; 0\;.
\end{aligned}$$

Jedes P vom Grad $< n$ läßt sich als $P(x) = \sum_{j=0}^{n-1} \alpha_j\, P_j(x)$ schreiben, es folgt

$$\ll P_n, P \gg \;=\; \sum_{j=0}^{n-1} \alpha_j \;\ll P_n, P_j \gg \;=\; 0\;.$$

b) Seien $x_1, \ldots, x_j$ die paarweise verschiedenen reellen Nullstellen von P_n in $(a, a+L)$ mit Vielfachheiten $n_1, \ldots, n_j$. Betrachte

$$P(x) = (x - x_1)^{m_1} \ldots (x - x_j)^{m_j} \quad \text{mit} \quad m_i = \begin{cases} 1 & \text{für ungerades } n_i \\ 0 & \text{für gerades } n_i \end{cases}$$

mit denselben Vorzeichenwechseln wie P_n auf $(a, a+L)$. Damit folgt $\ll P_n, P \gg \neq 0$. Falls $j < n$ gilt, so ist $\mathrm{grad}(P) = m_1 + \cdots + m_j < n$, und mit a) folgt der Widerspruch $\ll P_n, P \gg = 0$.

c) Mit den gewählten b_j gilt nach Satz 9.4 (mit $n \to n-1$), daß $G_n[f]$ bis zum Polynomgrad $n-1$ exakt ist. Für jedes Polynom P vom Grad $\leq 2n-1$ existiert eine Zerlegung

$$P(x) = \alpha(x) P_n(x) + \beta(x)$$

mit Polynomen $\alpha(x), \beta(x)$ vom Grad $\leq n-1$ (Polynomdivision mit Rest):

$$\int_a^{a+L} P(x)\,dx \;-\; G_n[P]$$

$$= \underbrace{\int_a^{a+L} \alpha(x)\,P_n(x)\,dx}_{=\ll \alpha, P_n \gg = 0} + \int_a^{a+L} \beta(x)\,dx - L \sum_{j=1}^{n} b_j \Big(\alpha(x_j)\,\underbrace{P_n(x_j)}_{0} + \beta(x_j)\Big)$$

$$= \int_a^{a+L} \beta(x)\,dx \;-\; G_n[\beta] \;=\; 0\;,$$

da die Quadratur bis zum Grad $n-1$ exakt ist. Damit ist G_n auch bis zum Grad $2n-1$ exakt. Es gilt $b_j = \int_0^1 L_j^*(c)\,dc$, aber auch

$$b_j = \sum_{k=1}^{n} b_k \underbrace{(L_j^*(c_k))^2}_{\delta_{kj}} \;=\; \int_0^1 (L_j^*(c))^2\,dc \;>\; 0\;,$$

da mit $\mathrm{grad}((L_j^*)^2) = 2n-2$ die Quadratur auf dem Standardintervall $[0,1]$ exakt ist.

Q.E.D.

Bezeichnung 9.18: *Die Orthogonalpolynome P_n heißen* **"Legendre-Polynome"** *über dem Intervall $[a, a+L]$. Sie haben die kompakte Darstellung*

$$P_n(x) = \frac{n!}{(2n)!}\,\frac{d^n}{dx^n}\,(x-a)^n (x-(a+L))^n \qquad \textit{(Rodriguez-Formel)}\;.$$

Die Literatur benutzt das Standardintervall $[-1,1]$, also $a=-1, L=2$. Sei $P_n^*(c)$ das n.te Legendre-Polynom über unserem Standardintervall $[0,1]$. Aus der Rodriguez-Darstellung folgt unmittelbar $P_n(a+cL) = L^n\,P_n^*(c)$.

Gauß-Daten 9.19: *Man berechnet:*

$$n=1: \quad P_1^*(c) = c - \tfrac{1}{2}\;, \quad c_1 = \tfrac{1}{2}\;, \quad b_1 = 1\;.$$

$$n=2: \quad P_2^*(c) = c^2 - c + \tfrac{1}{2}\;, \quad \begin{cases} c_1 = \frac{1}{2} - \frac{1}{2\sqrt{3}}\;, & b_1 = \frac{1}{2}\;, \\ c_2 = \frac{1}{2} + \frac{1}{2\sqrt{3}}\;, & b_2 = \frac{1}{2}\;. \end{cases}$$

$$n=3: \quad P_3^*(c) = (c-\tfrac{1}{2})\,(c^2-c+\tfrac{1}{10}) \ , \quad \left\{ \begin{array}{ll} c_1 = \frac{1}{2}-\frac{1}{2}\sqrt{\frac{3}{5}} \ , & b_1 = \frac{5}{18} \ , \\ c_2 = \frac{1}{2} \ , & b_2 = \frac{4}{9} \ , \\ c_3 = \frac{1}{2}+\frac{1}{2}\sqrt{\frac{3}{5}} \ , & b_3 = \frac{5}{18} \ . \end{array} \right.$$

$$n=4: \quad \left\{ \begin{array}{ll} c_1 = \frac{1}{2}-\frac{1}{2}\sqrt{15+2\sqrt{30}}, & b_1 = \frac{1}{4}-\frac{\sqrt{30}}{72} \ , \\ c_2 = \frac{1}{2}-\frac{1}{2}\sqrt{15-2\sqrt{30}}, & b_2 = \frac{1}{4}+\frac{\sqrt{30}}{72} \ , \\ c_3 = \frac{1}{2}+\frac{1}{2}\sqrt{15-2\sqrt{30}}, & b_3 = b_2 \ , \\ c_4 = \frac{1}{2}+\frac{1}{2}\sqrt{15+2\sqrt{30}}, & b_4 = b_1 \ . \end{array} \right.$$

$$n=5: \quad \left\{ \begin{array}{ll} c_1 = \frac{1}{2}-\frac{1}{6}\sqrt{5+2\sqrt{\frac{10}{7}}} \ , & b_1 = \frac{161}{900}-\frac{13\sqrt{70}}{1800} \ , \\ c_2 = \frac{1}{2}-\frac{1}{6}\sqrt{5-2\sqrt{\frac{10}{7}}} \ , & b_2 = \frac{161}{900}+\frac{13\sqrt{70}}{1800} \ , \\ c_3 = \frac{1}{2} \ , & b_3 = \frac{64}{225} \ , \\ c_4 = \frac{1}{2}+\frac{1}{6}\sqrt{5-2\sqrt{\frac{10}{7}}} \ , & b_4 = b_2 \ , \\ c_5 = \frac{1}{2}+\frac{1}{6}\sqrt{5+2\sqrt{\frac{10}{7}}} \ , & b_5 = b_1 \ . \end{array} \right.$$

Für $n>5$ müssen die Legendre-Nullstellen $c_1,\ldots,c_n$ numerisch berechnet werden. Die Gewichte $b_1,\ldots,b_n$ folgen dann als Lösung der ersten n linearen Gleichungen

$$\sum_{j=1}^{n} b_j\, c_j^{k-1} \ = \ \frac{1}{k} \ , \quad k=1,\ldots,n$$

aus Bemerkung 9.3.

Bemerkung 9.20: *Sei $m = a+\frac{L}{2}$ die Intervallmitte. Die Legendre-Polynome über $[a,a+L]$ erfüllen die Rekursionen (z.B. [Abramowitz, Stegun: Handbook of Math. Functions, Dover 1982])*

$$\begin{aligned} P_{n+1}(x) \ &= \ (x-m)\,P_n(x) \ - \ \frac{L^2}{4}\,\frac{n^2}{4n^2-1}\,P_{n-1}(x) \ , \\ P'_{n+1}(x) \ &= \ (x-m)P'_n(x) \ - \ \frac{L^2}{4}\,\frac{n^2}{4n^2-1}\,P'_{n-1}(x) \ + \ P_n(x) \ . \end{aligned}$$

Mit $P_0(x)=1$, $P_0'(x)=0$, $P_1(x)=x-m$, $P_1'(x)=1$ können so P_n und P_n' rekursiv an jeder Stelle numerisch stabil ausgewertet werden. Die Nullstellensuche über das Newton-Verfahren ist damit problemlos. Gute Startwerte für die

Nullstellen $(a <)\ x_1 < \ldots < x_n\ (< a+L)$ von P_n sind durch

$$x_j^{(0)} \;=\; m \;-\; \frac{L}{4}\Big[\;\cos\Big(\frac{j\,\pi}{n+1}\Big) \;+\; \cos\Big(\frac{(2j-1)\,\pi}{2n}\Big)\Big]\,, \qquad j=1,\ldots,n$$

gegeben.

Satz 9.21: (Fehler der Gauss-Quadratur)

Für $2n$-fach stetig differenzierbare Integranden gilt

$$\int_a^{a+L} f(x)\,dx \;-\; G_n[f] \;=\; \frac{\ll P_n, P_n \gg}{(2n)!}\; f^{(2n)}(\xi)$$

mit einem Zwischenwert $\xi \in (a, a+L)$ und

$$\ll P_n, P_n \gg \;=\; L^{2n+1}\int_0^1 (P_n^*(c))^2 dc \;=\; L^{2n+1}\;\frac{(n!)^4}{(2n)!(2n+1)!}\;.$$

Beweis: Wähle zu den Nullstellen $x_1,\ldots,x_n$ von P_n andere n Knoten $y_1,\ldots,y_n$ hinzu. Es sei P_{2n-1} das interpolierende Polynom durch $x_1,\ldots,y_n$, so daß nach Satz 7.12.a)

$$f(x) \;-\; P_{2n-1}(x) \;=\; \frac{f^{(2n)}(\xi(x))}{(2n)!}\;\prod_{j=1}^n (x-x_j)\;\prod_{j=1}^n (x-y_j)$$

mit einem Zwischenwert $\xi(x)$ folgt. Das Polynom P_{2n-1} wird durch G_n exakt integriert und liefert wegen der Interpolation an den Knoten x_j den Wert $G_n[P_{2n-1}] = G_n[f]$. Für den Quadraturfehler folgt die Darstellung

$$\int_a^{a+L} f(x)\;dx - G_n[f] \;=\; \frac{1}{(2n)!}\int_a^{a+L} f^{(2n)}(\xi(x))\;\prod_{j=1}^n (x-x_j)\;\prod_{j=1}^n (x-y_j)\;dx\;.$$

Wegen der stetigen Abhängigkeit des Integranden von $y_1,\ldots,y_n$ kann nun $y_j = x_j$ gewählt werden. Die Funktion $f^{(2n)}(\xi(x))$ ist stetig in x, die restlichen Faktoren bilden wegen $\prod_j (x-x_j) = P_n(x)$ die Funktion $(P_n(x))^2$ ohne Vorzeichenwechsel. Damit kann $f^{(2n)}$ über den Mittelwertsatz der Integralrechnung mit einem geeigneten Zwischenwert ξ aus dem Integral herausgezogen werden, welches dadurch zu $\ll P_n, P_n \gg$ wird. Die angegebenen Werte von $\int_0^1 (P_n^*(c))^2 dc$ sind in der Literatur wohlbekannt.

Q.E.D.

Bemerkung 9.22: *Die Ordnung (Abhängigkeit des Fehlers von der Intervalllänge L) ist bei der Gauß-Quadratur G_n etwa doppelt so groß wie bei der entsprechenden Newton-Cotes-Formel NC_{n-1}, welche dieselbe Anzahl n von Funktionsauswertungen benötigt.*

Für große Integrationsintervalle $[a,b]$ bzw. für stark variierende Integranden (d.h., $|f^{(2n)}|$ wächst schnell mit n), kann man durch Intervallzerlegung hohe Genauigkeiten erreichen:

Bemerkung 9.23: *Teilt man ein Integrationsintervall $[a,b]$ mittels*

$$X_i \;=\; a + i\,L\;, \quad i = 0,\ldots,N-1\;, \quad L \;=\; \frac{b-a}{N}$$

in N Intervalle $[a,X_1],\ldots,[X_i,X_{i+1}],\ldots,[X_{N-1},b]$ der Länge L auf und wendet G_n auf jedes Teilintervall an, so ergeben sich die **zusammengesetzten Gauß-Legendre-Formeln**

$$\int_a^b f(x)\,dx \;=\; L \sum_{i=1}^{N-1}\sum_{j=1}^{n} b_j\, f(X_i + c_j L) \;+\; F_{N,n}[f]$$

mit den Daten $c_1,\ldots,c_n,b_1,\ldots,b_n$ von G_n. Für den Quadraturfehler gilt nach Satz 9.21

$$F_{N,n}[f] \;=\; \frac{L^{2n+1}\,(n!)^4}{((2n)!)^2\,(2n+1)!}\sum_{i=0}^{N-1} f^{(2n)}(\xi_i)\;, \quad \xi_i \in (X_i,X_{i+1})\;,$$

also

$$|F_{N,n}[f]| \;\le\; \frac{(b-a)\;L^{2n}\;(n!)^4}{((2n)!)^2\,(2n+1)!}\max_{\xi\in[a,b]}|f^{(2n)}(\xi)|\;, \quad L \;=\; \frac{b-a}{N}\;.$$

Bei festem n kann der Fehler durch Wahl von hinreichend großem N beliebig klein gemacht werden.

9.3 Adaptive Quadratur

Es ist wünschenswert, die Quadratur so zu implementieren, daß Fehlerabschätzung automatisch vom Algorithmus vorgenommen werden und bei Bedarf hinreichend viele Stützpunkte gewählt werden. So ein Mechanismus sollte aus Effizienzgründen lokal (**"adaptiv"**) arbeiten: in Bereichen, wo der Integrand kritisch ist (starke Variationen), werden kleine Stützstellenabstände benötigt, in "harmlosen" Bereichen braucht man nur wenige Stützstellen. Automatische Fehlerschätzer können i.a. nicht mathematisch exakt arbeiten, sie basieren auf Heuristik, d.h., auf plausiblen Annahmen, die für große Klassen von –in der Praxis auftretenden– Integranden erfüllt sind.

Sei

$$I_S \approx \int_a^b f(x)\,dx$$

eine (grobe) Schätzung der Größenordnung des zu berechnenden Integrals, sei

$$Q(x_0, x_0 + L) \;:=\; L \sum_{i=1}^{n} b_i\, f(x_0 + c_i\, L) \;\approx\; \int_{x_0}^{x_0+L} f(x)\, dx$$

eine beliebige Quadraturformel mit n Knoten $c_1, \ldots, c_n$ und Gewichten $b_1, \ldots, b_n$. Sei `tau` die relative Maschinengenauigkeit der verwendeten Arithmetik. Mit folgender rekursiver Prozedur `adaptiveQuadratur` erhält man einen einfachen, aber wirkungsvollen adaptiven Mechanismus, der auf die nicht-adaptive Quadraturformel Q aufbaut.

Algorithmus 9.24:

```
adaptiveQuadratur:=prozedur(α,β)
lokaleVariablen: Q1,Q2,γ;
  γ:=(α+β)/2; Q1:= Q(α,β); Q2:= Q(α,γ)+Q(γ,β);
  if | Q1−Q2 | < tau∗I_S
    then RETURN(Q2)
    else RETURN( adaptiveQuadratur(α,γ)
                +adaptiveQuadratur(γ,β) )
end;
```

Der Aufruf `adaptiveQuadratur(a,b)` liefert eine numerische Approximation des gesuchten Integrals, für die i.a.

$$\left| \int_a^b f(x)\, dx - \texttt{adaptiveQuadratur(a,b)} \right| \;\approx\; \texttt{tau} * I_S$$

gilt, d.h., der relative Quadraturfehler liegt in der Größenordnung der Maschinengenauigkeit `tau`.

Erklärung: Durch die rekursive Halbierung des Integrationsintervalls $[a, b]$ wird die Quadraturformel Q auf Teilintervalle $[\alpha_j, \beta_j]$ angewendet, es gilt

$$\texttt{adaptiveQuadratur(a,b)} = \sum_j Q(\alpha_j, \beta_j) \quad \left(\approx \int_a^b f(x)\, dx \right) . \qquad (\#)$$

Sei $[\alpha, \beta]$ eines dieser Teilintervalle. In `adaptiveQuadratur`(α, β) wird das Ergebnis der Quadraturformel Q auf $[\alpha, \beta]$ mit der Summe der Quadraturen über die linke und rechte Intervallhälfte $[\alpha, \gamma]$ und $[\gamma, \beta]$ verglichen. Unterscheiden sich diese Werte um weniger als `tau`$*I_S$, so macht es keinen Unterschied, ob $Q(\alpha, \beta)$ in (#) durch $Q(\alpha, \gamma) + Q(\gamma, \beta)$ ersetzt wird: durch Rundung mit der Maschinengenauigkeit `tau` würde sich in beiden Fällen in der gesamten Summe derselbe Wert ergeben (vorausgesetzt, diese Summe liegt in der Tat in der Größenordnung von I_S). Damit braucht dieses Intervall $[\alpha, \beta]$ nicht mehr zur Verkleinerung des lokalen Quadraturfehlers halbiert zu werden.

Bemerkung 9.25: *Sei Q eine Quadratur der Ordnung p, d.h., es gelte eine Abschätzung der Form*

$$\left| \int_\alpha^\beta f(x)\, dx - Q(\alpha,\beta) \right| \leq (\beta-\alpha)^p \; const \tag{$*$}$$

mit einer vom Integrationsintervall unabhängigen Konstanten[2] *const. Dann folgt mit $\gamma = (\alpha+\beta)/2$:*

$$|\mathtt{Q1} - \mathtt{Q2}| = |Q(\alpha,\beta) - Q(\alpha,\gamma) - Q(\gamma,\beta)|$$

$$= \left| Q(\alpha,\beta) - \int_\alpha^\beta f(x)\, dx + \int_\alpha^\gamma f(x)\, dx - Q(\alpha,\gamma) + \int_\gamma^\beta f(x)\, dx - Q(\gamma,\beta) \right|$$

$$\leq \left((\beta-\alpha)^p + \frac{(\beta-\alpha)^p}{2^p} + \frac{(\beta-\alpha)^p}{2^p} \right) \; const = \left(1 + \frac{1}{2^{p-1}} \right) \; (\beta-\alpha)^p \; const\,.$$

Wurde das Intervall $[\alpha, \beta]$ durch N Halbierungsschritte von $[a, b]$ erzeugt, so gilt mit $\beta - \alpha = (b-a)/2^N$:

$$|\mathtt{Q1} - \mathtt{Q2}| \leq \frac{1}{2^{Np}} \left(1 + \frac{1}{2^{p-1}} \right) \; (b-a)^p \; const\,, \tag{$**$}$$

wo $(b-a)^p$ const als Quadraturfehler von Q über dem Gesamtintervall $[a, b]$ zu interpretieren ist. Jeder Halbierungsschritt verkleinert $|\mathtt{Q1}-\mathtt{Q2}|$ damit um einen Faktor $1/2^p$. Quadraturformeln hoher Ordnung (z.B., NC_n mit $p = n+2$ oder G_n mit $p = 2n+1$, n groß) erreichen das Abbruchkriterium $|\mathtt{Q1}-\mathtt{Q2}| < \mathtt{tau}{*}I_S$ damit schneller als Quadraturformeln niedrigerer Ordnung (wobei der Aufwand zur Berechnung der Quadratursummen $Q(\alpha, \beta)$ natürlich mit der Ordnung steigt).

[2] Für die Newton-Cotes-Formeln als auch für die Gauß-Quadratur gilt in der Tat nach den Sätzen 9.8 bzw. 9.21

$$\left| \int_\alpha^\beta f(x)\, dx - \mathrm{NC}_n(\alpha,\beta) \right| \leq (\beta-\alpha)^{n+2} \; d_n \; \max_{x\in[a,b]} |f^{(n+1)}(x)|$$

bzw.

$$\left| \int_\alpha^\beta f(x)\, dx - G_n(\alpha,\beta) \right| \leq \frac{(\beta-\alpha)^{2n+1} \; (n!)^4}{((2n)!)^2 \; (2n+1)!} \; \max_{x\in[a,b]} |f^{(2n)}(x)|$$

für jedes Teilintervall $[\alpha, \beta] \subset [a, b]$.

Bemerkung 9.26: *Im fall unglatter (z.B. unstetiger, aber beschränkter) Integranden f gilt $(*)$ wegen*

$$Q(\alpha,\beta) = (\beta - \alpha) \sum_{i=1}^{n} b_i \, f(\, \alpha + c_i \,(\beta - \alpha)\,)$$

immer noch mit $p = 1$:

$$\left| \int_\alpha^\beta f(x)\,dx - Q(\alpha,\beta) \right| \le \left| \int_\alpha^\beta f(x)\,dx \right| + |Q(\alpha,\beta)|$$

$$\le (\beta - \alpha) \left(1 + \sum_{i=1}^{n} |b_i| \right) \max_{x \in [a,b]} |f(x)| \ .$$

Mit $(**)$, $p = 1$, wird das Abbruchkriterium $|\mathtt{Q1}-\mathtt{Q2}|$ $<\mathtt{tau}*I_S$ *immer noch mit Sicherheit nach endlich vielen Halbierungsschritten erreicht, falls nicht gerade $I_S = 0$ gilt.* **Der Algorithmus `adaptiveQuadratur` terminiert stets!** *Allerdings werden viele Zerlegungsschritte benötigt, sodaß die Quadratur rechenaufwendig wird.*

Bemerkung 9.27: *Durch den rekursiven Aufbau werden zunächst von kleinen Intervallen stammende kleine Quadratursummen zusammengefaßt, bevor diese zu anderen Quadratursummen addiert werden. Dieser "baumartige[3]" Weg der Summation ist numerisch sehr stabil!*

Bemerkung 9.28: *Man hat keine Garantie, daß der gelieferte Quadraturwert wirklich*

$$\left| \int_a^b f(x)\,dx - \mathtt{adaptiveQuadratur(a,b)} \right| \;\approx\; \mathtt{tau} * I_S$$

erfüllt. Diese wäre dann der Fall, wenn das Abbruchkriterium $|\mathtt{Q1}-\mathtt{Q2}| < \mathtt{tau}*I_S$

$$\left| Q(\alpha,\gamma) + Q(\gamma,\beta) - \int_\alpha^\beta f(x)\,dx \right| \;<\; \mathtt{tau} * I_S$$

implizieren würde, da es dann innerhalb der Rundungsgenauigkeit nichts ausmacht, ob der exakte lokale Integralwert $\int_\alpha^\beta$ oder die numerische Approximation $Q(\alpha,\gamma) + Q(\gamma,\beta)$ in die Gesamtsumme $(\#)$ eingeht. Das Abbruchkriterium arbeitet unter der <u>heuristischen Annahme</u>, daß mit $\mathtt{Q1}= Q(\alpha,\beta)$ *und* $\mathtt{Q2}= Q(\alpha,\gamma) + Q(\gamma,\beta)$

$$\left| \mathtt{Q2} - \int_\alpha^\beta f(x)\,dx \right| \ll \left| \mathtt{Q1} - \int_\alpha^\beta f(x)\,dx \right|$$

[3] Man stellt sich die rekursive Zerlegung des Gesamtintervalls als binären Baum vor.

gilt, was in der Tat zu

$$\left|\texttt{Q2} - \int_\alpha^\beta f(x)\,dx\right| \ll \left|\texttt{Q1} - \int_\alpha^\beta f(x)\,dx\right| \approx |\texttt{Q1} - \texttt{Q2}| < \texttt{tau} * I_S$$

führt. Das Abbruchkriterium kann aber auch "zufällig" erfüllt sein. Betrachte dazu z.B. den Integranden

$$f(x) := \begin{cases} 1/\varepsilon & \text{für } x \in [0,\varepsilon] \\ 0 & \text{sonst.} \end{cases}$$

Ist ε sehr klein und benutzen weder $Q(0,1)$ noch $Q(0,1/2)$ Knoten in $[0,\varepsilon]$, so gilt $Q(0,1) = Q(0,1/2) = Q(1/2,1) = 0$, womit das Abbruchkriterium sofort nach dem ersten Halbierungsschritt erfüllt ist und der Wert `adaptiveQuadratur`$(0,1) = 0$ *geliefert wird, während $\int_0^1 f(x)\,dx = 1$ gilt.*

Beispiel 9.29: Die Funktion

$$f(x) = e^{-20\,x^2}\,\cos(20\,x)$$

wird mittels `adaptiveQuadratur` über $[-1,2\,]$ mit der groben Schätzung $I_S = 0.01$ und `tau` $= 10^{-5}$ integriert. Als unterliegende Quadraturregel Q wurde die Milne-Formel

$$Q(\alpha,\alpha+L) = \tfrac{L}{90}\Big(7\,f(\alpha) + 32\,f\Big(\alpha+\tfrac{L}{4}\Big) + 12\,f\Big(\alpha+\tfrac{L}{2}\Big) + 32\,f\Big(\alpha+\tfrac{3\,L}{4}\Big) + 7\,f(\alpha+L)\Big)$$

aus Beispiel 9.6 gewählt. Der gefundene Wert ist

$$\texttt{adaptiveQuadratur}(-1,2) = 0.0026704 \qquad \left(\int_{-1}^2 f(x)\,dx = 0.002670468928...\right).$$

Die in `adaptiveQuadratur` verwendeten Teilintervalle sind in der folgenden Graphik angedeutet: