Einführung in die Numerische Mathematik

Thomas Richter
thomas.richter@iwr.uni-heidelberg.de

Thomas Wick
thomas.wick@iwr.uni-heidelberg.de

Universität Heidelberg

30. Oktober 2012

Inhaltsverzeichnis

Literaturverzeichnis

[1] Jörg Bewersdorff. *Algebra für Einsteiger: Von der Gleichungsauflösung zur Galois-Theorie*. Wiesbaden, 2004.

[2] J.F. Epperson. *An introduction to numerical methods and analysis*. John Wiley & Sons, 2007.

[3] J. Douglas Faires and Richard L. Burdon. *Numerische Methoden*. Spektrum Akademischer Verlag, 1994.

[4] A. Grama, A. Gupta, G. Karypis, and V. Kumar. *Introduction to Parallel Computing*. Addison-Wesley, 2003. 2nd edition.

[5] G. Hämmerlin and K.-H. Hoffmann. *Numerische Mathematik*. Springer Verlag, 1992.

[6] M. Hanke-Bourgeois. *Grundlagen der numerischen Mathematik und des Wissenschaftlichen Rechnens*. Vieweg-Teubner Verlag, 2009.

[7] IEEE. IEEE 754-2008: Standard for floating-point arithmetic. Technical report, IEEE Standards Association, 2008. doi:10.1109/IEEESTD.2008.4610935.

[8] Rainer Kress. *Numerical Analysis*. Springer Verlag, 1998.

[9] R. Rannacher. Einführung in die numerische mathematik. Technical report, Universität Heidelberg, 2006. http://numerik.uni-hd.de/~lehre/notes.

[10] R. Rannacher. Numerik partieller Differentialgleichungen. Technical report, Universität Heidelberg, 2008. http://numerik.uni-hd.de/~lehre/notes.

[11] R. Rannacher. Numerik gewöhnlicher Differentialgleichungen. Technical report, Universität Heidelberg, 2011. http://numerik.uni-hd.de/~lehre/notes.

[12] T. Richter. Numerik partieller Differentialgleichungen. Technical report, Universität Heidelberg, 2011. http://numerik.uni-hd.de/~richter/teaching.shtml.

[13] H. R. Schwarz and N. Köckler. *Numerische Mathematik*. Vieweg-Teubner Verlag, 2011.

[14] Dirk Werner. *Funktionalanalysis*. Springer, 2004.

1 Einleitung

In der numerischen Mathematik werden Verfahren zum konkreten "Lösen" von mathematischen Problemen entworfen und analysiert. Dabei ist die numerische Mathematik eng mit anderen Zweigen der Mathematik verbunden und kann auch nicht von dem Anwendungsgebiet, also z.B. der Chemie, Physik oder Medizin getrennt werden. Der übliche Weg von Problem zur Lösung ist lang:

1. *Mathematische Modellierung* Das zugrundeliegende Problem wird mathematisch erfasst, es werden Gleichungen entwickelt, die das Problem beschreiben. Ergebnis des mathematischen Modells kann ein lineares Gleichungssystem sein, aber z.B. auch eine Differentialgleichung.

2. *Analyse des Modells* Das mathematische Modell muss auf seine Eigenschaften untersucht werden: existiert eine Lösung, ist diese Lösung eindeutig? Hier kommen alle Teilgebiete der Mathematik zum Einsatz, von der Linearen Algebra über Statistik, Gruppentheorie zur Theorie von partiellen Differentialgleichungen.

3. *Numerische Verfahrensentwicklung* Ein numerisches Lösungs- oder Approximationsverfahren wird für das Modell entwickelt. Viele mathematische Modelle (etwa Differentialgleichungen) können nicht exakt gelöst werden und oft kann eine Lösung, auch wenn sie existiert, nicht angegeben werden. Die Lösung einer Differentialgleichung ist eine Funktion $f : [a, b] \to \mathbb{R}$ und zur genauen Beschreibung müsste der Funktionswert an den unendlich-vielen Punkten des Intervalls $[a, b]$ bestimmt werden. Jede durchführbare Verfahrensvorschrift kann allerdings nur aus endlich vielen Schritten bestehen. Das Problem muss zunächst *diskretisiert* werden, also auf ein endlich-dimensionales reduziert werden. Die Numerische Mathematik befasst sich mit der Analyse der numerischen Verfahren, also mit Untersuchung von Konvergenz und Approximationsfehlern.

4. *Implementierung* Das numerische Verfahren muss auf einem Computer implementiert werden. Zur effizienten Implementierung müssen spezielle Algorithmen entwickelt werden. Um moderne Computer-Architekturen nutzen zu können muss das Verfahren z.B. für die Verwendung von Parallelrechnern modifiziert werden.

5. *Auswertung* Die numerischen Ergebnisse (etwa das Simulationsergebnis einer Flugzeugumströmung) müssen ausgewertet werden. Dies beinhaltet eine grafische Darstellung der Ergebnisse sowie in technischen Anwendungen z.B. die Auswertung von Kräften die nur indirekt gegeben sind. Anhand von Plausibilitätsanalysen muss die Qualität der Ergebnisse beurteilt werden. Unter Umständen führt diese Überprüfung zu neuen mathematischen Modellen oder modifizierten numerischen Verfahren.

Die Teilaspekte dürfen nicht getrennt voneinander gesehen werden. Ein effizienter Algorithmus ist nichts wert, wenn das zugrundeliegende Problem überhaupt keine wohldefinierte Lösung hat, ein numerisches Verfahren für ein Anwendungsproblem ist wertlos, wenn der Computer in absehbarer Zeit zu keiner Lösung kommt.

Das Ergebnis einer numerischen Aufgabe ist im Allgemeinen eine Zahl, oder eine endliche Menge von Zahlen. Beispiele für numerische Aufgaben sind das Berechnen der Nullstellen einer Funktion, die Berechnung von Integralen, die Berechnung der Ableitung einer Funktion in einem Punkt, aber auch komplexere Aufgaben wie das Lösen einer Differentialgleichung.

Zum Lösen von numerischen Aufgaben werden wir unterschiedliche Verfahren kennenlernen. Wir grenzen zunächst ein:

Definition 1.1 (Numerisches Verfahren, Algorithmus)**.** *Ein numerisches Verfahren ist eine Vorschrift zum Lösen oder zur Approximation einer mathematischen Aufgabe. Ein numerisches Verfahren heißt* direkt*, falls die Lösung bis auf Rundungsfehler exakt berechnet werden kann. Ein Verfahren heißt* approximativ *falls die Lösung nur angenähert werden kann. Ein Verfahren heißt* iterativ*, falls die Näherung durch mehrfache Ausführung des Verfahrens verbessert werden kann.*

Beispiele für direkte Lösungsverfahren sind die p/q-Formel zur Berechnung von Nullstellen quadratischer Polynome (siehe Kapitel 2) oder der Gauß'sche Eliminationsalgorithmus (Kapitel 4) zum Lösen von linearen Gleichungssysteme. Approximative Verfahren müssen z.B. zum Bestimmen von komplizierten Integralen oder auch zum Berechnen von Nullstellen allgemeiner Funktionen eingesetzt werden. Wir betrachten ein Beispiel eines direkten Verfahrens:

Beispiel 1.2 (Polynomauswertung)**.** *Es sei durch*

$$p(x) = a_0 + a_1 x + \cdots + a_n x^n$$

ein Polynom gegeben. Dabei sei $n \in \mathbb{N}$ sehr groß. Wir werten $p(x)$ in einem Punkt $x_0 \in \mathbb{R}$ mit dem trivialen Verfahren aus:

Algorithmus 1.3 (Polynom-Auswertung)**.**

1. *Für $i = 0, 1, \ldots, n$ berechne $y_i := a_i x_0^i$*
2. *Berechne $p = \sum_{i=0}^{N} y_i$.*

Wir berechnen den Aufwand *zur Polynom-Auswertung: In Schritt 1. des Algorithmus sind zur Berechnung der y_i i Multiplikationen notwendig, insgesamt*

$$0 + 1 + 2 + \cdots + n = \frac{n(n+1)}{2} = \frac{n^2}{2} + \frac{n}{2}.$$

In Schritt 2 weitere n Additionen notwendig. Der Gesamtaufwand des Algorithmus beträgt demnach $n^2/2 + n/2$ Multiplikationen sowie n Additionen. Wir fassen eine Addition und eine Multiplikation zu einer elementaren Operation *zusammen und erhalten zusammen als Aufwand der trivialen Polynomauswertung*

$$A_1(n) = \frac{n^2}{n} + \frac{n}{2}$$

elementare Operationen.

Wir schreiben das Polynom um

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

und leiten hieraus einen weiteren Algorithmus her:

Algorithmus 1.4 (Horner-Schema)**.**

1. *Setze* $p := a_n$
2. *In Schritt* $i = n - 1$ *bis* 0 *berechne* $p := a_i + x \cdot p$

Jeder der n Schritte des Verfahrens benötigt eine Multiplikation sowie eine Addition, also ergibt sich ein Aufwand von

$$A_2(n) = n$$

elementare Operationen. Das Horner-Schema *benötigt für die gleiche Aufgabe wesentlich weniger Operationen, man denke nur an Polynome* $n \gg 1000$*.*

Definition 1.5 (Numerischer Aufwand)**.** *Der* Aufwand *eines numerischen Verfahrens ist die Anzahl der notwendigen* elementaren Operationen*. Eine* elementare Operation *ist eine Addition und eine Multiplikation.*

Meist hängt der Aufwand eines Verfahrens von der Problemgröße ab. Die Problemgröße $N \in \mathbb{N}$ wird von Problem zu Problem definiert, beim Lösen eines linearen Gleichungssystems $Ax = b$ mit einer Matrix $A \in \mathbb{R}^{N \times N}$ ist die Größe der Matrix die Problemgröße. Beim Auswerten eines Polynoms $p(x) = a_n x^n + a_{n-1}x^{n-1} + \cdots + a_0$ in einem Punkt x ist die Problemgröße die Anzahl der Koeffizienten n.

Zur einfachen Schreibweise definieren wir:

Definition 1.6 (Landau-Symbole)**.** (i) *Es sei $g(n)$ eine Funktion mit $g \to \infty$ für $n \to \infty$. Dann ist $f \in O(g)$ genau dann, wenn*

$$\limsup_{n\to\infty} \left|\frac{f(n)}{g(n)}\right| < \infty$$

sowie $f \in o(g)$ genau dann, wenn

$$\lim_{n\to\infty} \left|\frac{f(n)}{g(n)}\right| = 0.$$

(ii) *Sei $g(h)$ eine Funktion mit $g(h) \to 0$ für $h \to 0$. Wir definieren wie oben $f \in O(g)$ sowie $f \in o(g)$.*

Einfach gesprochen: $f \in O(g)$, falls f höchstens so schnell gegen ∞ konvergiert wie g und $f \in o(g)$, falls g schneller als f gegen ∞ geht. Entsprechend gilt für $g \to 0$. dass $f \in O(g)$, falls f mindestens so schnell gegen Null konvergiert wie g und $f \in o(g)$ falls f schneller als g gegen Null konvergiert. Mit Hilfe der Landau-Symbole lässt sich der Aufwand eines Verfahrens einfacher charakterisieren. Im Fall der trivialen Polynomauswertung in Algorithmus 1.3 gilt für den Aufwand $A_1(n)$ in Abhängigkeit der Polynomgröße n:

$$A_1(n) \in O(n^2),$$

und im Fall des Horner-Schema's von Algorithmus 1.4

$$A_2(n) \in O(n).$$

Wir sagen: der Aufwand der trivialen Polynomauswertung wächst quadratisch, der Aufwand des Horner-Schema's linear. Weiter können mit den Landau-Symbolen Konvergenzbegriffe quantifiziert und verglichen werden. In den entsprechenden Kapiteln kommen wir auf diesen Punkt zurück.

Numerische Lösungen sind oft mit Fehlern behaftet. Fehlerquellen sind zahlreich: die Eingabe kann mit einem Messfehler versehen sein, das numerische Verfahren approximiert die Lösung nur (ist also kein direktes Verfahren), die Aufgabe kann nur mit Hilfe eines Computers oder Taschenrechners gelöst werden und ist mit *Rundungsfehlern* versehen. Wir definieren:

Definition 1.7 (Fehler)**.** *Sei $\tilde{x} \in \mathbb{R}$ die Approximation einer Größe $x \in \mathbb{R}$. Mit $|\delta x| = |\tilde{x} - x|$ bezeichnen wir den* absoluten Fehler *und mit $|\delta x|/|x|$ den* relativen Fehler*.*

Üblicherweise ist die Betrachtung des relativen Fehlers von größerer Bedeutung: denn ein absoluter Messfehler von $100m$ ist klein, versucht man den Abstand zwischen Erde und Sonne zu bestimmen, jedoch groß, soll der Abstand zwischen Mensa und Mathematikgebäude gemessen werden.

Messung	0.58s	0.61s	0.62s	0.54s	0.64s	0.598s
Messfehler (rel)	4%	0.5%	2%	10%	6%	1%
Größe	1.65m	1.82m	1.89m	1.43m	2.01m	1.76m
Fehler (abs)	0.15m	0.02m	0.09m	0.37m	0.21m	0.04m
Fehler (rel)	8%	1 %	5%	21%	12%	2%

Tabelle 1.1: Experiment 1: Größenbestimmung durch Fallenlassen eines Balles.

Messung	1.60s	1.48s	1.35s	1.53s	1.45s	1.482s
Messfehler (rel)	5%	2.5%	11%	<1%	5%	2%
Größe	2.53m	1.47m	0.48m	1.90m	1.23m	1.49m
Fehler (abs)	0.73m	0.33m	1.32m	0.10m	0.57m	0.31m
Fehler (rel)	40%	18%	73%	6%	32%	17%

Tabelle 1.2: Experiment 2: Größenbestimmung durch Hochwerfen des Balles.

1.1 Konditionierung von numerischen Problemen

Bei der Analyse von numerischen Verfahren spielen Fehler, insbesondere die Fortpflanzung von Fehlern eine entscheidende Rolle. Wir betrachten ein Beispiel:

Beispiel 1.8 (Größenbestimmung)**.** *Thomas will seine Größe h bestimmen, hat allerdings kein Maßband zur Verfügung. Dafür hat er eine Uhr, einen Ball und im Physikunterricht gut aufgepasst. Zur Lösung der Aufgabe hat er zwei Ideen:*

1. *Verfahren 1: Thomas lässt den Ball aus Kopfhöhe fallen und misst die Zeit t_0, bis der Ball auf dem Boden auskommt. Die Höhe berechnet er aus der Formel für den freien Fall,*
$$y(t) = h - \frac{1}{2}gt^2 \quad \Rightarrow \quad h = \frac{1}{2}gt_0^2,$$
mit der Gravitationskonstante $g = 9.81$.

2. *Verfahren 2: der Ball wird $2m$ über den Kopf geworfen und wir messen die Zeit bis der Ball wieder auf dem Boden angekommen ist. $s = 2m$ werden in $t' = \sqrt{2s/g}$ Sekunden zurückgelegt, hierfür benötigt der Ball eine Startgeschwindigkeit von $v_0 = gt' = \sqrt{2sg} \approx 6.26$ m/s. Es gilt für die Flugbahn:*
$$y(t) = h + v_0 t - \frac{1}{2}gt^2 \quad \Rightarrow \quad h = \frac{1}{2}gt_0^2 - v_0 t_0.$$

Das zweite Verfahren wird gewählt, weil die Zeit t_0, die der Ball in Verfahren 1 zum Fallen benötigt, sehr klein ist. Große Messfehler werden vermutet. Wir führen zunächst Algorithmus 1 durch und messen 5 mal (exakte Lösung $h = 1.80m$ und $t_0 \approx 0.606s$). Die Ergebnisse sind in Tabelle 1.1 zusammengefasst. In der letzten Spalte wurde als Zeit der

Mittelwert aller Messergebnisse gewählt. Dies geschieht in der Hoffnung den Messfehler zu optimieren.

Wir führen nun Algorithmus 2 durch und messen 5 mal (exakte Lösung $h = 1.80m$ und $t_0 \approx 1.519s$). In Tabelle 1.2 sind die Messergebnisse und ermittelten Größen zusammengefasst. In der letzten Spalte wird wieder der Mittelwert aller Messergebnisse betrachtet.

Trotz anfänglicher Zweifel erweist sich Algorithmus 1 als stabiler. Mögliche Fehlerquellen sind der Messfehler bei der Bestimmung der Zeit t_0 sowie bei Algorithmus 2 die Genauigkeit beim Erreichen der Höhe von 2m. In Algorithmus 1 führt der Messfehler zu etwa dem doppelten relativen Fehler in der Höhe. Bei Algorithmus 2 führen selbst kleine Fehler $\leq 1\%$ in den Messungen zu wesentlich größeren Fehlern im Ergebnis. Der immer noch kleine Fehler von 5% bei der ersten Messung führt zu einem Ergebnisfehler von etwa 40%. Auch bei Betrachtung des Mittelwerts über alle Messwerte ist die ermittelte Größe von $1.49m$ keine gute Näherung.

Wir wollen nun untersuchen, welche Auswirkung der Fehler in der Eingabe eines Algorithmus auf das Ergebnis hat. Hierzu sei die Aufgabe wieder allgemein durch die Vorschrift $A : x \mapsto y$ beschrieben. Die Eingabe x sei mit einem Fehler δx behaftet. Die gestörte Eingabe $\tilde{x} = x + \delta x$ liefert ein gestörtes Ergebnis $\tilde{y} = y + \delta y$:

$$\tilde{y} = A(\tilde{x}), \quad \delta y = \tilde{y} - y = A(\tilde{x}) - A(x) = A(x + \delta x) - A(x).$$

Wir teilen durch $y = A(x)$ und erweitern rechts mit δx sowie mit x

$$\frac{\delta y}{y} = \frac{A(x + \delta x) - A(x)}{\delta x} \frac{x}{A(x)} \frac{\delta x}{x}$$

Auf der linken Seite verbleibt der relative Fehler im Ergebnis, rechts steht der relative Eingabefehler multipliziert mit einem Differenzenquotienten für die ersten Ableitung der Aufgabe. Wir approximieren:

$$\left|\frac{\delta y}{y}\right| \approx \left|\frac{\partial A(x)}{\partial x} \frac{x}{A(x)}\right| \cdot \left|\frac{\delta x}{x}\right|,$$

und nennen die Größe

$$\kappa_{A,x} := \frac{\partial A(x)}{\partial x} \frac{x}{A(x)},$$

die *Konditionszahl* der Aufgabe $A(\cdot)$ in Bezug auf die Eingabe x. Die Konditionszahl beschreibt die relative Fehlerverstärkung einer Aufgabe in Bezug auf eine Eingabegröße. Wir definieren:

Definition 1.9 (Konditionszahl)**.** *Es sei $A : \mathbb{R}^n \to \mathbb{R}^m$. Wir nennen*

$$\kappa_{i,j} := \frac{\partial A_i(x)}{\partial x_j} \frac{x_j}{A_i(x)}$$

die relative Konditionszahl *der Aufgabe. Eine Aufgabe heißt* schlecht konditioniert*, falls $|\kappa_{i,j}| \gg 1$, ansonsten* gut konditioniert*. Im Fall $|\kappa_{i,j}| < 1$ spricht man von* Fehlerdämpfung*, ansonsten von* Fehlerverstärkung*.*

Wir setzen das Beispiel zur experimentellen Größenbestimmung fort:

Beispiel 1.10 (Größenbestimmung, Fortsetzung)**.**

1. *Verfahren 1: Zum Durchführen des Verfahrens $h(t_0)$ als Eingabe die gemessene Zeit t_0 erforderlich. Für die Konditionszahl gilt:*

$$\kappa_{h,t_0} := \frac{\partial h(t_0)}{\partial t_0}\frac{t_0}{h(t_0)} = gt_0\,\frac{t_0}{\frac{1}{2}gt_0^2} = 2.$$

   Ein relativer Fehler in der Eingabe $\delta t_0/t_0$ kann demnach einen doppelt so großen relativen Fehler in der Ausgabe verursachen.

2. *Verfahren 2: Wir bestimmen die Konditionszahl in Bezug auf die Geschwindigkeit die Zeit t_0:*

$$\kappa_{h,t_0} = (gt_0 - v_0)\frac{t_0}{\frac{1}{2}gt_0^2 - v_0t_0} = 2\frac{gt_0 - v_0}{gt_0 - 2v_0}$$

   Wir wissen, dass bei exaktem Wurf und ohne Messfehler, der Ball $t_0 \approx 1.519\,s$ unterwegs ist bei einer Startgeschwindigkeit von $v_0 \approx 6.26m/s$. Dies ergibt:

$$\kappa_{h,t_0} \approx \kappa_{h,v_0} \approx 8.$$

   Fehler in der Eingaben $\delta t_0/t_0$ sowie $\delta v_0/v_0$ werden um den Faktor 8 verstärkt. Die durch die Konditionszahlen vorhergesagten Fehlerverstärkungen lassen sich in Tabellen 1.1 und 1.2 zu Beispiel 1.8 gut wiederfinden.

Die Analyse von Fehlern und Fehlerfortpflanzungen spielen eine zentrale Rolle in der numerischen Mathematik. Fehler treten vielfältig auf, auch ohne ungenaue Eingabewerte. Oft sind numerische Algorithmen sehr komplex, setzen sich aus vielen Operationen zusammen. Bei der Approximation mit dem Computer treten unweigerlich *Rundungsfehler* auf. Da der Speicherplatz im Computer, bzw. die Anzahl der Ziffern auf dem Taschenrechner beschränkt sind, treten Rundungsfehler schon bei der bloßen Darstellung einer Zahl auf. Auch wenn es für die numerische Aufgabe

$$x^2 = 2, \quad \Leftrightarrow x = \pm\sqrt{2},$$

mit $x = \pm\sqrt{2}$ eine einfach anzugebene Lösung gibt, so kann diese auf einem Computer nicht exakt dargestellt werden:

$$\sqrt{2} = 1.41421356237309504880\ldots$$

Der naheliegende Grund für einen zwingenden Darstellungsfehler ist der beschränkte Speicher eines Computers. Ein weiterer Grund liegt in der Effizienz. Ein Computer kann nicht mit beliebig langen Zahlen rechnen. Grund ist die beschränkte Datenbandbreite (das sind die 8-Bit, 16-Bit, 32-Bit oder 64-Bit der Prozessoren). Operationen mit Zahlen in längerer Darstellung müssen zusammengesetzt werden, ähnlich dem schriftlichen Multiplizieren oder Addieren aus der Schule.

Computer speichern Zahlen gerundet in Binärdarstellung, also zur Basis 2:

$$\mathrm{rd}(x) = \pm \sum_{i=-n_1}^{n_2} a_i 2^i, \quad a_i \in \{0,1\}.$$

Die Genauigkeit der Darstellung, somit der Rundungsfehler hängt von der Anzahl der Ziffern, also von n_1 und n_2 ab. Die *Fixkommadarstellung* der Zahl im Binärsystem lautet:

$$\mathrm{rd}(x) = [a_{n_2} a_{n_2-1} \dots a_1 a_0 . a_{-1} a_{-2} \dots a_{-n_1}]_2$$

Praktischer ist die *Gleitkommadarstellung* von Zahlen. Hierzu wird die Binärdarstellung normiert und ein gemeinsamer Exponent eingeführt:

$$\mathrm{rd}(x) = \pm \left(\sum_{i=-n_1-n_2}^{0} a_{i+n_2} 2^i \right) 2^{n_2}, \quad a_i \in \{0,1\}.$$

Der führende Term (die a_i) heißt die *Mantisse* und wird von uns mit M bezeichnet, den *Exponenten* bezeichnen wir mit E. Der Exponent kann dabei eine positive oder negative Zahl sein. Zur Vereinfachung wird der Exponenten E als $E = e - b$, mit einer positiven Zahl $e \in \mathbb{N}$ und dem Bias $b \in \mathbb{N}$ geschrieben. Der Biaswert b wird in einer konkreten Zahldarstellung fest gewählt. Der variable Exponentanteil e wird selbst im Binärformat gespeichert. Es bleibt, die Anzahl der Binärstellen für Mantisse und Exponent zu wählen. Hinzu kommt ein Bit für das Vorzeichen $S \in \{+,-\}$. Die Gleitkommadarstellung im Binärsystem lautet:

$$\mathrm{rd}(x) = S \cdot [m_0 . m_{-1} m_{-2} \dots m_{-\#m}]_2 \cdot 2^{[e_{\#e} \dots e_1 e_0]_2 - b}$$

Die Mantisse wird üblicherweise mit $m_0 = 1$ normiert zu $M \in [1,2)$. D.h., die führende Stelle muss nicht explizit gespeichert werden.

Auf modernen Computern hat sich das *IEEE 754*-Format zum Speichern von Zahlen etabliert:

Definition 1.11 (Normalisierte Gleitkommazahlen im IEEE 754-Format)**.** *Das IEEE-754 Format [7] beschreibt die* Gleitkommadarstellung *einer Zahl im* Binärformat*:*

$$x = s \cdot M \cdot 2^{E-b},$$

mit einem Vorzeichen $s \in \{+, -\}$*, einer Mantisse* $M \in [1, 2)$ *sowie einem Exponenten* $E \in \mathbb{N}$ *mit Bias* $b \in \mathbb{N}$*. Die Gleitkommazahl wird in Binärdarstellung gespeichert:*

$$se_{\#e} \dots e_2 e_1 m_{\#m} \dots m_2 m_1$$

mit $\#e$ *Bit für den Exponenten und* $\#m$ *Bit für die Mantisse. Der Bias wird im Allgemeinen als* $2^{\#e-1} - 1$ *gewählt. Die Interpretation der Zahlen hängt vom Exponenten ab:*

Null *Im Fall* $E = 0$ *und* $M = 0$ *ist* $x = \pm 0$*. Die Unterscheidung zwischen* $+0$ *und* -0 *entsteht beim Runden kleinen Zahlen zur Null. Im direkten Vergleich gilt* $+0 = -0$*.*

Unendlich *Im Fall* $E = 2^{\#e} - 1$ *(also alle Bit im Exponenten gleich 1) und* $M = 0$ *ist* $x = \pm\infty$*. Unendlich entsteht etwa beim Teilen durch* 0 *(mit Vorzeichen!) oder falls das Ergebnis zu groß (oder negativ zu klein) ist um darstellbar zu sein.*

NaN *Im Fall* $E = 2^{\#e} - 1$ *und* $M > 0$ *steht der Wert für* Not a Number *und tritt zum Beispiel bei der Operation* $0/0$ *oder* $\infty - \infty$ *auf.*

Normalisierte Zahl *Im Fall* $0 \leq E < 2^{\#e-1}$ *steht die Zahl für*

$$s \cdot 1.m_{\#e} \dots m_2 m_1 \cdot 2^{E-b}.$$

Das Rechnen mit Gleitkommazahlen kann im Computer effizient realisiert werden. Im IEEE-754 Format wird die Gleitkommadarstellung durch *denormalisierte Zahlen* erweitert. Wir erwähnen dies zur Vollständigkeit:

Bemerkung 1.12 (Denormalisierte Zahlen)**.** *Denormalisierte Zahlen dienen zum Schließen der Lücke zwischen Null und der kleinsten positiven darstellbaren Zahl* $1.0 \dots 001 \cdot 2^{-b}$*. Im Fall* $E = 0$ *und* $M > 0$ *wird die Zahl interpretiert als:*

$$s \cdot 0.m_{\#e} \dots m_2 m_1 \cdot 2^{1-b}.$$

Die Genauigkeit bei der Rechnung mit denormalisierten Zahlen ist reduziert.

In Tabelle 1.3 sind verschiedene Gleitkommadarstellungen zusammengefasst, die historisch und aktuell benutzt werden. Derzeit wird fast ausschließlich das IEEE-Format verwendet. Hier sind zwei Darstellungen üblich, *single-precision* (in C++ float) und *double-precision* (in C++ double). Durch die Normierung der Mantisse wird ein Bit, das sogenannte *hidden-bit* gewonnen. Historisch wurden in Rechensystemern jeweils unterschiedliche Zahlendarstellungen gewählt. Während die ersten Computer noch im Prozessor integrierte Recheneinheiten für Gleitkomma-Arithmetik, die sogenannte *floating-point processing unit* (FPU)

	Größe	Vorzeichen	Exponent	Mantisse	Bias
einfache Genauigkeit (single)	32 Bit	1 Bit	8 Bit	23+1 Bit	127
doppelte Genauigkeit (double)	64 Bit	1 Bit	11 Bit	52+1 Bit	1023
Zuse Z1 (1938)	24 Bit	1 Bit	7 Bit	15 Bit	—
IBM 704 (1954)	36 Bit	1 Bit	8 Bit	27 Bit	128
i8087 Coprozessor (1980)	Erste Verwendung von IEEE (single + double)				
Intel 486 (1989)	Erste integrierte FPU in Standard PC				
NVIDIA G80 (2007)	GPU (single)				
NVIDIA Fermi (2010)	GPU (double)				

Tabelle 1.3: IEEE-754 Format in einfacher und doppelter Genauigkeit sowie Gleitkommaformate in aktueller und historischer Hardware.

hatten, verschwand diese zunächst wieder aus den üblichen Computern und war nur in speziellen Rechnern vorhanden. In Form von *Coprozessoren* konnte eine FPU nachgerüstet werden (z.B. der Intel 8087 zum Intel 8086). Der "486er" war der erste Prozessor für Heimcomputer mit integrierter FPU. Heute können Gleitkommaberechnungen effizient auf Grafikkarten ausgelagert werden. Die Prozessoren der Grafikkarten, die *graphics processing unit* (GPU) ist speziell für solche Berechnungen ausgelegt (z.B. schnelle Berechnungen von Lichtbrechungen und Spiegelungen, Abbilden von Mustern auf 3D-Oberflächen). Spezielle Steckkarten (z.B. NVIDIA Tesla), welche gleich mehrere GPU's enthalten werden in Höchstleistungssystemen eingesetzt. Die Genauigkeit der Darstellung ist im Wesentlichen von den in der Mantisse zu Verfügung stehenden Stellen bestimmt. Größte und kleinste darstellbare Zahlen sind durch die Stellen im Exponenten bestimmt. In numerischen Verfahren ist die Verwendung von doppelt-genauer Zahlendarstellung (double) üblich. Die Recheneinheiten moderner Computer nutzen intern eine erhöhte Genauigkeit zum Durchführen von elementaren Operationen. (80 Bit bei modernen Intel-CPU's). Gerundet wird erst nach Berechnung des Ergebnis.

Beispiel 1.13 (Gleitkommadarstellung)**.** *Wir gehen von vierstelliger Mantisse und vier Stellen im Exponent aus mit Bias* $2^{4-1} - 1 = 7$*.*

- *Die Zahl* $x = -96$ *hat zunächst negatives Vorzeichen, also* $S = 1$*. Die Binärdarstellung von 96 ist*

$$96_{10} = 1 \cdot 64 + 1 \cdot 32 + 0 \cdot 16 + 0 \cdot 8 + 0 \cdot 4 + 0 \cdot 2 + 0 \cdot 1 = 1100000_2,$$

  normalisiert

$$64_{10} = 1.1000_2 \cdot 2^{6_{10}} = 1.1_2 \cdot 2^{13_{10}-7_{10}} = 1.1_2 \cdot 2^{1101_2-b}.$$

  Als Gleitkommadarstellung ergibt sich 111011000_2*.*

- *Die Zahl* $x = -384$ *hat wieder negatives Vorzeichen und* $S = 1$*. Die Binärdarstellung von 384 ist:*

$$384_{10} = 1 \cdot 256 + 1 \cdot 128 + 0 \cdot 64 + 0 \cdot 32 + 0 \cdot 16 + 0 \cdot 8 + 0 \cdot 4 + 0 \cdot 2 + 0 \cdot 1 = 110000000_2,$$

also normalisiert

$$384_{10} = 1.1000 \cdot 2^{8_{10}} = 1.1000 \cdot 2^{15_{10}-7_{10}} = 1.1000_2 \cdot 2^{1111_2-b}.$$

Alle Bits im Exponenten sind 1. Der spezielle Wert $E = 1111_2$ *ist jedoch zur Speicherung von NaN (Not a Number) vorgesehen. Die Zahl* 384 *ist zu groß um in diesem Zahlenformat gespeichert zu werden. Stattdessen wird* $-\infty$ *oder Binär* 111110000_2 *gespeichert.*

- *Die Zahl* $x = 1/3 = 0.33333\ldots$ *ist positiv, also* $S = 0$*. Die Binärdarstellung von* $1/3$ *ist*

$$\frac{1}{3} = 0.01010101\ldots_2\,.$$

Normalisiert, mit vierstelliger Mantisse erhalten wir

$$\frac{1}{3} \approx 1.0101_2 \cdot 2^{-2} = 1.0101_2 \cdot 2^{5_2-7_2} = 1.0101_2 \cdot 2^{0101_2-b},$$

also die Binärdarstellung 0010101010_2*. Wir mussten bei der Darstellung runden und erhalten rückwärts*

$$1.0101_2 \cdot 2^{0101_2-7} = 1.3125 \cdot 2^{-2} = 0.328125,$$

mit dem relativen Fehler:

$$\left|\frac{\frac{1}{3} - 1.0101_2 \cdot 2^{0101_2-b}}{\frac{1}{3}}\right| \approx 0.016$$

- *Die Zahl* $x = \sqrt{2} \approx 1.4142135623\ldots$ *ist positiv, also* $S = 0$*. Gerundet gilt:*

$$\sqrt{2} \approx 1.0111_2$$

Diese Zahl liegt bereits normalisiert vor. Mit Bias $b = 7$ *gilt für den Exponenten* $e = 0 = 7 - 7$

$$\sqrt{2} \approx 1.0111_2 \cdot 2^{0111_2-b},$$

also 001110111*. Rückwärts in Dezimaldarstellung erhalten wir*

$$1.0111_2 \cdot 2^{0111_2-b} = 1.4375,$$

mit dem relativen Darstellungsfehler

$$\left|\frac{\sqrt{2} - 1.4375}{\sqrt{2}}\right| \approx 0.016.$$

- *Schließlich betrachten wir die Zahl* $x = -0.003$*. Mit* $S = 1$ *gilt:*

$$0.003_{10} \approx 0.00000000110001_2$$

und normalisiert

$$0.003_{10} \approx 1.10001_2 \cdot 2^{-9} = 1.110001_2 \cdot 2^{-2-7}.$$

Der Exponent $-9 = -2 - b$ ist zu klein und kann in diesem Format (also vier Stellen für den Exponenten) nicht dargestellt werden. Die Zahl kann hingegen denormalisiert dargestellt werden als:

$$0.003_{10} \approx \cdot 0.0110_2 \cdot 2^{0-7}.$$

Binär ergibt dies 100000110_2. Umrechnen in Dezimaldarstellung liefert:

$$0.0110_2 \cdot 2^{0-7} = 0.375 \cdot 2^{-7} \approx 0.0029,$$

mit dem relativen Darstellungsfehler

$$\left|\frac{0.003 - 0.0029}{0.003}\right| \approx 0.033.$$

Aus der begrenzten Anzahl von Ziffern ergibt sich zwangläufig ein Fehler bei der Durchführung von numerischen Algorithmen. Der relative Fehler, der bei der Computer-Darstellung $\tilde{x}$ einer Zahl $x \in \mathbb{R}$ entstehen kann

$$\left|\frac{\mathrm{rd}(x) - x}{x}\right|$$

ist durch die sogenannte *Maschinengenauigkeit* beschränkt:

Definition 1.14 (Maschinengenauigkeit)**.** *Die* Maschinengenauigkeit *eps ist der maximale relative Rundungsfehler der Zahldarstellung und wird bestimmt als:*

$$eps := \inf\{x > 0 : \ \mathrm{rd}(1 + x) > 1\}.$$

Sind im Zahlenformat denormalisierte Zahlen vorgesehen, so verschlechtert sich die Genauigkeit für $x \to 0$.

Da Rundungsfehler zwangsläufig auftreten, gelten grundlegende mathematische Gesetze wie das Assoziativgesetz oder das Distributivgesetz in Computern nicht mehr. Aufgrund von Rundungsfehlern spielt die Reihenfolge, in denen Operationen ausgeführt werden eine wichtige Rolle und verfälscht das Ergebnis. Auch ein einfacher Vergleich von Zahlen ist oft nicht möglich, die Abfrage **if** (3.8/10.0==0.38) kann durch Rundung das falsche Ergebnis liefern und muss durch Abfragen der Art **if** (|3.8/10.0 − 0.38| < eps) ersetzt werden.

Bemerkung 1.15. *Die Maschinengenauigkeit hat nichts mit der kleinsten Zahl zu tun, welche auf einem Computer darstellbar ist. Diese ist wesentlich durch die Anzahl der Stellen im Exponenten bestimmt. Die Maschinengenauigkeit wird durch die Anzahl der Stellen in der Mantisse bestimmt. Bei der üblichen Gleitkommadarstellung mit doppelter Genauigkeit (double in c++) gilt eps $\approx 10^{-16}$, bei einfacher Genauigkeiten, z.B. auf Grafikkarten gilt eps $\approx 10^{-8}$.*

Rundungsfehler treten bei jeder elementaren Operation auf. Daher kommt der Konditionierung der Grundoperationen, aus denen alle Algorithmen aufgebaut sind, eine entscheidende Bedeutung zu:

Beispiel 1.16 (Konditionierung von Grundoperationen, Auslöschung)**.**

1. *Addition, Subtraktion:* $A(x, y) = x + y$*:*

$$\kappa_{A,x} = \left| \frac{x}{x+y} \right| = \left| \frac{1}{1+\frac{y}{x}} \right|$$

Im Fall $x \approx -y$ *kann die Konditionszahl der Addition* ($x \approx y$ *bei der Subtraktion) beliebig groß werden. Ein Beispiel mit vierstelliger Rechnung:*

$$x = 1.021, \quad y = -1.019 \quad \Rightarrow \quad x + y = 0.002.$$

Jetzt sei $\tilde{y} = 1.020$ *gestört. Der relative Fehler in* y *ist sehr klein:* $|1.019-1.020|/|1.019| \leq 0.1\%$*. Wir erhalten das gestörte Ergebnis*

$$x + \tilde{y} = 0.001,$$

und einen Fehler von 100%*. Die enorme Fehlerverstärkung bei der Addition von Zahlen mit etwa gleichem Betrag wird* Auslöschung *genannt. Hier gehen wesentliche Stellen verloren. Auslöschung tritt üblicherweise dann auf, wenn das Ergebnis einer numerischen Operation verglichen mit den Eingabewerten einen sehr kleinen Betrag hat. Kleine relative Fehler in der Eingabe verstärken sich zu großen relativen Fehlern im Ergebnis.*

2. *Multiplikation:* $A(x, y) = x \cdot y$*. Die Multiplikation zweier Zahlen ist stets gut konditioniert:*

$$\kappa_{A,x} = \left| y \frac{x}{xy} \right| = 1.$$

3. *Division:* $A(x, y) = x/y$*. Das selbe gilt für die Division:*

$$\kappa_{A,x} = \left| \frac{1}{y} \frac{x}{\frac{x}{y}} \right| = 1, \quad \kappa_{A,y} = \left| \frac{x}{y^2} \frac{y}{\frac{x}{y}} \right| = 1.$$

4. *Wurzelziehen:* $A(x) = \sqrt{x}$*:*

$$\kappa_{A,x} = \left| \frac{1}{2\sqrt{x}} \frac{x}{\sqrt{x}} \right| = \frac{1}{2}.$$

Ein Fehler in der Eingabe wird im Ergebnis sogar reduziert.

Die meisten numerischen Algorithmen bestehen im Kern aus der wiederholten Ausführung dieser Grundoperationen. Der Aufwand von Algorithmen wird in der Anzahl der notwendigen elementaren Operationen (in Abhängigkeit von der Problemgröße) gemessen. Die

CPU	Jahr	Flops	Preis
Zuse Z3	1941	0.3	Einzelstücke
IBM 704	1955	$5 \cdot 10^3$	>10 000 000 Euro
Intel 8086 + 8087	1980	$50 \cdot 10^3$	2 000 Euro
i486	1991	$1.4 \cdot 10^6$	2 000 Euro
IPhone 4s	2011	$100 \cdot 10^6$	500 Euro
2xPentium III	2001	$800 \cdot 10^6$	2 000 Euro
Core i7	2011	$100 \cdot 10^9$	1 000 Euro
Helics I (Parallelrechner am IWR)	2002	$1 \cdot 10^{12}$	1 000 000 Euro
Nvidia GTX 580 (GPU)	2010	$1 \cdot 10^{12}$	500 Euro
K Computer	2011	$10 \cdot 10^{15}$	???

Tabelle 1.4: Gleitkomma-Geschwindigkeit sowie Anschaffungskosten einiger aktueller und historischer Computer.

Laufzeit eines Algorithmus hängt wesentlich von der Leistungsfähigkeit des Rechners ab. Diese wird in *FLOPS*, also *floating point operations per second* gemessen. In Tabelle 1.4 fassen wir die erreichbaren FLOPS für verschiedene Computer-Systeme zusammen. Die Leistung ist im Laufe der Jahre rapide gestiegen, alle zehn Jahre wird etwa der Faktor 1000 erreicht. Die Leistungssteigerung beruht zum einen auf effizienterer Hardware. Erste Computer hatten Register mit 8 Bit Breite, die in jedem Takt (das ist die MHz Angabe) verarbeitet werden konnten. Auf aktueller Hardware stehen Register mit 64 Bit zur Verfügung. Hinzu kommt eine effizientere Abarbeitung von Befehlen durch sogenanntes *Pipelining*: die übliche Abfolge im Prozessor beinhaltet "Speicher lesen, Daten bearbeiten, Ergebnis speichern". Das Pipelining erlaubt es dem Prozessor schon den Speicher für die nächste Operation auszulesen, während die aktuelle bearbeitet wird. So konnte die Anzahl der notwendigen Prozessortakte pro Rechenoperation erheblich reduziert werden. Die Kombination aus Intel 8086 mit FPU 8087 hat bei einem Takt von 8 Mhz und 50 000 Flops etwa 150 Takte pro Fließkommaoperation benötigt. Der 486er braucht bei 66 Mhz und etwa 1 000 000 Flops nur 50 Takte pro Operation. Der Pentium III liegt bei etwa 5 Takten pro Fließkommaoperation. In den letzten Jahren beruht die Effizienzsteigerung wesentlich auf einem sehr hohen Grad an Parallelisierung. Ein aktueller Core I7 Prozessor kann mehrere Operationen gleichzeitig durchführen. Eine Nvidia 580 GPU erreicht ihre Leistung mit über 500 Rechenkernen. Um diese Leistung effizient nutzen zu können müssen die Algorithmen entsprechend angepasst werden, so dass auch alle 500 Kerne ausgelastet werden. Kann ein Algorithmus diese spezielle Architektur nicht ausnutzen, so fällt die Leistung auf etwa ein Gigaflop zurück, also auf das Niveau des Pentium III aus dem Jahr 2001. Werden die 500 Kerne hingegen optimal ausgenutzt, so erreicht eine Grafikkarte die gleiche Leistung wie der Heidelberger Linux-Cluster helics aus dem Jahr 2002. Moderne Super-Computer wie der K Computer (schnellster Computer der Welt in 2012) vernetzen über 500 000 Kerne.

1.2 Rundungsfehleranalyse und Stabilität von numerischer Algorithmen

Beim Entwurf von numerischen Algorithmen für eine Aufgabe sind oft unterschiedliche Wege möglich, welche sich z.B. in der Reihenfolge der Verfahrensschritte unterscheiden. Unterschiedliche Algorithmen zu ein und derselben Aufgabe können dabei rundungsfehlerbedingt zu unterschiedlichen Ergebnissen führen:

Beispiel 1.17 (Distributivgesetz)**.** *Wir betrachten die Aufgabe* $A(x, y, z) = x \cdot z - y \cdot z = (x - y)z$ *und zur Berechnung zwei Vorschriften:*

$$\begin{aligned} a_1 &:= x \cdot z & a_1 &:= x - y, \\ a_2 &:= y \cdot z & a &:= a_1 \cdot z, \\ a &:= a_1 - a_2. \end{aligned}$$

Es sei $x = 0.519$*,* $y = 0.521$*,* $z = 0.941$*. Bei vierstelliger Arithmetik erhalten wir:*

$$\begin{aligned} a_1 &:= \mathrm{rd}(x \cdot z) = 0.4884 & b_1 &:= \mathrm{rd}(x - y) = -0.002 \\ a_2 &:= \mathrm{rd}(y \cdot z) = 0.4903 & b_2 &:= \mathrm{rd}(a_1 \cdot z) = -0.001882 \\ a_3 &:= \mathrm{rd}(a_1 - a_2) = -0.0019 \end{aligned}$$

Mit $A(x, y, z) = -0.001882$ *ergeben sich die relativen Fehler:*

$$\left|\frac{-0.0001882 - a_3}{0.0001882}\right| \approx 0.01, \quad \left|\frac{-0.0001882 - b_2}{0.0001882}\right| = 0.$$

Das Distributivgesetz gilt auf dem Computer nicht!

Die Stabilität hängt also entscheidend vom Design des Algorithmus ab. Eingabefehler, oder auch Rundungsfehler, die in einzelnen Schritten entstehen, werden in darauffolgenden Schritten des Algorithmus verstärkt. Wir analysieren nun beide Verfahren im Detail und gehen davon aus, dass in jedem der elementaren Schritte (wir haben hier nur Addition und Multiplikation) ein relativer Rundungsfehler ϵ mit $|\epsilon| \leq eps$ entsteht, also

$$\mathrm{rd}(x + y) = (x + y)(1 + \epsilon), \quad \mathrm{rd}(x \cdot y) = (x \cdot y)(1 + \epsilon),$$

Zur Analyse eines gegebenen Algorithmus verfolgen wir die in Rundungsfehler, welche in jedem Schritt entstehen und deren Akkumulation:

Beispiel 1.18 (Stabilität des Distributivgesetzes)**.** *Wir berechnen zunächst die Konditionszahlen der Aufgabe:*

$$\kappa_{A,x} = \left|\frac{1}{1 - \frac{y}{x}}\right|, \quad \kappa_{A,y} = \left|\frac{1}{1 - \frac{x}{y}}\right|, \quad \kappa_{A,z} = 1.$$

Für $x \approx y$ *ist die Aufgabe schlecht konditioniert. Wir starten mit Algorithmus 1 und schätzen in jedem Schritt den Rundungsfehler ab. Zusätzlich betrachten wir Eingabefehler (oder Darstellungsfehler) von* x, y *und* z*. Wir berücksichtigen stets nur Fehlerterme*

erster Ordnung und fassen alle weiteren Terme mit den Landau-Symbolen (für kleine ϵ) zusammen:

$$\begin{aligned}
a_1 =& x(1+\epsilon_x)z(1+\epsilon_z)(1+\epsilon_1) = xz(1+\epsilon_x+\epsilon_z+\epsilon_1+O(eps^2)) \\
&= xz(1+3\epsilon_1+O(eps^2)), \\
a_2 =& y(1+\epsilon_y)z(1+\epsilon_z)(1+\epsilon_2) = yz(1+\epsilon_y+\epsilon_z+\epsilon_2+O(eps^2)) \\
&= yz(1+3\epsilon_2+O(eps^2)), \\
a_3 =& (xz(1+3\epsilon_1) - yz(1+3\epsilon_2) + O(eps^2))(1+\epsilon_3) \\
&= (xz-yz)(1+\epsilon_3) + 3xz\epsilon_1 - 3yz\epsilon_2 + O(eps^2)
\end{aligned}$$

Wir bestimmen den relativen Fehler:

$$\left|\frac{a_3-(xz-yz)}{xz-yz}\right| = \frac{|(xz-yz)\epsilon_3 - 3xz\epsilon_1 + 3yz\epsilon_2|}{|xz-yz|} \leq eps + 3\frac{|x|+|y|}{|x-y|}eps.$$

Die Fehlerverstärkung dieses Algorithmus kann für $x \approx y$ groß werden und entspricht etwa (Faktor 3) der Konditionierung der Aufgabe. Wir nennen den Algorithmus daher stabil.

Wir betrachten nun Algorithmus 2:

$$\begin{aligned}
a_1 =& (x(1+\epsilon_x) - y(1+\epsilon_y))(1+\epsilon_1) \\
&= (x-y)(1+\epsilon_1) + x\epsilon_x - y\epsilon_y + O(eps^2) \\
a_2 =& z(1+\epsilon_z)((x-y)(1+\epsilon_1) + x\epsilon_x - y\epsilon_y + O(eps^2))(1+\epsilon_2) \\
&= z(x-y)(1+\epsilon_1+\epsilon_2+\epsilon_z+O(eps)^2) + zx\epsilon_x - zy\epsilon_y + O(eps^2).
\end{aligned}$$

Für den relativen Fehler gilt in erster Ordnung:

$$\left|\frac{a_2-(xz-yz)}{xz-yz}\right| = \frac{|z(x-y)(\epsilon_1+\epsilon_2+\epsilon_z) + zx\epsilon_x - zy\epsilon_y|}{|xz-yz|} \leq 3eps + \frac{|x|+|y|}{|x-y|}eps$$

Die Fehlerverstärkung kann für $x \approx y$ wieder groß werden. Der Verstärkungsfaktor ist jedoch geringer als bei Algorithmus 1. Insbesondere fällt auf, dass dieser zweite Algorithmus bei Fehlerfreien Eingabedaten keine Fehlerverstärkung aufweist. (Das ist der Fall $\epsilon_x = \epsilon_y = \epsilon_z = 0$).

Beide Algorithmen sind stabil, der zweite hat bessere Stabilitätseigenschaften als der erste.

Wir nennen zwei Algorithmen stabil, obwohl der eine wesentlich bessere Stabilitätseigenschaften hat. Der Stabilitätsbegriff dient daher oft zum relativen Vergleich verschiedener Algorithmen. Wir definieren

Definition 1.19 (Stabilität)**.** *Ein numerischer Algorithmus zum Lösen einer Aufgabe heißt* stabil*, falls die bei der Durchführung akkumulierten Rundungsfehler den durch die Kondition der Aufgabe gegebenen unvermeidlichen Fehler nicht übersteigen.*

Wir halten fest: für ein schlecht konditioniertes Problem existiert kein stabiler Algorithmus mit einer geringeren Fehlerfortpflanzung als durch die Konditionierung bestimmt. Für gut konditionierte Problem können jedoch sehr instabile Verfahren existieren.

Der wesentliche Unterschied zwischen beiden Algorithmen aus dem Beispiel ist die Reihenfolge der Operationen. In Algorithmus 1 ist der letzte Schritt eine Subtraktion, deren schlechte Kondition wir unter dem Begriff *Auslöschung* kennengelernt haben. Bereits akkumulierte Rundungsfehler zu Beginn des Verfahrens werden hier noch einmal wesentlich verstärkt. Bei Algorithmus 2 werden durch die abschließende Multiplikation die Rundungsfehler die zu Beginn auftreten nicht weiter verstärkt. Aus dem analysierten Beispiel leiten wir einer Regel her:

"Bei dem Entwurf von numerischen Algorithmen sollen schlecht konditionierte Operationen zu Beginn durchgeführt werden."

2 Nullstellenbestimmung

2.1 Motivation und Einordnung

Das Lösen von nichtlinearen Gleichungen spielt eine grundlegende Rolle in der Mathematik. Oft werden solche Aufgabenstellungen als Nullstellenproblem formuliert. Die Nullstellenbestimmung von linearen und quadratischen Polynomen ist das einfachste Beispiel und bereits aus der Schule bekannt. Insbesondere kann hierfür noch eine geschlossene Formel (Auflösen nach der Variablen, p-q-Formel, quadratische Ergänzung) angegeben werden. Aber schon bei Gleichungen dritten Grades gibt es keine geschlossene Lösungsformel, die lediglich mit reellen Zahlen arbeitet. Es sollte erwähnt werden, dass jedoch eine geschlossene Formel für kubische Gleichungen unter Zuhilfenahme der komplexen Zahlen existiert (entdeckt von Cardano im Jahr 1545). Eine Zusammenfassung findet der Leser in [1].

Um diese Art von Gleichungen (numerisch) zu lösen, formulieren wir die Aufgabe als Nullstellensuche (vergleiche zur p-q-Formel). Dies bedeutet, dass wir eine Nullstelle der Abbildung (Funktion)

$$f : D(f) \subset \mathbb{R} \to \mathbb{R},$$

suchen, wobei $D(f)$ den Definitionsbereich von f bezeichnet. Jede nichtlineare Gleichung

$$g(x) = y$$

kann durch die Transformation $f(x) := g(x) - y$ in eine Nullstellenaufgabe geschrieben werden:

$$f(x) = 0.$$

Das allgemeine Nullstellenproblem muss meist mit einem *Iterationsverfahren* gelöst werden. Ausgehend von einem Startwert x_0, der zufällig gewählt sein kann (aber Vorsicht!!, die akkurate Wahl eines Startwertes ist oft nicht trivial) besser aber klug bestimmt werden sollte, erhalten wir mit Hilfe der Iterationsvorschrift eine Folge von Punkten $x_k, k = 1, 2, 3, \ldots$, deren Grenzwert gegen die Nullstelle $\hat{x} \in \mathbb{R}$ konvergiert:

$$f(x_0) \to f(x_1) \to f(x_2) \cdots \to f(\hat{x}) = 0.$$

In der Praxis kann der unendliche Grenzprozess nicht durchgeführt werden. Stattdessen wird die Iteration bei einem $N \in \mathbb{N}$ gestoppt und das Folgenglied x_N wird als Approximation der Nullstelle $x_N \approx \hat{x}$ betrachtet. Hieraus ergeben sich unmittelbar wichtige Fragenstellungen, die wir in diesem Kapitel untersuchen wollen und uns in den weiteren Kapiteln wieder begegnen werden. Im Einzelnen sind dies:

- Im Sinne des ersten Kapitels 1: welche Aussagen können wir zur Stabilität und der Konditionierung der Nullstellenbestimmung machen?

- Konstruktion eines *effizienten* numerischen Verfahrens.

- Wie schnell konvergiert $(x_k)_{k\in\mathbb{N}}$ gegen $\hat{x}$? Das bedeutet insbesondere die Quantifizierung Konvergenzordnung und Konvergenzrate (Maße für die Konvergenzgeschwindigkeit). In Analysis I/II haben wir bisher immer von einer Folge $x_k \to \hat{x}$ gesprochen, die gegen den Grenzwert konvergiert. Diese Aussage ist in der Praxis aber nutzlos, da wir in endlicher Rechenzeit eine Lösung erhalten möchten.

- Konvergiert das ausgewählte numerische Verfahren auf beliebig großen Intervallen und mit beliebig gewählten Startwerten x_0, oder nur dann, wenn der Startwert x_0 bereits hinreichend nahe an der gesuchten Nullstelle ist? In anderen Worten: ist das Verfahren *lokal* oder *global* konvergent?

- Abschätzung der zu erwartenden Konvergenz mittels einer aussagekräftigen Fehlerschranke für die Approximation $|x_N - \hat{x}|$. Hier werden wir zwischen der *a priori* und *a posteriori* Fehleranalyse unterscheiden. Bei der a priori Fehleranalyse wird der Fehler $|x_N - \hat{x}|$ *vor* der Rechnung abgeschätzt. Dies gibt dem Numeriker eine grobe Schranke zum Verhalten der Fehlerentwicklung, die aber lediglich asymptotisch richtig ist - und somit für quantitative Aussagen oft nutzlos ist. Dagegen wird bei der a posteriori Fehleranalyse der Fehler $|x_N - \hat{x}|$ während der Rechnung abgeschätzt, alle bereits berechneten Approximation $x_0, x_1, \ldots, x_N$ können in die Abschätzung einfließen. A posteriori Abschätzungen lassen oft quantitative Aussagen zu, die zur Steuerung des numerischen Verfahrens genutzt werden können.

Im Rest dieses Kapitels werden wir die oben genannten Fragen anhand verschiedener Verfahren diskutieren und die Schlüsselbegriffe spezifizieren.

Generell lassen sich die Verfahren zur Nullstellensuche in ableitungsfreie Verfahren (z.B. Intervallschachtelung, Sekantenmethode, sukzessive Approximation) und ableitungsbehaftete Verfahren einteilen (Newton-artige Verfahren). Exemplarisch betrachten wir dazu die Intervallschachtelung und das klassische Newton-Verfahren. Die weiteren Verfahren werden am Ende des Kapitels zusammenfassend erläutert mit Hinweisen auf weiterführende Literatur.

Der Sinn dieses Kapitels ist einerseits die Brücke von bereits bekannten Problemstellungen in der Schule (hier: Nullstellenbestimmung) zu Aussagen der Analysis (z.B. Zwischenwertsatz, Konvergenz) zu schlagen. Gleichzeitig gehören die Algorithmen dieses Kapitels zu der großen Klasse von Fixpunktverfahren. Diese Verfahren werden wir ausführlich in Kapitel 5 besprechen. Insbesondere sind diese Verfahren auf höherdimensionale Funktionen $f \in \mathbb{R}^n$ sowie zum Lösen von Differentialgleichungen geeignet (siehe [11, 10] und entsprechende Hinweise auf die Fachliteratur).

2.2 Stabilität und Kondition

Wir werden zunächst auf die Stabilität und in Bezug auf Kapitel 1 die Konditionierung der Nullstellensuche eingehen. Als Beispiel seien Polynome zweiten Grades gewählt. Hier liegt mit der p-q-Formel ein direktes Lösungsverfahren zur Nullstellenberechnung vor. Trotz dieser auf den ersten Blick einfachen Aufgabe (da sogar exakt lösbar), können numerische Schwierigkeiten hieran demonstriert werden, die bereits mit logischem Vorstellungsvermögen einleuchtend sind.

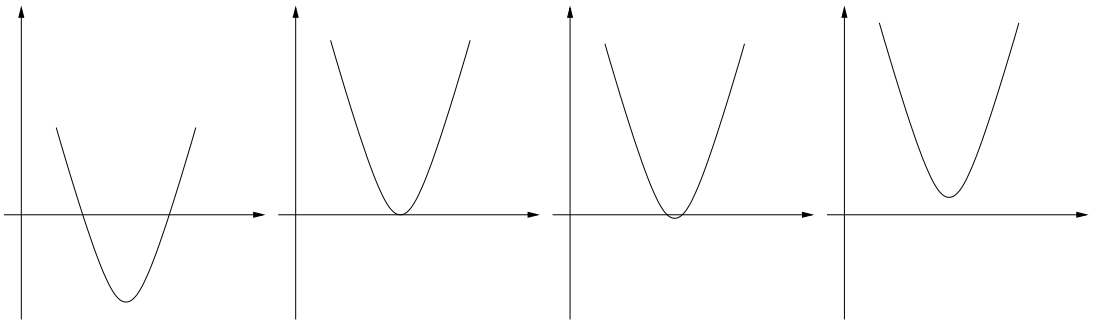

Abbildung 2.1: Die vier möglichen Situationen bei der Nullstellensuche quadratischer Funktionen: stabile Situation, doppelte Nullstelle, zwei Nullstellen nah beieinander und keine reellen Nullstellen.

Wir unterscheiden vier Situationen, wie in Abbildung 2.3 skizziert:

1. Stabile Situation: die beiden Nullstellen liegen hinreichend weit auseinander.

2. Doppelte Nullstelle, d.h. $f(\hat{x}) = f'(\hat{x}) = 0$: die Berechnung einer doppelten Nullstelle ist immer schlecht konditioniert, denn durch Rundungsfehler (Kapitel 1) könnte ein numerisches Verfahren unter Umständen nicht nur die eine einzige Nullstelle finden, sondern entweder zwei (siehe 3.) oder keine (siehe 4.) Nullstellen liefern.

3. Hinreichend nah beieinander liegende Nullstellen sind numerisch schwierig zu berechnen. Insbesondere gilt dies, falls mehr als eine einzige Nullstelle gesucht wird.

4. Hier liegen keine reellen Nullstellen vor. Allerdings können wir immer noch komplexe Nullstellen finden (Fundamentalsatz der Algebra). Die Suche nach komplexen Nullstellen kann ebenfalls schlecht konditioniert sein. Dies werden aber nicht weiter betrachten. Eine Beschreibung numerischer Verfahren zur Berechnung komplexer Nullstellen seien als Übungsaufgabe gestellt.

Beispiel 2.1 (Konditionierung der p-q-Formel)**.** *Wir berechnen die Konditionierung der Nullstellenberechnung mit der p-q-Formel in Abhängigkeit von den Koeffizienten. Dazu sei*

$$f(x) = x^2 - px + q = 0, \quad x \in \mathbb{R}.$$

Für die Wurzeln $x_{1,2}$ gilt

$$x_{1,2} = x_{1,2}(p,q) = \frac{p}{2} \pm \sqrt{\frac{p^2}{4} - q}$$

und wegen $f(x) = (x - x_1)(x - x_2)$ auch $p = x_1 + x_2$ sowie $q = x_1 \cdot x_2$ (Satz von Vieta). Die Konditionszahlen werden berechnet durch:

$$k_{11} = \frac{\partial x_1}{\partial p}\frac{p}{x_1} = \frac{1 + x_2/x_1}{1 - x_2/x_1},$$
$$k_{12} = \frac{\partial x_1}{\partial q}\frac{q}{x_1} = \frac{1}{1 - x_2/x_1}.$$

Entsprechende Resultate erhalten wir für k_{22} und k_{21}. Die Berechnung der Wurzeln von x_1, x_2 ist demnach schlecht konditioniert, falls $x_1/x_2 \sim 1$, d.h. wenn die Wurzeln vergleichsweise nah beieinander liegen. Dann sind nämlich die Konditionszahlen sehr groß.

Beispiel 2.2. *Das vorherige Beispiel wird mit konkreten Zahlenwerten weitergeführt. Es sei $p = 4$ und $q = 3.999$:*

$$f(x) = x^2 - 4x + 3.9999 = 0,$$

mit den Nullstellen $x_{1,2} = 2 \pm 0.01$. Dann gilt für die Konditionzahl

$$k_{12} = \frac{1}{1 - x_1/x_2} = 99.5.$$

Das entspricht einer möglichen Fehlerverstärkung um den Faktor 100. Bei einer Störung von p um 1% zu $\tilde{p} = 4.04$ erhalten wir die gestörten Nullstellen

$$\tilde{x}_1 \approx 2.304, \quad \tilde{x}_2 \approx 1.736,$$

mit relativen Fehlern von etwa 15%.

Bei einer Störung von p um minus 1% zu $\tilde{p} = 3.96$ erhalten wir keine reellen Nullstellen mehr - sondern die komplexen Nullstellen

$$\tilde{x}_1 \approx 1.98 \pm 0.28196i.$$

Die Nullstellenbestimmung kann also sehr schlecht konditioniert sein, falls die beiden Nullstellen nahe beieinander liegen. Wir untersuchen nun die Stabilität des gut konditionierten Falls mit Nullstellen die weit separiert sind. Es sei also ein Polynom

$$f(x) = x^2 - px + q,$$

mit $|q| \ll p^2/4$ gegeben. Die beiden Lösungen lauten:

$$x_{1,2} = \frac{p}{2} \pm \sqrt{\frac{p^2}{4} - q},$$

also $x_1 \approx p$ und $x_2 \approx 0$. Aus den vorherigen Beispielen haben wir gelernt, dass die Aufgabe für $|x_1/x_2| \gg 1$ gut konditioniert ist. Wir berechnen die Nullstellen mit der p/q-Formel:

1 $a_1 = p^2/4$
2 $a_2 = a_1 - q$
3 $a_3 = \sqrt{a_2} \geq 0\,.$

Im Fall $p < 0$ wird zur Vermeidung von Auslöschung zuerst die zweite Wurzel berechnet $\tilde{x}_2 = p/2 - a_3$ berechnet, ansonsten die erste $\tilde{x}_1 = p/2 + a_3$. Die (akzeptable) Fehlerfortpflanzung lautet in diesem Fall ($p > 0$):

$$\left|\frac{\Delta x_1}{x_1}\right| \leq \underbrace{\left|\frac{1}{1+2a_3/p}\right|}_{<1} \left|\frac{\Delta p}{p}\right| + \underbrace{\left|\frac{1}{1+p/2a_3}\right|}_{<1} \left|\frac{\Delta a_3}{a_3}\right|.$$

Die zweite Wurzel $\tilde{x}_1$ kann mit Hilfe von zwei unterschiedlichen Varianten bestimmt werden, die sich allerdings in ihren Stabilitätseigenschaften stark unterscheiden.

- Variante 1: $\tilde{x}_2 = p/2 - a_3$ von der p-q-Formel,
- Variante 2: $\tilde{x}_2 = q/\tilde{x}_1$ von Satz von Vieta.

Falls $|q| \ll p^2/4$, dann ist $a_3 \approx p/2$ (im Fall $p > 0$), und daher tritt bei Variante 1 automatisch Auslöschung auf. Die Rundungsfehler in p und a_3 übertragen sich wie folgt:

$$\left|\frac{\Delta x_2}{x_2}\right| \leq \underbrace{\left|\frac{1}{1-2a_3/p}\right|}_{\gg 1} \left|\frac{\Delta p}{p}\right| + \underbrace{\left|\frac{1}{1-p/2a_3}\right|}_{\gg 1} \left|\frac{\Delta a_3}{a_3}\right|.$$

Für $|q| \ll p^2/4$ ist dieser Algorithmus sehr instabil. Variante 2 hingegen erfordert lediglich eine Division $\tilde{x}_2 = q/\tilde{x}_1$. Diese ist immer gut konditioniert (vergleiche Beispiel 1.16):

$$\left|\frac{\Delta x_2}{x_2}\right| \leq \left|\frac{\Delta q}{q}\right| + \left|\frac{\Delta x_2}{x_2}\right|,$$

womit dieser Algorithmus stabil ist.

Bemerkung 2.3. *Die Quintessence dieser Betrachtungen ist, dass es zu gut konditionierten Problemen verschiedene Algorithmen mit unterschiedlichen Stabilitätseigenschaften geben kann. Auf den Punkt gebracht: im Sinne der Stabilität sollten bei der Lösung quadratischer Gleichungen nicht beide Wurzeln aus der p/q-Formel berechnet werden!*

Beispiel 2.4. *Es sollen die Nullstellen von*

$$x^2 - 4x + 0.01 = 0$$

bestimmt werden. Vierstellige Rechnung ergibt:

1 $a_1 = 4$
2 $a_2 = 3.99$
3 $a_3 = 1.998$

und damit (wegen $p > 0$) für die erste Nullstelle $\tilde{x}_1 = 3.998$ mit einem Fehler $|\Delta x_1/x| \approx 0.00013$. Die andere Wurzel kann mit den beiden Varianten berechnet werden:

- *Exakte Rechnung: $x_2 \approx 0.0002501564$,*
- *Variante 1: $\tilde{x}_2 = 0.002$, mit einem Fehler $|\Delta x_2/x_2| \approx 0.2$*
- *Variante 2: $\tilde{x}_2 = 0.002501$, mit einem Fehler $|\Delta x_2/x_2| \approx 0.00023$.*

Die berechneten Wurzeln der beiden Varianten unterscheiden sich stark. Hinweis: die 4-stellige Rechengenauigkeit dient hier nur zum Zwecke der Demonstration. Man mache sich aber bewusst, dass sich die Instabilität bei größeren Genauigkeiten überträgt.

2.3 Intervallschachtelung

Das einfachste Verfahren zur Nullstellenbestimmung ist die Intervallschachtelung, wurde bereits in Analysis I eingeführt, und basiert auf dem Zwischenwertsatz.

Es sei $f \in C[a, b]$ und $x, y \in [a, b]$ mit $x < y$. Falls $f(x)$ und $f(y)$ unterschiedliche Vorzeichen haben, d.h.,

$$f(x)f(y) < 0,$$

dann besitzt die Funktion f nach dem Zwischenwertsatz mindestens eine Nullstelle in (a, b). Das folgende Verfahren ermittelt eine Approximation dieser Nullstelle.

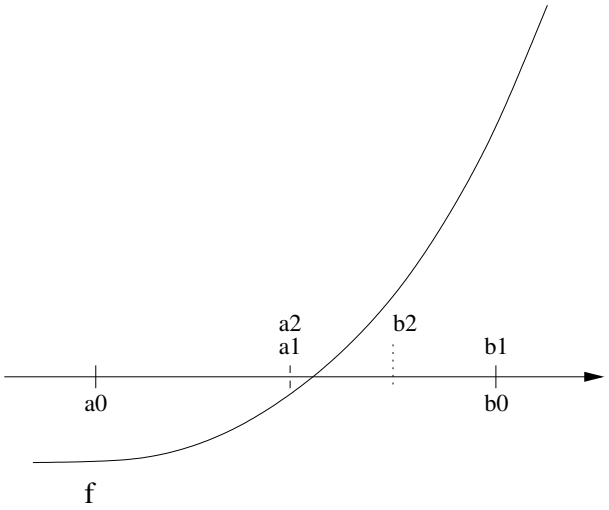

Abbildung 2.2: Geometrische Interpretation der Intervallschachtelung.

Satz 2.5 (Intervallschachtelung)**.** *Es sei $f \in C[a,b]$ mit $f(a)f(b) < 0$. Man definiere eine Folge von Intervallen $[a_n, b_n]$ mit $n = 0, 1, 2, \ldots, N$, durch*

$$[a_0, b_0] = [a, b]$$

und

$$[a_{n+1}, b_{n+1}] = \begin{cases} [a_n, x_n] & \text{falls } f(a_n)f(x_n) < 0, \\ [x_n, b_n] & \text{falls } f(x_n)f(b_n) < 0, \\ [x_n, x_n] & \text{falls } f(x_n) = 0, \end{cases}$$

wobei $x_n = \frac{1}{2}(a_n + b_n)$ der Mittelpunkt des Intervalls $[a_n, b_n]$ ist. Dann gilt, dass $x_n \to \hat{x}\,(n \to \infty)$ für eine Nullstelle $\hat{x}$ von f und

$$|x_n - \hat{x}| \leq \frac{b-a}{2^{n+1}}.$$

Diese Fehlerschranke verhält sich wie eine geometrische Folge mit dem Quotienten $\frac{1}{2}$ (Konvergenzrate) und hat die Konvergenzordnung $p = 1$ (mehr dazu in Abschnitt 2.6).

Beweis: Falls $f(x_n) = 0$ für ein $n \in \mathbb{N}$, dann ist die Nullstelle exakt gefunden. Im Folgenden wird dieser Fall nicht mehr betrachtet. Nach Konstruktion gilt

$$a \leq a_1 \leq \ldots \leq a_n \leq a_{n+1} \leq b_{n+1} \leq b_n \leq \ldots b_1 \leq b,$$

und damit

$$|b_n - a_n| = \frac{|b_{n-1} - a_{n-1}|}{2} = \frac{|b_{n-2} - a_{b-2}|}{2^2} = \ldots = \frac{|b-a|}{2^n}. \tag{2.1}$$

Somit ist (a_n) eine monoton wachsende und nach oben beschränkte Folge. Damit ist diese nach dem Monotoniekriterium konvergent (Analysis I). Analog erhalten wir die Konvergenz für die Folge (b_n), die monoton fallend und nach unten beschränkt ist. Somit gilt

$$\hat{x} := \lim_{n\to\infty} b_n = \lim_{n\to\infty}(a_n + (b_n - a_n)) = \lim_{n\to\infty} a_n + \lim_{n\to\infty}(b_n - a_n) = \lim_{n\to\infty} a_n.$$

Da $a_n \leq x_n \leq b_n$ für alle $n \in \mathbb{N}$ folgt $\lim_{n\to\infty} x_n = \hat{x}$. Aus der Stetigkeit von f folgt weiter

$$f(a_n)f(b_n) \leq 0 \quad \forall n \in \mathbb{N}$$

und dies impliziert

$$f(\hat{x})^2 = \lim_{n\to\infty} f(a_n)f(b_n) \leq 0 \quad \Rightarrow \quad f(\hat{x}) = 0.$$

Es bleibt die Konvergenzrate zu bestimmen. Es gilt $a_n \leq \hat{x} \leq b_n\,(\forall n)$ und

$$x_n - a_n = b_n - x_n = \frac{b-a}{2^{n+1}}.$$

Da $\hat{x} \in [a_n, x_n]$ oder $\hat{x} \in [x_n, b_n]$ folgt die Behauptung:

$$|x_n - \hat{x}| \leq \frac{b-a}{2^{n+1}}.$$

□

Bemerkung 2.6 (A priori Fehlerabschätzung)**.** *Die in Satz 2.5 angegebene Fehlerabschätzung*

$$|x_n - \hat{x}| \leq \frac{b-a}{2^{n+1}}.$$

ist eine a priori Fehlerabschätzung, da diese bereits vor der Rechnung angegeben werden kann.

Abschließend diskutieren wir noch eine a posteriori Fehlerabschätzung, die allerdings die stetige Differenzierbarkeit von f voraussetzt:

Satz 2.7 (A posteriori Fehlerabschätzung)**.** *Es sei $f \in C^1[a,b]$ mit $f(a)f(b) < 0$ und für alle $x \in [a,b]$ gilt*

$$0 < m \leq f'(x) \leq M.$$

Für die Nullstelle $\hat{x}$ von f und die in Satz 2.5 definierte Folge $(x_n)_{n\in\mathbb{N}}$ gilt dann

$$\frac{|f(x_n)|}{M} \leq |x_n - \hat{x}| \leq \frac{|f(x_n)|}{m}.$$

Beweis: Wird dem fleißigen Leser als Übungsaufgabe überlassen. □

Die a posteriori Fehlerschranke kann als Abbruchkriterium zum Beenden der Intervallschachtelung genutzt werden, allerdings nur, falls f stetig differenzierbar in $[a,b]$ ist. Unabhängig davon gibt es im Allgemeinen Fall drei sinnvolle Abbruchkriterien:

- $f(x_n) = 0$: d.h. die Nullstelle ist exakt gefunden.
- $[a_n, b_n] < TOL$
- $n = N_0$: d.h. Angabe einer Maximalanzahl von Iterationsschritten (mit dem Nachteil, dass die Nullstelle noch nicht genügend genau approximiert worden ist.

Beispiel 2.8. *Es sei $b - a = 1$, dann liefert die a priori Abschätzung*

$$|x_9 - \hat{x}| < 10^{-3}, \quad |x_{19} - \hat{x}| < 10^{-6}, \quad |x_{29} - \hat{x}| < 10^{-9}.$$

Das heißt in der 29-ten Iteration kann eine Genauigkeit von $TOL = 10^{-9}$ erwartet werden.

Wir halten fest, dass die Intervallschachtelung ein numerisch sehr stabiles Verfahren ist (d.h. bei Vorzeichenwechsel in $[a,b]$ liefert das Verfahren stets eine Nullstelle - Vorteil!), allerdings sehr langsam gegen die gesuchte Nullstelle konvergiert (Nachteil!), denn 29 Iterationen für eine Toleranz von 10^{-9} ist nicht besonders empfehlenswert. Zweiter Nachteil ist, dass gute Zwischenlösungen (je nachdem was das Abbruchkriterium ist) von dem Verfahren nicht wahrgenommen werden [3], S. 33, siehe praktische Übungen, Blatt 1 sowie Beispiel 2.7. Aufgrund seiner Stabilität wird das Intervallschachtelungsverfahren häufig als Startverfahren für andere Verfahren genutzt (siehe auch Abschnitt 2.7). Allerdings ist die Intervallschachtelung auf reelle Funktionen beschränkt (im Gegensatz zu den weiteren Verfahren, die wir im Anschluss kennen lernen werden) und das Auffinden doppelter Nullstellen ist nicht möglich.

2.4 Das Newton-Verfahren in 1D

Das Newtonverfahren ist eines der wichtigsten numerischen Verfahren zur Nullstellenbestimmung. Es ist nicht nur auf den Fall $\mathbb{R}^d$ mit $d > 1$ erweiterbar, sondern wird auch zur Lösung von Differentialgleichungen häufig genutzt, wobei die Differentialgleichungsaufgabe wiederum in Form eines Nullstellenproblems formuliert wird (siehe weiterführende Literatur [11, 10, 12] und enthaltene Literaturverweise).

Es sei $f \in C^1[a, b]$. Falls die erste Ableitung $f'(x)$ ohne große Probleme berechenbar ist (für 1D Probleme ist das gewöhnlich der Fall, allerdings kann das bei mehrdimensionalen Problemstellungen schwieriger werden - trotzdem oft auch noch möglich), dann stellt das Newton-Verfahren eine sehr effiziente Methodik zur Berechnung einer Nullstelle dar. Das Newton-Verfahren hat viele Ableger - es gibt nicht *das eine einzige* Newton-Verfahren. Das klassische Newton-Verfahren (auch Newton-Raphson-Verfahren genannt) ist durch folgende Überlegung motiviert. Die Funktion f wird im Näherungswert x_k linearisiert und der iterierte Wert x_{k+1} als Abzisse des Schnittpunktes der Tangente (an f) mit der x-Achse definiert.

2.4.1 Das klassische Newton-Verfahren

Verfahren 2.9 (Newton-Verfahren)**.** *Es sei $f \in C^1[a, b]$ und $x_0 \in [a, b]$ ein geeigneter Startwert. Die Tangente an f ist durch*

$$t(x) = f(x_k) + (x - x_k)f'(x_k), \quad k = 0, 1, 2, \ldots,$$

gegeben. Dann ist die Nullstelle x_{k+1} bestimmt durch

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}, \quad k = 0, 1, 2, \ldots. \tag{2.2}$$

Die Iteration wird bei Erfüllung des Abbruchkriteriums 2.11 beendet.

Diese Iteration ist möglich, solange $f'(x_k) \neq 0$.

Die Iteration 2.2 gehört (wie in der Motivation bereits erwähnt) zur Klasse der Fixpunktiterationen mit der Iterationsfunktion

$$F(x) := x - \frac{f(x)}{f'(x)}. \tag{2.3}$$

Für einen Fixpunkt $\hat{x} = F(\hat{x})$ gilt offenbar $f(\hat{x}) = 0$.

Als Hauptresultat des Abschnitts zeigen wir

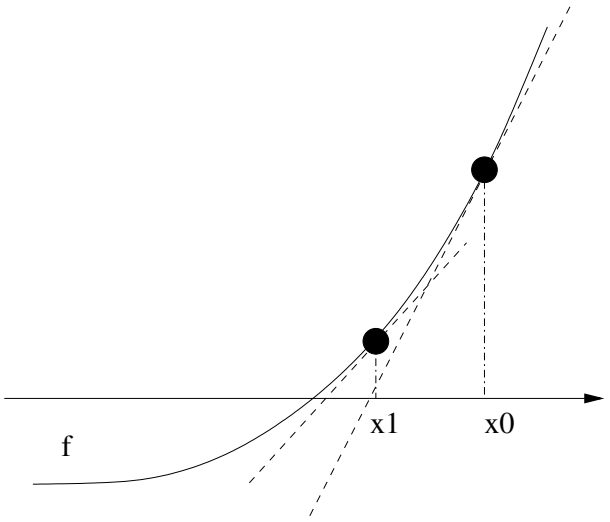

Abbildung 2.3: Geometrische Interpretation des Newton-Verfahrens.

Satz 2.10 (Newton-Verfahren)**.** *Die Funktion $f \in C^2[a,b]$ habe im Innern des Intervalls $[a,b]$ eine Nullstelle $\hat{x}$, und es seien*

$$m := \min_{a \le x \le b} |f'(x)| > 0, \quad M := \max_{a \le x \le b} |f''(x)|.$$

Es sei $\rho > 0$ so gewählt, dass

$$q := \frac{M}{2m}\rho < 1, \quad K_\rho(\hat{x}) := \{x \in \mathbb{R} : |x - \hat{x}| \le \rho\} \subset [a,b].$$

Dann sind für jeden Startpunkt $x_0 \in K_\rho(\hat{x})$ die Newton-Iterierten $x_k \in K_\rho(\hat{x})$ definiert und konvergieren gegen die Nullstelle $\hat{x}$. Desweiteren gelten die a priori Fehlerabschätzung

$$|x_k - \hat{x}| \le \frac{2m}{M} q^{2^k}, \quad k \in \mathbb{N},$$

und die a posteriori Fehlerabschätzung

$$|x_k - \hat{x}| \le \frac{1}{m}|f(x_k)| \le \frac{M}{2m}|x_k - x_{k+1}|^2, \quad k \in \mathbb{N}.$$

Beweis: Wir rekapitulieren zunächst einige Resultate aus der Analysis. Für zwei Punkte $x, y \in [a,b]$, $x \neq y$, gilt aufgrund des Mittelwertsatzes der Differentialrechnung

$$\left|\frac{f(x) - f(y)}{x - y}\right| = |f'(\zeta)| \ge m,$$

mit einem $\zeta \in [x,y]$. Daraus folgt

$$|x - y| \le \frac{1}{m}|f(x) - f(y)|.$$

Die Nullstelle $\hat{x}$ von f ist demnach die einzige in $[a, b]$.

Als zweites Hilfsmittel nutzen wir die Taylor-Formel mit Restglied zweiter Ordnung:

$$f(y) = f(x) + (y-x)f'(x) + (y-x)^2 \int_0^1 f''(x + s(y-x))(1-s)\, ds. \tag{2.4}$$

Wir setzen

$$R(y;x) := (y-x)^2 \int_0^1 f''(x + s(y-x))(1-s)\, ds = f(y) - f(x) - (y-x)f'(x). \tag{2.5}$$

Unter Berücksichtigung der Voraussetzung für f'' erhalten wir

$$|R(y;x)| \leq M|y-x|^2 \int_0^1 (1-s)\, ds = \frac{M}{2}|y-x|^2.$$

Für $x \in K_\rho(\hat{x})$ nutzen wir die Iterationsfunktion (5.2), so dass gilt

$$F(x) - \hat{x} = x - \frac{f(x)}{f'(x)} - \hat{x} = -\frac{1}{f'(x)}\big(f(x) + (\hat{x} - x)f'(x)\big).$$

Mit (2.5) und dort $y = \hat{x}$ folgt:

$$-R(\hat{x};x) = \big(f(x) + (\hat{x} - x)f'(x)\big).$$

Zusammenfassend erhalten wir

$$|F(x) - \hat{x}| \leq \frac{M}{2m}|x - \hat{x}|^2 \leq \frac{M}{2m}\rho^2 < \rho. \tag{2.6}$$

Dies impliziert, dass $F(x) \in K_\rho(\hat{x})$. Die Abbildung F ist eine Selbstabbildung in $K_\rho(\hat{x})$. Insbesondere bedeutet das, dass für $x_0 \in K_\rho(\hat{x})$ auch alle Newton-Iterierten x_k in $K_\rho(\hat{x})$ liegen. Wir setzen

$$\rho_k := \frac{M}{2m}|x_k - \hat{x}|,$$

so dass mit Hilfe von (2.6) folgt:

$$\rho_k \leq \rho_{k-1}^2 \leq \ldots \leq \rho_0^{2^k}, \quad |x_k - \hat{x}| \leq \frac{2m}{M}\rho_0^{2^k}.$$

Für

$$\rho_0 = \frac{M}{2m}|x_0 - \hat{x}| \leq \frac{M}{2m}\rho < 1$$

konvergieren die Iterierten $x_k \to \hat{x}$ für $k \to \infty$. Desweiteren haben wir die a priori Abschätzung gezeigt. Es bleibt die a posteriori Fehlerabschätzung zu beweisen. Hierzu nutzen wir die Taylor-Formel (2.4) und setzen dort $y = x_k, x = x_{k-1}$. Dann

$$f(x_k) = \underbrace{f(x_{k-1}) + (x_k - x_{k-1})f'(x_{k-1})}_{=0} + R(x_k; x_{k-1})$$

und

$$|x_k - \hat{x}| \leq \frac{1}{m}|f(x_k) - f(\hat{x})| \leq \frac{M}{2m}|x_k - x_{k-1}|^2,$$

wobei wir $f(\hat{x}) = 0$ ausgenutzt haben. Damit haben wir alles gezeigt. □

Satz 2.11 (Abbruchkriterium 1 des Newton-Verfahrens)**.** *Die Iteration wird beendet (d.h. wir haben eine akzeptable Lösung $x_{akzeptabel}$ gefunden), falls*

$$\frac{|x_{k+1} - x_k|}{|x_k|} < TOL, \tag{2.7}$$

wobei die Tolerenz TOL beispielsweise durch $TOL = 10^{-12}$ gegeben ist. Das Abbruchkriterium (2.7) misst den relativen Abstand zwischen zwei aufeinanderfolgenden Iterierten x_{k+1} und x_k; in anderen Worten, falls zwei Iterierte nah genug beieinander liegen, dann ist die Nullstelle näherungsweise bestimmt.

Bemerkung 2.12 (Alternatives Abbruchkriterium des Newton-Verfahrens)**.** *Aufgrund der Konstruktion des Newton-Verfahrens gilt:*

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}, \quad k = 0, 1, 2, \ldots,$$

so dass mit Hilfe der Voraussetzung aus Satz 2.10

$$m := \min_{a \le x \le b} |f'(x)| > 0,$$

gilt

$$|x_{k+1} - x_k| = \left| \frac{f(x_k)}{f'(x_k)} \right| \le \left| \frac{f(x_k)}{m} \right|.$$

Das heißt, ein kleines Residuum $f(x_k)$ kann ebenfalls (bei festem m) als (absolutes) Abbruchkriterium genutzt werden:

$$|f(x_k)| < TOL. \tag{2.8}$$

Bemerkung 2.13. *Für eine zweimal stetig differenzierbare Funktion f existiert zu jeder einfachen Nullstelle $\hat{x}$ stets eine (evtl. sehr kleine) Umgebung $K_\rho(\hat{x})$, für die die Voraussetzungen von Satz 2.10 erfüllt sind. Das Problem beim Newton-Verfahren ist also die Bestimmung eines im Einzugsbereich der Nullstelle $\hat{x}$ gelegenen Startpunktes x_0 (lokale Konvergenz). Dieser kann z.B. mit Hilfe der Intervallschachtelung (siehe entsprechende Bemerkung oben) berechnet werden. Dann konvergiert das Newton-Verfahren sehr schnell (quadratische Konvergenz) gegen die Nullstelle $\hat{x}$.*

Bemerkung 2.14. *Das Newton-Verfahren kann auch zur Bestimmung von doppelten Nullstellen genutzt werden, allerdings konvergiert das Verfahren dann nur noch schlicht linear, d.h. $p = 1$, was in Beispiel 2.31 explizit gezeigt wird.*

Bemerkung 2.15 (Einordnung des Newton-Verfahrens)**.** *Aufgrund der (möglichen) schnellen Konvergenz zählt das Newton-Verfahren zu den attraktivsten Verfahren in der numerischen Mathematik zur Ermittelung von Nullstellen. Aufgrund des möglicherweise sehr kleinen Einzugsbereich und damit der Abhängigkeit von der konkreten Aufgabenstellung kann das Newton-Verfahren nicht zu den black-box Lösern gezählt werden.*

Beispiel 2.16. *Falls der Startpunkt x_0 im Einzugsbereich des quadratischen Konvergenz liegt, dann konvergiert das Newton-Verfahren sehr schnell gegen die gesuchte Nullstelle. Es sei z.B. $q \leq \frac{1}{2}$, dann gilt nach nur 10 Iterationsschritten*

$$|x_{10} - \hat{x}| \leq \frac{2m}{M} q^{2^{10}} \sim \frac{2m}{M} 10^{-300}.$$

Diese Toleranz ist jenseits jeglicher Rechengenauigkeit, allerdings vergleiche man dieses Ergebnis mit der a priori berechneten erwarteten Toleranz der Intervallschachtelung!

Beispiel 2.17 (Wurzelberechnung)**.** *Die n-te Wurzel einer Zahl $a > 0$ ist die Nullstelle der Funktion $f(x) = x^n - a$. Das Newton-Verfahren zur Berechnung von $\hat{x} = \sqrt[n]{a} > 0$ lautet (mit $f'(x) = nx^{n-1}$):*

$$x_{k+1} = x_k - \frac{x_k^n - a}{n x_k^{n-1}} = \frac{1}{n}\left\{(n-1)x_k + \frac{a}{x_k^{n-1}}\right\},$$

bzw. als Defektkorrektur-Iteration:

$$\begin{aligned} n x_k^{n-1} \delta x &= x_k^n - a, \\ x_{k+1} &= x_k - \delta x. \end{aligned}$$

Folgende Spezialfälle leiten sich daraus ab:

$$\begin{aligned} x_{k+1} &= \frac{1}{2}\left\{x_k + \frac{a}{x_k}\right\}, \quad \textit{(Quadratwurzel)}, \\ x_{k+1} &= \frac{1}{3}\left\{2x_k + \frac{a}{x_k^2}\right\}, \quad \textit{(Kubikwurzel)}, \\ x_{k+1} &= (2 - a x_k) x_k, n = -1, \quad \textit{(Kehrwert)}. \end{aligned}$$

Da stets $f''(\hat{x}) = n(n-1)\hat{x}^{n-2} \neq 0$, erhalten wir die Konvergenzordnung $p = 2$. Die Fehlerkonstante beträgt

$$C = \left|\frac{f''(\hat{x})}{2f'(\hat{x})}\right| = \frac{1}{2\sqrt{2}}.$$

Aufgrund von Satz 2.10 konvergiert $x_k \to \hat{x} (k \to \infty)$, falls x_0 nahe genug bei $\hat{x}$ gewählt wird. Bei diesem Beispiel ist die Konvergenz allerdings für alle $x_0 > 0$ gesichert, da die Folge $(x_k)_{k \in \mathbb{N}}$ monoton fällt und notwendig gegen $\hat{x} = \sqrt[n]{a}$ konvergiert.

Bemerkung 2.18. *Mit dem in dem vorangegangen Beispiel wird auf vielen Rechnern die Wurzel $\sqrt[n]{a}$ berechnet. Desweiteren ermöglichte die Darstellung zur Berechnung des Kehrwertes, erstmals die Berechnung ohne Ausnutzung der Division sondern lediglich unter Zuhilfenahme von Multiplikationen und Subtraktionen.*

2.4.2 Das Newton-Verfahren als Defekt-Korrektur

In der Praxis wird das Newton-Verfahren oft als Defektkorrektur-Iteration ausgeführt. Wir definieren:

Definition 2.19 (Defekt)**.** *Es sei $\tilde{x} \in \mathbb{R}$ eine Approximation der Lösung von $f(x) = y$. Mit*

$$d(\tilde{x}) = y - f(\tilde{x}),$$

wird der Defekt *bezeichnet. Der Defekt wird auch als das* Residuum *bezeichnet.*

Es sei im Folgenden $y = 0$ (falls nicht transformieren wir die Gleichung entsprechend um), so dass wir wie üblich ein Nullstellenproblem lösen. Für die Approximation x_k ist durch $d_k := 0 - f(x_k)$ der Defekt der k-ten Iterierten gegeben und wir schreiben das Newton-Verfahren in der Form:

Definition 2.20 (Newton-Verfahren als Defektkorrektur)**.**

$$\begin{aligned} f'(x_k)\delta x = d_k, \quad d_k := -f(x_k), \\ x_{k+1} = x_k + \delta x, \quad k = 0, 1, 2, \ldots. \end{aligned}$$

Die Iteration wird bei Erfüllung des Abbruchkriteriums 2.11 beendet.

Diese Art der Berechnung kann bei vielen Fixpunktiterationsverfahren angewendet werden (nicht bei allen!) und wird uns wieder bei der iterativen Lösung von linearen Gleichungssystemen begegnen. Insbesondere hat das Defektkorrektur-Verfahren den Vorteil, dass die Ableitung nicht mehr im Nenner steht - oder anders gesprochen, wir müssen nicht mehr die Inverse der ersten Ableitung explizit angegeben. Für 1D Funktionen ist die explizite Angabe kein Problem. Wie verhält es sich aber bei höherdimensionalen Funktionen? (wir verweisen dazu auf Kapitel 5). Der Defekt $d(\tilde{x}) = y - f(\tilde{x})$ einer Gleichung ist eng mit dem Fehler $\tilde{x} - \hat{x}$ verbunden. Es gilt der folgende allgemeine Satz:

Satz 2.21 (Defekt-Fehler Abschätzung)**.** *Es sei $f \in C^1[a, b]$ eine differenzierbare Funktion und $\hat{x} \in (a, b)$ die Lösung von $f(x) = y$. Für die Approximation $\tilde{x} \in (a, b)$ gilt die Fehlerabschätzung*

$$|\tilde{x} - \hat{x}| \leq \frac{1}{m}|d(\tilde{x})|,$$

mit $m = \min_{[a,b]} |f'(x)|$.

Beweis: Es gilt mit einem $\xi \in [a, b]$:

$$d(\tilde{x}) = y - f(\tilde{x}) = \frac{f(\hat{x}) - f(\tilde{x})}{\hat{x} - \tilde{x}}(\hat{x} - \tilde{x}) = f'(\xi)(\hat{x} - \tilde{x}).$$

Hieraus folgt sofort die Behauptung. □

2.4.3 Das vereinfachte Newton-Verfahren

Häufig ändert sich im Verlauf der Newton-Iteration die Ableitung $f'(x)$ nicht mehr gravierend. In diesem Fall eignet sich das vereinfachte Newton-Verfahren. Die Idee ist recht einfach: berechne die Ableitung $f'(x)$ nur wenige Male (im Idealfall nur für x_0) und belasse diese für alle weiteren Schritte bei diesem Wert:

Definition 2.22 (Vereinfachtes Newton-Verfahren)**.** *Zu einem Startwert $x_0 \in \mathbb{R}$ werden Approximation durch die folgende Vorschrift gebildet:*

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(c)}, \quad k = 0, 1, 2, \ldots.$$

Hierbei ist c ein fest gewählter Punkt, etwa $x = x_0$.

Das vereinfachte Newton-Verfahren konvergiert nur noch linear. Die Konvergenz kann jedoch, falls $f'(c)$ eine gute Approximation für $f'(x_k)$ ist sehr gut sein. Das vereinfachte Newton-Verfahren spielt dann seine Vorteile aus, falls die erste Ableitung aufwendig zu berechnen ist (was bei den Beispielen in diesem Kapitel jedenfalls nicht der Fall ist, aber bei der Nullstellensuche im $\mathbb{R}^n$ (siehe Kapitel 5) oder bei Problemen mit Differentialgleichungen oder Optimierungsproblemen).

Dieses Verfahren kann in der theoretischen Analyse als ein Spezialfall der allgemeinen Fixpunktiteration aufgefasst werden. Wir verweisen hierzu auf Kapitel 5 und [9], S. 162.

2.4.4 Das gedämpfte Newton-Verfahren

Das Hauptproblem bei der Durchführung des Newton-Verfahrens ist die Bestimmung eines geeigneten Startwertes x_0 (siehe obige Bemerkungen), so dass quadratische Konvergenz erwartet werden kann. Deshalb wird in der Praxis häufig mit einer Variante gearbeitet, dem sogenannten *gedämpften* Newton-Verfahren:

Definition 2.23 (Gedämpftes Newton-Verfahren)**.** *Es sei x_0 ein geeigneter Startwert. Die Defektkorrektur-Iteration des gedämpften Newton-Verfahrens lautet:*

$$\begin{aligned} f'(x_k)\delta x &= f(x_k), \\ x_{k+1} &= x_k - \lambda_k \delta x, \quad k = 0, 1, 2, \ldots, \end{aligned}$$

wobei $\lambda_k \in (0, 1]$ den sog. Dämpfungsparameter bezeichnet.

Je kleiner λ_k gewählt wird, desto geringer ist der *Abstand* zwischen x_{k+1} und x_k. Ein kleines λ kann helfen, um Konvergenz zu sichern, obwohl der Einzugbereich der quadratischen Konvergenz noch nicht gesichert ist. So wird vermieden, dass die Iteration weit aus dem Einzugbereich herausspringt. Eine optimale Wahl von λ_k ist (abh. von der Problemstellung) sehr kompliziert werden. Zum Beispiel gibt es ausgeklügelte Algorithmen zur

Bestimmung von λ_k für Newton-Verfahren, die zur Lösung von Optimierungsproblemen genutzt werden (siehe Nocedal/Wright, Numerical Optimization). Oft hilft die *einfache* Strategie

$$\lambda_{k,0} = 1, \quad \lambda_{k,l+1} = \frac{\lambda_{k,l}}{2}, \quad l = 0, 1, 2, \ldots,$$

solange bis für das neue *Residuum* gilt

$$|f(x_{k+1})| < |f(x_k)|.$$

Dieser Algorithmus wird *Line-Search* genannt.

2.5 Weitere Verfahren zur Nullstellensuche

Im diesem Abschnitt fassen wir kurz weitere Verfahren zur Nullstellensuche zusammen. Zum tieferen Verständnis sei auf die angegebene Literatur verwiesen.

Sekantenverfahren

Das Sekantenverfahren gehört zur Klasse der Interpolationsmethoden und hat eine direkte Verbindung zu dem bereits kennengelernten Newtonverfahren. Geometrisch betrachtet wird bei dem Newton-Verfahren eine Tangente an die Funktion gelegt, während bei dem Sekantenverfahren eine Sekante als Approximation zur Ableitung $f'(x)$ genutzt wird. Hierdurch wird die explizite Berechnung von $f'(x)$ vermieden. Dies macht Sinn falls $f'(x)$ schwierig/aufwendig zu berechnen ist. Das Sekantenverfahren ist effizienter als die Nullstellenbestimmung mit Hilfe der Intervallschachtelung und auch einfacher zu realisieren als die sukzessive Approximation (Banachscher Fixpunktsatz); siehe nächster Abschnitt sowie Kapitel 5.

Konkret wird bei dem Sekantenverfahren die Ableitung $f'(x)$ durch einen *Differenzenquotienten* ersetzt, etwa durch

$$f'(x_k) = \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}}.$$

Die Iterationsvorschrift erfordert zwei Startwerte x_0 und x_1, so dass

$$x_{k+1} = x_k - \frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})} f(x_k).$$

Eine wichtige Beobachtung ist, dass die Sekantenmethode unter Umständen zur Auslöschung im Nenner neigt (zur Auslöschung siehe Kapitel 1). Analog zum Newton-Verfahren kann eine Analyse mit entsprechender Fehlerabschätzung durchgeführt werden. Hierzu verweisen wir auf [9], Satz 5.3 auf S. 167. Das Sekanten-Verfahren ist eine Kuriosität, da keine ganzzahlige Konvergenzordnung bewiesen werden kann. Stattdessen gilt asymptotisch:

$$|x_k - \hat{x}| \leq C q^{\gamma_k^k}, \quad \gamma_k \to \frac{1}{2}(1 + \sqrt{5}) \approx 1.6818,$$

also eine Ordnung von etwa 1.6. Die Konstante q hängt wie beim Newton-Verfahren vom Einzugbereich ab.

In Kombination mit der bereits diskutierten Intervallschachtelung in Abschnitt 2.3 erhalten wir ein stabilisiertes Verfahren:

Definition 2.24 (Regula falsi)**.** *Es sei f eine stetige Funktion auf $[a_0, b_0] := [a, b]$ mit $f(a)f(b) < 0$. Eine Approximation von $f(x_{n+1}) = 0$ wird mittels*

$$x_{n+1} := a_n - \frac{f(a_n)(b_n - a_n)}{f(b_n) - f(a_n)}$$

bestimmt. Dann ist das nächste Intervall gegeben durch

$$[a_{n+1}, b_{n+1}] = \begin{cases} [a_n, x_n] & \textit{falls } f(a_n)f(x_n) < 0, \\ [x_n, b_n] & \textit{falls } f(x_n)f(b_n) < 0, \\ [x_n, x_n] & \textit{falls } f(x_n) = 0, \end{cases}$$

In Worten bedeutet dies, dass das Verfahren approximierende Geraden (Sekanten) bildet, aber analog zur Intervallschachtelung die Nullstelle durch kleiner werdende Intervalle einschließt.

Die Konvergenzgeschwindigkeit ist vergleichbar zu den beiden Verfahren durch die die regula falsi definiert ist.

Sukzessive Approximation

Die sukzessive Approximation basiert auf dem Banachschen Fixpunktsatz. Das Verfahren eignet sich insbesondere zur Nullstellenbestimmung höherdimensionaler Funktionen. Daher verweisen wir auf Kapitel 5 für eine ausführliche Diskussion.

Berechnung komplexer Nullstellen

Nach dem Fundamentalsatz der Algebra besitzt jedes komplexe Polynom n-ten Grades, $n + 1$ Nullstellen. Die bisherigen Verfahren eignen sich nicht zur Berechnung von komplexen Nullstellen. Entsprechende Verfahren werden in der Literatur diskutiert und für eine Diskussion im Rahmen dieses Skriptums sei auf die Übungsaufgaben verwiesen.

2.6 Konvergenzbegriffe

In diesem Abschnitt wird die Konvergenz iterativer Verfahren anhand der bereits diskutierten Beispiele in einem allgemeinen Rahmen gefasst. Die theoretischen Resultate werden allgemein für Fixpunktpunktverfahren hergeleitet, so dass das Newton-Verfahren als ein Spezialfall aufgefasst werden kann. Die Aussagen dieses Abschnitts sind auf andere iterative Verfahren die wir später kennenlernen ebenfalls anwendbar (z.B. iterative Lösung von Gleichungssystemen).

Die folgenden Überlegungen sind durch die bereits bekannten Abschätzungen motiviert. Für die Intervallschachtelung gilt:

$$|x_n - \hat{x}| \leq \frac{b-a}{2^{n+1}},$$

mit der Konvergenzrate $\frac{1}{2}$ und der Konvergenzordnung $p = 1$ (lineare Konvergenz).

Das Newtonverfahren besitzt (lokal) in der Umgebung einer Nullstelle das Konvergenzverhalten

$$|x_k - \hat{x}| \leq c\,|x_{k-1} - \hat{x}|^2.$$

Wir sprechen von einem *quadratisch* konvergenten Verfahren oder auch von einem Verfahren *2-ter Ordnung*.

Allgemein gilt

Definition 2.25 (Konvergenzordnung)**.** *Wir nennen ein Iterationsverfahren zur Berechnung einer Nullstelle $\hat{x}$ von Konvergenz mit der Ordnung p mit $p \geq 1$, wenn gilt*

$$|x_k - \hat{x}| \leq c|x_{k-1} - \hat{x}|^p,$$

mit einer festen Konstanten $c > 0$. Im Fall $p = 1$, d.h. linearer Konvergenz heißt die beste Konstante $c \in (0, 1)$ die lineare Konvergenzrate. Gilt bei linearer Konvergenzrate zusätzlich $c_k \to 0$ $(k \to \infty)$, d.h.

$$|x_k - \hat{x}| \leq c_k\,|x_{k-1} - \hat{x}|,$$

so sprechen wir von superlinearer Konvergenz.

Grundsätzlich gilt (zumindest asymptotisch), dass superlinear konvergente Folgen schneller als (schlicht) lineare Folgen konvergieren. Außerdem konvergieren Verfahren mit Ordnung $p+1$ schneller als Verfahren mit der Ordnung p. Außerdem gilt, je kleiner die Konvergenzrate c ist, desto schneller ist die Konvergenz. Allerdings hat die Konvergenzordnung wesentlich größeren Einfluss auf die Konvergenzgeschwindigkeit, als die Konvergenzrate. Letztlich sollte bemerkt werden, dass Verfahren mit $p > 3$ sehr selten Anwendung finden. Sehr oft nutzt der Numeriker Newton-artige Verfahren, so dass $p \approx 2$ gilt.

Bemerkung 2.26. *Die Konvergenzordnung hat einen direkten praktischen Nutzen. Als Näherung erwartet wir bei einem iterativen Verfahren der Ordnung $p > 1$ zur Lösung des Problems $g(x) = 0$, dass sich die Anzahl der korrekten Dezimalstellen bei jeder Iteration ver-p-facht. Bei Verfahren erster Ordnung erwarten wir, dass sich die Anzahl der korrekten Dezimalstellen in etwa um den Summanden $|\log_{10}(g'(\hat{x}))|$ erhöht.*

Im Folgenden betrachten wir Fixpunktprobleme der Form

$$x_{k+1} = g(x_k), \quad k = 0, 1, 2, \ldots.$$

Bei Fixpunktiterationen mit stetig differenzierbarer Abbildung $g(\cdot)$ gilt mit dem Mittelwertsatz der Analysis:

$$\left|\frac{x_{k+1} - \hat{x}}{x_k - \hat{x}}\right| = \left|\frac{g(x_k) - g(\hat{x})}{x_k - \hat{x}}\right| \to |g'(\hat{x})| \quad (k \to \infty). \tag{2.9}$$

Hieraus implizieren wir, dass die lineare Konvergenzrate asymptotisch ($k \to \infty$) gerade gleich $|g'(\hat{x})|$ ist. Dementsprechend liegt im Falle $g'(\hat{x}) = 0$ (mindestens) superlineare Konvergenz vor.

Aus (2.9) können wir folgende Definition ableiten:

Definition 2.27. *Ein Fixpunkt $\hat{x}$ einer stetig differenzierbaren Abbildung $g(\cdot)$ heißt anziehend, falls $|g'(\hat{x})| < 1$ ist. Im Fall $|g'(\hat{x})| > 1$, wird ein Fixpunkt abstoßend genannt.*

Der folgende Satz liefert ein einfaches Kriterium zur Bestimmung der Konvergenzordnung einer differenzierbaren Fixpunktiteration. Er kann auch zur Konstruktion von Verfahren mit hoher Ordnung genutzt werden:

Satz 2.28 (Iterative Verfahren mit Ordnung $p \geq 2$)**.** *Die Funktion $g(\cdot)$ sei in einer Umgebung des Fixpunktes $\hat{x}$ p-mal stetig differenzierbar. Die Fixpunktiteration $x_{k+1} = g(x_k)$ hat genau dann die Ordnung p, wenn*

$$g'(\hat{x}) = \ldots = g^{(p-1)}(\hat{x}) = 0 \quad \textit{und } g^{(p)} \neq 0.$$

Beweis:

- Teil 1 (Rückrichtung)
  Es seien $g'(z) = \ldots = g^{(p-1)}(z) = 0$. Im Punkt z lautet die Taylorformel mit dem Restglied p-ter Ordnung:

  $$x_{k+1} - z = g(x_k) - g(z) = \sum_{j=1}^{p-1} \frac{(x_k - z)^j}{j!} g^{(j)}(z) + \frac{(x_k - z)^p}{p!} g^{(p)}(\xi_k).$$

  Damit erhalten wir die Abschätzung:

  $$|x_{k+1} - z| \leq \frac{1}{p!} \max |g^{(p)}| \; |x_k - z|^p.$$

- Teil 2 (Hinrichtung)
  Es sei nun umgekehrt die Iteration von p-ter Ordnung, sprich

  $$|x_{k+1} - z| \leq c|x_k - z|^p.$$

  Falls es ein minimales $m \leq p-1$ mit $g^{(m)}(z) \neq 0$ gäbe, aber $g^{(j)}(z) = 0$, $j = 1, \ldots, m-1$, so würde jede Iterationsfolge $(x_k)_{k\in\mathbb{N}}$ mit hinreichend kleinem $|x_0 - z| \neq 0$) notwendig gegen z wie

  $$|x_k - z| \leq \left|\frac{1}{m!} g^{(m)}(\xi_k)\right| \, |x_{k-1} - z|^m,$$

  also mit der Ordnung m. Hier finden wir aber einen Widerspruch zu der Annahme, dass die Iteration von der Ordnung p ist:

  $$|g^{(m)}(z)| = \lim_{k\to\infty} |g^{(m)}(\xi_k)| \leq c\, m! \lim_{k\to\infty} |x_k - z|^{p-m} = 0.$$

  Hieraus folgt, dass für $g'(z) = \ldots = g^{(p-1)}(z) = 0$, aber $g^{(p)}(z) \neq 0$, dass die Iteration nicht von höherer Ordnung als p sein kann.

□

Beispiel 2.29 (Darstellung des Newton-Verfahrens als Fixpunktverfahren)**.** *Die allgemeine Fixpunktiteration lautet*

$$x_{k+1} = g(x_k), \quad k = 0, 1, 2, \ldots. \tag{2.10}$$

Das bereits diskutierte Newton-Verfahren lässt sich in dieser Notation wie folgt herleiten. Es sei wie üblich f die Funktion des Nullstellenproblems und h eine hinreichend glatte Funktion, die in einer Umgebung der Nullstelle $\hat{x}$ nicht verschwindet. Es sei

$$g(x) := x + h(x)f(x), \tag{2.11}$$

mit der Fixpunkt-Eigenschaft $g(\hat{x}) = \hat{x}$. die zu (2.10) *zugehörige Iterationsfunktion. Anwendung der Produktregel und Satz 2.28 ergeben in $\hat{x}$:*

$$g'(\hat{x}) := 1 + h'(\hat{x}) \underbrace{f(\hat{x})}_{=0} + h(\hat{x})f'(\hat{x}) \stackrel{!}{=} 0.$$

Hieraus folgt die Bedingung an $h(x)$ in $\hat{x}$:

$$h(\hat{x}) = -\frac{1}{f'(\hat{x})}.$$

Wir wählen $h(x)$ nun für alle x gemäß diese Vorschrift, so dass

$$h(x) := -\frac{1}{f'(x)}.$$

Einsetzen in (2.11) *liefert die Iterationsfunktion*

$$g(x) := x - \frac{f(x)}{f'(x)},$$

und der Iterationsvorschrift

$$x_{k+1} := g(x_k) = x_k - \frac{f(x_k)}{f'(x_k)}, \quad k = 0, 1, 2, \ldots.$$

Beispiel 2.30 (Konvergenzordnung Newton für einfache Nullstellen)**.** *Im umgekehrten Fall zu Beispiel 2.29 sei das Newton-Verfahren nun als bekannt vorausgesetzt. Mit Hilfe des Satzes 2.28 kann explizit die Konvergenzordnung des Newton-Verfahrens ermittelt werden:*

$$g'(\hat{x}) = 1 - \frac{f'(\hat{x})^2 - f(\hat{x})f''(\hat{x})}{f'(\hat{x})^2} = 0,$$

und im Allgemeinen $g''(\hat{x}) \neq 0$. Damit ist das Newton-Verfahren (wie bereits vorher diskutiert) quadratisch konvergent.

Mehrfache Nullstellen

Das Newton-Verfahren kann auch zur Suche von doppelten Nullstellen verwendet werden, allerdings konvergiert es nur noch höchstens superlinear bei mehrfachen Nullstellen.

Es sei $\hat{x}$ eine zweifache Nullstelle der Funktion f, d.h. $f(\hat{x}) = f'(\hat{x}) = 0$ und $f''(\hat{x}) \neq 0$. Dann gilt mit Zwischenwerten $\zeta_k, \eta_k \in [x_k, \hat{x}]$:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} = x_k - \frac{f(x_k) - f(\hat{x})}{f'(x_k) - f'(\hat{x})} \frac{x_k - \hat{x}}{x_k - \hat{x}} = x_k - \frac{f'(\zeta_k)}{f''(\eta_k)}.$$

Beispiel 2.31 (Konvergenzordnung Newton für doppelte Nullstellen)**.** *Für das Newton-Verfahren zur Bestimmung einer einfachen Nullstelle der Funktion f lautet die Iterationsfunktion $g(x) = x - \frac{f(x)}{f'(x)}$. Mit Hilfe des Satzes 2.28 gilt*

$$g'(\hat{x}) = 1 - \frac{f'(\hat{x})^2 - f(\hat{x})f''(\hat{x})}{f'(\hat{x})^2} = 0.$$

Sei nun ebenfalls $f'(\hat{x}) = 0$ (doppelte Nullstelle). Hier können wir nun die Regel von L'Hopital verwenden

$$\lim_{x \to \hat{x}} g'(x) = f''(\hat{x}) \lim_{x \to \hat{x}} \frac{f(x)}{f'(x)^2} = f''(\hat{x}) \lim_{x \to \hat{x}} \frac{f'(x)}{2f'(x)f''(x)} = f''(\hat{x}) \frac{1}{2f''(\hat{x})} = \frac{1}{2}.$$

Damit ist das Newton-Verfahren nach Satz 2.28 zur Bestimmung doppelter Nullstellen nur linear konvergent.

Im Fall von mehrfachen Nullstellen lässt sich die quadratische Konvergenzordnung des Newton-Verfahrens durch einen Trick erreichen. Wir machen zur Iteration den Ansatz:

$$x_{k+1} = x_k - \omega \frac{f(x_k)}{f'(x_k)}$$

mit einem *Relaxationsparameter* ω, ähnlich dem gedämpften Newton-Verfahren. Angenommen, es liegt eine p-fache Nullstelle vor $f(\hat{x}) = f'(\hat{x}) = \ldots = f^{(p-1)}(\hat{x}) = 0$ und $f^{(p)}(\hat{x}) \neq 0$, so wählen wir $\omega = p$. Da $\omega > 1$ sprechen wir von *Überrelaxation*. Das so gewonnene Verfahren konvergiert wieder quadratisch. Der Beweis findet sich in der Literatur, etwa [9], S. 168. Der praktische Nutzen dieser Modifikation ist beschränkt, da man bei der Suche nach einer Nullstelle üblicherweise nicht deren Vielfachheit kennt.

2.7 Vier Verfahren im Vergleich

In diesem Beispiel vergleichen wir drei besprochene Verfahren hinsichtlich Konvergenz-Geschwindigkeit und -Ordnung sowie Anzahl der Iterationen um eine mögliche Fehlertoleranz zu erreichen. Wir vergleichen die Intervallschachtelung, das klassische Newton-Verfahren sowie das vereinfachte Newton-Verfahren. Zuletzt diskutieren wir das kombinierte Verfahren, in dem die Intervallschachtelung als Startiteration für das klassische Newton-Verfahren genutzt wird.

Konfiguration

Gesucht wird eine Nullstelle der Funktion (siehe auch [3], S. 34ff)

$$f(x) = x^3 + 4x^2 - 10$$

im Intervall $[1, 2]$. Die Abbruchtoleranz ist gegeben durch $TOL = 10^{-8}$. Da $f(1) = -5$ und $f(2) = 14$ kann die Intervallschachtelung angewendet werden. Als Startwert für das Newton-Verfahren sei $x_0 = 1$ gewählt. Die maximale Anzahl der Iterationen (sprich falls TOL nicht unterschritten wird) sei $N = 30$. Für das vereinfachte Newton-Verfahren sei die Ableitung durch $f'(x_0)$ approximiert. Die exakte Nullstelle ist auf zehn Dezimalstellen genau $\hat{x} = 1.3652300134$.

Diskussion Intervallschachtelung

Wir erkennen lineare Konvergenz mit einzelnen Ausschlägen, wie sehr gut in Grafik 2.4 zu sehen ist. Dies ist der oben diskutierte zweite Nachteil der Intervallschachtelung: ein u. U. sprunghaftes Konvergenzverhalten. Das Verfahren wird nach 27 Iterationen abgebrochen, da dann TOL unterschritten wird.

Diskussion Newton

Wir erkennen perfekte quadratische Konvergenz (siehe Grafik 2.5) und das Unterschreiten von TOL nach bereits 5 Iterationen. Die quadratische Konvergenz bedeutet in etwa eine Verdopplung in der Anzahl der richtigen Nachkommastellen in jedem Schritt, was gut in Tabelle 2.2 zu sehen ist. (Von Iteration 3 auf 4 ist die Konvergenz sogar noch etwas besser).

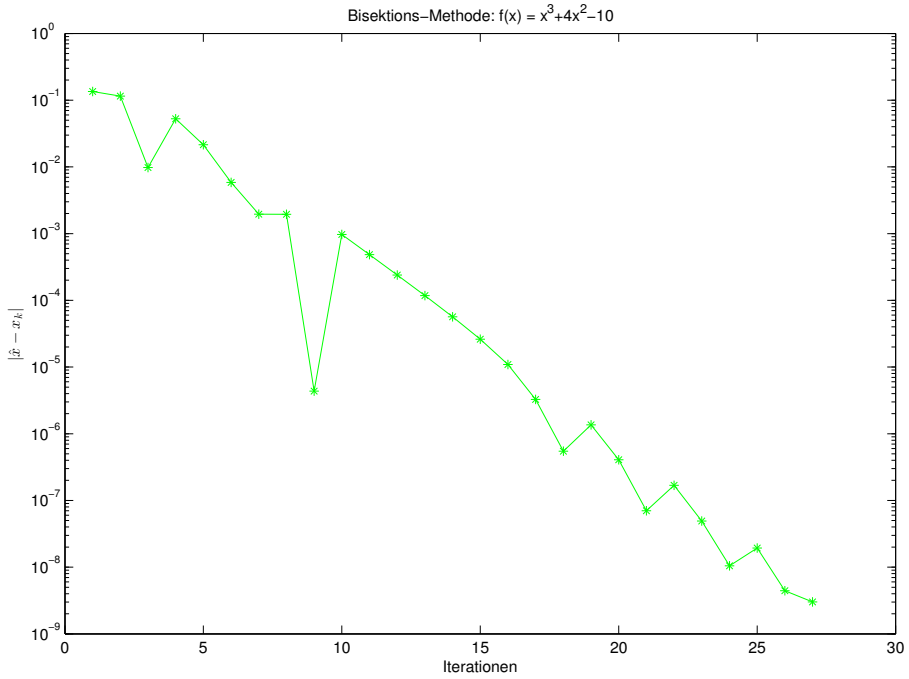

Abbildung 2.4: Fehler vs. Iterationen für das Intervallschachtelungsverfahren.

Diskussion vereinfachtes Newton

Analog zur Intervallschachtelung erkennen wir lineare Konvergenz des Verfahrens (siehe Grafik 2.6). Die TOL wird ebenfalls bei 27 Iterationen unterschritten. Die lineare Konvergenz spiegelt sich ebenfalls im Fehler $|\hat{x} - x_k|$ wieder. Lineare Konvergenz bedeutet, dass sich der Fehler in jeder Iteration etwas halbiert.

Diskussion Intervallschachtelung als Startiteration für Newton

Die Intervallschachtelung wird nach 3 Iterationen abgebrochen, um so einen besseren Startwert als $x_0 = 1$ für das Newton-Verfahren zu bekommen. Nach drei Iterationen ist $\tilde{x}_0 = 1.375$. Mit diesem Startwert wird nun das klassische Newton-Verfahren gestartet. Im Vergleich zu vorher können wir die Anzahl der Newton-Iterationen tatsächlich verringern und erreichen die TOL in 3 Schritten.

Tabelle 2.1: Nullstellenbestimmung mit Intervallschachtelung.

Iter	x_k	$\lvert\hat{x} - x_k\rvert$
1	1.500000000000000	0.134769986600000
2	1.250000000000000	0.115230013400000
3	1.375000000000000	0.009769986600000
4	1.312500000000000	0.052730013400000
5	1.343750000000000	0.021480013400000
6	1.359375000000000	0.005855013400000
7	1.367187500000000	0.001957486600000
8	1.363281250000000	0.001948763400000
9	1.365234375000000	0.000004361600000
10	1.364257812500000	0.000972200900000
11	1.364746093750000	0.000483919650000
12	1.364990234375000	0.000239779025000
13	1.365112304687500	0.000117708712500
14	1.365173339843750	0.000056673556250
15	1.365203857421875	0.000026155978125
16	1.365219116210938	0.000010897189062
17	1.365226745605469	0.000003267794531
18	1.365230560302734	0.000000546902734
19	1.365228652954102	0.000001360445898
20	1.365229606628418	0.000000406771582
21	1.365230083465576	0.000000070065576
22	1.365229845046997	0.000000168353003
23	1.365229964256287	0.000000049143713
24	1.365230023860931	0.000000010460931
25	1.365229994058609	0.000000019341391
26	1.365230008959770	0.000000004440230
27	1.365230016410351	0.000000003010351

Tabelle 2.2: Nullstellenbestimmung mit dem klassischen Newton-Verfahren.

Iter	x_k	$\lvert\hat{x} - x_k\rvert$
1	1.000000000000000	0.365230013400000
2	**1**.454545454545455	0.089315441145455
3	**1.36**8900401069519	0.003670387669519
4	**1.36523**6600202116	0.000006586802116
5	**1.3652300134**35367	0.000000000035367

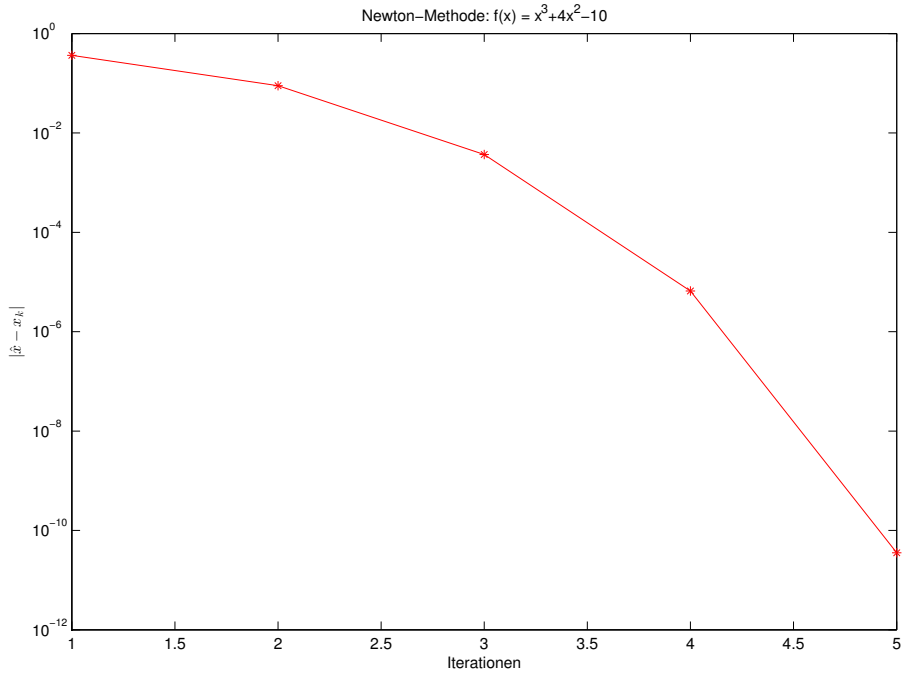

Abbildung 2.5: Fehler vs. Iterationen für das klassische Newton-Verfahren.

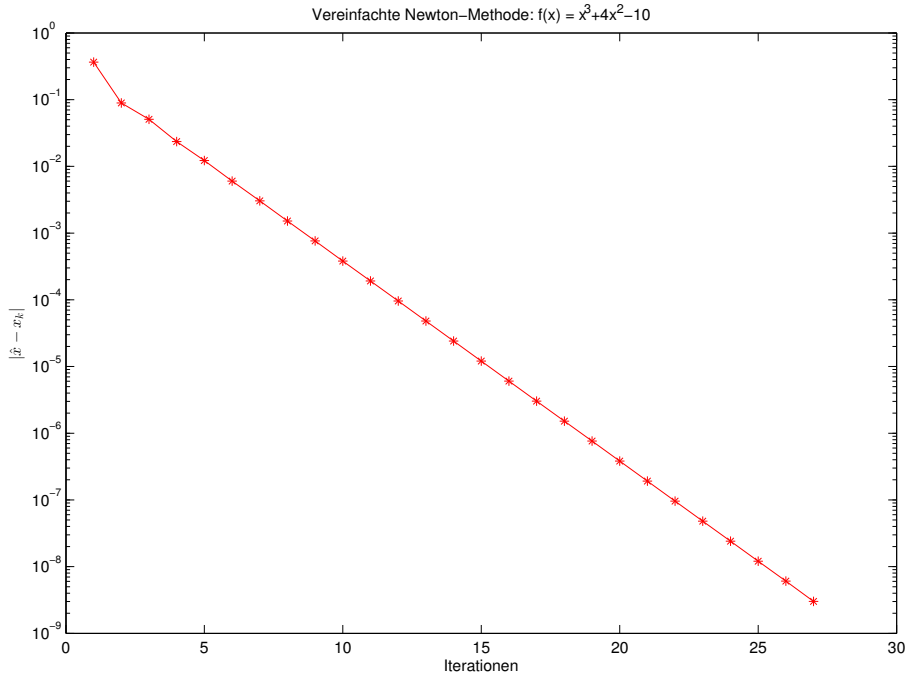

Abbildung 2.6: Fehler vs. Iterationen für das vereinfachte Newton-Verfahren.

Tabelle 2.3: Nullstellenbestimmung mit dem vereinfachten Newton-Verfahren.

Iter	x_k	$\lvert\hat{x} - x_k\rvert$
1	1.000000000000000	0.365230013400000
2	1.454545454545455	0.089315441145455
3	1.314527696195615	0.050702317204385
4	1.388762800510863	0.023532787110863
5	1.353026194519825	0.012203818880175
6	1.371237342903140	0.006007329503140
7	1.362192451689584	0.003037561710416
8	1.366745706220543	0.001515692820543
9	1.364468629666006	0.000761383733993
10	1.365611206121441	0.000381192721441
11	1.365038845782882	0.000191167617118
12	1.365325803195075	0.000095789795075
13	1.365181995089162	0.000048018310838
14	1.365254079370617	0.000024065970617
15	1.365217950694894	0.000012062705106
16	1.365236059360180	0.000006045960180
17	1.365226983049599	0.000003030350401
18	1.365231532280867	0.000001518880868
19	1.365229252128885	0.000000761271115
20	1.365230394983596	0.000000381583596
21	1.365229822164451	0.000000191235549
22	1.365230109271854	0.000000095871854
23	1.365229965368448	0.000000048031552
24	1.365230037495444	0.000000024095444
25	1.365230001344089	0.000000012055910
26	1.365230019463803	0.000000006063803
27	1.365230010381875	0.000000003018125

Tabelle 2.4: Nullstellenbestimmung mit Intervallschachtelung als Startiteration für das Newton-Verfahren.

Iter	x_k	$\lvert\hat{x} - x_k\rvert$
1	**1.3**75000000000000	0.009769986600000
2	**1.3652**76476101218	0.000046462701218
3	**1.36523001**4472403	0.000000001072403

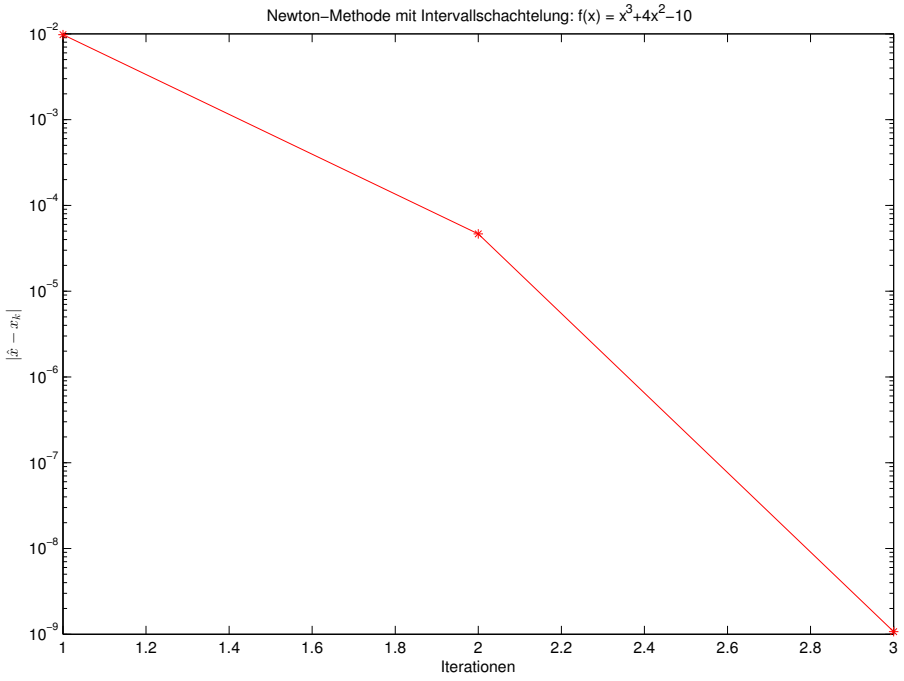

Abbildung 2.7: Fehler vs. Iterationen für das Newton-Verfahren mit dem Intervallschachtelungsverfahren als Startiteration.

3 Interpolation und Approximation

In dem ersten großen Kapitel beschäftigen wir uns mit der Frage, wie eine Reihe von Daten (z.B. aus physikalischen Messungen, experimentelle Beobachtungen, Börse, etc.) durch eine möglichst *einfache* Funktion $p(x)$ (z.B. Polynome) angenähert werden kann. Auf der anderen Seite soll geklärt werden, wie *komplizierte* Funktionen durch einfachere Funktionen aus einer bestimmten Klasse (z.B. Raum der Polynome) approximiert werden können.

Bei der *Interpolation* wird die approximierende Funktion p derart konstruiert, dass diese an den vorliegenden (diskreten) Daten exakt ist:

$$p(x_k) = y_k, \quad k = 0, 1, 2, 3, \ldots, n.$$

Die Stützstellen und Stützwerte (x_k, y_k) sind entweder diskrete Datenwerte (z.B. von Experimenten) oder durch Funktionen bestimmt $y_k = f(x_k)$. Mit Hilfe der Interpolation $p(x)$ können bis dato unbekannte Werte an Zwischenstellen $\xi \in (x_k, x_{k+1})$ oder das Integral bzw. die Ableitung von p bestimmt werden. Darüberhinaus hat die Interpolation eine wesentliche Bedeutung zur Entwicklung weiterer numerischer Verfahren, wie beispielsweise numerische Quadratur, Differentiation oder in weiterführenden Vorlesungen Entwicklung Finiter Elemente [10].

Die *Approximation* ist allgemeiner gefasst. Wieder wird eine einfache Funktion $p(x)$ (also wieder z.B. ein Polynom) gesucht, welche diskrete oder durch Funktionen bestimmte Datenwerte (x_k, y_k) möglichst gut approximiert. Im Gegensatz zur Interpolation wird jedoch nicht $p(x_k) = y_k$ explizit gefordert. Was die Approximation auszeichnet wird von Fall zu Fall entschieden. Möglich ist z.B. bei der Approximation einer Funktion f die beste Approximation bzgl. einer Norm

$$p \in P: \quad \|p - f\| = \min_{q \in P} \|q - f\|,$$

wobei P die Klasse der betrachteten einfachen Funktionen ist (also z.B. alle quadratischen Polynome). Eine Anwendung der Approximation ist das "Fitten" von diskreten Datenwerten, welche durch Experimente gegeben sind. Oft werden viele tausend Messwerte berücksichtigt, die von der zugrundeliegenden (etwa physikalischen) Formel jedoch aufgrund von statistischen Messfehlern nicht exakt im Sinne der Interpolation, sondern nur approximativ erfüllt werden sollen. Eine zweite Anwendung ist wieder die möglichst einfache Darstellung von gegebenen Funktionen.

Beispiel 3.1 (Interpolation: Personen in der Mensa mit einer Stichprobe)**.** *Die Studenten, die Numerik 0 bei Thomas hören, schließen eine Wette gegen die parallel verlaufende*

Vorlesung Einführung in die Wahrscheinlichkeitstheorie und Statistik ab. Es gilt die Anzahl der Mittagessenden Personen diesen Mittwoch um Punkt 12:00 zu schätzen. Allerdings gibt es ein Problem, dass um diese Zeit die Vorlesung der Numerik 0 stattfindet - und die möchte niemand missen.

Daher verabreden sich die Studenten kurz vor der Vorlesung um 11:00 Uhr und zählen alle Personen und dann wieder um 13:00 nach der Vorlesung. Folgende Zahlen haben sie ermittelt:

$$(t_1, y_1) = (11.00, 50), \quad (t_2, y_2) = (13.00, 350),$$

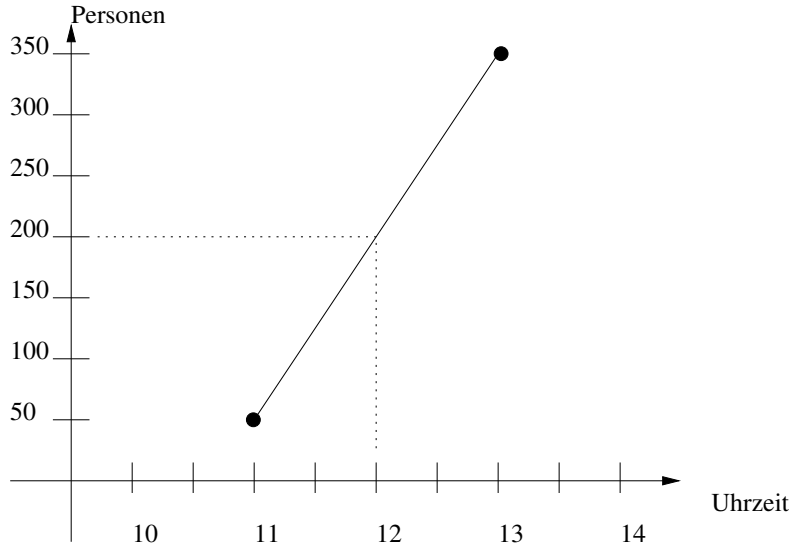

Abbildung 3.1: Lineare Interpolation zwischen zweier gegebener Messwerte.

In Vorlesung habe alle gut aufgepasst: durch Aufstellen der Interpolationsfunktion (hier eine schlichte Gerade) können die Studenten nun den Wert (Anzahl der Personen) durch Ablesen ermitteln. Die Schätzung sagt eine Anzahl von 200 *Personen voraus. Dieser Wert ist natürlich nur eine Schätzung da der Anstieg sich wahrscheinlich nicht linear verhält.*

Beispiel 3.2 (Approximation: Personen in der Mensa bei mehreren Stichproben)**.** *Die Studenten der Numerik 0 haben auch bei der Approximation gut aufgepasst und machen an nun jede Woche an verschiedenen Tagen Stichproben von der Anzahl der Personen in der Mensa. Die ermittelte Datenmenge kann nicht mehr interpoliert werden, allerdings stellen sie nun eine approximierende Funktion auf, die die wahrscheinliche Personenzahl am letzten Mittwoch in der Vorlesungszeit voraussagen soll.*

Ablesen der Regressionsgeraden ergibt eine Schätzung von ca. 250 *Personen in der Mensa.*

Grundsätzlich hat die Darstellung als einfache Funktion den großen Vorteil, dass Elementaroperationen (wie bei der Interpolation bereits angemerkt), wie Ableitungsbildung und Integration, viel einfacher ausgeführt werden können. Die Wahl der *einfachen* Funktion ist

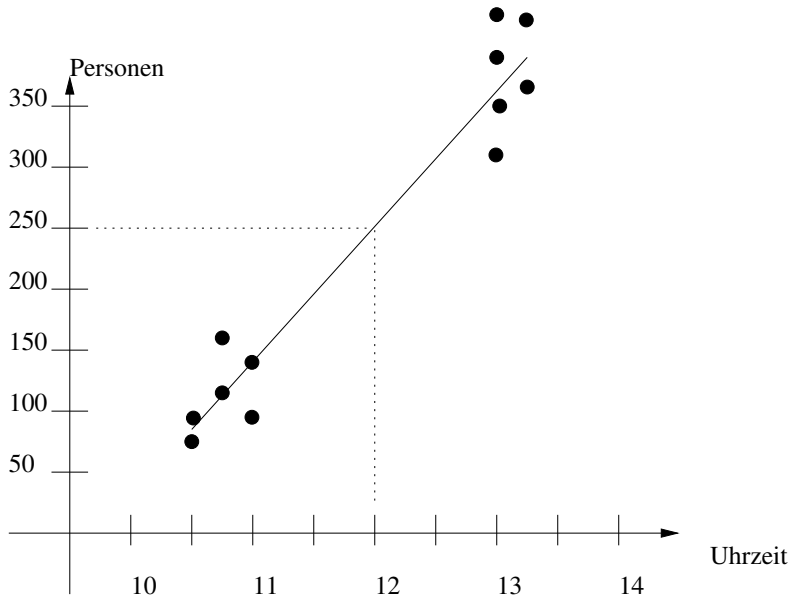

Abbildung 3.2: Approximation gegebener Messwerte.

eine der entscheidenden Fragen, die es zu klären gilt: Polynom, Spline, trigonometrische Funktion? Das hängt im wesentlichen von der gegebenen Aufgabenstellung ab.

Einen Zusammenhang zwischen Polynomen und stetigen Funktionen stellt der Weierstraßsche Approximationssatz her (Analysis I):

Satz 3.3 (Weierstraßsche Approximationssatz)**.** *Es sei $f \in C[a,b]$. Dann gibt es zu jedem $\varepsilon > 0$ ein auf $[a,b]$ definiertes Polynom p, so dass*

$$|f(x) - p(x)| < \varepsilon \quad \forall x \in [a,b].$$

Der Weierstraßsche Approximationssatz besagt zunächst, dass es möglich ist, jede stetige Funktion beliebig gut durch ein Polynom zu approximieren, hilft jedoch noch nicht bei der praktischen Durchführung. Auch hier gibt es einen einfachen Ansatz in der Analysis:

Satz 3.4 (Taylor-Entwicklung)**.** *Es sei $f \in C^{n+1}[a,b]$. Für das n-te Taylor-Polynom zu $x_0 \in (a,b)$*

$$t_n(x;x_0) := \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!}(x-x_0)^k$$

ist eine Approximation zu f in der Umgebung von x_0. Es gilt die Fehlerabschätzung:

$$f(x) - t_n(x;x_0) = \frac{f^{(n+1)}(\xi_x)}{(n+1)!}(x-x_0)^{n+1},$$

mit einer Zwischenstelle $\xi_x \in [a,b]$.

Hier ist $C([a,b])$ der Vektorraum der auf dem Intervall $[a,b]$ stetigen Funktionen und $C^k([a,b])$ der Raum der auf $[a,b]$ k-mal stetig differenzierbaren Funktionen. Die Taylor-Entwicklung ermöglicht eine konkrete Vorgehensweise zum Erstellen eines approximativen Polynoms. Wir sehen jedoch bereits, dass wir eine sehr starke Regularität von f benötigen. Darüber hinaus ist die Taylor-Entwicklung nur für Funktionen, nicht aber für diskrete Datenwerte (x_k, y_k) möglich.

Neben der Taylor-Entwicklung von Funktionen stellt die Analysis noch die *Fourier-Analyse* zur Approximation von periodischen Funktionen f mit Hilfe von trigonometrischen Polynomen zur Verfügung.

3.1 Polynominterpolation

Wir bezeichnen mit P_n den Vektorraum der Polynome vom Grad $\leq n$:

$$P_n := \left\{ p(x) = \sum_{k=0}^{n} a_k x^k \,|a_k \in \mathbb{R}, k = 0, \ldots, n \right\}.$$

Definition 3.5 (Lagrangesche Interpolationsaufgabe)**.** *Die Lagrangesche Interpolationsaufgabe besteht darin, zu $n+1$ paarweise verschiedenen Stützstellen (Knoten) $x_0, \ldots, x_n \in \mathbb{R}$ und zugehörigen gegebenen Stützwerten $y_0, \ldots, y_n \in \mathbb{R}$ ein Polynom $p \in P_n$ zu bestimmen, so dass die Interpolationsbedingung*

$$p(x_k) = y_k, \quad k = 0, \ldots, n,$$

erfüllt ist.

Satz 3.6. *Die Lagrangesche Interpolationsaufgabe ist eindeutig lösbar.*

Beweis: Die Eindeutigkeit wird zuerst nachgewiesen. Es seien $p_1, p_2 \in P_n$ zwei Lösungen. Dann gilt für das Differenzpolynom $p := p_1 - p_2 \in P_n$:

$$p(x_k) = 0, \quad k = 0, \ldots, n,$$

d.h. p hat $n+1$ Nullstellen und ist folglich das Nullpolynom. Dies kann mit Hilfe des Satzes von Rolle nachgewiesen werden. Zum Nachweis der Existenz betrachten wir die Gleichungen $p(x_k) = y_k, k = 0, \ldots, n$. Dies kann als ein lineares Gleichungssystem aufgefasst werden, welches $n+1$ Gleichungen für die $n+1$ unbekannten Koeffizienten $a_0, \ldots, a_n$ des Polynoms $p \in P_n$. Wegen der bereits gezeigten Eindeutigkeit des Interpolationspolynoms p (also der Injektivität) hat dieses System notwendigerweise eine Lösung (Surjektivität). □

Die gegebenen Stützwerte y_k können Werte einer gegebenen Funktion f sein, d.h.

$$f(x_k) = y_k, \quad k = 0, 1, \ldots, n,$$

oder auch beliebige diskrete Datenwerte

$$(x_k, y_k), \quad k = 0, 1, \ldots, n.$$

3.1.1 Lagrangesche Darstellung

In diesem Abschnitt verwenden wir zur Konstruktion des Interpolationspolynoms die Lagrangeschen Basispolynome

$$L_k^{(n)}(x) := \prod_{j=0, j\neq k}^{n} \frac{x - x_j}{x_k - x_j} \in P_n, \quad k = 0, 1, \ldots, n.$$

Die Basiseigenschaft möge der aufmerksame Leser selbst verifizieren. Als zweite Eigenschaft erhalten wir

$$L_k^{(n)}(x_l) = \delta_{kl} := \begin{cases} 1, & k = l \\ 0, & k \neq l, \end{cases}$$

wobei δ_{kl} das *Kronecker-Symbol* bezeichnet.

Satz 3.7 (Lagrangesche Darstellung)**.** *Das Polynom $p \in P_n$ gemäß*

$$p(x) := \sum_{k=0}^{n} y_k L_k^{(n)}(x)$$

erfüllt die Interpolationsbedingung $p(x_k) = y_k$.

Beweis: Zunächst gilt wegen $L_k^{(n)} \in P_n$ auch für die Linearkombination $p \in P_n$. Weiter folgt sofort:

$$p(x_j) = \sum_{k=0}^{n} y_k L_k^{(n)}(x_j) = \sum_{k=0}^{n} y_k \delta_{kj} = y_j.$$

□

Die Lagrangesche Darstellung des Interpolationspolynoms besticht durch ihre Einfachheit. Sie hat allerdings den großen Nachteil, das jedes Basispolynom $L_k^{(n)}(x)$ von sämtlichen Stützstellen $x_0, x_1, \ldots, x_n$ abhängt. Angenommen zur Steigerung der Genauigkeit soll eine Stützstelle x_{n+1} hinzugenommen werden, so müssen sämtliche Basispolynome ausgetauscht werden.

3.1.2 Newtonsche Darstellung

Die Newtonsche Darstellung der Lagrangeschen Interpolationsaufgabe löst dieses Problem durch eine andere Wahl von Basispolynomen:

$$\begin{aligned} N_0(x) &:= 1, \\ N_k(x) &:= \prod_{j=0}^{k-1}(x - x_j), \quad k = 1, \ldots, n. \end{aligned}$$

Das Basispolynom $N_k(x)$ hängt nur von den Stützstellen $x_0, \ldots, x_k$ ab. Bei Hinzunahme einer Stützstelle x_{k+1} müssen die ersten Basispolynome nicht geändert werden. Das Interpolationspolynom wird nach dem Ansatz

$$p(x) = \sum_{k=0}^{n} a_k N_k(x),$$

bestimmt. Für die Stützstelle x_k gilt $N_l(x_k) = 0$ für alle $l > k$. Wir gehen rekursiv vor:

$$\begin{aligned} y_0 &\stackrel{!}{=} p(x_0) = a_0, \\ y_1 &\stackrel{!}{=} p(x_1) = a_0 + a_1(x_1 - x_0), \\ &\vdots \\ y_n &\stackrel{!}{=} p(x_n) = a_0 + a_1(x_n - x_0) + \ldots + a_n(x_n - x_0)\cdots(x_n - x_{n-1}). \end{aligned}$$

Im Gegensatz zur vorher kennengelernten Lagrangeschen Darstellung kann ein weiteres Datenpaar (x_{n+1}, y_{n+1}) leicht hinzugefügt werden. Ist das Interpolationspolynom $p_n := p = \in P_n$ gegeben, so kann eine Stützstelle x_{n+1} einfach hinzugenommen werden:

$$y_{n+1} \stackrel{!}{=} p_{n+1}(x_{n+1}) = p_n(x_{n+1}) + a_{n+1}N_{n+1}(x_{n+1}) \quad \Rightarrow \quad a_{n+1} = \frac{y_{n+1} - p_n(x_{n+1})}{N_{n+1}(x_{n+1})}. \tag{3.1}$$

In der Praxis werden die Koeffizienten a_k auf eine numerisch stabilere und effizientere Weise bestimmt:

Satz 3.8. *Das Lagrangesche Interpolationspolynom zu den Punkten $(x_k, y_k), k = 0, \ldots, n$ lautet bezüglich der Newtonschen Polynombasis:*

$$p(x) = \sum_{k=0}^{n} y[x_0, \ldots, x_k]N_k(x).$$

Die Notation $y[x_0, \ldots, x_k]$ bezeichnet die dividierten Differenzen*, welche über die folgende rekursive Vorschrift definiert sind:*

```
1  für k = 0,...,n setze y[x_k] := y_k
2  für l = 1,...,n und
3      für k = 0,...,n-l berechne
4          y[x_k,...,x_{k+l}] := (y[x_{k+1},...,x_{k+l}] - y[x_k,...,x_{k+l-1}]) / (x_{k+l} - x_k)
```

BEWEIS: Wir bezeichnen mit $p_{k,k+l}$ in P_l das Polynom, welches die Interpolation zu den $l+1$ Punkte $(x_k, y_k), \ldots, (x_{k+l}, y_{k+l})$ darstellt. Insbesondere ist durch $p_{0,n}$ ist das gesuchte Polynom $p \in P_n$ gegeben. Wir zeigen, dass die folgende Aussage für alle $l = 0, \ldots n$ und $k = 0, \ldots, n-l$ gilt:

$$p_{k,k+l}(x) = y[x_k] + y[x_k, x_{k+1}](x - x_k) + \cdots + y[x_k, \ldots, x_{k+l}](x - x_k) \cdots (x - x_{k+l-1}). \quad (3.2)$$

Wir führen den Beweis durch Induktion nach dem Polynomgrad l. Für $l = 0$ gilt $p_{k,k}(x) = y[y_k] = y_k$. Angenommen, die Behauptung sei richtig für $l - 1 \geq 0$. D.h. insbesondere, das Polynome $p_{k,k+l-1}$ ist in Darstellung (3.2) gegeben und interpoliert die Punkte $(x_k, y_k), \ldots, (x_{k+l-1}, y_{k+l-1})$ und das Polynom $p_{k+1,k+l}$ interpoliert die Punkte (x_{k+1}, y_{k+1}), $\ldots$, (x_{k+l}, y_{k+l}). Dann ist durch

$$q(x) = \frac{(x - x_k)p_{k+1,k+l}(x) - (x - x_{k+l})p_{k,k+l-1}(x)}{x_{k+l} - x_k}, \quad (3.3)$$

ein Interpolationspolynom durch die Punkte $(x_k, y_k), \ldots, (x_{k+l}, y_{k+l})$ gegeben. Dies macht man sich durch Einsetzen von x_i in $q(x)$ deutlich. Für innere Punkte $i = k+1, \ldots, k+l-1$ gilt $p_{k,k+l-1}(x_i) = p_{k+1,k+l}(x_i) = y_i$, für x_k und x_{k+l} ist jeweils einer der beiden Faktoren gleich Null. Es gilt somit für das gesuchte Interpolationspolynom $p_{k,k+l} = q$. Nach Konstruktion (3.1) gilt jedoch auch die folgende Darstellung:

$$p_{k,k+l}(x) = p_{k,k+l-1}(x) + a(x - x_k) \cdots (x - x_{k+l-1}).$$

Koeffizientenvergleich des führenden Monoms x^n zwischen dieser Darstellung und (3.3) liefert mit Hilfe von (3.2) für $p_{k,k+l-1}$ sowie $p_{k+1,k+l}$:

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

□

Dieses rekursive Konstruktionsprinzip legt sofort einen Algorithmus zum Auswerten der Interpolation an einer Stelle $\xi \in \mathbb{R}$ nahe, ohne dass das gesuchte Interpolationspolynom $p(x)$ zuvor explizit bestimmt werden muss. Wir gehen hierzu von der Darstellung (3.3) aus:

Algorithmus 3.9 (Neville-Schema)**.** *Es seien die Stützstellenpaare* $(x_0, y_0), \ldots, (x_n, y_n)$ *gegeben sowie der Auswertungspunkt* ξ*. Berechne* $p(\xi) = p_{0,n}$*:*

```
1  für k = 0,...,n  setze  p_{k,k} := y_k
2  für l = 1,...,n  und
3      für k = 0,...,n-l  berechne
4          p_{k,k+l} := p_{k,k+l-1} + (ξ - x_k) (p_{k+1,k+l} - p_{k,k+l-1}) / (x_{k+l} - x_k)
```

Bei der Durchführung des Neville-Schemas erhalten wir mit $p_{k,l}$ automatisch die Approximationen der Interpolationspolynome durch $(x_k, y_k), \ldots, (x_{k+l}, y_{k+l})$. Wir führen das Neville-Schema exemplarisch durch:

Beispiel 3.10 (Neville-Schema). *Wir betrachten die Stützstellenpaare* $(0,0),(1,1),(2,8),(3,27)$*, welche von der Funktion* $f(x)=x^3$ *abgegriffen sind. Wir führen das Neville-Schema rekursiv aus:*

$$\begin{array}{cccccc} p_{00}=0 & & p_{11}=1 & & p_{22}=8 & & p_{33}=27 \\ & p_{01}=0.5 & & p_{12}=-2.5 & & p_{23}=-20.5 & \\ & & p_{02}=-0.25 & & p_{13}=2 & & \\ & & & p_{03}=0.125 & & & \end{array}$$

Die finale Approximation $p_{03}=0.125=0.5^3$ *ist exakt. Dies ist zu erwarten, da* $f\in P_3$*. Als Stichprobe betrachten wir die sehr schlechte Approximation* p_{23}*, welche sich durch das linearen Interpolationspolynom* $p_{2,3}(x)$ *durch die Stützstellen* $(2,8)$ *und* $(3,27)$*, also durch*

$$p_{2,3}(x)=8+\frac{27-8}{3-2}(x-2)=19x-30$$

ergibt. Es gilt $p_{2,3}(0.5)=-20.5$*.*

3.1.3 Interpolation von Funktionen und Fehlerabschätzungen

In diesem Abschnitt diskutieren wir die Interpolation von Funktionen. Die Punkte sind nun nicht mehr durch einen Datensatz gegeben, sondern durch Auswertung einer gegebenen Funktion f auf $[a,b]$:

$$y_k=f(x_k),\quad x_k\in[a,b],\ k=0,\dots,n.$$

Die Durchführbarkeit, also Existenz und Eindeutigkeit eines Interpolationspolynoms wurde bereits in den vorangehenden Abschnitten beantwortet. Bei der Interpolation von Funktionen stellt sich hier die Frage *wie gut* das Interpolationspolynom $p\in P_n$ die Funktion f auf $[a,b]$ approximiert.

Satz 3.11 (Interpolationsfehler mit differenziellem Restglied). *Es sei* $f\in C^{n+1}[a,b]$ *und* $p\in P_n$ *das Interpolationspolynom zu* f *in den* $n+1$ *paarweise verschiedenen Stützstellen* $x_0,\dots,x_n$*. Dann gibt es zu jedem* $x\in[a,b]$ *ein* $\xi\in(a,b)$*, so dass*

$$f(x)-p(x)=\frac{f^{n+1}(\xi)}{(n+1)!}\prod_{j=0}^{n}(x-x_j). \tag{3.4}$$

Insbesondere gilt

$$|f(x)-p(x)|\le\frac{\max_{\xi\in(a,b)}|f^{n+1}(\xi)|}{(n+1)!}\prod_{j=0}^{n}|x-x_j|. \tag{3.5}$$

Beweis: Falls x mit einer Stützstelle zusammenfällt, d.h. $x=x_k$ für ein $k\in\{0,\dots,n\}$, dann verschwindet der Fehler und wir sind fertig. Es sei daher $x\neq x_k$ für alle $k=0,1,\dots,n$

und $F \in C^{n+1}[a,b]$ mit

$$F(t) := f(t) - p_n(t) - K(x) \prod_{j=0}^{n} (t - x_j).$$

$K(x)$ sei so bestimmt, so dass $F(x) = 0$. Dies ist möglich, da

$$\prod_{j=0}^{n} (t - x_j) \neq 0 \quad \Rightarrow \quad K(x) = \frac{f(t) - p_n(t)}{\prod_{j=0}^{n}(t - x_j)}.$$

Dann besitzt $F(t)$ in $[a,b]$ mindestens $n+2$ verschiedene Nullstellen $x_0, x_1, \ldots, x_n, x$. Durch wiederholte Anwendung des Satzes von Rolle hat die Ableitung $F^{(n+1)}$ mindestens eine Nullstelle $\xi \in (a,b)$. Mit

$$0 = F^{(n+1)}(\xi) = f^{(n+1)}(\xi) - p^{(n+1)}(\xi) - K(x)(n+1)! = f^{(n+1)}(\xi) - 0 - K(x)(n+1)!.$$

Hieraus folgt die Behauptung mittels

$$K(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} \quad \Rightarrow \quad f(x) - p_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} \prod_{j=0}^{n} (x - x_j)$$

□

Satz 3.12 (Interpolationsfehler mit Integral-Restglied)**.** *Es sei $f \in C^{n+1}[a,b]$. Dann gilt für $x \in [a,b] \setminus \{x_0, \ldots, x_n\}$ die Darstellung*

$$f(x) - p(x) = f[x_0, \ldots, x_n, x] \prod_{j=0}^{n} (x - x_j),$$

mit den Interpolationsbedingungen $f[x_i, \ldots, x_{i+k}] := y[x_i, \ldots, x_{i+k}]$ und

$$f[x_0, \ldots, x_n, x] = \int_0^1 \int_0^{t_1} \cdots \int_0^{t_n} f^{(n+1)}(x_0 + t_1(x_1 - x_0) + \ldots + t(x - x_n))\, dt \ldots dt_2\, dt_1.$$

Beweis: Wird folgen. □

Für den Fehler der Lagrange-Interpolation können die folgenden Betrachtungen geführt werden. In (3.4) wird für großes n der Term $\frac{1}{(n+1)!}$ sehr klein. Das Produkt $\prod_{j=0}^{n}(x - x_j)$ wird klein, wenn die Stützstellen sehr dicht beieinander liegen. Sind alle Ableitungen von f gleichmäßig (bzgl. der Ableitungsstufe) beschränkt auf $[a,b]$, so gilt mit (3.5), dass

$$\max_{a \leq x \leq b} |f(x) - p(x)| \to 0, \quad n \to \infty.$$

Haben die Ableitungen der zu interpolierenden Funktion jedoch ein zu starkes Wachstumverhalten für $n \to \infty$, z.B.

$$f(x) = (1 + x^2)^{-1}, \quad |f^n(x)| \approx 2^n n! O(|x|^{-2-n}),$$

so konvergiert die Interpolation nicht gleichmäßig auf $[a,b]$.

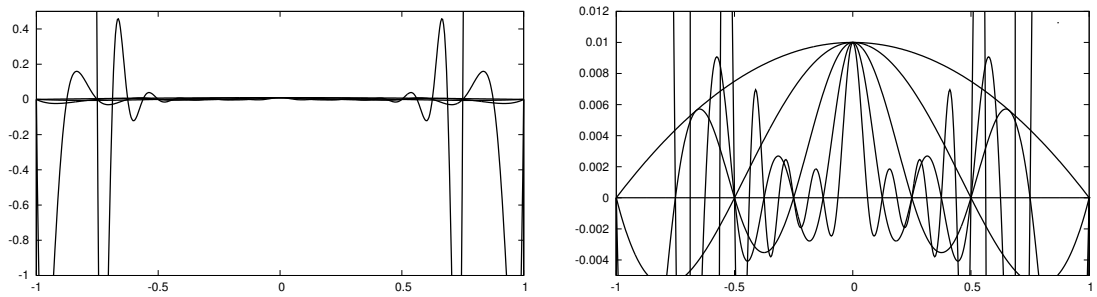

Abbildung 3.3: Interpolationspolynome $\tilde{p}_m(x) \in P_{2m}$. Links: das komplette Interpolationsintervall. Rechts: Ausschnitt nahe $y = 0$.

Beispiel 3.13. *Die Funktion $f(x) = |x|, x \in [-1, 1]$ werde mit Hilfe der Lagrange-Interpolation in den Stützstellen*

$$x_k = -1 + kh, \quad k = 0, \ldots, 2m, \quad h = 1/m, \quad x \neq x_k$$

interpoliert. Dies ergibt das globale Verhalten

$$p_m \nrightarrow f(x), \quad m \to \infty.$$

Zwar ist f in diesem Beispiel nicht differenzierbar, dennoch ist dieses Verhalten der Lagrange-Interpolation auch bei anderen Beispielen zu beobachten. Man betrachte z.B. die Funktion

$$f(x) = (1 + x^2)^{-1}, \quad x \in [-5, 5].$$

Wir fassen die bisherigen Ergebnisse zusammen:

Bemerkung 3.14. *Der Approximationssatz von Weierstraß besagt, dass jede Funktion $f \in C([a, b])$ durch ein Polynom beliebig gut approximiert werden kann. Die Analysis gibt jedoch keine Hilfestellung bei der konkreten Durchführung der Approximation. Die Lagrangesche Interpolation ist eine Möglichkeit zur Approximation. Die Qualität dieser Approximation wird jedoch wesentlich durch die Regularität der Daten, also durch f bestimmt. Eine gleichmäßige Approximation von Funktionen mit Lagrangeschen Interpolationspolynomen ist im Allgemeinen nicht möglich.*

Die Lagrangesche Interpolation "krankt" demnach an den gleichen Einschränkungen der Taylor-Entwicklung, Satz 3.4. Von der Möglichkeit, eine nur stetige Funktion $f \in C([a, b])$ beliebig gut zu approximieren sind wir noch weit entfernt.

Ein zweiter Nachteil der Lagrange-Interpolation ist die fehlende Lokalität. Eine Störung von in einer Stützstelle $(\tilde{x}_k, \tilde{y}_k)$ hat Auswirkung auf alle Lagrange-Polynome und insbesondere auf das gesamte Interpolationsintervall. Wir betrachten hierzu ein Beispiel:

Beispiel 3.15 (Globaler Fehlereinfluss)**.** *Wir suchen das Interpolationspolynom zu der Funktion $f(x) = 0$ in den $2m + 1$ Stützstellen*

$$x_k = -1 + kh, \quad k = -m, \dots, m, \quad h = \frac{1}{m}.$$

Das exakte Interpolationspolynom $p \in P_{2m}$ ist natürlich durch $p = 0$ gegeben. Wir nehmen an, dass die Funktionsauswertung in Nullpunkt gestört ist:

$$y_k = \begin{cases} 0 & k \neq 0 \\ \epsilon & k = 0 \end{cases},$$

mit einem kleinen ϵ. Das gestörte Interpolationspolynom ist in Lagrangescher Darstellung gegeben durch

$$\tilde{p}(x) = \prod_{i=-m, i\neq 0}^{m} \frac{x - x_i}{x_i}.$$

In Abbildung 3.3 zeigen wir $\tilde{p}_m \in P_{2m}$ für die Fälle $m = 1, 2, 4, 8$ mit einer Störung $\epsilon = 0.01$. Trotz dieser kleinen Störung an der Stelle $x = 0$ weichen die Interpolationspolynom am Intervallrand sehr stark von $y_k = 0$ ab. In der unteren Abbildung sieht man, dass für kleine Polynomgrade, also $m = 1$ und $m = 2$ der maximale Fehler auf dem Intervall nicht größer als die anfängliche Störung $\epsilon = 0.01$ ist. Die Lagrangesche Interpolation ist instabil für große Polynomgrade.

Hermite Interpolation

Zum Abschluss der Funktionsinterpolation erwähnen wir noch eine Verallgemeinerung, die sogenannte *Hermitesche Interpolationsaufgabe*. Diese unterscheidet sich von der Lagrange-Interpolation durch die Möglichkeit neben Funktionswerten $p(x_k) = f(x_k)$ auch Gleichheit von Ableitungswerten $p^{(i)}(x_k) = f^{(i)}(x_k)$ zu fordern. Wir fassen zusammen:

Satz 3.16 (Hermite Interpolation)**.** *Es sei $f \in C^{(n+1)}([a, b])$. Es seien $x_0, \dots, x_m$ paarweise verschiedene Stützstellen und $\mu_k \in \mathbb{N}$ für $k = 0, \dots, m$ ein Ableitungsindex. Ferne gelte $n = m + \sum_{k=0}^{m} \mu_k$. Das Hermitesche Interpolationspolynom zu*

$$k = 0, \dots, m: \quad p^{(i)}(x_k) = f^{(i)}(x_k), \quad i = 0, \dots, \mu_k,$$

ist eindeutig bestimmt und zu jedem $x \in [a, b]$ existiert eine Zwischenstelle $\xi \in [a, b]$, so dass gilt:

$$\begin{aligned} f(x) - p(x) &= f[\underbrace{x_0, \dots, x_0}_{\mu_0+1}, \dots, \underbrace{x_m, \dots, x_m}_{\mu_m+1}, x] \prod_{k=0}^{m} (x - x_k)^{\mu_k+1} \\ &= \frac{1}{(n+1)!} f^{(n+1)}(\xi) \prod_{k=0}^{m} (x - x_k)^{\mu_k+1}. \end{aligned}$$

3.2 Spline Interpolation

Ein wesentlicher Defekt der *globalen* Interpolation aus dem vorherigen Abschnitt ist, dass die interpolierenden Polynome starke Oszillationen zwischen den Stützstellen mit immer größeren Werten annehmen. Der Grund ist die generische Steifheit, die durch die *implizite* Forderung von C^∞-Übergängen in den Stützstellen gegeben ist. Die Steifheit kann dadurch reduziert werden, dass die globale Funktion als *stückweise* polynomiale (Teil)-Funktionen bzgl. der Zerlegung $a = x_0 < x_1 < \ldots < x_n < b$ zusammengesetzt wird. In den Stützstellen x_i werden dann geeignete Differenzierbarkeitseigenschaften gefordert. Am häufigsten werden in der Literatur sogenannte *kubische* Splines verwendet. Allerdings hat eine spezielle Klasse, die linearen Splines, eine besondere Bedeutung für die numerische Lösung von Differentialgleichungen mit Hilfe der Finite Elemente Methode [10, 12].

Das wesentliche Ziel dieses Abschnitts liegt im Verständnis der *stückweisen* Approximation. Dieses Konzept erlaubt die gleichmäßige Abschätzung der Konvergenz und verhindert Oszillationen am Rand des Intervalls, wie sie in Beispiel 3.15 auftreten.

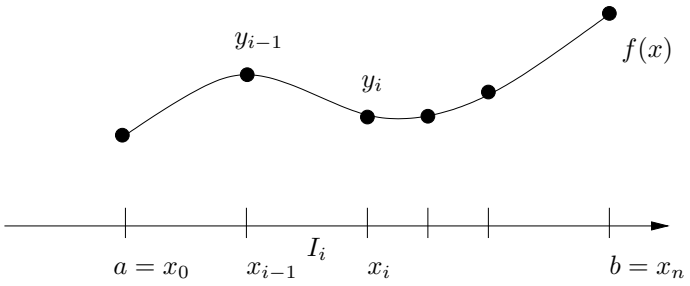

Abbildung 3.4: Spline-Interpolation.

Das globale Intervall (wie vorher $[a, b] =: I$) wird in Teilintervalle $I_i = [x_{i-1}, x_i]$ mit der Länge $h_i = x_i - x_{i-1}$ unterteilt. Die Feinheit der gesamten Intervallunterteilung wird durch $h := \max_{i=1,\ldots n} h_i$ charakterisiert. Zur Definition der Splinefunktion (kurz Spline) seien die Vektorräume von stückweisen polynomialen Funktionen wie folgt gegeben:

$$S_h^{k,r}[a, b] := \{p \in C^r[a, b],\, p|_{I_i} \in P_k(I_i)\}, \quad k, r \in \{0, 1, 2, \ldots\}.$$

Zu einem Satz gegebener Stützwerte (die wie in Abschnitt 3.1 aus gegebenen Datenwerten oder Funktionswerten stammen können) in dem Gesamtintervall I, wird eine Interpolierende $p \in S_h^{k,r}[a, b]$ mit Hilfe von geeigneten Interpolationsbedingungen bestimmt.

Wir diskutieren nun einige Beispiele, wobei der Fokus auf der Idee des ganzen Prozesses liegt und weniger auf der Beweisführung bei Fragen zu Existenz, Eindeutigkeit und Fehlerabschätzung.

Beispiel 3.17 (Stückweise lineare Interpolation)**.** *Die stückweise lineare Lagrange-Interpolierende (d.h. $k = 1, r = 0$) approximiert eine gegebene Funktion f auf $[a, b]$ durch einen Polygonzug*

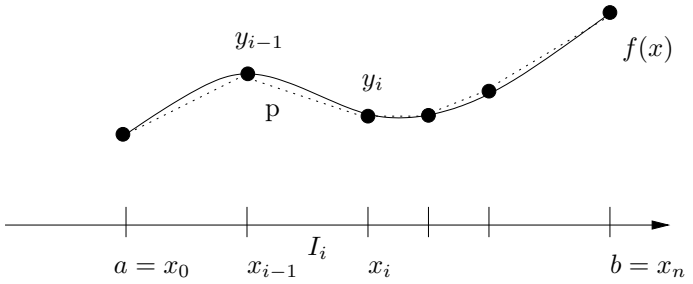

Abbildung 3.5: Stückweise lineare Interpolation p einer Funktion f.

in den Stützstellen x_i, $i = 0, \ldots, n$:

$$p \in S_h^{1,0}[a,b] = \{p \in C[a,b],\ p|_{I_i} \in P_1(I_i)\},$$

mit den Interpolationsbedingungen

$$p(x_i) = f(x_i), \quad i = 0, \ldots, n.$$

Anwendung der Fehlerabschätzung für die Lagrange-Interpolation separat auf jedem I_i liefert die globale Fehlerabschätzung

$$\max_{x\in[a,b]} |f(x) - p(x)| \leq \frac{1}{2}h^2 \max_{x\in[a,b]} |f''(x)|. \tag{3.6}$$

Für Schrittweite gilt

$$\max_{x\in[a,b]} |f(x) - p(x)| \to 0 \quad (h \to 0),$$

gleichmäßig auf dem gesamten Intervall. Im Gegensatz hierzu erhalten wir für $n \to \infty$ also für größer werdenden Polynomgrad keine gleichmäßige Konvergenz!

Bemerkung 3.18. *Wir halten fest, dass eine gleichmäßige Approximation für Splines durch* (3.6) *gewährleistet ist, falls $h \to 0$ (d.h. eine immer größer werdende Anzahl von Stützstellen). Es ist wichtig, dass eine größere Anzahl von Stützstellen bei der Lagrange-Interpolation nicht hilft, da hier die Anzahl Stützstellen an den Polynomgrad gekoppelt sind. D.h. eine größere Anzahl von Stützstellen impliziert automatisch einen höheren Polynomgrad. Allerdings können höhere Ableitungen der zu interpolierenden Funktion f starkes Wachstum haben, wodurch die gleichmäßige Fehlerabschätzung in* (3.5) *nichtig wird.*

Die Interpolierende des vorangegangenen Beispiels wird mit Hilfe der sogenannten Knotenbasis von $S_h^{(1,0)}([a,b])$ konstruiert. Diese Knotenbasis besteht aus den *Hutfunktionen* $\phi_i \in S_h^{(1,0)}[a,b], i = 0, \ldots, n$, die durch die Bedingung

$$\phi_i(x_j) = \delta_{ij}$$

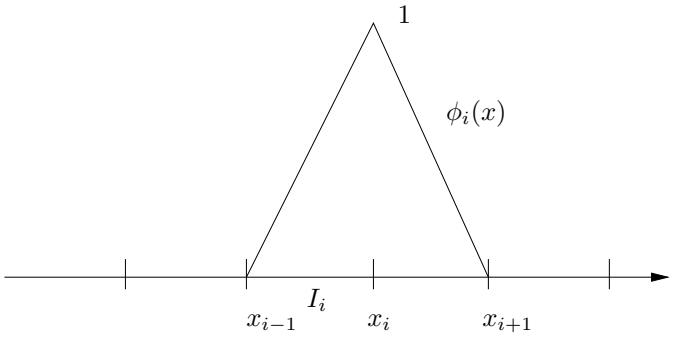

Abbildung 3.6: Lineare Knotenbasisfunktion - auch Hutfunktion genannt.

eindeutig bestimmt sind. Die Interpolierende p von f erlaubt dann die Darstellung in Form von

$$p(x) = \sum_{i=0}^{n} f(x_i)\phi(x_i).$$

Diese Konstruktion stellt die Analogie zur Lagrangeschen Darstellung des Lagrange-Interpolationspolynoms her. Im Gegensatz zu dieser *globalen* Sichtweise arbeiten wir aber bei den Splines *lokal*, da die Hutfunktionen ϕ_i nur in den direkt an der jeweiligen Stützstelle x_i angrenzenden Teilintervallen von Null verschieden ist (siehe Abbildung 3.6).

Bemerkung 3.19. *Aus den bisherigen Betrachtungen der linearen Splines erhalten wir die Erkenntnis, dass eine höhere globale Glattheit (sprich Differenzierbarkeit) offensichtlich nicht sinnvoll zu erzielen ist. Man mache sich nochmals klar, dass an den Stützstellen keine Differenzierbarkeit vorliegen kann (wegen $r = 0$) und versuche sich den Fall $r = 1$ vorzustellen.*

Bemerkung 3.20. *Bei Interesse an höheren Vorlesungen zur Numerik sollte das Konstruktionsprinzip der linearen Splines nicht vergessen werden, da es die Grundlage der Finite Elemente Technik bildet [10, 12].*

Wie in Bemerkung 3.19 erläutert, ist die Konstruktion von linearen Splines mit höheren globalen Glattheitseigenschaften nicht ohne weiteres möglich. Erhöht man jedoch den lokalen Polynomgrad auf jedem Teilintervall, so kann mit Hilfe der Interpolationsbedingungen höhere Glattheit erzielt werden. Dies führt auf die kubischen Splines, d.h. $k = 3$.

Beispiel 3.21 (Stückweise kubische Interpolation)**.** *Es sei auf jedem Teilintervall I_i ein kubisches Polynom vorgeschrieben, d.h. $k = 3$ mit $r = 0$, so dass $S_h^{(3,0)}[a, b]$. Die Interpolationsbedingungen für den Fall $r = 0$ (d.h. globale Stetigkeit) lauten*

$$p(x_i) = f(x_i).$$

Zur eindeutigen Bestimmung eines kubischen Polynoms sind in jedem I_i vier Bedingungen notwendig, so dass zwei zusätzliche Interpolationspunkte vorgegeben werden:

$$p(x_{ij}) = f(x_{ij}),$$

wobei $x_{ij} \in I_i$, $i = 1, \ldots,$ $j = 1, 2$. Durch diese stückweise kubische Lagrange-Interpolation ist eindeutig ein global stetiger Spline $p \in S_h^{(3,0)}[a, b]$ festgelegt.

Anstatt die Interpolationsbedingung durch Zwischenwerte $x_{ij} \in (x_{i-1}, x_i)$ anzureichern ist es auch möglich Ableitungswerte vorzugeben:

Beispiel 3.22 (Stückweise kubische Hermite-Interpolation)**.** *Es sei wiederum auf jedem Teilintervall I_i ein kubisches Polynom vorgeschrieben, d.h. $k = 3$ und dieses Mal mit $r = 1$, so dass $S_h^{(3,1)}[a, b]$. Die Interpolationsbedingungen für den Fall $r = 1$ (d.h. globale Stetigkeit und einmalige globale Differenzierbarkeit) lauten*

$$p(x_i) = f(x_i), \quad p'(x_i) = f'(x_i)$$

Durch diese vier Bedingungen $p \in S_h^{(3,1)}[a, b]$ eindeutig festgelegt.

Bemerkung 3.23. *Nutzt man zusätzlich zu den Punktwerten die Ableitungsinformation in den Stützstellen zur Konstruktion einer Interpolierenden, so wie in Beispiel 3.2 geschehen, dann spricht man von Hermite-Interpolation. Diese Interpolationsaufgabe kann für globale und lokale Interpolationen genutzt werden, siehe Satz 3.16*

Satz 3.24. *Für den kubischen Spline p (d.h. $k = 3$) mit $r = 0$ oder $r = 1$ zur Approximation der Funktion f gilt die globale Fehlerabschätzung*

$$\max_{x \in [a,b]} |f(x) - p(x)| \leq \frac{1}{4!} h^4 \max_{x \in [a,b]} |f^{(4)}(x)|.$$

Zur eindeutigen Bestimmung einer stückweise kubischen Funktion sind auf jedem Intervall I_i vier Bedingungen erforderlich. Dennoch unterscheiden sich die Sätze an Bedingungen in den beiden betrachteten Beispielen. Bei der stückweise kubischen Lagrange Interpolation in Beispiel werden in den n Intervallen jeweils vier Bedingungen $p(x_{ij}) = f(x_{ij})$ gestellt, wobei an den Intervallenden x_i die gleiche Interpolationsbedingung doppelt auftaucht. Insgesamt ist die stückweise kubische Lagrange Interpolation durch $3n + 1$ Bedingungen gegeben. Die stückweise Hermite Interpolation in Beispiel hat auch vier Bedingungen pro Intervall. An den Intervallenden treten nun jedoch 2 Bedingungen doppelt auf, so dass sich global $2n + 2$ Bedingungen ergeben.

Soll die globale Regularität der Interpolation weiter gesteigert werden, so schlägt der triviale Ansatz zusätzlich $p''(x_i) = f''(x_i)$ zu fordern fehl, da dies zu sechs Bedingungen pro Teilintervall führen würde (bei nur vier Unbekannten von eines kubischen Polynoms).

Es ist dennoch möglich für eine stückweise kubische Interpolation global $C^2([a,b])$ zu erreichen. Dies wird durch den *kubischen Spline* realisiert, welcher auf dem Prinzip der *Energieminimierung* basiert. Zu Stützstellen x_i für $i = 0, \ldots, n$ wird eine Funktion $s \in S_h^{(3,2)}$ gesucht mit der Eigenschaft:

$$s(x_i) = f(x_i) = y_i, \quad i = 0, \ldots, n, \quad \int_a^b s''(x)^2 \, \mathrm{d}x = \min!.$$

Die Ableitungswerte von $s(x)$ müssen in den Stützstellen keiner Interpolationsbedingung genügen, $s(x)$ muss global lediglich stetig zweimal differenzierbar sein (d.h., die Ableitungswerte von $s(x)$ müssen an den Enden der Teilintervalle stetig sein) und die Energie, also das Integral über die zweiten Ableitungen muss minimiert werden.

Hieraus ist wohl auch der Begriff Spline entsprungen, da seine Übersetzung ins Deutsche der *Biegestab* ist. Man stelle sich einen elastischen Stab vor, welcher an gewissen Punkten festgehalten wird, dazwischen jedoch eine freie Form annehmen darf. Nach physikalischen Grundprinzipien wird diese Form die Energie des Stabes minimieren. Das Energieminimierungsprinzip ist höchst wichtig und tritt in der Natur und Alltag häufig auf (z.B. Seifenhäute). Man mache sich klar, dass eine minimale Energie den geringsten Energieverbrauch bedeutet und damit *Treibstoffkosten* gespart werden kann.

Diese Form des Splines ist die am häufigsten genutzte Spline-Interpolationsaufgabe und hat z.B. große Anwendungen in der Computergrafik.

Mathematisch ausgedrückt bedeutet Energieminimierung im Falle des Splines:

$$\int_a^b s''(x)^2 \, dx = \min!$$

bzgl. aller möglichen interpolierenden (hinreichend glatten) Funktionen.

Definition 3.25 (Kubischer Spline)**.** *Eine Funktion $s_n : [a,b] \to \mathbb{R}$ wird* kubischer *Spline bzgl. Zerlegung $a = x_0 < x_1 < \ldots < x_n = b$ genannt, wenn gilt*

i) $s_n \in C^2[a,b]$.

ii) $s_n|_{[x_{i-1},x]} \in P_3, \quad i = 1, \ldots n.$

Falls zusätzlich

iii) $s_n''(a) = s_n''(b) = 0$

gilt, so heißt der Spline natürlich.

Satz 3.26 (Existenz, Energieminimierung und Fehlerabschätzung kubischer Splines)**.** *Es sei f eine gegebene zu interpolierende Funktion oder $y_0, \ldots, y_n$ eine Reihe von Messwerten. Durch $x_0, x_1, \ldots, x_n$ seien $n+1$ paarweise verschiedene Stützstellen gegeben. Der durch die Interpolationsvorschrift*

$$s_n(x_i) = f(x_i) = y_i \in \mathbb{R}, \quad i = 0, \ldots, n, \quad s_n''(a) = y_a \in \mathbb{R}, \quad s_n''(b) = y_b \in \mathbb{R}$$

beschriebene kubische Spline existiert. Er ist eindeutig bestimmt durch die Vorgabe der zweiten Ableitungsinformationen.
Sei weiter $g \in C^2([a,b])$ beliebig mit $g(x_i) = f(x_i) = y_i$ mit $x_i \in [a,b]$ sowie $y_a = g''(a)$ und $y_b = g''(b)$. Dann gilt das Prinzip der Energieminimierung:

$$\int_a^b |s_n''(x)|^2 \, dx \leq \int_a^b |g''(x)|^2 \, dx, \quad \forall g \in C^2[a,b].$$

Falls $f \in C^4[a,b]$, so gilt die Fehlerabschätzung

$$\max_{a \leq x \leq b} |f(x) - s_n(x)| \leq \frac{1}{2} h^4 \max_{a \leq x \leq b} |f^{(4)}(x)|.$$

Beweis: Siehe [9], S. 43-47. □

3.3 Numerische Differentiation

In diesem Abschnitt befassen wir uns mit einer einfachen numerischen Aufgabe: zu einer gegebenen Funktion $f : [a,b] \to \mathbb{R}$ soll in einem Punkt $x_0 \in (a,b)$ die Ableitung $f'(x_0)$ oder die n-te Ableitung $f^{(n)}(x_0)$ berechnet werden. Wir gehen davon aus, dass es mit vertretbarem Aufwand nicht möglich ist, die Funktion $f(x)$ symbolisch zu Differenzieren und die Ableitung an der Stelle x_0 auszuwerten. Wir benötigen also Approximationsverfahren zur Bestimmung der Ableitung. Die grundlegende Idee wird durch das folgende Verfahren beschrieben:

Verfahren 3.27 (Numerische Differentiation)**.** *Die Funktion $f : [a,b] \to \mathbb{R}$ wird durch ein Polynom $p(x)$ interpoliert. Die n-te Ableitung $f^{(n)}(x_0)$ wird durch die n-te Ableitung des Polynoms $p^{(n)}(x_0)$ approximiert.*

Im Folgenden entwickeln wir einige einfache Verfahren zur Approximation von Ableitungen, welche auf der Interpolation beruhen.

Lineare Interpolation Wir interpolieren $f(x)$ linear in den Stützstellen x_0 sowie $x_0 + h$:

$$p_1(x) = f(x_0)\frac{x_0 + h - x}{h} + f(x_0 + h)\frac{x - x_0}{h}, \quad p_1'(x) = \frac{f(x_0 + h) - f(x_0)}{h}.$$

Wir approximieren in $x = x_0$ und erhalten mit Taylorentwicklung von $f(x_0 + h)$ um x_0

$$p_1'(x_0) = \frac{f(x_0 + h) - f(x_0)}{h} = f'(x_0) + \frac{h}{2} f''(x_0) + O(h^2),$$

bei gegebener Differenzierbarkeit von f eine Approximation der ersten Ableitung von erster Ordnung, also $p_1'(x_0) = f'(x_0) + O(h)$. Diese erste Approximation heißt *einseitiger Differenzenquotient* . Dabei kann zur Approximation natürlich auch eine Stützstelle $x_0 - h$ links von der Auswertungsstelle gewählt werden:

$$p_1(x) = \frac{f(x_0) - f(x_0 - h)}{h} = f'(x_0) + \frac{h}{2} f''(x_0) + O(h^2).$$

Insbesondere bei der Diskretisierung von gewöhnlichen Differentialgleichungen haben sich für diese beiden Varianten die Begriffe *Vorwärtsdifferenzenquotient* und *Rückwärtsdifferenzenquotient* etabliert (je nachdem, ob die weitere Stützstelle nach vorne $x_0 + h$ oder zurück $x_0 - h$ greift).

Aus Symmetriegründen erscheint es ebenso sinnvoll, das Interpolationspolynom in der Mitte zwischen den beiden Stützstellen auszuwerten. Wir legen daher ein lineares Interpolationspolynom durch die beiden Stützstellen $x_0 - h$ und $x_0 + h$:

$$\hat{p}_1(x) = f(x_0 - h)\frac{x_0 + h - x}{2h} + f(x_0 + h)\frac{x - x_0 + h}{2h}, \quad \hat{p}_1'(x) = \frac{f(x_0 + h) - f(x_0 - h)}{2h}.$$

Wir approximieren wieder bei x_0 und erhalten mit Taylorentwicklung beider Terme um den Mittelpunkt x_0 wegen

$$f(x_0 \pm h) = f(x_0) \pm h f'(x_0) + \frac{h^2}{2} f''(x_0) \pm \frac{h^3}{6} f'''(x_0) + \frac{h^4}{24} f^{(iv)}(x_0) + O(h^5),$$

die bessere Approximation von zweiter Ordnung:

$$\hat{p}_1'(x_0) = \frac{f(x_0 + h) - f(x_0 - h)}{2h} = f'(x_0) + \frac{h^2}{6} f'''(x_0) + O(h^4). \tag{3.7}$$

Diese wichtige Approximation wird *zentraler Differenzenquotient* genannt. Das Ausnutzen von Symmetrie beim Entwurf von numerischen Verfahren führt oft zu einer besseren Konvergenzordnung als zunächst erwartet. So ein Verhalten wird im Allgemeinen *Superapproximation* genannt.

Eine lineare Interpolation $p_1 \in P_1$ eignet sich nicht zur Approximation höherer Ableitungen, da p_1' konstant ist.

Quadratische Interpolation Wir interpolieren $f(x)$ mit Hilfe der drei Stützstellen $x_0 - h$, x_0, $x_0 + h$ durch ein quadratisches Polynom:

$$\begin{aligned} p_2(x) = {} & f(x_0 - h)\frac{(x_0 - x)(x_0 + h - x)}{2h^2} \\ & + f(x_0)\frac{(x - x_0 + h)(x_0 + h - x)}{h^2} + f(x_0 + h)\frac{(x - x_0 + h)(x - x_0)}{2h^2}. \end{aligned}$$

In $x = x_0$ gilt für erste und zweite Ableitungen:

$$p_2'(x_0) = \frac{f(x_0+h) - f(x_0-h)}{2h}, \quad p_2''(x_0) = \frac{f(x_0+h) - 2f(x_0) + f(x_0-h)}{h^2}.$$

Für die erste Ableitung ergibt sich erstaunlicherweise wieder der zentralen Differenzenquotienten, der schon durch lineare Interpolation hergeleitet wurde. Dies ist der Ursprung der Bezeichnung *Superapproximation*: mit Hilfe der linearen Interpolierenden wird ein Ergebnis erreicht, das eigentlich erst bei quadratischer Interpolierender zu erwarten wäre.

Für die zweite Ableitung erhalten wir mit Taylorentwicklung:

$$p_2''(x_0) = \frac{-2f(x_0) + f(x_0-h) + f(x_0+h)}{h^2} = f''(x_0) + \frac{1}{12}h^2 f^{(iv)}(x_0) + O(h^4)$$

den zentrale Differenzenquotient für die zweite Ableitung.

Wir können auf der Basis von p_2 auch einen einseitigen Differenzenquotienten für die zweite Ableitung herleiten. Dies erfolgt durch Approximation von $f''(x_0-h) \approx p_2''(x_0-h)$. Wieder mit Taylorentwicklung erhalten wir

$$p''(x_0-h) = f''(x_0-h) + hf'''(x_0-h) + O(h^2)$$

lediglich eine Approximation erster Ordnung. Neben der Ordnung des Interpolationspolynoms $p(x)$ kommt es entscheidend auf die entsprechende Wahl der Stützstellen an.

Wir betrachten ein Beispiel:

Beispiel 3.28. *Es sei*

$$f(x) = \tanh(x).$$

Wir suchen eine Approximation von

$$f'\left(\frac{1}{2}\right) \approx 0.7864477329659274, \quad f''\left(\frac{1}{2}\right) \approx -0.7268619813835874.$$

Zur Approximation verwenden wir für die vier bisher diskutieren Differenzenquotienten zu verschiedenen Schrittweiten $h > 0$:

In Abbildung 3.7 tragen wir die Fehler der Approximationen gegenüber der Schrittweite auf. Hier ist deutlich der Unterschied zwischen linearer und quadratischer Ordnung in h zu erkennen.

Stabilität

Abschließend untersuchen wir die Stabilität der Differenzenapproximation. Die Koeffizienten der verschiedenen Formeln wechseln das Vorzeichen, somit besteht die Gefahr der Auslöschung. Exemplarisch führen wir die Stabilitätsanalyse für die zentralen Differenzenapproximation zur Bestimmung der ersten Ableitung durch. Wir gehen davon aus, dass die

h	$\frac{f(\frac{1}{2}+h)-f(\frac{1}{2})}{h}$	$\frac{f(\frac{1}{2}+h)-f(\frac{1}{2}-h)}{2h}$	$\frac{f(\frac{1}{2}+2h)-f(\frac{1}{2}+h)+f(\frac{1}{2})}{h^2}$	$\frac{f(\frac{1}{2}+h)-2f(\frac{1}{2})+f(\frac{1}{2}-h)}{h^2}$
$\frac{1}{2}$	**0.**598954	**0.7**61594	$-$**0.**623692	$-$**0.**650561
$\frac{1}{4}$	**0.**692127	**0.78**0461	$-$**0.7**45385	$-$**0.7**06667
$\frac{1}{8}$	**0.7**39861	**0.78**4969	$-$**0.7**63733	$-$**0.721**740
$\frac{1}{16}$	**0.7**63405	**0.786**079	$-$**0.7**53425	$-$**0.725**577
$\frac{1}{32}$	**0.7**75004	**0.786**356	$-$**0.7**42301	$-$**0.726**540
$\frac{1}{64}$	**0.78**0747	**0.7864**25	$-$**0.7**35134	$-$**0.726**782
Exakt	0.786448	0.786448	-0.726862	-0.726862

Tabelle 3.1: Differenzenapproximation von $f'(1/2)$ (zwei Tabellen links) und $f''(1/2)$ (rechts) der Funktion $f(x) = \tanh(x)$. Dabei ist jeweils der einseitige bzw. der zentrale Differenzenquotient genutzt worden.

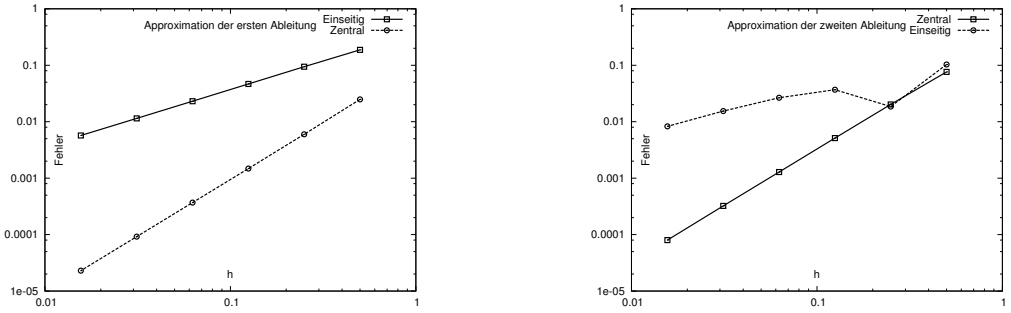

Abbildung 3.7: Fehler bei der Differenzenapproximation der ersten (links) und zweiten (rechts) Ableitung von $f(x) = \tanh(x)$ im Punkt $x_0 = \frac{1}{2}$.

Funktionswerte an den beiden Stützstellen nur fehlerhaft (mit relativem Fehler $|\epsilon| \leq eps$) ausgewertet werden können und erhalten:

$$\begin{aligned}\tilde{p}'(x_0) &= \frac{f(x_0+h)(1+\epsilon_1) - f(x_0-h)(1+\epsilon_2)}{2h}(1+\epsilon_3) \\ &= \frac{f(x_0+h) - f(x_0-h)}{2h}(1+\epsilon_3) + \frac{\epsilon_1 f(x_0+h) - \epsilon_2 f(x_0-h)}{2h} + O(eps^2).\end{aligned}$$

Für den relativen Fehler gilt

$$\left|\frac{\tilde{p}'(x_0) - p'(x_0)}{p'(x_0)}\right| \leq eps + \frac{|f(x_0+h)| + |f(x_0-h)|}{|f(x_0+h) - f(x_0-h)|} eps + O(eps^2).$$

Im Fall $f(x_0+h) \approx f(x_0-h)$ also $f'(x_0) \approx 0$ kann der Fehler beliebig stark verstärkt werden. Je kleiner h gewählt wird, umso größer wird dieser Effekt, denn:

$$f(x_0+h) - f(x_0-h) = 2hf'(x_0) + O(h^2).$$

Dieses Ergebnis ist gegenläufig zur Fehlerabschätzung für den Differenzenquotienten (3.7). Bei der numerischen Approximation müssen beide Fehleranteile addiert werden:

$$\begin{aligned}\frac{|f'(x_0)-\tilde{p}'(x_0)|}{|f'(x_0)|} &\leq \frac{|f'(x_0)-p'(x_0)|}{|f'(x_0)|} + \frac{|p'(x_0)-\tilde{p}'(x_0)|}{|f'(x_0)|} \\ &\leq \frac{1}{3}h^2 + O(h^4) + \frac{\max_\xi |f(\xi)|}{|f'(x_0)|h} eps + O(eps^2).\end{aligned}$$

Für kleines h steigt der Rundungsfehleranteil. Der Gesamtfehler wird minimal, falls beide Fehleranteile balanciert sind, also im Fall:

$$h^2 \approx \frac{eps}{h} \quad \Rightarrow \quad h \approx \sqrt[3]{eps}.$$

3.4 Richardson Extrapolation zum Limes

Eine wichtige Anwendung der Interpolation ist die *Extrapolation zum Limes*. Die Idee lässt sich am einfachsten anhand eines Beispiels erklären. Wir wollen die Ableitung $f'(x_0)$ einer Funktion f im Punkt x_0 mit Hilfe des einseitigen Differenzenquotienten berechnen:

$$a(h) := \frac{f(x_0+h)-f(x_0)}{h}.$$

Der Schrittweitenparameter h bestimmt die Qualität der Approximation $a(h) \approx f'(x_0)$. Für $h \to 0$ gilt (bei Vernachlässigung von Rundungsfehlern) $a(h) \to f'(x_0)$. An der Stelle $h = 0$ lässt sich $a(h)$ jedoch nicht auswerten. Die Idee die Extrapolation zum Limes ist es nun, ein Interpolationspolynom $p(x)$ durch die Stützstellenpaare $(h_i, a(h_i))$ für eine Folge von Schrittweiten $h_0, h_1, \ldots, h_n$ zu legen und den Wert $p(0)$ als Approximation für $a(0)$ zu verwenden.

Es stellt sich die grundlegende Frage: hat die Interpolierende in den Stützstellen $h_0, \ldots, h_n$ auch Aussagekraft außerhalb des Intervalls $I := [\min_i h_i, \max_i h_i]$? Beispiel 3.15 lässt dies zunächst nicht vermuten. Hier war die Approximationseigenschaft durch Oszillationen in Punkten zwischen den Stützstellen schon am Rande des Intervalls I stark gestört. Wir betrachten dennoch ein einfaches Beispiel:

Beispiel 3.29 (Extrapolation des einseitigen Differenzenquotienten)**.** *Zu $f \in C^3([a,b])$ sei*

$$a(h) = \frac{f(x_0+h)-f(x_0)}{h} = f'(x_0) + \frac{h}{2}f''(x_0) + \frac{h^2}{6}f'''(\xi_{x_0,h}), \tag{3.8}$$

die einseitige Approximation der ersten Ableitung. Wir legen durch die Stützstellen $(h, a(h))$ sowie $(h/2, a(h/2))$ das lineare Interpolationspolynom,

$$p(t) = \left(\frac{t-\frac{h}{2}}{h-\frac{h}{2}}\right) a(h) + \left(\frac{t-h}{\frac{h}{2}-h}\right) a\left(\frac{h}{2}\right),$$

und werten dieses an der Stelle $t = 0$ als Approximation von $a(0)$ aus:

$$a(0) \approx p(0) = 2a\left(\frac{h}{2}\right) - a(h).$$

Für $a(h/2)$ sowie $a(h)$ setzen wir die Taylor-Entwicklung (3.8) ein und erhalten mit

$$\begin{aligned} p(0) &= 2\left(f'(x_0) + \frac{h}{4}f''(x_0) + \frac{h^2}{24}f'''(\xi_{x_0,h/2})\right) - \left(f'(x_0) + \frac{h}{2}f''(x_0) + \frac{h^2}{6}f'''(\xi_{x_0,h})\right) \\ &= f'(x_0) + O(h^2), \end{aligned}$$

eine Approximation der ersten Ableitung von zweiter Ordnung in der Schrittweite h.

Dieses Extrapolationsprinzip lässt sich mit weiteren Stützstellen und Interpolation mit Polynomen von größerem Grad fortsetzen.

Dieses Beispiel zeigt, dass durch das Prinzip der Extrapolation es grundsätzlich möglich ist, die Ordnung eines Verfahrens durch geschickte Kombination der Ergebnisse $a(h_i)$ zu verbessern. Eine solche Vorgehensweise, bei der Ergebnisse eines numerischen Verfahrens weiter verarbeitet werden, wird *Postprocessing* genannt.

Satz 3.30 (Einfache Extrapolation)**.** *Es sei $a(h) : \mathbb{R}_+ \to \mathbb{R}$ eine $n+1$ mal stetig differenzierbare Funktion mit der Summenentwicklung*

$$a(h) = a_0 + \sum_{j=1}^{n} a_j h^j + a_{n+1}(h)h^{n+1},$$

mit Koeffizienten $a_j \in \mathbb{R}$ sowie $a_{n+1}(h) = a_{n+1} + o(1)$. Weiter sei $(h_k)_{k=0,1,\ldots}$ mit $h_k \in \mathbb{R}_+$ eine monoton fallende Schrittweitenfolge mit der Eigenschaft

$$0 < \frac{h_{k+1}}{h_k} \leq \rho < 1. \tag{3.9}$$

Es sei $p_n^{(k)} \in P_n$ das Interpolationspolynom zu $(h_k, a(h_k)), \ldots, (h_{k+n}, a(h_{k+n}))$. Dann gilt:

$$a(0) - p_n^{(k)}(0) = O(h_k^{n+1}) \quad (k \to \infty).$$

Beweis: In Lagrangescher Darstellung gilt

$$p_n^{(k)}(t) = \sum_{i=0}^{n} a(h_{k+i}) L_{k+i}^{(n)}(t), \quad L_{k+i}^{(n)} = \prod_{l=0, l\neq i}^{n} \frac{t - h_{k+l}}{h_{k+i} - h_{k+l}}. \tag{3.10}$$

Für die Lagrangeschen Basispolynome gilt die Beziehung:

$$\sum_{i=0}^{n} h_{k+i}^r L_{k+i}^{(n)}(0) = \begin{cases} 1 & r = 0, \\ 0 & r = 1, \ldots, n, \\ (-1)^n \prod_{i=0}^{n} h_{k+i} & r = n+1. \end{cases} \tag{3.11}$$

Der Nachweis erfolgt durch Analyse der Lagrangeschen Interpolation von t^r für verschiedene Exponenten r. Wir setzen die Entwicklung von $a(h)$ in die Polynomdarstellung (3.10) ein und erhalten für $t = 0$:

$$p_n^{(k)}(0) = \sum_{i=0}^{n} \left\{ a_0 + \sum_{j=1}^{n} a_j h_{k+i}^j + a_{n+1}(h_{k+i}) h_{k+i}^{n+1} \right\} L_{k+i}^{(n)}(0).$$

Mit (3.11) und $a_{n+1}(h_{k+i}) = a_{n+1} + o(h_{k+i})$ folgt:

$$p_n^{(k)}(0) = a_0 + a_{n+1}(-1)^n \prod_{i=0}^{n} h_{i+k} + \sum_{i=0}^{n} o(1) h_{k+i}^{n+1} L_{k+i}^{(n)}(0).$$

Mit der Schrittweitenbedingung (3.13) $h_{k+i} \leq \rho^i h_k$ gilt für die Lagrangeschen Basispolynome in $t = 0$:

$$|L_{k+i}^{(n)}(0)| = \prod_{l=0, l\neq i}^{n} \left| \frac{1}{\frac{h_{k+i}}{h_{k+l}} - 1} \right| \leq \prod_{l=0, l\neq i}^{n} \left| \frac{1}{\rho^{i-l} - 1} \right| = c(\rho, n). \tag{3.12}$$

Insgesamt folgt mit $h_{k+i} \leq h_k$

$$p_n^{(k)} = a(0) + (-1)^n a_{n+1} h_k^{n+1} + o(h_k^{n+1}).$$

□

Die Schrittweitenbedingung ist notwendig zum Abschätzen von $|L_{k+i}^{(n)}| = O(1)$ und verhindert das starke Oszillieren der Basisfunktionen bei $t = 0$ wie in Beispiel 3.15 beobachtet. Eine zulässige Schrittweitenfolgen zur Extrapolation ist $h_i = \frac{h_0}{2^i}$. Die einfache Wahl $h_i = h_0/i$ hingegen ist nicht zulässig, da hier $h_{k+1}/h_k \to 1$ gilt und eine Abschätzung in Schritt (3.12) nicht mehr gleichmäßig in k möglich ist.

Um die Extrapolationsvorschrift auf ein gegebenes Verfahren anzuwenden, werden die Approximationen $p_n^{(k)}(0)$ mit Hilfe des Neville-Schemas aus Algorithmus berechnet. Wir führen hierzu Beispiel 3.29 fort:

Beispiel 3.31 (Extrapolation des Differenzenquotienten)**.** *Wir approximieren die Ableitung von* $f(x) = \tanh(x)$ *an der Stelle* $x_0 = 0.5$ *und berechnen die Approximationen mit dem einseitigen Differenzenquotienten. Exakter Wert* $\tanh'(0.5) \approx 0.78645$*:*

h	$a(h)$	p_{kk}	$p_{k,k+1}$	$p_{k,k+2}$	$p_{k,k+3}$
2^{-1}	**0.**59895	**0.**59895	**0.78**530	**0.78**836	**0.786**50
2^{-2}	**0.**69213	**0.**69213	**0.78**759	**0.786**73	
2^{-3}	**0.7**3986	**0.7**3986	**0.786**95		
2^{-4}	**0.7**6341	**0.7**6341			

Die Richardson-Extrapolation spielt ihre Stärke erst dann voll aus, wenn die zugrundeliegende Funktion $a(h)$ eine spezielle Reihenentwicklung besitzt, welche z.B. nur gerade

Potenzen von h beinhaltet. Für den zentralen Differenzenquotienten gilt bei hinreichender Regularität von f:

$$a(h) := \frac{f(x_0+h)-f(x_0-h)}{2h} = f'(x_0) + \sum_{i=1}^{n} \frac{f^{(2i+1)}x_0}{(2i+1)!}h^{2i} + O(h^{(2n+1)}).$$

Die Idee ist es nun, diese Entwicklung bei der Interpolation auszunutzen und nur Polynome in $1, h^2, h^4, \ldots$ zu berücksichtigen. Der zentrale Differenzenquotient ist kein Einzelfall. Bei der numerischen Quadratur werden wir das Romberg-Verfahren kennenlernen, welches ebenso auf der Extrapolation einer Vorschrift mit Entwicklung in h^2 beruht. Auch bei der Approximation von gewöhnlichen Differentialgleichungen lernen wir mit dem Gragg'schen Extrapolationsverfahren eine Methode kennen, die auf diesem Prinzip beruht. Daher formulieren wir ohne Beweis denn allgemeinen Satz zur Richardson-Extrapolation:

Satz 3.32 (Richardson-Extrapolation zum Limes)**.** *Es sei $a(h) : \mathbb{R}_+ \to \mathbb{R}$ für ein $q > 0$ eine $q(n+1)$ mal stetig differenzierbare Funktion mit der Summenentwicklung*

$$a(h) = a_0 + \sum_{j=1}^{n} a_j h^{qj} + a_{n+1}(h)h^{q(n+1)},$$

mit Koeffizienten $a_j \in \mathbb{R}$ sowie $a_{n+1}(h) = a_{n+1} + o(1)$. Weiter sei $(h_k)_{k=0,1,\ldots}$ eine monoton fallende Schrittweitenfolge $h_k > 0$ mit der Eigenschaft

$$0 < \frac{h_{k+1}}{h_k} \leq \rho < 1. \tag{3.13}$$

Es sei $p_n^{(k)} \in P_n$ (in h^q) das Interpolationspolynom zu $(h_k^q, a(h_k)), \ldots, (h_{k+n}^q, a(h_{k+n}))$. Dann gilt:

$$a(0) - p_n^{(k)}(0) = O(h_k^{q(n+1)}) \quad (k \to \infty).$$

Beweis: Der Beweis ist eine einfache Modifikation von Satz 3.31 und kann im Wesentlichen mit Hilfe der Substitution $\tilde{h}_k = h_k^q$ übertragen werden. Für Details, siehe [9]. □

Prinzipiell erlaubt die Richardson-Extrapolation bei hinreichender Regularität eine beliebige Steigerung der Verfahrensordnung. Üblicherweise werden jedoch nur einige wenige Extrapolationsschritte aus den letzten Verfahrenswerten $h_k, h_{k+1}, \ldots, h_{k+n}$ verwendet. Die Extrapolation wird am einfachsten mit dem modifizierten Neville-Schema durchgeführt.

Algorithmus 3.33 (Modifiziertes Neville-Schema zur Extrapolation)**.** *Für $h_0, h_1, \ldots, h_n$ sei $a(h_i)$ bekannt. Berechne:*

1 $\quad$ *für* $k = 0, \ldots, n$ *setze* $a_{k,k} := a(h_k)$
2 $\quad$ *für* $l = 1, \ldots, n$ *und*
3 $\quad\quad$ *für* $k = 0, \ldots, n-l$ *berechne*
4 $\quad\quad\quad$ $a_{k,k+l} := a_{k,k+l-1} - \frac{a_{k+1,k+l} - a_{k,k+l-1}}{\frac{h_{k+l}^q}{h_k^q} - 1}$

Wir schließen die Richardson-Extrapolation mit einem weiteren Beispiel ab:

Beispiel 3.34 (Extrapolation des zentralen Differenzenquotienten)**.** *Wir approximieren die Ableitung von* $f(x) = \tanh(x)$ *an der Stelle* $x = 0.5$ *mit dem zentralen Differenzenquotienten und Extrapolation. Exakter Wert ist* $\tanh'(0.5) \approx 0.7864477329$*:*

h	$a(h) = p_{kk}$	$p_{k,k+1}$	$p_{k,k+2}$	$p_{k,k+3}$
2^{-1}	**0.7**61594156	**0.786**7493883	**0.7864**537207	**0.7864477**443
2^{-2}	**0.78**0460580	**0.7864**721999	**0.786447**8377	
2^{-3}	**0.78**4969295	**0.78644**93603		
2^{-4}	**0.786**079344			

Bereits die jeweils erste Extrapolation $p_{k,k+1}$ *liefert weit bessere Ergebnisse als die Extrapolation des einseitigen Differenzenquotienten. Die beste Approximation verfügt über 8 richtige Stellen.*

3.5 Numerische Integration

Die numerische Integration (oder auch numerische Quadratur) dient zur approximativen Berechnung von Integralen. Mögliche Gründe sind

- Die Stammfunktion eines Integrals lässt sich nicht durch eine elementare Funktion ausdrücken. Z.B.
$$\int_0^\infty \exp(-x^2)\,\mathrm{d}x, \quad \int_0^\infty \frac{\sin(x)}{x}\,\mathrm{d}x.$$
- Eine Stammfunktion existiert in geschlossener Form, aber die Berechnung ist derart aufwendig, so dass numerische Methoden vorzuziehen sind.
- Der Integrand ist nur an diskreten Stellen bekannt; beispielsweise bei Messreihendaten.

Bei der numerischen Integration basieren die Methoden auf den bereits kennengelernten Interpolatonsmethoden. Sie sind somit eine klassische Anwendung der der Polynom-Interpolation sowie Spline-Interpolation. Erstere führen auf die sogenannten *interpolatorischen Quadraturformeln* während die Spline-Interpolation auf die *stückweise interpolatorische Quadratur* (die in der Literatur auch häufig unter dem Namen der zusammengesetzten oder summierten Quadratur zu finden ist).

Bei der interpolatorischen Quadratur einer Funktion $f(x)$ auf dem Intervall $[a, b]$ wird zunächst eine Interpolationspolynom zu gegebenen Stützstellen $x_0, x_1, \ldots, x_n$ kreiert, welches dann integriert wird (basiert dementsprechend auf Abschnitt 3.1). Bei der Gauß Quadratur wird die Position der Stützstellen $x_i \in [a, b]$ im Intervall so bestimmt, dass die resultierende Integrationsformel eine *optimale* Ordnung erzielt. Zuletzt betrachten wir als Anwendung der Extrapolation zum Limes (Abschnitt 3.4), dass sogenannte Romberg'sche Integrationsverfahren in Abschnitt 3.5.4.

Definition 3.35 (Quadraturformel)**.** *Es sei $f \in C[a,b]$. Unter einer* numerischen Quadraturformel *zur Approximation von $I(f) := \int_a^b f(x)\,dx$ verstehen wir die Vorschrift:*

$$I^n(f) := \sum_{i=0}^{n} \alpha_i f(x_i),$$

mit $n+1$ paarweise verschiedenen Stützstellen $x_0, \ldots, x_n$ *sowie $n+1$* Quadraturgewichten $\alpha_0, \ldots, \alpha_n$.

Zunächst bringen wir zur Veranschaulichung einige einfache Beispiele:

Definition 3.36 (Boxregel)**.** *Die Box-Regel zur Integration auf $[a,b]$ ist die einfachste Quadraturformel und basiert auf Interpolation von $f(x)$ mit einem konstanten Polynom $p(x) = f(a)$ und Integration von $p(x)$ (siehe Abbildung 3.8):*

$$I^0(f) = (b-a)f(a).$$

Neben dieser linksseitigen Boxregel *existiert mit $I^0(f) = (b-a)f(b)$ auch die* rechtsseitige Boxregel.

Die Boxregel hat ihre Bedeutung bei der Herleitung des Riemann-Integrals. Die Boxregel ist vergleichbar mit dem einseitigen Differenzenquotienten. Eine bessere Quadraturformel erhalten wir durch Ausnutzen von Symmetrieeigenschaften:

Definition 3.37 (Mittelpunktsregel)**.** *Zur Herleitung der Mittelpunktregel wird die Funktion $f(x)$ in der Mitte des Intervalls mit einem konstanten Polynom interpoliert:*

$$I^0(f) = (b-a)f\left(\frac{a+b}{2}\right).$$

Siehe Abbildung 3.8.

Diese beiden Regeln basieren auf einer Interpolation von Grad Null. Zur Herleitung der Trapezregel wird $f(x)$ in den beiden Intervallenden linear interpoliert:

Definition 3.38 (Trapezregel)**.** *Die Trapezregel ist durch Integration der linearen Interpolation in $(a, f(a))$ sowie $(b, f(b))$ gebildet:*

$$I^1(f) = \frac{b-a}{2}(f(a) + f(b)).$$

Siehe auch Abbildung 3.8

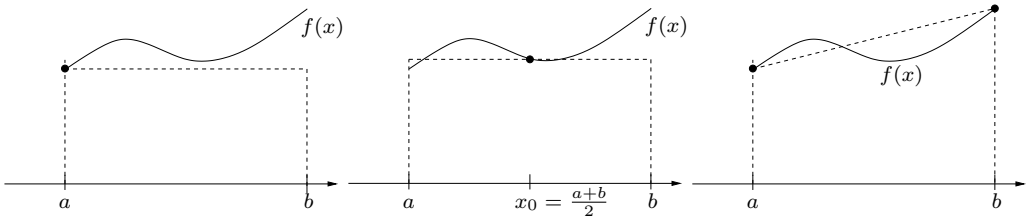

Abbildung 3.8: Linksseitige Boxregel, Mittelpunktsregel und Trapezregel zur Integralapproximation.

3.5.1 Interpolatorische Quadratur

Die interpolatorischen Quadraturformeln werden über die Konstruktion eines geeigneten Interpolationspolynoms hergeleitet. Zu den gegebenen Stützstellen $a \leq x_0 < \ldots < x_n \leq b$ wird gemäß Abschnitt 3.1 das Lagrangsche Interpolationspolynom als Approximation der Funktion f gebildet:

$$p_n(x) = \sum_{i=0}^{n} f(x_i) L_i^{(n)}(x).$$

Dieses wird dann integriert:

$$I^{(n)}(f) := \int_a^b p_n(x)\,\mathrm{d}x = \sum_{i=0}^{n} f(x_i) \underbrace{\int_a^b L_i^{(n)}(x)\,\mathrm{d}x}_{=\alpha_i}.$$

Die Quadraturgewichte

$$\alpha_i = \int_a^b L_i^{(n)}(x)\,\mathrm{d}x \tag{3.14}$$

hängen offensichtlich nicht von der zu integrierenden Funktion $f(x)$ ab, dafür vom Intervall $[a, b]$ sowie von den Stützstellen $x_0, \ldots, x_n$. Dies impliziert die Frage, ob durch geschickte Verteilung der Stützstellen die Qualität der Gewichte verbessert werden kann. (Als Vorwegnahme auf den nach-nachfolgenden Abschnitt kann diese Frage in der Tag mit *Ja* beantwortet werden).

Bevor wir einzelne Quadratur-Formeln analysieren und nach möglichst leistungsfähigen Formeln suchen, können wir zunächst ein einfaches aber doch allgemeines Resultat herleiten:

Satz 3.39 (Lagrange-Quadratur)**.** *Für die interpolatorischen Quadraturformeln $I^n(f)$ mit $n+1$ paarweise verschiedenen Stützstellen $x_0, x_1, \ldots, x_n$ gilt zur Approximation des Integrals $I(f) = \int_a^b f(x)\,dx$ die Fehlerdarstellung*

$$I(f) - I^n(f) = \int_a^b f[x_0, \ldots, x_n, x] \prod_{j=0}^{n} (x - x_j)\,dx,$$

mit Newtonscher Restglieddarstellung.

Beweis: Der Beweis folgt unmittelbar durch Integration der entsprechenden Fehlerabschätzung für die Lagrange-Interpolation in Satz 3.12. □

Hieraus folgt eine wichtige Eigenschaft der interpolatorischen Quadraturformeln:

Korollar 3.40. *Die interpolatorische Quadraturformel $I^{(n)}(\cdot)$ ist exakt für alle Polynome vom Grad n.*

Beweis: Folgt direkt aus der Konstruktion der interpolatorischen Quadraturformeln, da für jedes $f \in P_n$ sofort $p = f$ gilt. □

Die Integrierbarkeit von Polynomen wird genutzt, um die Ordnung von Quadraturregeln zu charakterisieren:

Definition 3.41 (Ordnung von Quadraturregeln)**.** *Eine Quadraturformel $I^{(n)}(\cdot)$ wird (mindestens) von der Ordnung m genannt, wenn durch sie mindestens alle Polynome aus P_{m-1} exakt integriert werden. D.h. die interpolatorischen Quadraturformeln $I^{(n)}(\cdot)$ zu $n+1$ Stützstellen sind mindestens von der Ordnung $n+1$.*

Im Folgenden werden wir die bereits eingeführten einfachen Quadraturformeln näher analysieren und ihre Fehlerabschätzung sowie Ordnung bestimmen. Hierzu werden wir die Newtonsche Integraldarstellung des Interpolationsfehlers aus Satz 3.39 nutzen.

Satz 3.42 (Boxregel)**.** *Es sei $f \in C^1[a,b]$. Die Boxregel (Definition 3.36) ist von erster Ordnung und es gilt die Fehlerabschätzung:*

$$I^0(f) := (b-a)f(a), \quad I(f) - I^0(f) = \frac{(b-a)^2}{2} f'(\xi),$$

mit einer Zwischenstelle $\xi \in (a,b)$.

Beweis: Die Boxregel basiert auf der Interpolation mit einem konstanten Polynom, hat daher erste Ordnung. Aus Satz 3.39 folgt mit $x_0 = a$:

$$I(f) - I^0(f) = \int_a^b f[a,x](x-a)\,\mathrm{d}x,$$

wobei mit Satz 3.12 weiter gilt:

$$I(f) - I^0(f) = \int_a^b \int_0^1 f'(a + t(x-a))(x-a)\,\mathrm{d}t\,\mathrm{d}x.$$

Wir wenden den Mittelwertsatz der Integralrechnung zweimal an und erhalten

$$I(f) - I^0(f) = f'(\xi) \int_a^b \int_0^1 (x-a)\,\mathrm{d}t\,\mathrm{d}x = \frac{1}{2} f'(\xi)(b-a)^2,$$

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

Die Boxregel ist die einfachste Quadraturformel. In Analogie zur Diskussion bei den Differenzenquotienten können wir bei der Mittelpunktsregel durch geschickte Ausnutzung der Symmetrie eine bessere Approximationsordnung erwarten. Um diese bessere Ordnung zu erreichen müssen wieder Superapproximationeigenschaften genutzt werden, welche im Allgemeinen etwas Mehraufwand erfordern:

Satz 3.43 (Mittelpunktsregel)**.** *Es sei $f \in C^2[a,b]$. Die Mittelpunktsregel (Definition 3.37) ist von Ordnung 2 und es gilt die Fehlerabschätzung:*

$$I^0(f) := (b-a) f\left(\frac{a+b}{2}\right), \quad I(f) - I^0(f) = \frac{(b-a)^3}{24} f''(\xi),$$

mit einer Zwischenstelle $\xi \in (a,b)$.

Beweis: Aufgrund der Interpolation mit konstantem Polynom ist zunächst ist nur erste Ordnung zu erwarten. Es gilt mit Satz 3.4:

$$I(f) - I^0(f) = \int_a^b f\left[\frac{a+b}{2}, x\right]\left(x - \frac{a+b}{2}\right) \mathrm{d}x.$$

Angenommen $\bar{f}(x)$ ein lineares Polynom. Dann ist $\bar{f}'$ konstant und also $\bar{f}\,[(a+b)/2, x] = \bar{f}[(a+b)/2, \bar{x}]$ mit $\bar{x} \in [a,b]$. Aus Symmetriegründen folgt:

$$I(r\bar{f}) - I^0(\bar{f}) = \underbrace{\bar{f}[(a+b)/2, \bar{x}]}_{\text{konstant}} \underbrace{\int_a^b \left(x - \frac{a+b}{2}\right) \mathrm{d}x}_{=0} = 0.$$

Zur Herleitung der Fehlerabschätzung soll diese Argumentation weiter genutzt werden. Das Restglied einer beliebigen linearen Funktion $\bar{f}$ verschwindet. Daher können wir ein beliebiges $\bar{f}$ (welches wir weiter unten spezifizieren) in die Fehleridentität (3.15) für allgemeines $f \in C^2([a,b])$ einschieben:

$$\begin{aligned} I(f) - I^0(f) &= \int_a^b f\left[\frac{a+b}{2}, x\right]\left(x - \frac{a+b}{2}\right) \mathrm{d}x \\ &= \int_a^b \left(f\left[\frac{a+b}{2}, x\right] - \bar{f}\left[\frac{a+b}{2}, x\right]\right)\left(x - \frac{a+b}{2}\right) \mathrm{d}x. \end{aligned}$$

Es gilt mit Taylor-Entwicklung von f' um den Mittelwert:

$$\begin{aligned} f\left[\frac{a+b}{2}, x\right] &= \int_0^t f'\left(\frac{a+b}{2} + t\left(x - \frac{a+b}{2}\right)\right) \mathrm{d}t \\ &= \int_0^t \left(f'\left(\frac{a+b}{2}\right) + t\left(x - \frac{a+b}{2}\right) f''(\xi)\right) \mathrm{d}t, \end{aligned}$$

mit einem Mittelwert ξ. Wir wählen nun $\bar{f}$ so, dass $\bar{f}[(a+b)/2, x] = f'((a+b)/2)$. Dann folgt (mit nochmaliger Anwendung des Mittelwertsatzes der Integralrechnung):

$$I(f) - I^0(f) = f''(\xi) \int_a^b \int_0^1 t \left(x - \frac{a+b}{2} \right)^2 \mathrm{d}t \, \mathrm{d}x = \frac{(b-a)^3}{24} f''(\xi).$$

□

Die höhere Ordnung der Mittelpunktsregel erhalten wir nur durch Einschieben der Null in Form einer linearen Funktion $\bar{f}$. Mit dem gleichen Aufwand wie die Boxregel, also mit einer Auswertung der Funktion $f(x)$ erreicht die Mittelpunktsregel die doppelte Ordnung.

Satz 3.44 (Trapezregel)**.** *Es sei $f \in C^2([a,b])$. Die Trapezregel (Definition 3.38) ist von Ordnung 2 und es gilt die Fehlerabschätzung:*

$$I^1(f) := \frac{b-a}{2}(f(a) + f(b)), \quad I(f) - I^1(f) = \frac{(b-a)^3}{12} f''(\xi),$$

mit einer Zwischenstelle $\xi \in (a,b)$.

Beweis: Mit Satz 3.4 gilt:

$$I^1(f) - I(f) = \int_a^b f[a,b,x](x-a)(x-b) \, \mathrm{d}x. \tag{3.15}$$

Für $f[a,b,x]$ gilt bei zweimaliger Anwendung des Mittelwertsatzes der Integralrechnung:

$$f[a,b,x] = \int_0^1 \int_0^{t_1} f''(a + t_1(b-a) + t(x-b)) \, \mathrm{d}t \, \mathrm{d}t_1 = \frac{1}{2} f''(\xi_{a,b,x}),$$

mit einem Zwischenwert ξ_x. Hiermit erhalten wir mit nochmaliger Anwendung des Mittelwertsatzes (da $(x-a)(x-b) \leq 0$) die Restgliedabschätzung:

$$I(f) - I^1(f) = \frac{1}{2} \int_a^b f''(\xi_x)(x-a)(x-b) \, \mathrm{d}x = \frac{(b-a)^3}{12} f''(\xi),$$

mit einem weiteren Zwischenwert $\xi \in (a,b)$. □

Die Verwendung eines quadratischen Interpolationspolynoms führt auf die Simpson Regel:

Satz 3.45 (Simpson-Regel)**.** *Es sei $f \in C^4[a,b]$. Die* Simpson-Regel*, basierend auf Interpolation mit quadratischem Polynom:*

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

ist von Ordnung vier und es gilt:

$$\int_a^b f(x) = I(f) - I^2(f) = \frac{f^{(4)}(\xi)}{2880} (b-a)^5,$$

wobei $\xi \in (a,b)$.

Beweis: Der Beweis folgt analog zu den Beweisen zu Mittelpunkts- und Trapezregel und sei dem Leser als Übung gestellt. Dabei müssen wieder das Superapproximationsprinzip ausgenutzt werden. □

Bemerkung 3.46. *Wie bei der Mittelpunktsregel ist die Ordnung der Simpson-Regel eine Potenz höher als zu erwarten wäre. Treibt man dieses Spiel weiter und konstruiert noch höhere Quadraturformeln, dann erkennen wir ein allgemeines Prinzip: bei Quadratur mit geraden Interpolationspolynomen, wird die Fehlerordnung um eine Potenz erhöht. Dies folgt jeweils aus Symmetriegründen.*

Bei allen bisherigen Quadraturformeln sind die Stützstellen gleichmäßig im Intervall $[a, b]$ verteilt. Diese Formeln werden *Newton-Cotes-Formeln* genannt:

Definition 3.47 (Newton-Cotes-Formeln)**.** *Quadraturregeln mit äquidistant im Intervall $[a, b]$ verteilten Stützstellen heißen* Newton-Cotes-Formeln*. Gehören die Intervallenden a sowie b zu den Stützstellen, so heißen die Formeln* abgeschlossen*, ansonsten* offen*.*

In Tabelle 3.2 fassen wir einige gebräuchliche Newton-Cotes Formeln zusammen. Ab $n \geq 7$ bei den abgeschlossenen Formeln und ab $n = 2$ bei den offenen Formeln, treten negative Quadraturgewichte α_i auf. Dies führt zu dem Effekt, dass auch bei rein positiven Integranden Auslöschung auftreten kann. Daher sind diese Formeln aus numerischer Sicht nicht mehr anwendbar.

Tabelle 3.2: Eigenschaften und Gewichte der Newton-Cotes Formeln.

n	Gewichte α_i		Name
0	1	offen	Boxregel
1	$\frac{1}{2}, \frac{1}{2}$	geschlossen	Trapezregel
2	$\frac{1}{6}, \frac{4}{6}, \frac{1}{6}$	geschlossen	Simpson-Regel
3	$\frac{3}{24}, \frac{9}{24}, \frac{9}{24}, \frac{3}{24}$	geschlossen	Newton's 3/8-Regel
4	$\frac{14}{180}, \frac{64}{180}, \frac{24}{180}, \frac{64}{180}, \frac{14}{180}$	geschlossen	Milne's-Regel

3.5.2 Stückweise interpolatorische Quadraturformeln

Die interpolatorische Quadratur aus dem vorangegangenen Abschnitt beruht auf der Integration eines Interpolationspolynoms p. Für dieses gilt die Fehlererabschätzung (Satz 3.11):

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

Die Genauigkeit der Quadratur kann prinzipiell auf zwei Wege verbessert werden: der Vorfaktor kann durch Wahl von mehr Stützstellen klein gehalten werden, denn es gilt

$1/(n+1)! \to 0$. Dies führt jedoch nur dann zur Konvergenz des Fehlers, falls die Ableitungen $f^{(n)}$ nicht zu schnell wachsen, siehe Beispiel 3.15. Als zweite Option bleibt das Produkt der Stützstellen. Eine einfache Abschätzung besagt wegen $x, x_0, \ldots, x_n \in [a,b]$:

$$\left| \prod_{i=0}^{n} (x - x_i) \right| \leq (b-a)^{n+1}.$$

Interpolationsfehler (und somit auch Quadraturfehler) lassen sich durch Potenzen der Intervallgröße beschränken. Dies wird zur Entwicklung von stabilen und konvergenten Quadraturformeln entsprechend dem Vorgehen der summierten Interpolation (Abschnitt 3.2) genutzt. Hierzu sei

$$a = y_0 < y_1 < \cdots < y_m = b, \quad h_i := y_i - y_{i-1},$$

eine Zerlegung des Intervalls $[a,b]$ in m Teilintervalle mit Schrittweite h_i. Auf jedem dieser m Teilintervalle wird die Funktion f mit Hilfe einer Quadraturformel approximiert:

$$I_h^n(f) = \sum_{i=1}^{m} I^n_{[y_{i-1}, y_i]}(f).$$

Aus Satz 3.39 folgt sofort eine allgemeine Fehlerabschätzung für die summierten Quadraturformeln

Satz 3.48 (Summierte Quadratur für n ungerade)**.** *Es sei $f \in C^{n+1}([a,b])$ sowie $a = y_0 < \cdots < y_m = b$ eine Zerlegung des Intervalls mit Schrittweiten $h_i := y_i - y_{i-1}$ sowie $h := \max h_i$. Für die die summierte Quadraturformeln $I_h^n(f)$ gilt die Fehlerabschätzung:*

$$I(f) - I_h^n(f) \leq c \max_{[a,b]} |f^{(n+1)}| h^{n+1},$$

mit einer Konstante $c(n) > 0$.

Beweis: Der Beweis folgt einfach aus Kombination von Satz 3.39 sowie der differentiellen Restglieddarstellung der Lagrange-Interpolation. □

Satz 3.49 (Summierte Quadratur für n gerade)**.** *Es sei $f \in C^{n+1}([a,b])$ sowie $a = y_0 < \cdots < y_m = b$ eine Zerlegung des Intervalls mit Schrittweiten $h_i := y_i - y_{i-1}$ sowie $h := \max h_i$. Für die die summierte Quadraturformeln $I_h^n(f)$ gilt die Fehlerabschätzung:*

$$I(f) - I_h^n(f) \leq c \max_{[a,b]} |f^{(n+2)}| h^{n+2},$$

mit einer Konstante $c(n) > 0$.

Beweis: Der Beweis folgt einfach aus Kombination von Satz 3.39 sowie der differentiellen Restglieddarstellung der Lagrange-Interpolation. □

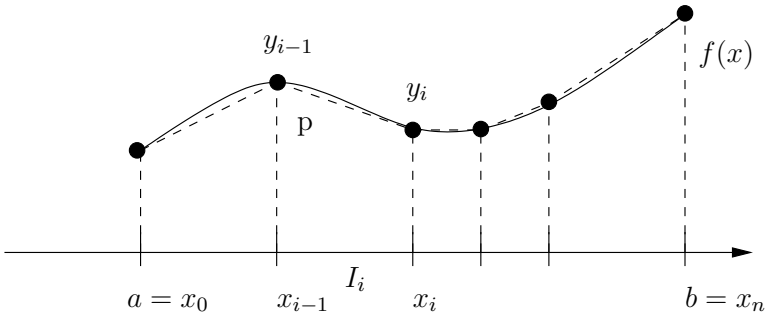

Abbildung 3.9: Summierte Trapezregel zur Integralapproximation.

Bemerkung 3.50. *Von den einfachen Quadraturregeln überträgt sich demnach die verbesserte Konvergenzordnung bei geraden n auf die summierten Quadraturformeln.*

Aus diesen Resultaten kann ein wichtiges Ergebnis abgelesen werden: für $h \to 0$, also für steigende Feinheit der Intervallzerlegung konvergiert die summierte Quadratur für beliebiges n. Mit Hilfe von summierten Quadraturregeln lassen sich somit auch Funktionen mit schnell wachsenden Ableitungen integrieren. Darüber hinaus eignen sich summierte Regeln auch zur Integration von Funktionen, die nur eine geringe Regularität, etwa $f \in C^1([a,b])$ aufweisen. Da es möglich ist die Ordnung n klein zu halten sind summierte Quadraturregeln numerisch stabiler. Zur Veranschaulichung betrachten wir in Abbildung 3.9 die summierte Trapezregel

Im folgenden konkretisieren wir einige einfache summierte Quadraturregeln. Hierzu betrachten wir ausschließlich äquidistante Zerlegungen des Intervalls $[a,b]$:

$$a = y_0 < \cdots < y_m = b, \quad h := \frac{b-a}{m}, \quad y_i := a + ih, \quad i = 0, \ldots, m. \tag{3.16}$$

Satz 3.51 (Summierte Boxregel)**.** *Es $f \in C^1[a,b]$ sowie durch (3.16) eine äquidistante Zerlegung des Intervalls gegeben. Für die summierte Boxregel*

$$I_h^0(f) = h \sum_{i=1}^{m} f(y_i),$$

gilt die Fehlerabschätzung:

$$|I(f) - I_h^0(f)| \leq \frac{b-a}{2} h \max_{[a,b]} |f'|.$$

Beweis: Aus Satz 3.42 folgern wir durch Summation über die Teilintervalle

$$I(f) - I_h^0(f) = \sum_{i=1}^{m} \frac{h^2}{2} f'(\xi_i),$$

mit Zwischenstellen $\xi_i \in [y_{i-1}, y_i]$. Das Ergebnis folgt nun mit $(b-a) = \sum_i h_i$ sowie durch Übergang zum Maximum. □

Mit der summierten Boxregel haben wir also sogar für nur stückweise stetig differenzierbare Funktionen eine konvergente Quadraturformel für $h \to 0$. Entsprechend gilt für die Trapezregel:

Satz 3.52 (Summierte Trapezregel)**.** *Es $f \in C^2[a,b]$ sowie durch (3.16) eine äquidistante Zerlegung des Intervalls gegeben. Für die summierte Trapezregel*

$$I_h^1(f) = \frac{h}{2}\sum_{i=1}^{m}(f(y_{i-1}) + f(y_i)) = \frac{h}{2}f(a) + h\sum_{i=1}^{m-1} f(y_i) + \frac{h}{2}f(b)$$

gilt die Fehlerabschätzung:

$$|I(f) - I_h^1(f)| \leq \frac{b-a}{12}h^2 \max_{[a,b]} |f''|.$$

Beweis: Übung! □

Die summierte Trapezregel ist besonders attraktiv, da sich die Stützstellen überschneiden: der Punkt y_i ist sowohl Stützstelle im Teilintervall $[y_{i-1}, y_i]$ als auch $[y_i, y_{i+1}]$ und $f(y_i)$ muss nur einmal bestimmt werden. Darüber hinaus eignet sich die summierte Trapezregel als Grundlage von *adaptiven Quadraturverfahren*, bei denen die Genauigkeit Stück für Stück dem Problem angepasst wird: wurde auf einer Zerlegung zur Schrittweite h die Approximation $I_h^1(f)$ bestimmt, so kann die Genauigkeit durch Intervallhalbierung $I_{h/2}^1(f)$ einfach gesteigert werden. Alle Stützstellen $f(y_i)$ können weiter verwendet werden, die Funktion $f(x)$ muss lediglich in den neuen Intervallmitten berechnet werden. Ein entsprechendes Resultat gilt für die summierte Simpsonregel:

$$\begin{aligned} I_h^2(f) &= \sum_{i=1}^{m} \frac{h}{6r}\left(f(y_{i-1}) + 4f\left(\frac{y_{i-1}+y_i}{2}\right) + f(y_i)\right) \\ &= \frac{h}{6}f(a) + \sum_{i=1}^{m}\frac{h}{3}f(y_i) + \sum_{i=1}^{m}\frac{2h}{3}f\left(\frac{y_{i-1}+y_i}{2}\right) + \frac{h}{6}f(b). \end{aligned}$$

Bei geschickter Anordnung sind nur $2m+2$ statt $3m$ Funktionsauswertungen notwendig.

3.5.3 Gauß-Quadratur

Wie bereits diskutiert, sind die interpolatorischen Quadraturformeln

$$I^{(n)}(f) = \sum_{i=0}^{n} \alpha_i f(x_i)$$

zu den Stützstellen $x_0, \ldots, x_n \in [a, b]$ nach Konstruktion mindestens von der Ordnung $n + 1$. D.h.:

$$I(p) = I^{(n)}(p) \quad \forall p \in P_n.$$

Wir haben jedoch mit der Mittelpunktsregel $n = 0$ und der Simpsonregel $n = 2$ bereits Quadraturformeln kennengelernt, die exakt sind für alle Polynome P_{n+1}, die also von der Ordnung $n + 2$ sind.

Wir untersuchen in diesem Abschnitt die Frage, ob die Ordnung mit anderen bisher noch nicht kennengelernten Methoden weiter erhöht werden kann. Bisher lag die Freiheit lediglich in der Wahl des Polynomgrads und somit in der Anzahl der Stützstellen. Die Frage nach der optimalen Verteilung der Stützstellen im Intervall $[a, b]$ wurde bisher nicht untersucht. In diesem Abschnitt werden wir mit der *Gauß-Quadratur* interpolatorische Quadraturformeln kennenlernen, welche durch optimale Positionierung der Stützstellen die maximale Ordnung $2n + 2$ erreichen.

Satz 3.53 (Ordnungsbarriere der Quadratur)**.** *Eine interpolatorische Quadraturformel zu $n + 1$ Stützstellen kann höchstens die Ordnung $2n + 2$ haben.*

Beweis: Angenommen $I^{(n)}(\cdot)$ mit den Stützstellen $x_0, \ldots, x_n$ wäre von höherer Ordnung, insbesondere exakt für Polynome P_{2n+2}, also für

$$p(x) = \prod_{j=0}^{n} (x - x_j)^2 \in P_{2n+2}.$$

Die eindeutige Interpolierende $p_n \in P_n$ mit $p_n(x_i) = p(x_i) = 0$ hat $n + 1$ Nullstellen und stellt somit die Nullfunktion dar. Dies ergibt einen Widerspruch:

$$0 < \int_a^b p(x)\, dx = I^{(n)}(p) = I^{(n)}(p_n) = 0.$$

□

Im Folgenden untersuchen wir interpolatorische Quadraturregeln genauer und versuchen Bedingungen herzuleiten, unter denen die höchst mögliche Ordnung $2n + 2$ wirklich erreicht werden kann. Wir müssen demnach eine Quadraturformel in $n + 1$ Stützstellen finden, welche exakt ist für alle Polynome aus P_{2n+1}. Hierzu wählen wir zunächst eine Quadraturformel mit den $2n + 2$ Stützstellen $x_0, x_1, \ldots, x_n, x_{n+1}, \ldots, x_{2n+1}$. Von dieser wissen wir, dass sie auf jeden Fall der Ordnung $2n + 2$ ist. Es gilt in Newtonscher Darstellung des Interpolationspolynoms

$$\begin{aligned} I(f) - I^{2n+1}(f) &= I(f) - \sum_{i=0}^{2n+1} f[x_0, \ldots, x_i] \int_a^b \prod_{j=0}^{i-1} (x - x_j)\, \mathrm{d}x \\ &= I(f) - I^n(f) - \sum_{i=n+1}^{2n+1} f[x_0, \ldots, x_i] \int_a^b \prod_{j=0}^{i-1} (x - x_j)\, \mathrm{d}x, \end{aligned}$$

wobei $I^n(\cdot)$ diejenige Quadraturregel ist, welche durch die ersten $n+1$ Stützstellen $x_0, \ldots, x_n$ gegeben ist. Wir versuchen nun die Stützstellen $x_0, \ldots, x_{2n+1}$ so zu bestimmen, dass das Restglied

$$\sum_{i=n+1}^{2n+1} f[x_0, \ldots, x_i] \int_a^b \prod_{j=0}^{i-1} (x - x_j) \, \mathrm{d}x = 0$$

für alle Funktionen $f \in C^{2n+2}([a,b])$ verschwindet. In diesem Fall gilt:

$$I(f) - I^{2n+1}(f) = I(f) - I^n(f) \quad \Rightarrow \quad I^{2n+1}(f) = I^n(f).$$

und die resultierende Formel wäre von Ordnung $2n+2$ bei nur $n+1$ Stützstellen. Es gilt für $n < i < 2n+2$:

$$\int_a^b \underbrace{\prod_{j=0}^{i-1} (x - x_j)}_{\in P_{2n+1}} \, \mathrm{d}x = \int_a^b \underbrace{\prod_{j=0}^{n} (x - x_j)}_{\in P_{n+1}} \underbrace{\prod_{j=n+1}^{i-1} (x - x_j)}_{P_n} \, \mathrm{d}x.$$

Die Polynome aus dem zweiten Produkt

$$\left\{ 1, x - x_{n+1}, (x - x_{n+1})(x - x_{n+2}), \ldots, \prod_{j=n+1}^{2n} (x - x_j) \right\},$$

bilden eine Basis des Polynomraums P_n. Ziel ist es, die ersten Stützstellen $x_0, \ldots, x_n$ so zu positionieren, dass das Integral

$$\int_a^b \prod_{i=0}^{n} (x - x_i) q(x) \, \mathrm{d}x = 0 \quad \forall q \in P_n \tag{3.17}$$

für alle q aus P_n verschwindet. Es ist zunächst unklar, ob eine solche Stützstellenwahl überhaupt möglich ist. Mit

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

schreibt sich die Aufgabe kurz:

$$\text{Suche } p_{n+1} \in P_{n+1} : \quad (p_{n+1}, q)_{L^2([a,b])} = 0 \quad \forall q \in P_n,$$

mit dem L^2-Skalarprodukt:

$$(f, g)_{L^2([a,b])} := \int_a^b f(x) \, g(x) \, \mathrm{d}x.$$

Geometrisch bedeutet diese Aufgabe: suche ein Polynom $p_{n+1} \in P_{n+1}$, welches auf allen Polynomen $q \in P_n$ orthogonal steht.

Bevor wir die allgemeine Lösbarkeit dieser Aufgabe betrachten untersuchen wir zunächst als einfaches Beispiel die Fälle $n = 0$ sowie $n = 1$:

Beispiel 3.54 (Optimale Stützstellenwahl durch Orthogonalisierung)**.** *Wir wollen die Polynome $p_1(x) = (x - x_0)$ sowie $p_2(x) = (x - x_0)(x - x_1)$ bestimmen, so dass sie im L^2-Skalarprodukt orthogonal auf allen Polynomen von Grad 0 bzw. 1 stehen. Ohne Einschränkung betrachten wir das Intervall $[-1, 1]$:*

Für $n = 0$ gilt mit $q \equiv \alpha_0 \in P_0$

$$0 \stackrel{!}{=} (x - x_0, \alpha_0)_{L^2([-1,1])} = -2x_0\alpha_0 \quad \forall \alpha_0 \in \mathbb{R},$$

also muss $x_0 = 0$ gelten.

Im Fall $n = 1$ gilt mit $q = \alpha_0 + \beta_0 x$:

$$0 \stackrel{!}{=} ((x - x_0)(x - x_1), \alpha_0 + \beta_0 x)_{L^2[-1,1]} = \frac{-2(x_0 + x_1)}{3}\beta_0 + \frac{2(1 + 3x_0x_1)}{3}\alpha_0 \quad \forall \alpha_0, \beta_0 \in \mathbb{R}.$$

Aus $\alpha_0 = 1$ sowie $\beta_0 = 0$ folgern wir $x_0 = -x_1$ und aus $\alpha_0 = 0$ und $\beta_0 = 1$ folgt hiermit $x_{1/2} = \pm\frac{1}{\sqrt{3}}$. Die optimalen Gewichte bestimmen wir gemäß Formel (3.14) zu:

$$\alpha_0^0 = \int_{-1}^{1} 1 \, dx = 2, \quad \alpha_0^1 = \int_{-1}^{1} \frac{x - \frac{-1}{\sqrt{3}}}{\frac{1}{\sqrt{3}} - \frac{-1}{\sqrt{3}}} \, dx = 1, \quad \alpha_1^1 = \int_{-1}^{1} \frac{x - \frac{1}{\sqrt{3}}}{\frac{-1}{\sqrt{3}} - \frac{1}{\sqrt{3}}} \, dx = 1,$$

und wir erhalten die beiden ersten Gauß-Quadraturformeln:

$$I_G^0(f) = 2f(0), \quad I_G^1(f) = f\left(-\frac{1}{\sqrt{3}}\right) + f\left(\frac{1}{\sqrt{3}}\right).$$

Die erste Formel ist gerade die Mittelpunktsregel, von der wir bereits die Ordnung zwei (also $2n + 2 = 2 \cdot 0 + 2$) kennen. Die zweite Formel hat die Ordnung 4, dies kann einfach durch Integration der Monome $1, x, x^2, x^3$ überprüft werden.

Gauß-Legendre-Quadratur

In diesem Abschnitt werden allgemeine Resultate zur Existenz und Eindeutigkeit von Gauß'schen Quadraturregeln $I_G^n(f)$ untersucht. Aus Konventionsgründen betrachtet man dabei das Intervall $[-1, 1]$ (zur Transformation auf allgemeine Intervalle sei auf Bemerkung 3.71 verwiesen).

Definition 3.55 (Gauß-Quadratur)**.** *Eine Quadraturformel zur Integration einer Funktion $f : [a, b] \to \mathbb{R}$ mit*

$$I_G^n(f) = \sum_{k=0}^{n} a_k f(x_k)$$

mit $n+1$ Quadratur-Stützstellen wird Gauß-Quadraturformel genannt, falls alle Polynome $p \in P_{2n+1}$ exakt integriert werden, d.h. falls gilt:

$$\int_{-1}^{1} p(x) \, dx = \sum_{k=0}^{n} a_k p(x_k) \quad \forall p \in P_{2n+1}.$$

Nach dieser Definition sind die beiden Quadraturregeln aus dem vorherigen Beispiel Gauß'sche Quadraturformeln. Wir wollen im Folgenden die Existenz von Gauß'schen Quadraturregeln mit allgemeiner Ordnung, also für beliebige Stützstellenzahl $n \in \mathbb{N}$ untersuchen. Zunächst weisen wir nach, dass die Stützstellen der Gauß-Quadratur stets Nullstellen von orthogonalen Polynomen sein müssen:

Satz 3.56. *Es seien $x_0, \dots, x_n$ paarweise verschiedene Quadratur-Stützstellen einer Gauß-Quadraturformel $I^n(\cdot)$. Dann gilt die Orthogonalitätsbeziehung*

$$\int_{-1}^{1} p_{n+1}(x) q(x)\, dx = 0 \quad \forall q \in P_n, \quad p_{n+1}(x) := (x - x_0) \cdots (x - x_n).$$

Beweis: Das Polynom $p_{n+1} q \in P_{2n+1}$ wird von der Gauß-Regel exakt integriert. Aus $p_{n+1}(x_k) = 0$ folgt:

$$\int_{-1}^{1} p_{n+1}(x) q(x)\, dx = \sum_{k=0}^{n} a_k p_{n+1}(x_k) q(x_k) = 0 \quad \forall q \in P_n.$$

Dies ist gerade die Orthogonalitätsbeziehung. □

Falls eine interpolatorische Quadratur-Formel also die Gauß-Ordnung $2n + 2$ besitzt, so müssen die Stützstellen die Nullstellen eines orthogonalen Polynoms $p_{n+1} \in P_{n+1}$ sein. Es bleibt, die Rückrichtung zu zeigen, dass also durch die Nullstellen von orthogonalen Polynomen auch immer Gauß-Regeln gegeben sind:

Satz 3.57. *Es seien $x_0, \dots, x_n$ paarweise verschiedene Quadratur-Stützstellen. Das Polynom*

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

sei $L^2([-1, 1])$-orthogonal auf allen Polynomen $q \in P_n$. Dann ist durch

$$I_G^n(f) = \sum_{k=0}^{n} \alpha_k f(x_k), \quad \alpha_k = \int_{-1}^{1} \prod_{j=0, j \neq k}^{n} \frac{x - x_j}{x_k - x_j}\, dx,$$

eine Gauß-Quadraturformel $I^n(f)$ zur Integration einer Funktion $f : [a, b] \to \mathbb{R}$ gegeben.

Beweis: Zu gegebener Funktion $f \in C^{2n+2}([-1, 1])$ sei $p_n \in P_n$ das Interpolationspolynom durch die Stützstellen $x_0, \dots, x_n$ und p_{2n+1} ein Interpolationspolynom durch die Stützstellen $x_0, \dots, x_n$ sowie durch weitere (verschiedene) Stützstellen $x_{n+1}, \dots, x_{2n+1}$. Durch Integration dieses p_{2n+1} ist eine Quadraturformel der Ordnung $2n + 2$ gegeben. Es

gilt wie in der Einleitung zur Gauß-Quadratur:

$$\begin{aligned} p_{2n+1}(x) &= \sum_{i=0}^{2n+1} f[x_0,\dots,x_i] \prod_{j=0}^{i-1}(x-x_i) \\ &= p_n(x) + \sum_{i=n+1}^{2n+1} f[x_0,\dots,x_i] \underbrace{\prod_{j=0}^{n}(x-x_j)}_{=p_{n+1}} \underbrace{\prod_{j=n+1}^{i-1}(x-x_j)}_{=q\in P_n}. \end{aligned}$$

Aufgrund der angenommenen Orthogonalität von p_{n+1} auf $q \in P_n$ folgt, dass bereits durch Integration von p_n eine Quadraturformel der Ordnung $2n+2$, also eine Gauß-Quadratur gegeben ist. □

Diese beiden Sätze besagen, dass die Charakterisierung von Gauß'schen Quadraturregeln durch Nullstellen orthogonaler Polynome eindeutig ist. Zum Abschluss müssen wir noch zeigen, dass überhaupt orthogonale Polynome zu beliebiger Stützstellenzahl n existieren, und dass diese Polynome auch reelle Nullstellen haben!

Satz 3.58. *Es existiert eine eindeutige Folge $(p_n)_n$ von Polynomen $p_n \in P_n$ mit*

$$\begin{aligned} p_0(x) &\equiv 1, \\ p_n(x) &= x^n + r_{n-1}(x), \quad n = 1,2,\dots, \end{aligned}$$

mit $r_{n-1} \in P_{n-1}$, die die Orthogonalitätsbeziehung

$$\int_{-1}^{1} p_n(x)p_m(x)\,dx = 0, \quad 0 \le m < n,$$

erfüllen. Durch $\{p_k, k = 0,\dots,n\}$ ist dann eine Orthogonalbasis des P_n gegeben.

Beweis: Die Aussage folgt unter Ausnutzung des Satzes von Gram-Schmidt für die Monombasis $\{1, x, x^2, \dots\}$ von $P_{n+1}[a,b]$, Satz 4.51 Mit

$$\begin{aligned} p_0 &:= 1, \quad k = 1,\dots,n+1 \\ p_k(x) &:= x^k - \sum_{j=0}^{k-1} \frac{(x^k, p_j)}{\|p_j\|^2} p_j(x), \end{aligned}$$

wird dann $\{p_0,\dots,p_{n+1}\}$ ein Orthogonalsystem in $P_{n+1}[a,b]$, unter Ausnutzung des L^2-Skalarprodukts

$$(f,g)_{L^2([-1,1])} := \int_{-1}^{1} f(x)g(x)\,\mathrm{d}x.$$

Denn vollständigen Beweis des Gram-Schmidt-Algorithmus werden wir später nachtragen. Die spezielle Version findet der Leser in [9], S. 87. □

Satz 3.59 (Nullstellen orthogonaler Polynome)**.** *Die bezüglich des $L^2([-1,1])$-Skalarprodukts orthogonalen Polynome p_n besitzen reelle, einfache Nullstellen im Intervall $(-1,1)$.*

Beweis: Der Beweis wird über Widerspruchsargumente geführt. Es sei darauf hingewiesen, dass p_n ein reellwertiges Polynom (nach Standardvoraussetzung in diesem Kapitel) ist. Wir nehmen zunächst an, dass p_n eine reelle Nullstelle λ hat, die nicht im Intervall $(-1,1)$ liegt. Dazu definieren wir

$$q(x) := \frac{p_n(x)}{x-\lambda}.$$

Es gilt $q \in P_{n-1}$ (da $p_n \in P_n$) und deshalb ist $q \perp p_n$. Dies impliziert

$$0 = (q, p_n) = \int_{-1}^{1} \frac{p_n^2(x)}{x-\lambda}\, dx.$$

Dies ist aber ein Widerspruch, da $p_n(x)^2$ auf $[-1,1]$ stets positiv ist (da p_n reellwertig ist), noch die Nullfunktion ist, und darüber hinaus der Faktor $(x-\lambda)^{-1}$ keinen Vorzeichenwechsel haben kann, da λ außerhalb von $[-1,1]$ liegt. Deshalb kann p_n keine Nullstelle außerhalb von $(-1,1)$ haben.

Im zweiten Teil zeigen wir, dass im Intervall $(-1,1)$ nur einfache und reelle Nullstellen liegen. Wir arbeiten wieder mit einem Widerspruchsargument. Wir nehmen an, dass p_n eine Nullstelle λ besitzt, die entweder mehrfach oder nicht reell ist. In beiden Fällen definieren wir

$$q(x) := \frac{p_n(x)}{(x-\lambda)(x-\bar{\lambda})} = \frac{p_n(x)}{|x-\lambda|^2}.$$

Das Polynom $q(x)$ liegt in P_{n-2}. Der komplexwertige Fall ist hier deshalb eingeschlossen, da dann auch $\bar{\lambda}$ eine Nullstelle von p_n ist. Daher gilt

$$0 = (p_n, q) = \int_{-1}^{1} \frac{p_n^2(x)}{|x-\lambda|^2}\, dx.$$

Mit der gleichen Argumentation wir vorher erhalten wir, dass p_n positiv in $(-1,1)$ und auch nicht identisch zur Nullfunktion ist. Daher sind alle Nullstellen von p_n einfach und reell. □

Mit diesen Vorarbeiten sind wir in der Lage den Hauptsatz dieses Abschnitts zu aufzustellen:

Satz 3.60 (Existenz und Eindeutigkeit der Gaußquadratur)**.** *Für jedes $n = 0, 1, \ldots$ existiert eine eindeutige Gaußquadraturformel der Ordnung $2n+2$ zu $n+1$ paarweise verschiedenen Stützstellen, die als Nullstellen des orthogonalen Polynoms $p_{n+1} \in P_{n+1}$ gegeben sind.*

Beweis: Der Beweis besteht aus der Zusammenfassung der bisherigen Resultate:

- Satz 3.58 liefert die eindeutige Existenz des orthogonalen Polynoms $p_{n+1} \in P_{n+1}$.

- Satz 3.59 garantiert, dass dieses Polynom $n+1$ paarweise verschiedene, reelle Nullstellen besitzt.
- Satz 3.57 besagt, dass durch Wahl der Nullstellen als Stützstellen bei entsprechenden Gewichten eine Quadraturformel der Ordnung $2n+2$, also eine Gauß-Quadratur gegeben ist, siehe Definition 3.55.
- Schließlich besagt Satz 3.56, dass nur durch diese Wahl der Stützstellen eine Gauß-Quadraturformel erzeugt wird, liefert also die Eindeutigkeit des Konstruktionsprinzips.

Damit ist der Beweis geführt. □

Ein Nachteil der Newton-Cotes-Formeln sind negative Gewichte ab $n \geq 7$ für geschlossene Formeln. Im nächsten Resultat zeigen wir, dass die Gaußquadraturgewichte stets positiv sind:

Satz 3.61 (Gewichte der Gauß-Quadratur)**.** *Die Gewichte der Gaußquadratur sind stets positiv.*

Beweis: Es sei durch $I_G^n(f) = \sum_k \alpha_k f(x_k)$ die Gauß-Quadratur zu den $n+1$ Stützstellen $x_0, \ldots, x_n$ gegeben. Für die Lagrangeschen Basispolynome gilt

$$L_i^{(n)}(x) = \prod_{j \neq i} \frac{x - x_j}{x_i - x_j}, \quad L_i^{(n)}(x_k) = \delta_{ik}.$$

Es gilt $L_i^{(n)} \in P_n$, also wird auch $(L_i^{(n)})^2$ von der Gauss-Formel exakt integriert:

$$0 < \int_{-1}^{1} (L_i^{(n)}(x))^2 \, \mathrm{d}x = \sum_{k=0}^{n} \alpha_k (L_i^{(n)}(x_k))^2 = \alpha_k.$$

□

Die Gauß'schen Quadraturformeln haben neben der hohen Ordnung den weiteren Vorteil, dass bei positiven Integranden keine Auslöschung auftreten wird. Sie sind numerisch sehr stabil.

Weiter haben wir gesehen, dass die die Newton-Cotes Formeln unter dem gleichen Mangel wie die Lagrangesche Interpolation leidet: falls die Ableitungen des Integranden zu schnell steigen, so muss keine Konvergenz bei steigendem Polynomgrad vorliegen. Im Fall der Gauss-Quadratur erhalten wir hingegen Konvergenz des Integrals:

Satz 3.62 (Konvergenz der Gauss-Quadratur)**.** *Es sei $f \in C^\infty([-1,1])$. Die Folge $(I^{(n)}(f))_n$, $n = 1, 2, \ldots$ der Gaußformeln zur Berechnung von*

$$I(f) = \int_{-1}^{1} f(x) \, dx.$$

ist konvergent:

$$I^{(n)}(f) \to I(f) \quad (n \to \infty).$$

BEWEIS: Es gilt

$$I^{(n)}(f) = \sum_{i=0}^{n} \alpha_i^{(n)} f(x_i^{(n)}), \quad \alpha_i^{(n)} > 0, \quad \sum_{i=0}^{n} \alpha_i^{(n)} = 2.$$

Es sei $\epsilon > 0$. Nach dem Approximationssatz von Weierstraß gibt es ein $p \in P_m$ (wobei m hinreichend groß), so dass

$$\max_{-1 \le x \le 1} |f(x) - p(x)| \le \frac{\epsilon}{4}.$$

Für hinreichend großes $2n + 2 > m$ gilt $I(p) - I_G^n(p) = 0$. Damit folgern wir

$$|I(f) - I^{(n)}(f)| \le \underbrace{|I(f-p)|}_{\le \frac{\epsilon}{4} \cdot 2} + \underbrace{|I(p) - I^{(n)}(p)|}_{=0} + \underbrace{|I^{(n)}(p-f)|}_{\le \frac{\epsilon}{4} \cdot 2} \le \epsilon.$$

Da $\epsilon > 0$ beliebig gewählt worden ist, muss $I^{(n)}(f) \to I(f)$ für $n \to \infty$ konvergieren. □

Schließlich beweisen wir noch ein optimales Resultat für den Fehler der Gauß-Quadratur:

Satz 3.63 (Fehlerabschätzung der Gauß-Quadratur)**.** *Es sei $f \in C^{2n+2}[a,b]$. Dann lautet die Restglieddarstellung zu einer Gaußquadraturformel der Ordnung $2n+2$:*

$$R^{(n)}(f) = \frac{f^{(2n+2)}(\xi)}{(2n+2)!} \int_a^b p_{n+1}^2(x)\, dx$$

für $\xi \in [a,b]$ und mit $p_{n+1} = \prod_{j=0}^{n}(x - \lambda_j)$.

BEWEIS: Der Beweis erfolgt unter Zuhilfenahme der Hermite-Interpolation (siehe Satz 3.16). Hiernach existiert ein Polynom $h \in P_{2n+1}$ zu einer zu interpolierenden Funktion $f \in C^{2n+2}[-1,1]$, welches die Hermitesche Interpolationsaufgabe löst, mit den Interpolationsbedingungen

$$h(\lambda_i) = f(\lambda_i), \quad h'(\lambda_i) = f'(\lambda_i), \quad i = 0, \ldots, n.$$

Hierzu lautet die (bereits bekannte) Restglieddarstellung

$$f(x) - h(x) = f[\lambda_0, \lambda_0, \ldots, \lambda_n, \lambda_n, x] \prod_{j=0}^{n} (x - \lambda_j)^2.$$

Wendet man nun die Gaußsche-Quadraturformel auf $h(x)$ an, dann gilt zunächst $I_G^{(n)}(h) = I(h)$, und weiter unter Anwendung des Mittelwertsatzes der Integralrechnung

$$\begin{aligned} I(f) - I_G^{(n)}(f) &= I(f-h) - I_G^{(n)}(f-h) \\ &= \int_{-1}^{1} f[\lambda_0, \lambda_0, \ldots, \lambda_n, \lambda_n, x] \prod_{j=0}^{n} (x - \lambda_j)^2\, dx - \sum_{i=0}^{n} a_i [f(\lambda_i) - h(\lambda_i)] \\ &= \frac{f^{(2n+2)}(\xi)}{2n+2)!} \int_{-1}^{1} \prod_{j=0}^{n} (x - \lambda_j)^2\, dx. \end{aligned}$$

Der Mittelwertsatz darf hier angewendet werden, da stets $\prod_{j=0}^{n}(x-\lambda_j)^2 \geq 0$. Der Term $[f(\lambda_i)-h(\lambda_i)]=0$, da hier die Interpolationsbedingungen ausgenutzt worden sind. Damit ist alles gezeigt. □

Bemerkung 3.64 (Zur Regularität in der Fehlerformel)**.** *Es bleibt zu bemerken, dass zur Gültigkeit der Fehlerabschätzung 3.63 eine vergleichsweise hohe Regularität an die Funktion f gestellt wird: $f \in C^{2n+2}[a,b]$. Um Aussagen über die Fehlerentwicklung bei geringerer Regularität zu erhalten, sollte man wieder zu summierten Formeln (siehe Splines und stückweise interpolatorische Quadratur) übergehen.*

Nachdem wir nun das theoretische Fundament der Gaußquadratur bereitgestellt haben verbleibt im finalen Schritt die explizite Angabe der Stützstellen sowie der (Quadratur)-Gewichte. Hierzu nutzen wir die Legendre-Polynome, die grundsätzlich bei orthogonalen Polynomen eine herausragende Rolle spielen.

Legendre-Polynome

Satz 3.65 (Legendre-Polynome)**.** *Die* Legendre-Polynome $L_n \in P_n$ *mit*

$$L_n(x) := \frac{1}{2^n n!}\frac{d^n}{dx^n}(x^2-1)^n,$$

sind die bezüglich des $L^2([-1,1])$-Skalarprodukt orthogonalisierten Monome $\{1,x,x^2,\dots\}$.

Beweis: Zunächst gilt $L_n \in P_n$, denn n-fache Ableitung von x^{2n} liefert das höchste Monom x^n. Die Orthogonalitätseigenschaft folgt durch mehrfache Anwendung von partieller Integration aus Ausnutzen der Tatsache, dass der Term $(x^2-1)^n$ an den beiden Intervallenden n-fache Nullstellen besitzt. □

Mit den Legendre-Polynomen existiert eine explizite Darstellung für die orthogonalen Polynome bezüglich des $L^2([-1,1])$-Skalarprodukts. Diese Polynome unterscheiden sich von den Polynomen aus Satz 3.58 lediglich durch die Normierung. Für die orthogonalen Polynome $p_n(x)$ aus Satz 3.58 gilt, dass der Koeffizient vor dem höchsten Monom 1 ist. Für die Legendre-Polynome gilt die Normierungseigenschaft $L_n(1)=1$.

Die ersten Legendre-Polynome lauten:

$$\begin{aligned}
&L_0(x) = 1, && \\
&L_1(x) = x, && x_0 = 0 \\
&L_2(x) = \frac{1}{2}(3x^2-1), && x_{0/1} = \pm\sqrt{\frac{1}{3}}, \\
&L_3(x) = \frac{1}{2}(5x^3-3x), && x_0 = 0,\ x_{1/2} = \pm\sqrt{\frac{3}{5}} \\
&L_4(x) = \frac{1}{8}(35x^4-30x^2+3), && x_{0/1} \approx \pm 0.861136,\ x_{2/3} \approx \pm 0.339981
\end{aligned}$$

Für die Nullstellen (also die Quadratur-Stützstellen) existiert keine geschlossene Formel. Im Fall $n > 3$ können die Nullstellen nur noch numerisch berechnet werden.

Die entsprechenden Quadraturgewichte können einfach über die Formel (3.14) berechnet werden

$$\alpha_k = \int_{-1}^{1} \prod_{j=0}^{n} \frac{x - x_j}{x_k - x_j} \, \mathrm{d}x.$$

In Tabelle 3.3 fassen wir die ersten wichtigen Gauß-Legendre Regeln zusammen.

$n-1$	x_i	α_i	Ordnung
0	0	2	2
1	$\pm\sqrt{\frac{1}{3}}$	1	4
2	0	$\frac{8}{9}$	
	$\pm\sqrt{\frac{3}{5}}$	$\frac{5}{9}$	6
3	± 0.8611363116	0.347854845	
	± 0.3399810436	0.652145154	8
4	0	0.568888889	
	± 0.9061798459	0.236926885	
	± 0.5384693101	0.478628670	10

Tabelle 3.3: Einige Gauß-Legendre Quadratur-Regeln.

Zuletzt sollte ein (kleiner) Nachteil der Gauß-Legendre Formeln erwähnt werden. Für jedes Legendre-Polynom $L_n(x)$ erhalten wir neue Stützstellen $x_0, \ldots, x_n$, die im Allg. mit den bereits berechneten Stützstellen der Polynome $L_n(x), n = 0, \ldots, n-1$ nicht übereinstimmen.

Gauß-Tschebyscheff Quadratur

Die Gauß'schen Quadraturformeln sind so konstruiert, dass Polynome bis zu einem bestimmten Grad exakt integriert werden können. Oft sind hingegen die zu integrierenden Funktionen keine Polynome, man denke z.B. an

$$f(x) = \sqrt{1 - x^2},$$

auf dem Intervall $[-1, 1]$. Die Ableitungen dieser Funktion haben an den Intervallenden Singularitäten. Die einfache Gauß-Quadratur ist zur Approximation von $\int_{-1}^{1} f(x) \, \mathrm{d}x$ nicht geeignet, da hohe Regularität vorausgesetzt wird.

Die Idee der Gauß-Quadratur kann nun verallgemeinert werden. Dazu sollen Quadraturregeln konstruiert werden, welche Funktionen nach der Art

$$p_\omega(x) = \omega(x) p(x),$$

mit einem Polynom $p(x)$ möglichst exakt integrieren. Die Funktion $\omega(x)$ nennen wir Gewichtsfunktion. Diese Gewichtsfunktion fließt nun in das Konstruktionsprinzip von speziellen Gauß-Quadraturformeln ein. Zunächst formulieren wir den hierfür zentralen Satz:

Satz 3.66 (Gewichtetes Skalarprodukt)**.** *Durch $\omega \in L^\infty([-1,1])$ sei eine positive*

$$\omega(x) > 0 \quad \textit{fast überall},$$

Gewichtsfunktion gegeben. Dann ist durch

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

ein Skalarprodukt gegeben.

Beweis: Übung! □

Die bisher bewiesenen Sätze 3.56, 3.57, 3.58, 3.59 sowie 3.61 gelten alle auch bei Verwendung dieses gewichteten Skalarprodukts. Das Übertragen der Beweise überlassen wir als Übung. Die Gauß-Tschebyscheff Quadratur wählt als spezielles Gewicht die Funktion

$$\omega(x) = \frac{1}{\sqrt{1-x^2}}.$$

Diese Funktion legt ein starkes Gewicht auf die beiden Intervallenden. Es gilt der folgende zunächst erstaunliche Satz:

Satz 3.67 (Tschebyscheff-Polynome)**.** *Die* Tschebyscheff-Polynome $T_n \in P_n$ *mit*

$$T_n(x) := \cos(n \arccos x), \quad -1 \le x \le 1,$$

sind die bezüglich des $(\cdot,\cdot)_\omega$ Skalarprodukts orthogonalisierten Monome $\{1, x, \dots\}$ mit der Gewichtsfunktion

$$\omega(x) = \frac{1}{\sqrt{1-x^2}}.$$

Die n paarweise verschiedenen Nullstellen des n-ten Tschebyscheff-Polynoms $T_n(x)$ sind gegeben durch

$$x_k = \cos\left(\frac{2k+1}{2n}\pi\right), \quad k = 0, \dots, n-1.$$

Beweis: *(i)* Wir müssen zunächst nachweisen, dass durch $T_n(x)$ überhaupt Polynome gegeben sind. Wir führen den Beweis induktiv durch Herleiten einer Iterationsformel. Es gilt:

$$T_0(x) = 1, \quad T_1(x) = x,$$

also insbesondere $T_0 \in P_0$ und $T_1 \in P_1$. Aus dem Additionstheorem für die Kosinusfunktion

$$\cos((n+1)y) + \cos((n-1)y) = 2\cos(nt)\cos(y),$$

erhalten wir mit $y = \arccos x$ eine zweistufige Rekursionsformel:

$$T_{n+1}(x) + T_{n-1}(x) = 2xT_n(x) \quad \Rightarrow \quad T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x).$$

Hieraus schließen wir induktiv $T_n \in P_n$ mit

$$T_n(x) = 2^{n-1}x^n + \dots.$$

(ii) Wir weisen nun die Orthogonalität der Tschebyscheff-Polynome bzgl. des gewichteten Skalarprodukts auf $[-1, 1]$ nach. Mit $x = \cos(t)$ gilt unter Ausnutzung der Orthogonalität trigonometrischer Polynome:

$$\int_{-1}^{1} \frac{T_n(x)T_m(x)}{\sqrt{1-x^2}}\,dx = \int_0^{\pi} \cos(nt)\,\cos(mt)\,dt = \begin{cases} \pi, & n = m = 0, \\ \frac{\pi}{2}, & n = m > 0, \\ 0, & n \neq m. \end{cases}$$

(iii) Die Nullstellen der Tschebyscheff-Polynome können elementar berechnet werden. Diese Nullstellen sind die Stützstellen der entsprechenden Quadratur. □

Zum Aufstellen von konkreten Gauß-Tschebyscheff-Quadraturformeln benötigen wir noch die entsprechenden Quadraturgewichte. Da die Tschebyscheff-Polynome gewichtet mit ω exakt integriert werden müssen gilt:

$$\sum_{k=0}^{n-1} a_k T_m(x_k) = \int_{-1}^{1} \frac{T_m(x)}{\sqrt{1-x^2}}\,\mathrm{d}x, \quad m = 0, \dots, n-1.$$

Es folgt:

$$\sum_{k=0}^{n-1} a_k \cos\frac{(2k+1)m}{2n}\pi = \begin{cases} \pi, & m = 0, \\ 0, & m = 1, \dots, n-1. \end{cases}$$

Mit Hilfe von trignometrischen Argumenten (siehe [8]) erhalten wir die Lösung des linearen Gleichungssystems:

$$a_k = \frac{\pi}{n}, \quad k = 0, \dots, n-1. \tag{3.18}$$

Damit haben wir alle Komponenten gesammelt, um die Gauß-Tschebyscheff Formel inklusive Restglied anzugeben:

Verfahren 3.68 (Gauß-Tschebyscheff-Formel)**.** *Die Gauß-Tschebyscheff-Formel vom Grad $2n$ mit Restglied lautet*

$$\int_{-1}^{1} \frac{f(x)}{\sqrt{1-x^2}}\,dx = \frac{\pi}{n}\sum_{k=0}^{n-1} f\left(\cos\frac{2k+1}{2n}\pi\right) + \frac{\pi}{2^{2n-1}(2n)!}f^{(2n)}(\xi)$$

mit $\xi \in [-1, 1]$.

BEWEIS: Die Quadraturformel folgt sofort aus Einsetzen der Stützstellen und Gewichte in die allgemeine Definition einer Quadraturformel. Die Restglieddarstellung folgt aus Satz 3.63. □

Der große Vorteil der Gauß-Tschebyscheff Quadratur ist die einfache Möglichkeit, die Quadraturformeln für beliebige Ordnung explizit aufzustellen. Stützstellen sowie Gewichte sind einfach zu bestimmen.

Beispiel 3.69. *Wir verwenden die Gauss-Tschebyscheff-Quadratur zur Berechnung des Integrals (halber Kreisinhalt)*

$$I(f) = \int_{-1}^{1} \sqrt{1-x^2}\, dx = \int_{-1}^{1} \omega(x) \underbrace{(1-x^2)}_{=:f(x)}\, dx,$$

mit dem Gauss-Tschebyscheff-Gewicht $\omega(x)$ und $(1-x^2) \in P_2$. Aufgrund von P_2 wählen wir $n = 2$ im Verfahren 3.68 zur exakten Berechnung des Integrals. D.h.

$$\int_{-1}^{1} \omega(x) f(x)\, dx = \frac{\pi}{2} \sum_{k=0}^{1} f\left(\cos \frac{2k+1}{2 \cdot 2}\pi\right) + \frac{\pi}{2 \cdot (2)!} f^{(2)}(\xi)$$

mit $\xi \in (-1,1)$ und den Quadraturgewichten $a_0 = a_1 = \frac{\pi}{2}$ (berechnet mit Formel 3.18). Unter Vernachlässigung des Restglieds erhalten wir also

$$\begin{aligned} I(f) &= \int_{-1}^{1} \sqrt{1-x^2}\, dx = \int_{-1}^{1} \omega(x)(1-x^2)\, dx \\ &= \frac{\pi}{2} \sum_{k=0}^{1} \left(1 - (\cos \frac{2k+1}{4}\pi)^2\right) \\ &= \frac{\pi}{2} \left(\left(1 - \cos\left(\frac{1}{4}\pi\right)^2\right) + \left(1 - \cos\left(\frac{3}{4}\pi\right)^2\right)\right) \\ &= \frac{\pi}{2} \left((1-0.5) + (1-0.5)\right) \\ &= \frac{\pi}{2}. \end{aligned}$$

Gauß-Quadratur mit beliebigen Gewichten $\omega(x) > 0$

Es verbleibt die Existenz, Eindeutigkeit und Fehlerabschätzung für die Gauß-Formeln zu zeigen, deren Gewicht $\omega(x) \neq 1 \geq 0$ ist. In diese Kategorie fällt insbesondere die Gauß-Tschebyscheff Quadratur mit

$$\omega(x) = \frac{1}{\sqrt{1-x^2}}.$$

Hierzu gilt:

Satz 3.70 (Existenz, Eindeutigkeit und Fehlerabschätzung)**.** *Es sei $w(x) \geq 0$ fast überall. Dann lassen sich mit Hilfe des gewichteten Skalarprodukts*

$$(f,g)_w := \int_a^b \omega(x) f(x) g(x)\, dx, \quad w(x) > 0.$$

der Satz 3.60 (inklusive der darin verwendeten Sätze) und Satz 3.62 analog beweisen.

Beweis: Nach Verifizierung der Skalarproduktеigenschaften von $(f,g)_w$, übertrage man die Beweise der Sätze 3.60 und 3.62 unter Hinzunahme der Gewichtsfunktion. □

Bemerkung 3.71 (Normierung des Integrationsintervalls)**.** *Die Gauß-Quadratur mit Legendre-Polynomen und Tschebyscheff Polynomen ist lediglich auf dem Intervall $[-1,1]$ definiert. Allerdings bedeutet dies keine Einschränkung für den allgemeinen Fall $[a,b]$. Jedes endlich-dimensionale Intervall $[a,b]$ kann durch die Transformation*

$$x = 2\frac{t-a}{b-a} - 1, \quad t \in [a,b]$$

in $[-1,1]$ überführt werden. Mit der Wahl $x \in [-1,1]$ wird dann die Integration in $[-1,1]$ durchgeführt, um anschließend durch die Rücktransformation

$$t = \frac{(x+1)(b-a)}{2} + a, \quad x \in [-1,1]$$

die (berechneten) Werte für $t \in [a,b]$ zu erhalten.

Korollar 3.72 (Transformation des Integrationsintervalls)**.** *Mit den Voraussetzungen aus Bemerkung 3.71 gilt für das (exakte) Integral*

$$I(f) = \int_a^b f(t)\, dt = \frac{b-a}{2} \int_{-1}^1 f\left[\frac{(x+1)(b-a)}{2} + a\right] dx.$$

Die Quadraturformel zur Integration von $I(f)$ lautet dann

$$I^n(f) = \frac{b-a}{2} \sum_{k=1}^n a_k f\left[\frac{(x_k+1)(b-a)}{2} + a\right].$$

3.5.4 Romberg-Quadratur

Die Idee der Richardson Extrapolation in Abschnitt 3.4, Approximationsprozesse hoher Ordnung zu erzielen, die auf Basismethoden niedriger Ordnung basieren, wird hier auf die numerische Quadratur angewendet. Konkret werden wir die summierte Trapezregel zur Extrapolation nutzen. Die Trapezregel gehört zu den sehr einfachen Verfahren mit niedriger Ordnung. Wir fassen die wesentlichen Ergebnisse aus Abschnitt 3.5.2 zusammen.

Auf einer gleichmäßigen Zerlegung des Intervalls $[a,b]$ in N Teilintervalle mit Schrittweite $h = 1/N$ ist die summierte Trapezregel zur Approximation von $\int_a^b f \,\mathrm{d}x$ gegeben durch:

$$I_h(f) = h\left(\frac{1}{2}f(a) + \sum_{j=1}^{N-1} f(x_j) + \frac{1}{2}f(b)\right), \quad x_j := a + jh,$$

und liefert im Falle $f \in C^2([a,b])$ die Fehlerabschätzung:

$$I(f) - I_h(f) = \frac{b-a}{12}h^2 f^{(2)}(\xi),$$

mit einer Zwischenstelle $\xi \in [a,b]$. Um die Extrapolationsmethode erfolgreich anwenden zu können brauchen wir Kenntnis über die weitere Fehlerentwicklung der Trapezregel. Die Basis hierzu ist die Euler-Maclaurinsche Summenformel wurde, die eine Entwicklung des Fehlers in geraden Potenzen von h zeigt:

Satz 3.73 (Euler-Maclaurinsche Summenformel)**.** *Falls $f^{2m+2}[a,b]$, dann gilt die Euler-Maclaurinsche Summenformel*

$$I(f) - I_h(f) = \sum_{k=1}^{m} h^{2k}\frac{B_{2k}}{(2k)!}\left(f^{(2k-1)}(b) - f^{(2k-1)}(a)\right) + h^{2m+2}\frac{b-a}{(2m-2)!}B_{2m+2}f^{(2m+2)}(\xi),$$

mit $\xi \in [a,b]$ und den Bernoulli-Zahlen B_{2k}.

Beweis: Seminarthema! □

Die Bernoulli-Zahlen sind definiert als die Koeffizienten der Taylorreihendarstellung von

$$\frac{x}{e^x - 1} = \sum_{k=0}^{\infty}\frac{B_k}{k!}x^k = 1 - \frac{1}{2}x + \frac{1}{6}\frac{x^2}{2!} - \frac{1}{30}\frac{x^4}{4!} + \frac{1}{42}\frac{x^6}{6!} + \ldots,$$

und genügen der Rekursionsformel

$$B_0 = 0, \quad B_k = -\sum_{j=0}^{k-1}\frac{k!}{j!(k-j+1)!}B_j, \quad k = 1,2,\ldots.$$

Die ersten Bernoulli-Zahlen sind gegeben als:

$$1, -\frac{1}{2}, \frac{1}{6}, 0, -\frac{1}{30}, 0, \frac{1}{42}, 0, -\frac{1}{30}, \ldots.$$

Ausgenommen $B_1 = -\frac{1}{2}$ gilt für jede zweite (ungerader Index) Bernoulli-Zahl $B_{2k+1} = 0$. Ansonsten folgen sie keinem erkennbaren Gesetz Für große k wachsen die Bernoulli-Zahlen sehr schnell an und verhalten sich asymptotisch wie

$$|B_{2k}| \sim 2(2k)!(2\pi)^{-2k}, \quad k \to \infty.$$

Die Euler-Maclaurinsche Summenformel besagt, dass die Trapezregel eine Entwicklung in geraden Potenzen von h besitzt. Daher eignet sie sich gleich in zweifacher Hinsicht optimal zur Extrapolation: aufgrund der quadratischen Fehlerentwicklung gewinnen wir in jedem Extrapolationsschritt zwei Ordnungen Genauigkeit und aufgrund der Stützstellenwahl an den Enden der Teilintervalle x_{j-1} und x_j können bereits berechnete Werte $f(x_j)$ zu einer Schrittweite h bei der nächst feineren Approximation zu $h/2$ weiter verwendet werden. Zur Approximation von

$$a(0) \approx \int_a^b f(x)\,\mathrm{d}x, \quad a(h) := I_h(f) = h\left(\frac{1}{2}f(a) + \sum_{j=1}^{N-1} f(x_j) + \frac{1}{2}f(b)\right),$$

verwenden wir das Extrapolationsprinzip aus Abschnitt 3.4:

Verfahren 3.74 (Romberg-Quadratur)**.**

1) Berechne für eine Folge von Schrittweiten $(h_k)_{k\in\mathbb{N}}$ mit

$$\frac{h_{k+1}}{h_k} \leq \rho < 1,$$

die Approximationen

$$a(h_k), \quad k = 0, \ldots, m.$$

2) Extrapoliere die Werte $(h_k^2, a(h_k)), k = 0, \ldots, m$ mit Polynomen in h^2.

Als Schrittweitenfolge kann mit einem $h > 0$ die einfache Vorschrift

$$h_k = 2^{-k}h,$$

verwendet werden. Diese Folge wird *Romberg-Folge* genannt und hat den Vorteil, dass bereits berechnete Stützstellen weiter verwendet werden können. Der Nachteil dieser Folge ist das schnelle Wachstum der Anzahl der Stützstellen. Die Extrapolation selbst wird mit dem modifizierten Neville-Schema 3.33 durchgeführt.

Die Diagonalelemente $a_{k,k}$ sind gerade die Näherungen zu $a(0)$. Basierend auf Satz 3.32 zur allgemeinen Richardson-Extrapolation erhalten wir die Fehlerabschätzung:

Satz 3.75 (Romberg-Quadratur)**.** *Es sei $f^{2m+2}[a,b]$ sowie $h > 0$ gegeben. Das Romberg-Verfahren zur Schrittweitenfolge $h_k = 2^{-k}h,\ k = 0, \ldots, m$ liefert nach m Extrapolationsschritten die Approximation:*

$$I(f) - a_{m,m} = O(h^{2m+2}).$$

Beweis: Der Beweis folgt durch Kombination von Satz 3.32 über die Richardson-Extrapolation mit der Euler-Maclaurinschen Summenformel aus Satz 3.73. □

Bemerkung 3.76. *Anstatt der Romberg-Folge $h_k = 2^{-k}h$ kann auch mit der Burlisch-Folge*

$$h_k = \begin{cases} h \cdot 2^{-\frac{k}{2}} & k \text{ gerade} \\ \frac{h}{3} 2^{-\frac{k-1}{2}} & k \text{ ungerade} \end{cases}$$

gearbeitet werden, da diese wesentlich weniger Funktionsauswertungen benötigt. Bei $h = 1$ sind die ersten Folgenglieder gegeben durch:

$$\frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \frac{1}{6}, \frac{1}{8}, \frac{1}{12}, \ldots,$$

Integration periodischer Funktionen

Im Falle periodischer Funktionen, d.h. es sei $f^{2m+2}(-\infty, \infty)$ mit dem Periodenintervall $[a, b]$ und

$$f^{(2k-1)}(a) = f^{(2k-1)}(b), \quad k = 1, 2, \ldots$$

kann die Euler-Maclaurinsche Summenformel *entscheidend* vereinfacht werden. Aus

$$I(f) - I_h(f) = \sum_{k=1}^{m} h^{2k} \frac{B_{2k}}{(2k)!} \left(f^{(2k-1)}(b) - f^{(2k-1)}(a) \right) + h^{2m+2} \frac{b-a}{(2m-2)!} B_{2m+2} f^{(2m+2)}(\xi),$$

mit

$$I_h(f) = h \left(\frac{1}{2} f(a) + \sum_{j=1}^{N-1} f(x_j) + \frac{1}{2} f(b) \right), \quad x_j := a + jh,$$

folgt dann

$$I(f) - I_h(f) = h^{2m+2} \frac{b-a}{(2m-2)!} B_{2m+2} f^{(2m+2)}(\xi) = O(h^{2m+2}),$$

mit

$$I_h(f) = h \left(\frac{1}{2} f(a) + \sum_{j=1}^{N-1} f(x_j) + \frac{1}{2} f(b) \right) = h \left(f(a) + \sum_{j=1}^{N-1} f(x_j) \right) = h \left(\sum_{j=0}^{N-1} f(x_j) \right)$$

mit $x_j := a + jh$. Die summierte Trapezregel *vereinfacht* sich dadurch zur summierten linksseitigen Boxregel.

Falls $f \in C^\infty(-\infty, \infty)$, dann konvergiert die summierte Trapezregel (d.h. bei $[a, b]$-periodischen Funktionen, die summierte Boxregel) schneller gegen den Integralwert, für $h \to 0$, als jede andere Quadraturregel.

3.6 Approximationstheorie

Bisher haben wir uns im Wesentlichen mit der Interpolation beschäftigt. Die Approximation ist weiter gefasst: wir suchen eine einfache Funktion $p \in P$ (dabei ist der Funktionenraum P meist wieder der Raum der Polynome), welche die *beste Approximation* zu gegebener Funktion f oder zu gegebenen diskreten Daten $y_i \in \mathbb{R}$ darstellt. Der Begriff beste Approximation ist dabei weit gefasst und wir bezeichnen ihn mit dem minimalen Abstand von p zu f (oder zu den diskreten Daten). Den Abstand zwischen Funktionen können wir in Normen messen und die allgemeine Approximationsaufgabe besteht nun im Auffinden von $p \in P$, so dass

$$\|f - p\| = \min_{q \in P} \|f - q\|.$$

Die Wahl der Norm $\|\cdot\|$ ist dabei zunächst beliebig. Es stellt sich jedoch heraus, dass die mathematische Approximationsaufgabe je nach betrachteter Norm einer sehr unterschiedliche Vorgehensweise bedarf. Eine spezielle Approximationsaufgabe haben wir bereits kennengelernt:

Bemerkung 3.77 (Lagrange-Interpolation als Approximation)**.** *Angenommen, in den $n+1$ paarweise verschiedenen Stützstellen $x_0, x_1, \ldots, x_n$ soll für das Polynom $p_n \in P_n$ gelten $p(x_i) = f(x_i)$, dann definieren wir die Norm:*

$$|p|_n := \max_{i=0,\ldots,n} |p(x_i)|.$$

Man kann zeigen, dass dies wirklich eine Norm auf dem Polynomraum P_n ist. Die Approximationsaufgabe: suche $p \in P_n$, so dass

$$|p - f|_n = \min_{q \in P_n} |q - f|_n,$$

ist gerade die Lagrange-Interpolationsaufgabe.

In diesem Abschnitt werden wir uns hingegen hauptsächlich mit der L^2-Norm

$$\|f\|_{L^2([a,b]} := \left(\int_a^b f(x)^2 \, \mathrm{d}x \right)^{\frac{1}{2}},$$

sowie mit der Maximumsnorm

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

befassen. Die beste Approximation in der L^2-Norm taucht in der Analysis in Form der *Fourier-Entwicklung* in trigonometrischen Polynomen auf, Konvergenz wird hier bezüglich der L^2-Norm

$$\|f - f_n\|_{L^2([a,b])} \to 0 \quad (n \to \infty),$$

gezeigt. Die Approximation bezüglich der Maximumsnorm ist für die praktische Anwendung von großer Bedeutung: denn eine Approximation, die den Fehler gleichmäßig auf dem gesamten Intervall $[a, b]$ unter Kontrolle hält, schließt die typischen Oszillationen der Lagrange-Interpolation an den Intervallenden (siehe Beispiel 3.15) systematisch aus. Es zeigt sich allerdings, das gerade dieser für die Anwendung wichtige Approximationsbegriff mathematisch schwer zu greifen ist.

3.6.1 Gauss-Approximation: Beste Approximation in der L^2-Norm

Zunächst betrachten wir die beste Approximation einer Funktion $f : [a, b] \to \mathbb{R}$ mit Polynomen $p \in P$ bezüglich der L^2-Norm:

$$\|f - p\|_{L^2([a,b])} = \min_{\phi \in P} \|f - \phi\|_{L^2([a,b])}.$$

Die L^2-Norm kann mittels $\|f\|_{L^2} = (f, f)^{\frac{1}{2}}$ über das L^2-Skalarprodukt definiert werden. Vektorräume mit Skalarprodukt heißen *Prähilbertraum*. Speziell in reellen Vektorräumen spricht man von *euklidischen Räumen*, im komplexen auch von *unitären Räumen*. Das Skalarprodukt dient zur Beschreibung von Orthogonalitätsbeziehungen. Für die Approximation bezüglich der L^2-Norm gilt die folgende Charakterisierung:

Satz 3.78 (Approximation und Orthogonalität)**.** *Es sei $f \in C[a, b]$ und $S \subset C[a, b]$ ein endlich dimensionaler Unterraum. Auf S sei durch $(\cdot, \cdot)$ ein Skalarprodukt und durch $\|\cdot\|$ die induzierte Norm gegeben. Dann ist $p \in S$ beste Approximation zu $f \in C[a, b]$*

$$\|f - p\| = \min_{\phi \in S} \|f - \phi\|,$$

genau dann, wenn der Fehler $f - p$ orthogonal auf dem Raum S steht:

$$(f - p, \phi) = 0 \quad \forall \phi \in S.$$

Beweis: Es sei $p \in S$ eine beste Approximation Dann besitzt die quadratische Funktion

$$F_\phi(t) := \|f - p - t\phi\|^2, \quad t \in \mathbb{R}$$

für jedes fest gewählte $\phi \in S$ bei $t = 0$ ein Minimum. Also gilt die für Minima differenzierbarer Funktionen notwendige Bedingung:

$$\begin{aligned} 0 = F'(0) &= \frac{\partial}{\partial t} \|f - p - t\phi\|^2 \big|_{t=0} \\ &= -(f - p - t\phi, \phi) - (\phi, f - p - t\phi) \big|_{t=0} \\ &= -2(f - p, \phi) \quad \forall \phi \in S. \end{aligned}$$

Geometrisch kann man dies so ausdrücken, dass der Fehler $f - p$ senkrecht auf dem approximierenden Raum S steht im Sinne des L^2-Skalarprodukts.

Umgekehrt gelte nun die Beziehung

$$(f - p, \phi) = 0 \quad \forall \phi \in S,$$

für ein $p \in S$. Dann erhält man durch Erweitern mit ϕ und Anwendung der Cauchy-Schwarz-Ungleichung

$$\|f-p\|^2 = (f-p, f-p) = (f-p, f-\phi+\phi-p) = (f-p, f-\phi)+(f-p, \underbrace{\phi - p}_{\in S}) \leq \|f-p\| \, \|f-\phi\|.$$

Hier haben wir die Orthogonalitätsbeziehung für $\phi - p \in S$ verwendet. Weiter folgt:

$$\|f - p\| \leq \|f - \phi\| \Rightarrow \|f - p\| = \inf_{\phi \in S} \|f - \phi\|$$

und damit ist auch p beste Approximation per Definitionem. □

Wir können die beste Approximation $p \in S$ durch eine Orthogonalitätsbeziehung beschreiben. Dieser Zusammenhang ist der Schlüssel zur Analyse der Gauss-Approximation und auch zur praktischen numerischen Realisierung. Wir können sofort den allgemeinen Satz beweisen:

Satz 3.79 (Allgemeine Gauß-Approximation)**.** *Es sei H ein Vektorraum mit Skalarprodukt $(\cdot, \cdot)$ und $S \subset H$ ein endlich-dimensionaler Teilraum. Dann existiert zu jedem $f \in H$ eine eindeutig bestimmte beste Approximation $p \in S$ in der induzierten Norm $\| \cdot \|$:*

$$\|f - p\| = \min_{\phi \in S} \|f - \phi\|.$$

Beweis: Für den Beweis nutzen wir die Charakterisierung der Bestapproximation mit Hilfe des Skalarprodukts aus Satz 3.78.

(i) Eindeutigkeit.

Seien $p_1, p_2 \in S$ zwei Bestapproximationen. Dann gilt notwendigerweise

$$(f - p_1, \phi) = 0 \quad \text{und} \quad (f - p_2, \phi) = 0$$

für alle $\phi \in S$. Dies impliziert

$$(p_1 - p_2, \phi) = 0 \quad \phi \in S.$$

Da dies für alle $\phi \in S$ gilt, können wir ein spezielles ϕ geschickt wählen. Eine kluge Wahl ist $\phi := p_1 - p_2$, woraus dann

$$(p_1 - p_2, p_1 - p_2) = \|p_1 - p_2\|^2 = 0$$

und aufgrund der Definitheit der Norm (im Teilraum $S \in H$) $p_1 = p_2$ folgt.

(ii) Existenz

Der endlich-dimensionale Teilraum $S \subset H$ besitzt eine Basis $\{\phi_1, \ldots, \phi_n\}$ mit $n := \dim(S)$. Die beste Approximation $p \in S$ stellen wir als Linearkombination in dieser Basis dar:

$$p = \sum_{k=1}^{n} \alpha_k u_k,$$

mit eindeutig bestimmten Koeffizienten $\alpha_k \in \mathbb{R}$. Dieser Ansatz für p wird in die Orthogonalitätsbedingung eingesetzt:

$$(f - p, \phi) = (f - \sum_{k=1}^{n} \alpha_k \phi_k, \phi) = (f, \phi) - \sum_{k=1}^{n} \alpha_k (\phi_k, \phi) = 0 \quad \forall \phi \in S.$$

Wir durchlaufen mit ϕ nach und nach alle Basisvektoren und erhalten (Superpositionsprinzip) das lineare Gleichungssystem mit n Gleichungen und n Variablen:

$$\sum_{k=1}^{n} \underbrace{(\phi_k, \phi_i)}_{=a_{ki}} \underbrace{\alpha_k}_{x} = \underbrace{(f, \phi_i)}_{=b_i} \quad i = 1, \ldots, n, \tag{3.19}$$

in dem der Koeffizientenvektor $x = (\alpha_k)_{k=1,\ldots,n}$ die gesuchte unbekannten Lösung ist. Die sogenannte Systemmatrix $A = (a_{ij})_{i,j=1}^{n}$ ist die *Gramsche Matrix* der Basis $\{\phi_1, \ldots, \phi_n\}$ und stets regulär. A ist folglich injektiv. Weiter ist A symmetrisch und also auch positiv definit. Das Gleichungssystem $Ax = b$ ist somit für jede rechte Seite b (d.h. für jedes $f \in H$) eindeutig lösbar. Damit ist über die Orthogonalitätsbedingung eindeutig ein Element $p \in P$ bestimmt, welches aufgrund von Teil Satz 3.78 die Bestapproximationseigenschaft besitzt. □

Der Beweis liefert sofort ein Konstruktionsprinzip zur Bestimmung der Bestapproximation $p \in S$. Wir stellen zu gegebener Basis $\{\phi_1, \ldots, \phi_n\}$ das lineare $n \times n$ Gleichungssystem auf:

$$\sum_{j=1}^{n} a_{ij} \alpha_j = b_i, \quad i = 1, \ldots, n,$$

mit $a_{ij} = (\psi_i, \psi_j)$ und berechnen den Ergebnisvektor $x = (\alpha_j)_j$. Die gesuchte Bestapproximation $p \in S$ ist dann in der Basisdarstellung gegeben als

$$p = \sum_{i=1}^{n} \alpha_i \psi_i.$$

Zur numerischen Realisierung muss zunächst das lineare Gleichungssystem aufgestellt werden, d.h. insbesondere müssen die Einträge der Systemmatrix (numerisch) integriert werden. Schließlich ist das lineare Gleichungssystem $Ax = b$ zu lösen. Das Lösen von linearen Gleichungssystemen stellt einen eigenen Schwerpunkt in der numerischen Mathematik dar und wir befassen uns damit in Kapitel 4. Es zeigt sich, dass dieser Lösungsweg numerische sehr instabil ist. Angenommen, wir starten die Konstruktion mit der Monombasis $\{1, x, \ldots, x^n\}$. Dann ist die durch

$$a_{ij} = \int_{-1}^{1} x^i \, x^j \, \mathrm{d}x,$$

gegebene Matrix die sogenannte *Hilbert-Matrix*

$$H_n = \begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \cdots & \frac{1}{n} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \cdots & \frac{1}{n+1} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \ddots & & \frac{1}{n+2} \\ \frac{1}{4} & \frac{1}{5} & \ddots & & \ddots & \vdots \\ \vdots & & & \ddots & & \vdots \\ \frac{1}{n} & \frac{1}{n+1} & \frac{1}{n+2} & \frac{1}{n+3} & \cdots & \frac{1}{2n-1} \end{pmatrix}.$$

Das Lösen eines Gleichungssystems mit der Hilbert-Matrix ist äußerst schwer. Dies liegt an der *Konditionszahl* der Hilbertmatrix, welche angibt, wie sich Rundungsfehler beim Lösungsprozess verstärken. Schon für $n = 4$ liegt die Fehlerverstärkung der Hilbert-Matrix bei 15 000. Diese Matrix muss also unbedingt vermieden werden.

Auch wenn die Wahl der Basisvektoren $\{\phi_1, \ldots, \phi_n\}$ keinen Einfluss auf das Ergebnis hat, so bestimmt sie doch wesentlich den Aufwand bei der Realisierung. Angenommen, die Basis sei ein Orthonormalsystem, d.h. es gelte $(\phi_i, \phi_j) = \delta_{ij}$ für alle $i, j = 1, \ldots, n$. Dann gilt für die Systemmatrix in (3.19)

$$a_{ij} = (\phi_i, \phi_j) = \delta_{ij} \quad \Rightarrow \quad A = I.$$

Die Systemmatrix ist die Einheitsmatrix und die gesuchten Koeffizienten lassen sich sofort ablesen

$$\alpha_i = (f, \phi_i),$$

womit die Bestapproximation durch die Relation

$$p = \sum_{i=1}^{n} (f, \phi_i)\phi_i,$$

trivial gegeben ist. Für dieses vereinfachte Vorgehen benötigen wir zunächst eine Orthonormalbasis von $S \subset H$. Die Orthonormalisierung kann mit Hilfe des bereits diskutieren Gram-Schmidt-Algorithmus durchgeführt werden. Bei Verwendung der Monombasis des $S = P_n$ führt dieser Algorithmus auf die Legendre-Polynome. Wir fassen zusammen:

Korollar 3.80 (Gauss-Approximation mit Polynomen)**.** *Es sei $f \in C[-1, 1]$. Die Bestapproximation $p \in P_n$ bezüglich der L^2-Norm im Raum der Polynome von Grad n eindeutig bestimmt durch:*

$$p(x) = \sum_{i=0}^{n} \frac{1}{\|L_i\|^2_{L^2[-1,1]}} (L_i, f) L_i(x),$$

mit den Legendre-Polynomen

$$L_i(x) := \frac{1}{2^i i!} \frac{d^i}{dx^i} (x^2 - 1)^i.$$

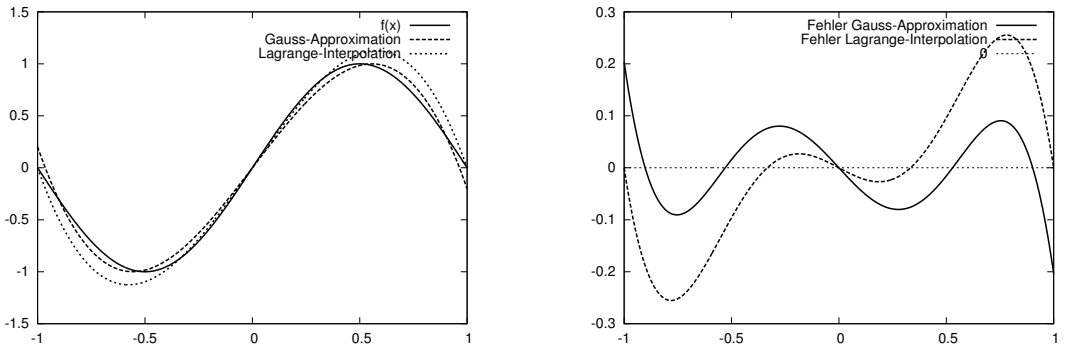

Abbildung 3.10: Gauss-Approximation sowie Lagrange-Interpolation (jeweils kubisch) der Funktion $f(x) = \sin(\pi x)$. Rechts: Fehler der Approximationen.

Die Normierung mit $\|L_n\|^{-2}$ ist notwendig, da die Legendre-Polynome bezüglich der L^2-Norm nicht normiert sind, vergleiche Satz 3.65.

Die Eigenschaft von $p \in P_n$ die beste Approximation zu f im Raum der Polynome zu sein, bedeutet auch, dass p bezüglich der L^2-Norm eine bessere Approximation ist als jede Lagrange-Interpolation zu beliebiger Wahl von Stützstellen $x_0, \ldots, x_n$ in $[-1, 1]$. Hieraus können wir auf triviale Weise eine einfache Fehlerabschätzung herleiten. Dazu sei $f \in C^{n+1}([a, b])$ und $p \in P_n$ die Bestapproximation. Es gilt:

$$\|f - p\|_{L^2([a,b])} \leq \frac{\|f^{n+1}\|_\infty}{(n+1)!} \min_{x_0,\ldots,x_n \in [a,b]} \left(\int_{-1}^{1} \prod_{j=0}^{n} (x - x_j)^2 \, \mathrm{d}x \right)^{\frac{1}{2}}. \tag{3.20}$$

Das Minimum des zweiten Ausdrucks ist nicht einfach zu bestimmen, es können alle Stützstellen im Intervall frei variiert werden. Wir werden aber später auf diesen Punkt zurückkommen und diese Lücke schließen.

Beispiel 3.81 (Gauss-Approximation vs. Lagrange-Interpolation)**.** *Wir approximieren die Funktion $f(x) = \sin(\pi x)$ auf dem Intervall $I = [-1, 1]$ mit Polynomen vom Grad drei. Zunächst erstellen wir in den äquidistant verteilten vier Stützstellen*

$$x_i = -1 + \frac{2i}{3}, \quad i = 0, \ldots, 3$$

das zugehörige Lagrangesche Interpolationspolynom. Aufgrund von $f(x_0) = f(x_3) = 0$ gilt:

$$\begin{aligned} p_L(x) &= \sin(x_1) L_1^{(3)}(x) + \sin(x_2) L_2^{(3)}(x) \\ &= \frac{27\sqrt{3}}{16}(x - x^3). \end{aligned}$$

Als zweite Approximation bestimmen wir die Gauß-Approximation in der L^2-Norm. Hierzu verwenden wir die Darstellung über die normierten Legendre-Polynome $\tilde{L}_n(x)$ auf $[-1, 1]$.

Es gilt aus Symmetriegründen:

$$\int_{-1}^{1} \tilde{L}_0(x)\, f(x)\, dx = \int_{-1}^{1} \tilde{L}_2(x)\, f(x)\, dx = 0,$$

sowie

$$\int_{-1}^{1} \tilde{L}_1(x)\, f(x)\, dx = \frac{\sqrt{6}}{\pi}, \quad \int_{-1}^{1} \tilde{L}_3(x)\, f(x)\, dx = \frac{\sqrt{14}(\pi^2 - 15)}{\pi^3}.$$

Hieraus erhalten wir die Gauß-Approximation

$$\begin{aligned} p_G(x) &= \frac{\sqrt{6}}{\pi}\tilde{L}_1(x) + \frac{\sqrt{14}(\pi^2 - 15)}{\pi^3}\tilde{L}_3(x) \\ &= \frac{5x(7\pi^2x^2 - 105x^2 - 3\pi^2 + 63)}{2\pi^3}. \end{aligned}$$

Für die beiden Approximationen gilt:

$$\|f - p_L\|_{L^2([-1,1])} \approx 0.2, \quad \|f - p_G\|_{L^2([-1,1])} \approx 0.1,$$

d.h., die Gauß-Approximation liefert in der L^2-Norm ein doppelt so gutes Ergebnis. In Abbildung 3.10 zeigen wir beide Approximationen, die Funktion $f(x)$ sowie die Fehler $f(x)-p_l(x)$ sowie $f(x)-p_g(x)$. Hier sehen wir zunächst, dass die Lagrange-Interpolation an den Stützstellen die Interpolationsbedingung, also $f(x_i) = p_L(x_i) = 0$ erfüllt, wo hingegen die Gauss-Approximation gerade am Rand einen großen Fehler aufweist. Der maximale Fehler im Intervall ist hingegen bei der Gauß-Approximation etwas geringer.

Diskrete Gauß-Approximation

Gegeben sei nun eine Messreihe $(x_i, y_i), i = 1, 2, \ldots, m$. Wir suchen eine approximierende Funktion $p \in S$, wobei üblicherweise wieder $S = P_n$ ein einfacher Polynomraum ist. Wir suchen die beste Approximation $p \in S$ bezüglich der euklidischen Norm:

$$|p - y|_2 := \left(\sum_{i=1}^{m} |p(x_i) - y_i|^2 \right)^{\frac{1}{2}} = \min_{\phi \in S} |\phi - y|_2.$$

Im Gegensatz zur Lagrangeschen Interpolationsaufgabe fordern wir in den Stützstellen x_i nicht $p(x_i) = y_i$. Die Diskrete Gauß-Approximation wird üblicherweise zur Approximation von diskreten Messwerten verwendet. Dann gilt meist $m \gg n$, wenn zum Beispiel eine lineare Funktion $p(x) = \alpha_0 + \alpha_1 x$ gesucht wird, die viele (tausend) Messwerte approximiert. Die euklidische Vektornorm ist aus dem euklidischen Skalarprodukt abgeleitet:

$$|x|_2 = (x, x)_2^{\frac{1}{2}},$$

daher kann die bisher entwickelte Theorie unmittelbar angewendet werden. Schlüssel zur Bestimmung der Bestapproximation ist wieder gemäß Satz 3.78 die Charakterisierung über die Orthogonalität:

$$|p - y|_2 = \min_{\phi \in S} |\phi - y|_2 \quad \Leftrightarrow \quad (p - y, \phi)_2 = 0 \quad \forall \phi \in \mathbb{R}^n.$$

Alle Eigenschaften der kontinuierlichen Gauß-Approximation übertragen sich und wir können gleich den Existenzsatz formulieren:

Satz 3.82 (Diskrete Gauß-Approximation)**.** *Es seien durch (x_i, y_i) für $i = 1, \dots, m$ diskrete Datenwerte gegeben. Weiter sei S ein endlich dimensionaler Funktionenraum mit Basis $\{\phi_1, \dots, \phi_n\}$. Die diskrete Gauß-Approximation $p \in S$*

$$|p - y|_2 := \left(\sum_{i=1}^{m} |p(x_i) - y_i|^2 \right)^{\frac{1}{2}} = \min_{\phi \in S} |\phi - y|_2,$$

ist eindeutig bestimmt.

Beweis: Über die Basisdarstellung ist S äquivalent zum euklidischen Raum $\mathbb{R}^n$. Eindeutigkeit und Existenz folgen nun in $\mathbb{R}^n$ wie im Beweis zu Satz 3.79. □

Die Konstruktion der diskreten Gauß-Approximation folgt wieder über die Basisdarstellung

$$p(x) = \sum_{i=1}^{n} \alpha_i \phi_i(x)$$

aus der beschreibenden Orthogonalitätsbeziehung:

$$(p - y, \phi_k)_2 = \sum_{i=1}^{m} (p(x_i) - y_i)\phi_k(x_i) = 0, \quad k = 1, \dots, n$$

Setzen wir für p die Basisdarstellung ein, so erhalten wir das Gleichungssystem:

$$\sum_{j=1}^{n} \alpha_j \underbrace{\sum_{i=1}^{m} \phi_j(x_i)\phi_k(x_i)}_{=(\phi_j, \phi_k)_2} = \underbrace{\sum_{i=1}^{m} y_i \phi_k(x_i)}_{=(y, \phi_k)_2}, \quad k = 1, \dots, n.$$

Der gesuchte Koeffizientenvektor $\alpha = (\alpha_k)_{k=1}^n$ ist gegeben als Lösung des Gleichungssystems:

$$\begin{pmatrix} (\phi_0, \phi_0)_2 & (\phi_0, \phi_1)_2 & \cdots & (\phi_0, \phi_n)_2 \\ (\phi_1, \phi_0)_2 & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ (\phi_n, \phi_0)_2 & \cdots & \cdots & (\phi_n, \phi_n)_2 \end{pmatrix} \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_n \end{pmatrix} = \begin{pmatrix} (y, \phi_0)_2 \\ (y, \phi_1)_2 \\ \vdots \\ (y, \phi_n)_2. \end{pmatrix}$$

Aus der allgemeinen Darstellung kann eine spezielle und häufig genutzte Approximation abgeleitet werden: die Gauß'sche Ausgleichsrechnung:

Beispiel 3.83 (Lineare Ausgleichsrechnung)**.** *Wir suchen die lineare Approximation $p(x) \in P_1$*

$$p(x) = \alpha_0 + \alpha_1 x,$$

zu gegebenen Messwerten. Es seien $\phi_0(x) \equiv 1, \phi_1(x) = x$ *dann ergibt sich das folgende lineare Gleichungssystem:*

$$\begin{pmatrix} (\phi_0,\phi_0)_2 & (\phi_0,\phi_1)_2 \\ (\phi_1,\phi_0)_2 & (\phi_1,\phi_1)_2 \end{pmatrix} \begin{pmatrix} \alpha_0 \\ \alpha_1 \end{pmatrix} = \begin{pmatrix} (y,\phi_0)_2 \\ (y,\phi_1)_2, \end{pmatrix} \Longleftrightarrow \begin{pmatrix} m & \sum x_i \\ \sum x_i & \sum x_i^2 \end{pmatrix} \begin{pmatrix} \alpha_0 \\ \alpha_1 \end{pmatrix} = \begin{pmatrix} \sum y_i \\ \sum x_i y_i. \end{pmatrix}$$

Dieses System ist regulär, falls $m \geq 2$ *und es mindestens zwei Stützwerte* $x_i \neq x_j$ *gibt.*

Wir wenden die lineare Ausgleichsrechnung auf konkrete Messdaten an:

Beispiel 3.84 (Fortsetzung der Mensazählung)**.** *In der Mensazählung haben wir zu gewissen Uhrzeiten* $x_i, i = 0, \ldots, 7$ *(natürlich außerhalb der Vorlesungszeiten) die Anzahl der Personen* $y_i, i = 0, \ldots, 7$ *in der Mensa gezählt. Die folgenden Messdaten haben wir somit erhalten:*

$$\begin{aligned} x_i[hour] &= 10.30, 10.35, 10.45, 11.00, 13.00, 13.02, 13.14, 13.15, \\ y_i[Anz.\ Pers.] &= 60, 70, 107, 90, 300, 325, 325, 350. \end{aligned}$$

Dann ist $m = 8$ *und* $\sum x_i = 94.41$ *und* $\sum x_i^2 = 1.1275e+03$ *und* $\sum x_i y_i = 2.0455e+04$. *Durch lösen des linearen Gleichungssystems erhalten wir die beiden Unbekannten*

$$\alpha_0 = -904.6498, \quad \alpha_1 = 93.8905.$$

Damit erhalten wir die lineare Ausgleichsgerade (illustriert in Abbildung 3.11)

$$g(x) = -904.6498 + 93.8905x.$$

Verallgemeinerung der Gauß-Approximation Die diskrete Gauß-Approximation sowie die Approximation von Funktionen lassen sich durch die Verwendung von gewichteten Skalarprodukten und entsprechenden gewichteten induzierten Normen vereinfachen.

Die Verwendung eines Gewichtes dient dazu, die Approximationsgenauigkeit der Gauß-Approximierenden an den Intervallenden zu verbessern. Für jedes integrable und positive $\omega(x) > 0$ ist durch

$$(f,g)_\omega := \int_a^b f(x)g(x)\omega(x)\,\mathrm{d}x,$$

wieder ein Skalarprodukt mit entsprechender Norm

$$\|f\|_\omega = (f,f)_\omega^{\frac{1}{2}},$$

gegeben. Wir die Gewichtsfunktion $w(x) = \frac{1}{\sqrt{1-x^2}}$ verwendet, so legt die Bestapproximation ein größeres Gewicht auf die Intervallenden. Die hieraus abgeleiteten orthonormalen Polynome sind die bereits diskutierten Tschebyscheff-Polynome.

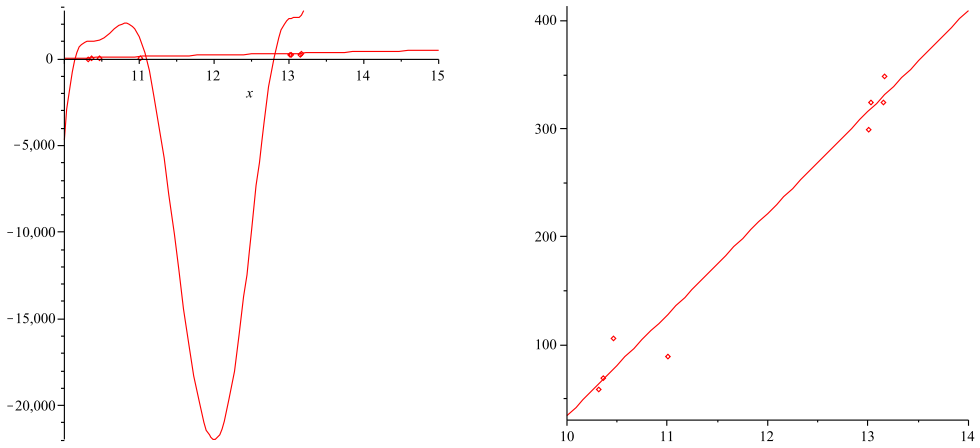

Abbildung 3.11: Lineare Ausgleichsgerade der Gauß-Approximation und entsprechendes Interpolationspolynom im Vergleich (linkes Bild). Die Polynominterpolation macht nur wenig Sinn, da gegen 12.00 Uhr mehr als minus 20000 Menschen in der Mensa sind!!! Lineare Ausgleichsgerade der Gauß-Approximation und gegebene Messdaten sind im rechten Bild zu sehen.

Auch die diskrete Gauß-Approximation lässt sich diesbezüglich verallgemeinern. Hierzu sei durch $\omega \in \mathbb{R}^n$ ein Vektor mit $\omega_i > 0$ gegeben. Dann ist für $x, y \in \mathbb{R}^n$ durch

$$(x,y)_\omega = \sum_{i=1}^{n} x_i y_i \omega_i, \quad |x|_\omega = (x,x)_\omega^{\frac{1}{2}},$$

ein gewichtetes Skalarprodukt mit entsprechender Norm gegeben. Gerade für die diskrete Approximation spielen die Gewichte eine große Rolle in der Anwendung: Angenommen für die Stützstellen und Stützwerte sind Abschätzungen für den Messfehler bekannt. Dann können Stützwerte mit kleinerem Messfehler stärker gewichtet werden.

Sämtliche Sätze zur Gauß-Approximation wurden für beliebige von Skalarprodukten induzierte Normen bewiesen. Daher übertragen sich alle Eigenschaften auch auf die gewichteten Normen.

4 Numerische Lineare Algebra

In der linearen Algebra wird die Struktur von linearen Abbildungen $T : V \to W$ zwischen endlich-dimensionalen Vektorräumen untersucht. In der *numerischen linearen Algebra* befassen wir uns mit einigen praktischen Fragestellungen der linearen Algebra. Schwerpunkt der Anwendung ist das Lösen von linearen Gleichungen, also das Auffinden von $x \in V$, so dass für ein $b \in W$ gilt $T(x) = b$. Weiter werden wir Verfahren zur Berechnung von Eigenwerten linearer Abbildungen sowie zur Orthogonalisierung von Vektoren kennenlernen. Die meisten Probleme der linearen Algebra treten als Teilprobleme anderer Verfahren auf. Große lineare Gleichungssysteme müssen zur Diskretisierung von Differentialgleichungen, aber z.B. auch bei der Approximation von Funktionen gelöst werden. Effiziente numerische Quadraturregeln benötigen zur Konstruktion die Nullstellen orthogonaler Polynome. Orthogonalisierungsverfahren spielen aber auch eine Rolle bei der Lösung von großen Gleichungssystemen.

4.1 Grundlagen der linearen Algebra

Wir sammeln zunächst einige Definitionen und grundlegende Resultate. Es sei V stets ein Vektorraum über dem Körper $\mathbb{K}$. Üblicherweise betrachten wir den Raum der reellwertigen Vektoren $V = \mathbb{R}^n$.

Definition 4.1 (Basis)**.** *Eine Teilmenge $B \subset V$ eines Vektorraums über $\mathbb{K}$ heißt* Basis*, falls sich jedes Element $v \in V$ eindeutig als Linearkombination der Basisvektoren $B = \{v_1, \ldots, v_n\}$ darstellen lässt:*

$$v = \sum_{i=1}^{n} \alpha_i v_i, \quad \alpha_i \in \mathbb{K}.$$

Die eindeutige Darstellbarkeit jedes $v \in V$ durch Basisvektoren erlaubt es, den Vektorraum V mit dem Vektorraum der Koeffizientenvektoren $\alpha \in \mathbb{K}^n$ zu identifizieren. Daher können wir uns in diesem Abschnitt im wesentlichen auf diesen Raum (bzw. auf $\mathbb{R}^n$) beschränken. Alle Eigenschaften und Resultate übertragen sich auf V.

Definition 4.2 (Norm)**.** *Eine Abbildung $\|\cdot\| : V \to \mathbb{R}_+$ heißt* Norm*, falls sie die folgenden drei Eigenschaften besitzt:*

1. Definitheit:	$\|x\| \geq 0, \quad \|x\| = 0 \quad \Rightarrow \quad x = 0,$	
2. Linearität:	$\|\alpha x\| = \lvert\alpha\rvert \, \|x\|$	$\forall x \in V,\ \alpha \in \mathbb{K},$
3. Dreiecksungleichung:	$\|x + y\| \leq \|x\| + \|y\|,$	$\forall x, y \in V.$

Ein Vektorraum mit Norm heißt *normierter Raum*. Häufig verwendete Normen sind die *Maximumsnorm* $\|\cdot\|_\infty$, die *euklidische Norm* $\|\cdot\|_2$ sowie die l_1*-Norm* $\|\cdot\|_1$:

$$\|x\|_\infty := \max_{i=1,\dots,n} |x_i|, \quad \|x\|_2 := \left(\sum_{i=1}^n x_i^2\right)^{\frac{1}{2}}, \quad \|x\|_1 := \sum_{i=1}^n |x_i|.$$

Im Vektorraum $\mathbb{R}^n$ sowie in allen endlich-dimensionalen Vektorräumen gilt der folgende wichtige Satz:

Satz 4.3 (Normäquivalenz)**.** *Zu zwei beliebigen Normen $\|\cdot\|$ sowie $\|\cdot\|'$ im endlich-dimensionalen Vektorraum V existiert eine Konstante $c > 0$ so dass gilt:*

$$\frac{1}{c}\|x\| \le \|x\|' \le c\|x\| \quad \forall x \in V.$$

Dieser Satz bedeutet, dass alle Normen in endlich-dimensionalen Vektorräumen äquivalent sind. Da Normen wesentlich für den Konvergenzbegriff sind, bedeutet dieses Resultat, dass eine Folge $x_n \to x$, welche bzgl. einer Norm $\|\cdot\|$ konvergiert auch bzgl. jeder anderen Norm $\|\cdot\|'$ konvergiert. Auch dieser Zusammenhang ist typisch für endlich-dimensionale Räume und gilt z.B. nicht in Funktionenräumen. So gilt z.B. für die Funktionenfolge $f_n(x) = \exp(-nx^2)$:

$$\|f_n - 0\|_{L^2([-1,1])} \to 0, \qquad \text{jedoch } \|f_n - 0\|_\infty \not\to 0,$$

bezüglich der L^2-Norm sowie der Maximumsnorm:

$$\|f\|_{L^2([-1,1])} := \left(\int_{-1}^1 f(x)\,\mathrm{d}x\right)^{\frac{1}{2}}, \quad \|f\|_\infty := \sup_{x\in[-1,1]} |f(x)|.$$

Neben Normen spielen Räume, in denen ein *Skalarprodukt* existiert eine wichtige Rolle:

Definition 4.4 (Skalarprodukt)**.** *Eine Abbildung $(\cdot,\cdot) : V \times V \to \mathbb{K}$ heißt Skalarprodukt, falls sie die folgenden Eigenschaften besitzt:*

1. Definitheit:	$(x,x) > 0$		$\forall x \in V,\ x \neq 0,$
	$(x,x) = 0$	$\Rightarrow \quad x = 0$	
2. Linearität:	$(x, \alpha y + z) = \alpha(x,y) + (x,z)$		$\forall x,y,z \in V,\ \alpha \in \mathbb{K},$
3. Symmetrie:	$(x,y) = \overline{(y,x)}$		$\forall x,y \in V.$

In reellen Räumen gilt die echte Symmetrie $(x,y) = (y,x)$. Das bekannteste Skalarprodukt ist das euklidische Skalarprodukt für Vektoren $x, y \in \mathbb{R}^n$.

$$(x,y)_2 = x^T y = \sum_{i=1}^n x_i y_i.$$

Vektorräume mit Skalarprodukt werden *Prähilberträume* genannt. (Ein *Prähilbertraum* heißt *Hilbertraum*, falls er vollständig, also ein *Banachraum* ist). Komplexe Vektorräumen mit Skalarprodukt nennt man auch *unitäre Räume*, im Reellen spricht man von *euklidischen Räumen*.

Skalarprodukte sind eng mit Normen verwandt:

Satz 4.5 (Induzierte Norm)**.** *Es sei V ein Vektorraum mit Skalarprodukt. Dann ist durch*

$$\|x\| = \sqrt{(x,x)}, \quad x \in V,$$

auf V die induzierte Norm *gegeben.*

Beweis: Übung! □

Die euklidische Norm ist die vom euklidischen Skalarprodukt induzierte Norm:

$$\|x\|_2 = (x,x)_2^{\frac{1}{2}}.$$

Einige wichtige Sätze gelten für Paare aus Skalarprodukt und induzierter Norm:

Satz 4.6. *Es sei V ein Vektorraum mit Skalarprodukt $(\cdot,\cdot)$ und induzierter Norm $\|\cdot\|$. Dann gilt die* Cauchy-Schwarz Ungleichung*:*

$$|(x,y)| \leq \|x\|\,\|y\| \quad \forall x, y \in V,$$

sowie die Parallelogrammidentität*:*

$$\|x+y\|^2 + \|x-y\|^2 = 2\|x\|^2 + 2\|y\|^2 \quad \forall x, y \in V.$$

Beweis: Übung. □

Mit Hilfe des Skalarproduktes 4.4 können wir den Begriff der *Orthogonalität* einführen: zwei Vektoren $x, y \in V$ heißen orthogonal, falls $(x,y) = 0$.

Eine der Aufgaben der numerischen linearen Algebra ist die Orthogonalisierung (oder auch Orthonormalisierung) von gegebenen Systemen von Vektoren:

Definition 4.7 (Orthonormalbasis)**.** *Eine Basis $B = \{v_1, \ldots, v_n\}$ von V heißt* Orthogonalbasis *bezüglich des Skalarproduktes $(\cdot,\cdot)$, falls gilt:*

$$(\phi_i, \phi_j) = 0 \quad \forall i \neq j,$$

und Orthonormalbasis *falls gilt:*

$$(\phi_i, \phi_j) = \delta_{ij},$$

mit dem Kronecker-Symbol δ_{ij}.

Mit unterschiedlichen Skalarprodukten existieren unterschiedliche Orthonormalbasen zu ein und demselben Vektorraum V. Orthogonalität stimmt dann im Allgemeinen nicht mit dem geometrischen Orthogonalitätsbegriff des euklidischen Raums überein:

Beispiel 4.8 (Skalarprodukte und Orthonormalbasis)**.** *Es sei $V = \mathbb{R}^2$. Durch*

$$(x, y)_2 := x_1 y_1 + x_2 y_2, \quad (x, y)_\omega = 2\, x_1 y_1 + x_2 y_2,$$

sind zwei verschiedene Skalarprodukte gegeben. Das erste ist das euklidische Skalarprodukt. Die Skalarprodukteigenschaften des zweiten sind einfach zu überprüfen. Durch

$$x_1 = \frac{1}{\sqrt{2}}(1, 1)^T, \quad x_2 = \frac{1}{\sqrt{2}}(-1, 1)^T,$$

ist eine Orthonormalbasis bezüglich $(\cdot, \cdot)_2$ gegeben. Es gilt jedoch:

$$(x_1, x_2)_\omega = \frac{1}{2}(2 - 1) = \frac{1}{\sqrt{2}} \neq 0.$$

Eine Orthonormalbasis erhalten wir z.B. durch

$$x_1 = \frac{1}{\sqrt{3}}(1, 1)^T, \quad x_2 = \frac{1}{2}(-1, 2)^T,$$

Orthonormalbasen werden für zahlreiche numerische Verfahren benötigt, bei der Gauß'schen Quadratur, bei der Gauß-Approximation von Funktionen und z.B. für die QR-Zerlegung einer Matrix.

Wir betrachten nun den Vektorraum aller $\mathbb{R}^{n\times m}$-Matrizen. Auch dieser Vektorraum ist endlich-dimensional und prinzipiell können wir den Vektorraum der $n \times m$-Matrizen mit dem Vektorraum der (nm)-Vektoren identifizieren. Von besonderem Interesse ist für uns der Vektorraum der quadratischen $\mathbb{R}^{n\times n}$-Matrizen:

Definition 4.9 (Eigenwerte, Eigenvektoren)**.** *Die* Eigenwerte λ *einer Matrix $A \in \mathbb{R}^{n\times n}$ sind definiert als Nullstellen des charakteristischen Polynoms:*

$$\det(A - \lambda I) = 0.$$

Die Menge aller Eigenwerte einer Matrix heißt das Spektrum

$$\sigma(A) := \{\lambda \in \mathbb{C},\ \lambda \text{ Eigenwert von } A\}.$$

Der Spektralradius $\mathrm{spr} : \mathbb{R}^{n\times n} \to \mathbb{R}_+$ *ist der betragsmäßig größte Eigenwert:*

$$\mathrm{spr}(A) := \max\{|\lambda|,\ \lambda \in \sigma(A)\}.$$

Ein Element $w \in \mathbb{R}^n$ heißt Eigenvektor *zu Eigenwert λ, falls gilt:*

$$Aw = \lambda w.$$

Bei der Untersuchung von linearen Gleichungssystemen $Ax = b$ stellt sich zunächst die Frage, ob ein solches Gleichungssystem überhaupt lösbar ist und ob die Lösung eindeutig ist. Wir fassen zusammen:

Satz 4.10 (Reguläre Matrix)**.** *Für eine quadratische Matrix $A \in \mathbb{R}^{n\times n}$ sind die folgenden Aussagen äquivalent:*

1. *Die Matrix A ist regulär .*
2. *Die transponierte Matrix A^T ist regulär.*
3. *Die Inverse A^{-1} ist regulär.*
4. *Das lineare Gleichungssystem $Ax = b$ ist für jedes $b \in \mathbb{R}^n$ eindeutig lösbar.*
5. *Es gilt* $\det(A) \neq 0$.
6. *Alle Eigenwerte von A sind ungleich Null.*

Wir definieren weiter:

Definition 4.11 (positiv definit)**.** *Eine Matrix $A \in \mathbb{K}^{n\times n}$ heißt positiv definit, falls*

$$(Ax, x) > 0 \quad \forall x \neq 0.$$

Umgekehrt können wir aus der definierenden Eigenschaft der positiven Definitheit einer Matrix ablesen: falls A positiv definit ist, so ist durch $(A\cdot, \cdot)$ ein Skalarprodukt gegeben. Es gilt:

Satz 4.12 (Positiv definite Matrizen)**.** *Es sei $A \in \mathbb{R}^{n\times n}$ eine symmetrische Matrix. Dann ist A genau dann positiv definit, falls alle (reellen) Eigenwerte von A positiv sind. Dann sind alle Diagonalelemente von A positiv und das betragsmäßig größte Element steht auf der Diagonalen.*

Beweis: *(i)* Es sei A eine symmetrische Matrix mit einer Orthonormalbasis aus Eigenvektoren $w_1, \ldots, w_n$. A sei positiv definit. Dann gilt für beliebigen Eigenvektor w_i mit Eigenwert λ_i:

$$0 < (Aw_i, w_i) = \lambda_i(w_i, w_i) = \lambda_i.$$

Umgekehrt seien alle λ_i positiv. Für $x = \sum_{i=1}^n \alpha_i w_i$ gilt:

$$(Ax, x) = \sum_{i,j}(\lambda_i\alpha_i\omega_i, \alpha_j\omega_j) = \sum_i \lambda_i\alpha_i^2 > 0.$$

(ii) A sei nun eine reelle, positiv definite Matrix. Es sei e_i der i-te Einheitsvektor. Dann gilt:

$$0 < (Ae_i, e_i) = a_{ii}.$$

D.h., alle Diagonalelemente sind positiv.

(iii) Entsprechend wählen wir nun $x = e_i - \text{sign}(a_{ij})e_j$. Wir nehmen an, dass $a_{ji} = a_{ij}$ das betragsmäßig größte Element der Matrix sei. Dann gilt:

$$0 < (Ax, x) = a_{ii} - \text{sign}(a_{ij})(a_{ij} + a_{ji}) + a_{jj} = a_{ii} + a_{ji} - 2|a_{ij}| \leq 0.$$

Aus diesem Widerspruch folgt die letzte Aussage des Satzes: das betragsmäßig größte Element muss ein Diagonalelement sein. □

Für Normen auf dem Raum der Matrizen definieren wir weitere Struktureigenschaften:

Definition 4.13 (Matrixnormen)**.** *Eine Norm* $\|\cdot\| : \mathbb{R}^{n\times n} \to \mathbb{R}_+$ *heißt* Matrizenorm*, falls sie submultiplikativ ist:*

$$\|AB\| \leq \|A\|\,\|B\| \quad \forall A, B \in \mathbb{R}^{n\times n}.$$

Sie heißt verträglich *mit einer Vektornorm* $\|\cdot\| : \mathbb{R}^n \to \mathbb{R}_+$*, falls gilt:*

$$\|Ax\| \leq \|A\|\,\|x\| \quad \forall A \in \mathbb{R}^{n\times n},\ x \in \mathbb{R}^n.$$

Eine Matrixnorm $\|\cdot\| : \mathbb{R}^{n\times n} \to \mathbb{R}_+$ *heißt von einer Vektornorm* $\|\cdot\| : \mathbb{R}^n \to \mathbb{R}_+$ induziert*, falls gilt:*

$$\|A\| := \max_{x\neq 0} \frac{\|Ax\|}{\|x\|}.$$

Es ist leicht nachzuweisen, dass jede von einer Vektornorm induzierte Matrixnorm mit dieser auch verträglich ist. Verträglich mit der euklidischen Norm ist aber auch die *Frobeniusnorm*

$$\|A\|_F := \left(\sum_{i,j=1}^{n} a_{ij}^2\right)^{\frac{1}{2}},$$

welche nicht von einer Vektor-Norm induziert ist. Für allgemeine Normen auf dem Vektorraum der Matrizen gilt nicht notwendigerweise $\|I\| = 1$, wobei $I \in \mathbb{R}^{n\times n}$ die Einheitsmatrix ist. Dieser Zusammenhang gilt aber für jede von einer Vektornorm induzierte Matrixnorm. Wir fassen im folgenden Satz die wesentlichen induzierten Matrixnormen zusammen:

Satz 4.14 (Induzierte Matrixnormen)**.** *Die aus der euklidischen Vektornorm, der Maximumsnorm sowie der* l_1*-Norm induzierten Matrixnormen sind die* Spektralnorm $\|\cdot\|_2$*, die maximale Zeilensumme* $\|\cdot\|_\infty$*, sowie die maximale Spaltensumme* $\|\cdot\|_1$*:*

$$\|A\|_2 = \sqrt{\text{spr}(A^T A)}, \quad \text{spr}(B) := \max\{|\lambda|,\ \lambda \textit{ ist Eigenwert von } B\},$$
$$\|A\|_\infty = \max_{i=1,\dots,n} \sum_{j=1}^{m} |a_{ij}|,$$
$$\|A\|_1 = \max_{j=1,\dots,m} \sum_{i=1}^{n} |a_{ij}|.$$

Beweis: *(i)* Es gilt:

$$\|A\|_2^2 = \sup_{x\neq 0} \frac{\|Ax\|_2^2}{\|x\|_2^2} = \sup_{x\neq 0} \frac{(Ax, Ax)}{\|x\|_2^2} = \sup_{x\neq 0} \frac{(A^TAx, x)}{\|x\|_2^2}$$

Die Matrix A^TA ist symmetrisch und hat als solche nur reelle Eigenwerte. Sie besitzt eine Orthonormalbasis $\omega_i \in \mathbb{R}^n$ von Eigenvektoren mit Eigenwerten $\lambda_i \geq 0$. Alle Eigenwerte λ_i sind größer gleich Null, denn:

$$\lambda_i = \lambda_i(\omega_i, \omega_i) = (A^TA\omega_i, \omega_i) = (A\omega_i, A\omega_i) = \|A\omega_i\|^2 \geq 0.$$

Es sei $x \in \mathbb{R}^n$ beliebig mit Basisdarstellung $x = \sum_i \alpha_i\omega_i$. Es gilt dann wegen $(\omega_i, \omega_j)_2 = \delta_{ij}$ die Beziehung $\|x\|_2^2 = \sum_i \alpha_i^2$ sowie mit dem Koeffizientenvektor $\alpha \in \mathbb{R}^n$:

$$\begin{aligned}\|A\|_2^2 &= \sup_{\alpha\neq 0} \frac{(\sum_i \alpha_i A^TA\omega_i, \sum_i \alpha_i\omega_i)}{\sum_i \alpha_i^2} = \sup_{\alpha\neq 0} \frac{(\sum_i \alpha_i\lambda_i\omega_i, \sum_i \alpha_i\omega_i)}{\sum_i \alpha_i^2} \\ &= \sup_{\alpha\neq 0} \frac{\sum_i \lambda_i\alpha_i^2}{\sum_i \alpha_i^2} \leq \max_i \lambda_i.\end{aligned}$$

Es sei nun umgekehrt durch λ_k der größte Eigenwert gegeben. Dann gilt für $\alpha_i = \delta_{ki}$:

$$0 \leq \max_i \lambda_i = \lambda_k = \sum_i \lambda_i\alpha_i^2 \leq \|A\|_2^2.$$

(ii) Wir zeigen das Ergebnis exemplarisch für die Maximumsnorm:

$$\|Ax\|_\infty = \sup_{\|x\|_\infty=1} \left(\max_i \sum_{j=1}^m a_{ij}x_j \right).$$

Jede Summe nimmt ihr Maximum an, falls $|x_j| = 1$ und falls das Vorzeichen x_j so gewählt wird, dass $a_{ij}x_j \geq 0$ für alle $j = 1, \ldots, m$. Dann gilt:

$$\|Ax\|_\infty = \max_i \sum_{j=1}^m |a_{ij}|.$$

□

Als Nebenresultat erhalten wir, dass jeder Eigenwert betragsmäßig durch die Spektralnorm der Matrix A beschränkt ist. Es gilt sogar mit beliebiger Matrixnorm und verträglicher Vektornorm für einen Eigenwert λ mit zugehörigem Eigenvektor $w \in \mathbb{R}^n$ von A:

$$|\lambda| = \frac{|\lambda|\,\|w\|}{\|w\|} = \frac{\|Aw\|}{\|w\|} \leq \frac{\|A\|\,\|w\|}{\|w\|} = \|A\|.$$

Eine einfache Schranke für den betragsmäßig größten Eigenwert erhält man also durch Analyse beliebiger (verträglicher) Matrixnormen.

Aus Satz 4.14 folgern wir weiter, dass für symmetrische Matrizen die $\|\cdot\|_2$-Norm mit dem Spektralradius der Matrix selbst übereinstimmt, daher der Name Spektralnorm.

4.2 Lösungsmethoden für lineare Gleichungssysteme

Das Lösen von linearen Gleichungssystemen ist eine der wichtigsten numerischen Aufgaben. Viele Probleme sind nicht unmittelbar als lineares Gleichungssystem formuliert, in vielen Anwendungen treten allerdings ständig (unter Umständen sehr große) lineare Gleichungssysteme auf. Groß bedeutet in Anwendungen, dass Gleichungssysteme mit vielen Millionen Unbekannten gelöst werden müssen. Wir werden uns in diesem Abschnitt ausschließlich mit reellwertigen Matrizen $A \in \mathbb{R}^{n\times n}$ befassen. Methoden zum Lösen von linearen Gleichungsystemen klassifiziert man als *direkte Verfahren*, welche die Lösung des Gleichungssystems unmittelbar und bis auf Rundungsfehlereinflüsse exakt berechnen und *iterative Verfahren*, welche die Lösung durch eine Fixpunktiteration approximieren. Hier befassen wir uns ausschließlich mit direkten Methoden. Iterative Verfahren sind Gegenstand von Kapitel 5.

Als einführendes Beispiel betrachten wir das einfache Gleichungssystem

$$\begin{pmatrix} 0.988 & 0.959 \\ 0.992 & 0.963 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 0.087 \\ 0.087 \end{pmatrix}$$

mit der Lösung $(x, y)^T = (3, -3)^T$. Wir bestimmen die Lösung numerisch durch Gauß-Elimination mit dreistelliger Rechengenauigkeit. Die Gauß-Elimination setzen wir dabei als bekannt voraus:

$$\left(\begin{array}{cc|c} 0.988 & 0.959 & 0.087 \\ 0.992 & 0.963 & 0.087 \end{array}\right) \quad \times 0.992/0.998$$

$$\left(\begin{array}{cc|c} 0.988 & 0.959 & 0.087 \\ 0.988 & 0.960 & 0.0866 \end{array}\right) \quad \downarrow -$$

$$\left(\begin{array}{cc|c} 0.988 & 0.959 & 0.087 \\ 0 & 0.001 & -0.0004 \end{array}\right)$$

Mit Hilfe der Gauß-Elimination haben wir die Matrix A auf eine Dreiecksgestalt transformiert. Die rechte Seite b wurde entsprechend modifiziert. Das resultierende Dreiecksystem kann nun sehr einfach durch Rückwärtseinsetzen gelöst werden (bei dreistelliger Rechnung):

$$0.001y = -0.0004 \;\Rightarrow\; y = -0.4, \quad 0.988x = 0.087 - 0.959 \cdot (-0.4) \approx 0.471 \;\Rightarrow\; x = 0.477$$

Wir erhalten also $(x, y) = (0.477, -0.4)$. Der relative Fehler der numerischen Lösung beträgt somit fast 90%. Die numerische Aufgabe, ein Gleichungssystem zu lösen scheint also entweder generell sehr schlecht konditioniert zu sein (siehe Kapitel 1), oder aber das Eliminationsverfahren zur Lösung eines linearen Gleichungssystems ist numerisch sehr instabil und nicht gut geeignet. Der Frage nach der Konditionierung und Stabilität gehen wir im folgenden Abschnitt auf den Grund.

In der praktischen Anwendung treten sehr große Gleichungssysteme $Ax = b$ auf. Bei der numerischen Approximation von partiellen Differentialgleichungen müssen Matrizen $A \in \mathbb{R}^{n\times n}$ der Dimension $n > 1\,000\,000$ invertiert werden. (Man nennt das Lösen eines linearen Gleichungssystems oft invertieren, auch wenn die Inverse A^{-1} nicht wirklich aufgestellt wird). Hinzu kommt, dass ein solches lineares Gleichungssystem oft wiederholt (viele 1 000 mal) gelöst werden muss (z.B. bei der Diskretisierung von instationären Differentialgleichungen, oder bei nichtlinearen Gleichungen). Neben der Stabilität des Lösungsprozesses wird auch die numerische Effizienz eine große Rolle spielen. Man versuche, eine 20×20-Matrix mit dem Gauß'schen Eliminationsverfahren zu invertieren!

4.2.1 Störungstheorie & Stabilitätsanalyse von linearen Gleichungssystemen

Zu einer quadratischen, regulären Matrix $A \in \mathbb{R}^{n\times n}$ sowie einem Vektor $b \in \mathbb{R}^n$ betrachten wir das lineare Gleichungssystem

$$Ax = b.$$

Durch numerische Fehler, Rundungsfehler oder bloße Eingabefehler zum Beispiel durch Messungenauigkeiten liegen sowohl A als auch b nur gestört vor:

$$\tilde{A}\tilde{x} = \tilde{b}.$$

Dabei sei $\tilde{A} = A + \delta A$ sowie $\tilde{b} = b + \delta b$, mit Störung δA sowie δb.

Wir kommen nun zur Kernaussage dieses Abschnitts und wollen die Fehlerverstärkung beim Lösen von linearen Gleichungssystemen betrachten. Fehler können dabei in der Matrix A als auch in der rechten Seite b auftauchen. Wir betrachten zunächst Störungen der rechten Seite:

Satz 4.15 (Störung der rechten Seite)**.** *Durch $x \in \mathbb{R}^n$ sei die Lösung des linearen Gleichungsystems $Ax = b$ gegeben. Es sei δb eine Störung der rechten Seite $\tilde{b} = b + \delta b$ und $\tilde{x}$ die Lösung des gestörten Gleichungssystems. Dann gilt:*

$$\frac{\|\delta x\|}{\|x\|} \leq \operatorname{cond}(A)\frac{\|\delta b\|}{\|b\|},$$

mit der Konditionszahl *der Matrix*

$$\operatorname{cond}(A) = \|A\| \cdot \|A^{-1}\|.$$

Beweis: Es sei $\|\cdot\|$ eine beliebige Matrixnorm mit verträglicher Vektornorm $\|\cdot\|$. Für die Lösung $x \in \mathbb{R}^n$ und gestörte Lösung $\tilde{x} \in \mathbb{R}^n$ gilt:

$$\tilde{x} - x = A^{-1}(A\tilde{x} - Ax) = A^{-1}(\tilde{b} - b) = A^{-1}\delta b.$$

Also:

$$\frac{\|\delta x\|}{\|x\|} \leq \|A^{-1}\|\frac{\|\delta b\|}{\|x\|} \cdot \frac{\|b\|}{\|b\|} = \|A^{-1}\|\frac{\|\delta b\|}{\|b\|} \cdot \frac{\|Ax\|}{\|x\|} \leq \underbrace{\|A\| \cdot \|A^{-1}\|}_{=:\operatorname{cond}(A)} \frac{\|\delta b\|}{\|b\|}.$$

□

Bemerkung 4.16 (Konditionszahl einer Matrix)**.** *Die Konditionszahl einer Matrix spielt die entscheidende Rolle in der numerischen Linearen Algebra. Betrachten wir etwa die Konditionierung der Matrix-Vektor Multiplikation $y = Ax$ so erhalten wir bei gestörter Eingabe $\tilde{x} = x + \delta x$ wieder:*

$$\frac{\|\delta y\|}{\|y\|} \leq \mathrm{cond}(A)\frac{\|\delta x\|}{\|x\|}$$

Die Konditionszahl einer Matrix hängt von der gewählten Norm ab. Da jedoch alle Matrixnormen im $\mathbb{R}^{n\times n}$ äquivalent sind, sind auch alle Konditionsbegriffe äquivalent. Mit Satz 4.14 folgern wir für symmetrische Matrizen für den Spezialfall $\mathrm{cond}_s(A)$:

$$\mathrm{cond}_2(A) = \|A\|_2 \cdot \|A^{-1}\|_2 = \frac{\max\{|\lambda|,\ \lambda \textit{ Eigenwert von } A\}}{\min\{|\lambda|,\ \lambda \textit{ Eigenwert von } A\}}.$$

Wir betrachten nun den Fall, dass die Matrix A eines linearen Gleichungssystems mit einer Störung δA versehen ist. Es stellt sich zunächst die Frage, ob die gestörte Matrix $\tilde{A} = A + \delta A$ überhaupt noch regulär ist.

Hilfsatz 4.17. *Es sei durch $\|\cdot\|$ eine von der Vektornorm induzierte Matrixnorm gegeben. Weiter sei $B \in \mathbb{R}^{n\times n}$ eine Matrix mit $\|B\| < 1$. Dann ist die Matrix $I + B$ regulär und es gilt die Abschätzung:*

$$\|(I + B)^{-1}\| \leq \frac{1}{1 - \|B\|}.$$

Beweis: Es gilt:

$$\|(I + B)x\| \geq \|x\| - \|Bx\| \geq (1 - \|B\|)\|x\|$$

Da $1 - \|B\| > 0$ ist durch $I + B$ eine injektive Abbildung gegeben. Also ist $I + B$ eine reguläre Matrix. Weiter gilt:

$$\begin{aligned} 1 = \|I\| &= \|(I + B)(I + B)^{-1}\| = \|(I + B)^{-1} + B(I + B)^{-1}\| \\ &\geq \|(I + B)^{-1}\| - \|B\|\,\|(I + B)^{-1}\| = \|(I + B)^{-1}\|(1 - \|B\|) > 0. \end{aligned}$$

□

Mit diesem Hilfsatz können wir im Folgenden auch auf die Störung der Matrix eingehen:

Satz 4.18 (Störung der Matrix)**.** *Es sei $x \in \mathbb{R}^n$ die Lösung des linearen Gleichungssystems $Ax = b$ und $\tilde{A} = A + \delta A$ einer gestörte Matrix mit $\|\delta A\| \leq \|A^{-1}\|^{-1}$. Für die gestörte Lösung $\tilde{x}$ gilt:*

$$\frac{\|\delta x\|}{\|x\|} \leq \frac{\mathrm{cond}(A)}{1 - \mathrm{cond}(A)\|\delta A\|/\|A\|}\frac{\|\delta A\|}{\|A\|}.$$

Beweis: Wir betrachten den Fall, dass die rechte Seite nicht gestört ist: $\delta b = 0$. Dann gilt für Lösung x sowie gestörte Lösung $\tilde{x}$ und Fehler $\delta x := \tilde{x} - x$:

$$\begin{aligned} &(A + \delta A)\tilde{x} = b \\ &(A + \delta A)x = b + \delta A x \end{aligned} \quad \Rightarrow \quad \delta x = -[A + \delta A]^{-1}\delta A x.$$

Da laut Voraussetzung $\|A^{-1}\delta A\| \leq \|A^{-1}\| \, \|\delta A\| < 1$ folgt mit Hilfsatz 4.17:

$$\|\delta x\| \leq \|A^{-1}[I + A^{-1}\delta A]^{-1}\delta A\| \, \|x\| \leq \frac{\|A^{-1}\|}{1 - \|A^{-1}\delta A\|} \|\delta A\| \, \|x\|.$$

Das Ergebnis erhalten wir durch zweimaliges Erweitern mit $\|A\|/\|A\|$ sowie mit der Voraussetzung $0 \leq \|A^{-1}\delta A\| < 1$. □

Diese beiden Störungssätze können einfach kombiniert werden, um gleichzeitig die Störung durch rechte Seite und Matrix abschätzen zu können:

Satz 4.19 (Störungssatz für lineare Gleichungssysteme)**.** *Es sei $x \in \mathbb{R}^n$ die Lösung des linearen Gleichungssystems $Ax = b$ mit einer regulären Matrix $A \in \mathbb{R}^{n\times n}$. Für die Lösung $\tilde{x} \in \mathbb{R}^n$ des gestörten Systems $\tilde{A}\tilde{x} = \tilde{b}$ mit Störungen $\delta b = \tilde{b} - b$ und $\delta A = \tilde{A} - A$ gilt unter der Voraussetzung*

$$\|\delta A\| < \frac{1}{\|A^{-1}\|}$$

die Abschätzung:

$$\frac{\|\delta x\|}{\|x\|} \leq \frac{\operatorname{cond}(A)}{1 - \operatorname{cond}(A)\|\delta A\|/\|A\|} \left(\frac{\|\delta b\|}{\|b\|} + \frac{\|\delta A\|}{\|A\|} \right),$$

mit der Konditionszahl

$$\operatorname{cond}(A) = \|A\| \, \|A^{-1}\|.$$

Beweis: Wir kombinieren die Aussagen von Satz 4.15 und 4.18. Hierzu sei x die Lösung von $Ax = b$, $\tilde{x}$ die gestörte Lösung $\tilde{A}\tilde{x} = \tilde{b}$ und $\hat{x}$ die Lösung zu gestörter rechter Seite $A\hat{x} = \tilde{b}$. Dann gilt:

$$\|x - \tilde{x}\| \leq \|x - \hat{x}\| + \|\hat{x} - \tilde{x}\| \leq \operatorname{cond}(A) \frac{\|\delta b\|}{\|b\|} + \frac{\operatorname{cond}(A)}{1 - \operatorname{cond}(A)\frac{\|\delta A\|}{\|A\|}} \frac{\|\delta A\|}{\|A\|}.$$

Die Aussage folgt mit (beachte $\|\delta A\| < \|A^{-1}\|^{-1}$)

$$0 \leq \operatorname{cond}(A) \frac{\|\delta A\|}{\|A\|} < \|A\| \, \|A^{-1}\| \, \frac{1}{\|A\| \, \|A^{-1}\|} \leq 1.$$

□

Mit diesem Ergebnis kehren wir zum einführenden Beispiel aus Abschnitt 4.2 zurück:

$$A = \begin{pmatrix} 0.988 & 0.959 \\ 0.992 & 0.963 \end{pmatrix}$$

$$\Rightarrow \quad A^{-1} \approx \begin{pmatrix} 8301 & -8267 \\ -8552 & 8517 \end{pmatrix}$$

In der maximalen Zeilensummennorm $\|\cdot\|_\infty$ gilt:

$$\|A\|_1 = 1.955, \quad \|A^{-1}\|_1 \approx 17079, \quad \operatorname{cond}_1(A) \approx 33370.$$

Das Lösen eines linearen Gleichungssystems mit der Matrix A ist also äußerst schlecht konditioniert. Hieraus resultiert der enorme Rundungsfehler im Beispiel zu Beginn des Kapitels. Wir halten hier fest: bei großer Konditionszahl ist die Konditionierung des Problems sehr schlecht, d.h. der große Fehler ist immanent mit der Aufgabe verbunden und nicht unbedingt auf ein Stabilitätsproblem des Verfahrens zurückzuführen.

4.2.2 Das Gauß'sche Eliminationsverfahren und die LR-Zerlegung

Das wichtigste Verfahren zum Lösen eines linearen Gleichungsystems $Ax = b$ mit quadratischer Matrix $A \in \mathbb{R}^{n\times n}$ ist das Gauß'sche Eliminationsverfahren: durch Elimination der Einträge unterhalb der Diagonale wird die Matrix $A \in \mathbb{R}^{n\times n}$ in den ersten $n-1$ Schritten auf eine obere rechte Dreiecksgestalt gebracht:

$$\begin{pmatrix} * & * & * & \cdots & * \\ * & * & * & & * \\ * & * & * & & * \\ \vdots & & & \ddots & \vdots \\ * & * & * & \cdots & * \end{pmatrix} \to \begin{pmatrix} * & * & * & \cdots & * \\ 0 & * & * & & * \\ 0 & * & * & & * \\ \vdots & & & \ddots & \vdots \\ 0 & * & * & \cdots & * \end{pmatrix} \to \cdots \to \begin{pmatrix} * & * & * & \cdots & * \\ 0 & * & * & & * \\ 0 & 0 & * & & * \\ \vdots & & & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & * \end{pmatrix}$$

Mit der Dreiecksmatrix $R \in \mathbb{R}^{n\times n}$ ($R = (r_{ij})_{i,j=1}^n$ mit $r_{ij} = 0$ für $i > j$) kann das reduzierte Gleichungssystem

$$Rx = \tilde{b},$$

durch *Rückwärtseinsetzen* gelöst werden. Wir betrachten zunächst diese Rückwärtseinsetzen:

Algorithmus 4.20 (Rückwärtseinsetzen)**.** *Es sei $\mathbb{R}^{n\times n}$ eine rechte obere Dreiecksmatrix. Die Lösung $x \in \mathbb{R}^n$ von $Rx = b$ ist gegeben durch:*

1. *Setze $x_n = r_{nn}^{-1} b_n$*
2. *Für $i = n-1, \ldots, 1$*

$$x_i = r_{ii}^{-1} \left(b_i - \sum_{j=i+1}^{n} r_{ij} x_j \right).$$

Es gilt:

Satz 4.21 (Rückwärtseinsetzen)**.** *Es sei $\mathbb{R} \in \mathbb{R}^{n\times n}$ eine rechte obere Dreiecksmatrix mit $r_{ii} \neq 0$. Dann ist die Matrix R regulär und das Rückwärtseinsetzen erfordert*

$$N_R(n) = \frac{n^2}{2} + O(n)$$

arithmetische Operationen.

BEWEIS: Es gilt $\det(R) = \prod r_{ii} \neq 0$. Also ist die Matrix R regulär.

Jeder Schritt der Rückwärtseinsetzen besteht aus Additionen, Multiplikationen und Division durch die Diagonalelemente. Bei $r_{ii} \neq 0$ ist jeder Schritt durchführbar.

Zur Berechnung von x_i sind $n-(i+1)$ Multiplikationen und Additionen notwendig. Hinzu kommt eine Division pro Schritt. Dies ergibt:

$$n + \sum_{i=1}^{n-1} \big(n - (i+1)\big) = n + (n-1)n - (n-1) - \frac{n(n-1)}{2} = \frac{n^2 - n + 2}{2}.$$

□

Die Transformation von A auf Dreiecksgestalt geschieht durch zeilenweise Elimination:

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \ddots & a_{3n} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{nn} \end{pmatrix} \to \begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & \dots & a_{2n}^{(1)} \\ 0 & a_{32}^{(1)} & a_{33}^{(3)} & \ddots & a_{3n}^{(1)} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & a_{n2}^{(1)} & a_{n3}^{(1)} & \dots & a_{nn}^{(1)} \end{pmatrix} \to$$

$$\to \begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & \dots & a_{2n}^{(1)} \\ 0 & 0 & a_{33}^{(2)} & \ddots & a_{3n}^{(2)} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & a_{n3}^{(2)} & \dots & a_{nn}^{(2)} \end{pmatrix} \to \dots \to A^{(n-1)} =: R.$$

Beginnend mit $A^{(0)} := A$ werden sukzessive Matrizen $A^{(i)}$ erzeugt, mit $A^{(n-1)} =: R$. Dabei wird in Schritt i des Verfahrens die Spalte i-te Spalte von $A^{(i-1)}$ unterhalb der Diagonalen eliminiert. Dies geschieht durch Subtraktion des $g_k^{(i)}$-fachen der i-ten Zeile von der k-ten. Hierbei gilt:

$$g_k^{(i)} := \frac{a_{ki}^{(i-1)}}{a_{ii}^{(i-1)}}.$$

Im i-ten Eliminationschritt bleiben die ersten $i-1$ Zeilen und Spalten unverändert. Der i-te Eliminationsschritt lässt sich kompakt in Form einer Matrix-Matrix Multiplikation schreiben

$$A^{(i)} = F^{(i)} A^{(i-1)},$$

mit der Eliminationsmatrix (alle nicht spezifizierten Einträge sind Null):

$$F^{(i)} := \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & -g_{i+1}^{(i)} & \ddots & \\ & & \vdots & & \ddots \\ & & -g_n^{(i)} & & 1 \end{pmatrix}, \quad g_k^{(i)} := \frac{a_{ki}^{(i-1)}}{a_{ii}^{(i-1)}}.$$

Mehrfache Anwendung von Eliminationsmatrizen führt zu der Darstellung:

$$R = A^{(n-1)} = F^{(n-1)}A^{(n-2)} = F^{(n-1)}F^{(n-2)}A^{(n-3)} = \underbrace{F^{(n-1)} \cdots F^{(1)}}_{=:F} A^{(0)} = FA. \quad (4.1)$$

Matrizen mit der Gestalt der Eliminationsmatrizen $F^{(i)}$ heißen *Frobeniusmatrix*. Es gilt der folgende Satz:

Satz 4.22 (Frobeniusmatrix)**.** *Jede Frobeniusmatrix $F^{(i)} \in \mathbb{R}^{n\times n}$ ist regulär und es gilt:*

$$F^{(i)} := \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & -g_{i+1} & \ddots & \\ & & \vdots & & \ddots \\ & & -g_n & & 1 \end{pmatrix} \quad \Rightarrow \quad [F^{(i)}]^{-1} := \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & g_{i+1} & \ddots & \\ & & \vdots & & \ddots \\ & & g_n & & 1 \end{pmatrix}.$$

Für zwei Frobeniusmatrizen $F^{(i_1)}$ und $F^{(i_2)}$ mit $i_1 < i_2$ gilt:

$$F^{(i_1)}F^{(i_2)} = F^{(i_1)} + F^{(i_2)} - I = \begin{pmatrix} 1 & & & & & & & \\ & \ddots & & & & & & \\ & & 1 & & & & & \\ & & -g_{i_1+1}^{(i_1)} & 1 & & & & \\ & & \vdots & & \ddots & & & \\ & & \vdots & & & 1 & & \\ & & \vdots & & & -g_{i_2+1}^{(i_2)} & \ddots & \\ & & \vdots & & & \vdots & & \ddots \\ & & -g_n^{(i_1)} & & & -g_n^{(i_2)} & & 1 \end{pmatrix}$$

Beweis: Nachrechnen! □

Bei der Multiplikation von Frobeniusmatrizen ist darauf zu achten, dass diese nicht kommutativ ist. Es gilt:

$$F^{(i_2)}F^{(i_1)} \neq F^{(i_1)} + F^{(i_2)} - I \text{ für } i_1 < i_2!$$

Aus dem Multiplikationsverhalten von Frobeniusmatrizen können wir für $i_1 < i_2 < i_3$ eine einfache Verallgemeinerung ableiten:

$$\begin{aligned} F^{(i_1)}F^{(i_2)}F^{(i_3)} &= F^{(i_1)}(F^{(i_2)} + F^{(i_3} - I)F^{(i_1)}F^{(i_2)} + F^{(i_1)}F^{(i_3} - F^{(i_1)} \\ &= F^{(i_1)} + F^{(i_2)} - I + F^{(i_1)} + F^{(i_3)} - I - F^{(i_1)} \\ &= F^{(i_1)} + F^{(i_2)} + F^{(i_3)} - 2I. \end{aligned}$$

Wir setzen nun (4.1) fort und mit $F^{-(i)} := [F^{(i)}]^{-1}$ gilt bei Verwendung von Satz 4.22 zunächst, dass F als Produkt von regulären Matrizen selbst regulär ist. Also folgt:

$$A = F^{-1}R = [F^{(n-1)} \cdots F^{(1)}]^{-1}R = \underbrace{F^{-(1)} \cdots F^{-(n-1)}}_{=:L} R.$$

Die Matrix L ist nach der Verallgemeinerung von Satz 4.22 eine untere Dreiecksmatrix mit Diagonaleinträgen 1:

$$L = \begin{pmatrix} 1 & & & & & \\ g_2^{(1)} & 1 & & & & \\ g_3^{(1)} & g_3^{(2)} & 1 & & & \\ g_4^{(1)} & g_4^{(2)} & g_4^{(3)} & 1 & & \\ \vdots & \vdots & \ddots & \ddots & \ddots & \\ g_n^{(1)} & g_n^{(2)} & \cdots & \cdots & g_n^{(n-1)} & 1. \end{pmatrix}$$

Wir fassen zusammen:

Satz 4.23 (LR-Zerlegung)**.** *Es sei $A \in \mathbb{R}^{n\times n}$ eine quadratische, reguläre Matrix. Angenommen alle bei der Elimination auftretenden Diagonalelemente $a_{ii}^{(i-1)}$ seien ungleich Null. Dann existiert die eindeutig bestimmte LR-Zerlegung in eine rechte obere reguläre Dreiecksmatrix $R \in \mathbb{R}^{n\times n}$ sowie in eine linke untere reguläre Dreiecksmatrix $L \in \mathbb{R}^{n\times n}$ mit Diagonaleinträgen 1. Die Aufwand zur Durchführung der LR-Zerlegung beträgt*

$$\frac{1}{3}n^3 + O(n^2)$$

arithmetische Operationen.

Beweis: *(i) Eindeutigkeit.* Angenommen, es existieren zwei LR-Zerlegungen

$$A = L_1R_1 = L_2R_2 \quad \leftrightarrow \quad L_2^{-1}L_1 = R_2R_1^{-1}.$$

Das Produkt von Dreiecksmatrizen ist wieder eine Dreiecksmatrix, also müssen beide Produkte Diagonalmatrizen sein. Das Produkt $L_2^{-1}L_1$ hat nur Einsen auf der Diagonale, also folgt

$$L_2^{-1}L_1 = R_2R_1^{-1} = I,$$

und somit $L_1 = L_2$ und $R_1 = R_2$.

(ii) Durchführbarkeit. Jeder Schritt der Elimination ist durchführbar, solange nicht durch $a_{ii}^{(i-1)} = 0$ geteilt werden muss. Die Matrix F ist per Konstruktion regulär und somit existiert auch die Matrix L.

(iii) Aufwand. Im i-ten Eliminationsschritt

$$A^{(i)} = F^{(i)}A^{(i-1)}$$

sind zunächst $n-(i+1)$ arithmetische Operationen zur Berechnung der $g_j^{(i)}$ für $j = i+1,\dots,n$ notwendig. Die Matrix-Matrix Multiplikation betrifft nur alle Elemente a_{kl} mit $k > i$ sowie $l > i$. Es gilt:

$$a_{kl}^{(i)} = a_{kl}^{(i-1)} + g_k^{(i)} a_{ik}^{(i)}, \quad k,l = i+1,\dots,n.$$

Hierfür sind $(n-(i+1))^2$ arithmetische Operationen notwendig. Insgesamt summiert sich der Aufwand in den $n-1$ Schritten zu:

$$N_{LR}(n) = \sum_{i=1}^{n-1} \{n-(i+1)+(n-(i+1))^2\} = \sum_{i=1}^{n-1} \{n^2 - n - i(1+2n) + i^2\},$$

und mit den bekannten Summenformeln folgt:

$$N_{LR}(n) = (n-1)n^2 - (n-1)n - \frac{n(n-1)}{2}(1+2n) + \frac{(n-1)n(2n-1)}{6} = \frac{1}{3}n^3 - 2n^2 + \frac{5}{3}n.$$

□

Die LR-Zerlegung kann nun zum Lösen von linearen Gleichungssystemen verwendet werden:

Algorithmus 4.24 (Lösen von linearen Gleichungsystemen mit der LR-Zerlegung)**.** *Es sei $A \in \mathbb{R}^{n\times n}$ eine reguläre Matrix, für welche die LR-Zerlegung existiert.*

1. *Berechne die LR-Zerlegung $A = LR$ in linke untere und rechte obere Dreiecksmatrix.*
2. *Löse das Gleichungssystem $Ax = b$ durch Vorwärts- und Rückwärtseinsetzen:*

$$(i) \quad Ly = b$$
$$(ii) \quad Rx = y.$$

Die Vorwärtselimination läuft entsprechend der Rückwärtseinsetzen in Algorithmus 4.20 und gemäß Satz 4.21 benötigt sie $O(n^2)$ Operationen. Das eigentliche Lösen eines linearen Gleichungsystems ist weit weniger aufwendig als das Erstellen der Zerlegung. In vielen Anwendungsproblemen, etwa bei der Diskretisierung von parabolischen Differentialgleichungen, müssen sehr viele Gleichungssysteme mit unterschiedlichen rechten Seiten aber identischen Matrizen hintereinander gelöst werden. Hier bietet es sich an, die Zerlegung nur einmal zu erstellen und dann wiederholt anzuwenden.

Bemerkung 4.25 (Praktische Aspekte)**.** *Die Matrix L ist eine linke untere Dreiecksmatrix mit Einsen auf der Diagonale. Die bekannten Diagonalelemente müssen demnach nicht gespeichert werden. Ebenso müssen die Nullelemente der Matrizen $A^{(i)}$ unterhalb der Diagonale nicht gespeichert werden. Es bietet sich an, die Matrizen L und R in der gleichen quadratischen Matrix zu speichern. In Schritt i gilt dann:*

$$\tilde{A}^{(i)} = \left(\begin{array}{ccccccc} a_{11} & a_{12} & a_{13} & \cdots & \cdots & \cdots & a_{1n} \\ \boldsymbol{l_{21}} & a_{22}^{(1)} & a_{23}^{(1)} & & & & \vdots \\ \boldsymbol{l_{31}} & \boldsymbol{l_{32}} & a_{33}^{(2)} & & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & & & \vdots \\ \hline \vdots & & & \boldsymbol{l_{i+1,i}} & a_{i+1,i+1}^{(i)} & \cdots & a_{i+1,n}^{(i)} \\ \vdots & & & \vdots & & \ddots & \vdots \\ \boldsymbol{l_{n1}} & \boldsymbol{l_{n2}} & \boldsymbol{\cdots} & \boldsymbol{l_{n,i}} & a_{n,i+1}^{(i)} & \cdots & a_{nn}^{(i)} \end{array}\right)$$

Dabei sind die fett gedruckten Werte die Einträge von L. Die Werte oberhalb der Linie ändern sich im Verlaufe des Verfahrens nicht mehr und bilden bereits die Einträge L sowie R.

Pivotierung Das Element $a_{ii}^{(i-1)}$ wird das *Pivot-Element* genannt. Bisher musste dieses Element stets ungleich Null sein. Dies ist jedoch für reguläre Matrizen nicht zwingend notwendig. Wir betrachten als Beispiel die Matrix

$$A := \begin{pmatrix} 1 & 4 & 2 \\ 2 & 8 & 1 \\ 1 & 2 & 1 \end{pmatrix}.$$

Im ersten Schritt zur Erstellung der LR-Zerlegung ist $a_{11}^{(0)} = 1$ und es gilt:

$$A^{(1)} = F^{(1)}A = \begin{pmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 4 & 2 \\ 2 & 8 & 1 \\ 1 & 2 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 4 & 2 \\ 0 & 0 & -3 \\ 0 & -2 & -1 \end{pmatrix}.$$

An dieser Stelle bricht der Algorithmus ab, denn es gilt $a_{22}^{(1)} = 0$. Wir könnten den Algorithmus jedoch mit der Wahl $a_{32}^{(i)} = -2$ als neues Pivot-Element weiterführen. Dies geschieht systematisch durch Einführen einer *Pivotisierung*. Im i-ten Schritt des Verfahrens wird zunächst ein geeignetes Pivot-Element a_{ki} in der i-ten Spalte gesucht. Die k-te und i-te Zeile werden getauscht und die LR-Zerlegung kann nicht weiter durchgeführt werden. Das

Tauschen von k-ter und i-ter Zeile erfolgt durch Multiplikation mit einer Pivot-Matrix:

$$P^{ki} := \begin{pmatrix} 1 & & & & & & & & & \\ & \ddots & & & & & & & & \\ & & 1 & & & & & & & \\ & & & 0 & 0 & \dots & 0 & 1 & & \\ & & & 0 & 1 & & & 0 & & \\ & & & \vdots & & \ddots & & \vdots & & \\ & & & 0 & & & 1 & 0 & & \\ & & & 1 & 0 & \cdots & 0 & 0 & & \\ & & & & & & & & 1 & \\ & & & & & & & & & \ddots \\ & & & & & & & & & & 1 \end{pmatrix}$$

Es gilt $p^{ki}_{jj} = 1$ für $j \neq k$ und $j \neq i$ sowie $p^{ki}_{ki} = p^{ki}_{ik} = 1$, alle anderen Elemente sind Null. Wir fassen einige Eigenschaften von P zusammen:

Satz 4.26 (Pivot-Matrizen)**.** *Es sei $P = P^{kl}$ die Pivot-Matrix mit $P^{kl}_{ii} = 1$ für $i \neq k, l$ und $P^{kl}_{kl} = P^{kl}_{lk} = 1$. Die Anwendung $P^{kl}A$ von links tauscht k-te und l-te Zeile von A, die Anwendung AP^{kl} von rechts tauscht k-te und l-te Spalte. Es gilt:*

$$P^2 = I \quad \Leftrightarrow \quad P^{-1} = P.$$

Beweis: Übung. □

In Schritt i der LR-Zerlegung suchen wir nun zunächst das Pivot-Element:

Algorithmus 4.27 (Pivot-Suche)**.** *In Schritt i suche Index $k \geq i$, so dass*

$$|a_{ki}| = \max_{j \geq i} |a_{ji}|.$$

Bestimme die Pivot-Matrix als $P^{(i)} := P^{ki}$.

Im Anschluss bestimmen wir $A^{(i)}$ als

$$A^{(i)} = F^{(i)} P^{(i)} A^{(i-1)}.$$

Die Pivotisierung sorgt dafür, dass alle Elemente $g^{(i)}_k = a^{(i-1)}_{ki} / a^{(i-1)}_{ii}$ von $F^{(i)}$ im Betrag durch 1 beschränkt sind. Insgesamt erhalten wir die Zerlegung:

$$R = A^{(n-1)} = F^{(n-1)} P^{(n-1)} \cdots F^{(1)} P^{(1)} A. \tag{4.2}$$

Die Pivot-Matrizen kommutieren nicht mit A oder den $F^{(i)}$. Daher ist ein Übergang zur LR-Zerlegung nicht ohne weitere möglich. Wir definieren:

$$\tilde{F}^{(i)} := P^{(n-1)} \cdots P^{(i+1)} F^{(i)} P^{(i+1)} \cdots P^{(n-1)}.$$

Die Matrix $\tilde{F}^{(i)}$ entsteht durch mehrfache Zeilen- und Spaltenvertauschung von $F^{(i)}$. Dabei werden nur Zeilen und Spalten $j > i$ vertauscht. Die Matrix $\tilde{F}^{(i)}$ hat die gleiche Besetzungsstruktur wie $F^{(i)}$ und insbesondere nur Einsen auf der Diagonale. D.h, sie ist wieder eine Frobeniusmatrix und Satz 4.22 gilt weiter. Es sind lediglich die Einträge in der i-ten Spalte unterhalb der Diagonale permutiert. Für die Inverse gilt entsprechend:

$$\tilde{L}^{(i)} := [\tilde{F}^{(i)}]^{-1} = P^{(n-1)} \cdots P^{(i+1)} L^{(i)} P^{(i+1)} \cdots P^{(n-1)}.$$

Die gleichzeitige Vertauschung von Zeilen und Spalten lässt die Diagonalelemente unverändert. Die Matrix $\tilde{L}^{(i)}$ ist wieder eine Frobeniusmatrix, es werden nur die Elemente der Spalte $l_{ij}, i > j$ permutiert. Wir formen (4.2) durch geschicktes Einfügen von Permutationsmatrizen um:

$$R = \tilde{F}^{(n-1)} \tilde{F}^{(n-2)} \cdots \tilde{F}^{(1)} \underbrace{P^{(n-1)} \cdots P^{(1)}}_{=:P} A$$

Diesen Prozess mache man sich anhand eines einfachen Beispiels klar:

$$\begin{aligned} R &= F^{(3)} P^{(3)} F^{(2)} P^{(2)} F^{(1)} P^{(1)} A \\ &= F^{(3)} P^{(3)} F^{(2)} \underbrace{P^{(3)} P^{(3)}}_{=I} P^{(2)} F^{(1)} \underbrace{P^{(2)} P^{(3)} P^{(3)} P^{(2)}}_{=I} P^{(1)} A \\ &= \underbrace{F^{(3)}}_{=\tilde{F}^{(3)}} \underbrace{P^{(3)} F^{(2)} P^{(3)}}_{=\tilde{F}^{(2)}} \underbrace{P^{(3)} P^{(2)} F^{(1)} P^{(2)} P^{(3)}}_{=\tilde{F}^{(1)}} \underbrace{P^{(3)} P^{(2)} P^{(1)}}_{=P} A \end{aligned}$$

Mit $\tilde{L}^{(i)} = [\tilde{F}^{(i)}]^{-1}$ gilt dann:

$$\underbrace{\tilde{L}^{(1)} \cdots \tilde{L}^{(n-1)}}_{=:\tilde{L}} R = PA.$$

Da $\tilde{L}^{(i)}$ wieder Frobeniusmatrizen sind, gilt weiter mit Satz 4.22:

$$\begin{aligned} \tilde{L} &= \tilde{L}^{(1)} \cdots \tilde{L}^{(n-1)} \\ &= \sum_{i=1}^{n-1} \tilde{L}^{(i)} - (n-2)I \\ &= P^{(n-1)} \left(L^{(n-1)} + P^{(n-1)} \left(L^{(n-2)} + \cdots + P^{(2)} F^{(1)} P^{(2)} \right) \cdots P^{(n-2)} \right) P^{(n-1)} - (n-2)I. \end{aligned}$$

Beim Erstellen der LR-Zerlegung müssen also nicht nur die $A^{(i)}$ sondern auch die bisher berechneten $L^{(i)}$ permutiert werden.

Wir fassen zusammen:

Satz 4.28 (LR-Zerlegung mit Pivotisierung)**.** *Es sei $A \in \mathbb{R}^{n \times n}$ eine reguläre Matrix. Es existiert eine LR-Zerlegung*

$$PA = LR,$$

wobei P ein Produkt von Pivot-Matrizen ist, L eine untere Dreiecksmatrix mit Diagonaleinträgen eins und R eine rechte obere Dreiecksmatrix. Die LR-Zerlegung ohne Pivotisierung $P = I$ ist eindeutig, falls sie existiert.

BEWEIS: Übung. □

Die Pivotisierung dient einerseits dazu, die Durchführbarkeit der LR-Zerlegung sicherzustellen. Auf der anderen Seite kann durch geeignete Pivotisierung die Stabilität der Gauß-Elimination verbessert werden. Durch Wahl eines Pivot-Elements a_{ki} mit maximaler relativer Größe (bezogen auf die Zeile) kann die Gefahr der Auslöschung verringert werden.

Beispiel 4.29 (LR-Zerlegung ohne Pivotierung)**.** *Es sei:*

$$A = \begin{pmatrix} 2.3 & 1.8 & 1.0 \\ 1.4 & 1.1 & -0.7 \\ 0.8 & 4.3 & 2.1 \end{pmatrix}, \quad b = \begin{pmatrix} 1.2 \\ -2.1 \\ 0.6 \end{pmatrix},$$

und die Lösung des linearen Gleichungssystems $Ax = b$ ist gegeben durch (Angabe mit fünfstelliger Genauigkeit):

$$x \approx \begin{pmatrix} 0.34995 \\ -0.98024 \\ 2.1595 \end{pmatrix}.$$

Für die Matrix A gilt $\operatorname{cond}_\infty(A) = \|A\|_\infty \|A^{-1}\|_\infty \approx 7.2 \cdot 1.2 \approx 8.7$. *Die Aufgabe ist also gut konditioniert, eine Fehlerverstärkung um höchstens eine Stelle ist zu erwarten. Wir erstellen zunächst die LR-Zerlegung (dreistellige Rechnung). Dabei schreiben wir die Einträge von L fettgedruckt in die Ergebnismatrix:*

$$F^{(1)} = \begin{pmatrix} 1 & 0 & 0 \\ -\frac{1.4}{2.3} & 1 & 0 \\ -\frac{0.8}{2.3} & 0 & 1 \end{pmatrix} \approx \begin{pmatrix} 1 & 0 & 0 \\ -0.609 & 1 & 0 \\ -0.348 & 0 & 1 \end{pmatrix}, \quad [L^{(1)}, A^{(1)}] \approx \left(\begin{array}{ccc} 2.3 & 1.8 & 1.0 \\ \mathbf{0.609} & 0.0038 & -1.31 \\ \mathbf{0.348} & 3.67 & 1.75 \end{array}\right)$$

Im zweiten Schritt gilt:

$$F^{(2)} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -\frac{3.67}{0.0038} & 1 \end{pmatrix} \approx \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -966 & 1 \end{pmatrix}, \quad [L^{(2)}L^{(1)}, A^{(2)}] \approx \left(\begin{array}{ccc} 2.3 & 1.8 & 1.0 \\ \mathbf{0.609} & 0.0038 & -1.31 \\ \mathbf{0.348} & \mathbf{966} & 1270 \end{array}\right)$$

Die LR-Zerlegung ergibt sich als:

$$L = \begin{pmatrix} 1 & 0 & 0 \\ 0.609 & 1 & 0 \\ 0.348 & 966 & 1 \end{pmatrix}, \quad R := \begin{pmatrix} 2.3 & 1.8 & 1.0 \\ 0 & 0.0038 & -1.31 \\ 0 & 0 & 1270 \end{pmatrix}$$

Wir lösen das Gleichungssystem nun durch Vorwärts- und Rückwärtseinsetzen:

$$A\tilde{x} = L \underbrace{R\tilde{x}}_{=y} = b.$$

Zunächst gilt:

$$y_1 = 1.2,\ y_2 = -2.1 - 0.609 \cdot 1.2 \approx -2.83,\ y_3 = 0.6 - 0.348 \cdot 1.2 + 966 \cdot 2.83 \approx 2730.$$

Und schließlich:

$$\tilde{x}_3 = \frac{2730}{1270} \approx 2.15,\ \tilde{x}_2 = \frac{-2.83 + 1.31 \cdot 2.15}{0.0038} \approx -3.55,\ \tilde{x}_1 = \frac{1.2 + 1.8 \cdot 3.55 - 2.15}{2.3} \approx 2.37.$$

Für die Lösung $\tilde{x}$ gilt:

$$\tilde{x} = \begin{pmatrix} 2.37 \\ -3.55 \\ 2.15 \end{pmatrix}, \quad \frac{\|\tilde{x} - x\|_2}{\|x\|_2} \approx 1.4,$$

d.h. ein Fehler von 140 *Prozent, obwohl wir nur Rundungsfehler und noch keine gestörte Eingabe betrachtet haben.*

Dieses Negativbeispiel zeigt die Bedeutungder Pivotisierung. Im zweiten Schritt wurde als Pivot-Element mit 0.0038 ein Wert Nahe bei 0 gewählt. Hierdurch entstehen Werte von sehr unterschiedlicher Größenordnung in den Matrizen L und R. Dies wirkt sich ungünstig auf die weitere Stabilität aus.

Beispiel 4.30 (LR-Zerlegung mit Pivotisierung)**.** *Wir setzen das Beispiel in Schritt 2 fort und suchen zunächst das Pivot-Element:*

$$[L^{(1)}, A^{(1)}] = \left(\begin{array}{ccc} 2.3 & 1.8 & 1.0 \\ \mathbf{0.609} & 0.0038 & -1.31 \\ \mathbf{0.348} & \boxed{3.67} & 1.75 \end{array}\right), \quad P^{(2)} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

Also, pivotisiert:

$$[\tilde{L}^{(1)}, \tilde{A}^{(1)}] = \left(\begin{array}{ccc} 2.3 & 1.8 & 1.0 \\ \mathbf{0.348} & 3.67 & 1.75 \\ \mathbf{0.609} & 0.0038 & -1.31 \end{array}\right).$$

Weiter folgt nun:

$$F^{(2)} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -\frac{0.0038}{3.67} & 1 \end{pmatrix} \approx \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -0.00104 & 1 \end{pmatrix}$$

$$[L^{(2)}\tilde{L}^{(1)}, \tilde{A}^{(2)}] \approx \left(\begin{array}{ccc} 2.3 & 1.8 & 1.0 \\ \mathbf{0.348} & 3.67 & 1.75 \\ \mathbf{0.609} & \mathbf{0.001040} & -1.31 \end{array}\right)$$

Wir erhalten die Zerlegung:

$$\tilde{L}R = PA, \quad \tilde{L} := \begin{pmatrix} 1 & 0 & 0 \\ 0.348 & 1 & 0 \\ 0.609 & 0.00104 & 1 \end{pmatrix}, \quad R := \begin{pmatrix} 2.3 & 1.8 & 1.0 \\ 0 & 3.67 & 1.75 \\ 0 & 0 & -1.31 \end{pmatrix}, \quad P := \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}.$$

Das Lineare Gleichungssystem lösen wir in der Form:

$$PAx = \tilde{L} \underbrace{Rx}_{=y} = Pb.$$

Zunächst gilt für die rechte Seite: $\tilde{b} = Pb = (1.2, 0.6, -2.1)^T$ *und Vorwärtseinsetzen in* $\tilde{L}y = \tilde{b}$ *ergibt*

$$y_1 = 1.2, \quad y_2 = 0.6 - 0.348 \cdot 1.2 \approx 0.182, \quad y_3 = -2.1 - 0.609 \cdot 1.2 - 0.00104 \cdot 0.182 \approx -2.83.$$

Als Näherung $\tilde{x}$ *erhalten wir:*

$$\tilde{x} = \begin{pmatrix} 0.35 \\ -0.98 \\ 2.16 \end{pmatrix}$$

mit einem relativen Fehler

$$\frac{\|\tilde{x} - x\|_2}{\|x\|_2} \approx 0.0002,$$

also von nur 0.02% *statt* 140%.

Die Beispiele zeigen, dass die berechnete LR-Zerlegung in praktischer Anwendung natürlich keine echte Zerlegung, sondern aufgrund von Rundungsfehlern nur eine Näherung der Matrix $A \approx LR$ ist. Man kann durch Berechnung von LR leicht die Probe machen und den Fehler $A - LR$ bestimmen.

Die LR-Zerlegung ist eines der wichtigsten direkten Verfahren zum Lösen von linearen Gleichungssystemen. Der Aufwand zur Berechnung der LR-Zerlegung steigt allerdings mit dritter Ordnung sehr schnell. Selbst auf modernen Computern übersteigt die Laufzeit für große Gleichungssysteme schnell eine sinnvolle Grenze:

n	Operationen	Zeit
100	300 000	30 μs
1 000	$300 \cdot 10^6$	30 ms
10 000	$300 \cdot 10^9$	30 s
100 000	$300 \cdot 10^{12}$	10 h
1 000 000	$300 \cdot 10^{15}$	1 Jahr

Tabelle 4.1: Rechenzeit zum Erstellen der LR-Zerlegung einer Matrix $A \in \mathbb{R}^{n \times n}$ auf einem Rechner mit 10 GigaFLOPS.

Bei der Diskretisierung von partiellen Differentialgleichungen treten Gleichungssysteme mit $n = 10^6 \sim 10^9$ auf. Die Matrizen verfügen dann aber über Struktureigenschaften wie Symmetrie, oder über ein besonders dünnes Besetzungsmuster (in jeder Zeile sind nur einige wenige ungleich Null). Die linearen Gleichungsysteme, die bei der Finite-Elemente Diskretisierung der Laplace-Gleichung (beschreibt die Ausdehnung einer Membran) entstehen haben z.B. unabhänging von n nur 5 Einträge pro Zeile. Die so entstehenden linearen Gleichungsysteme lassen sich bei effizienter Implementierung der LR-Zerlegung auch bei $n = 1\,000\,000$ in weniger als einer Minute lösen.

4.2.3 LR-Zerlegung für diagonaldominante Matrizen

Satz 4.28 besagt, dass die LR-Zerlegung für beliebige reguläre Matrizen mit Pivotierung möglich ist. Es gibt allerdings auch viele Matrizen, bei denen die LR-Zerlegung ohne Pivotisierung stabil durchführbar ist. Beispiele hierfür sind positiv definite oder *diagonaldominante Matrizen*:

Definition 4.31 (Diagonaldominanz)**.** *Eine Matrix $A \in \mathbb{R}^{n\times n}$ heißt* diagonaldominant*, falls*

$$|a_{ii}| \geq \sum_{j\neq i} |a_{ij}|, \quad i = 1, \ldots, n.$$

Eine diagonaldominante Matrix hat das betragsmäßig größte Element auf der Diagonalen, bei regulären Matrizen sind die Diagonalelemente zudem ungleich Null.

Satz 4.32 (LR-Zerlegung diagonaldominanter Matrizen)**.** *Sei $A \in \mathbb{R}^{n\times n}$ eine reguläre, diagonaldominante Matrix. Dann ist die LR-Zerlegung ohne Pivotierung durchführbar und alle auftretenden Pivot-Elemente $a_{ii}^{(i-1)}$ sind von Null verschieden.*

Beweis: Wir führen den Beweis über Induktion und zeigen, dass alle Untermatrizen $A_{kl>i}^{(i)}$ wieder diagonaldominant sind. Für eine diagonaldominante Matrix gilt

$$|a_{11}| \geq \sum_{j>1} |a_{1j}| \geq 0,$$

und da A regulär ist auch zwingend $|a_{11}| > 0$. Der erste Schritt der LR-Zerlegung ist durchführbar.

Es sei nun A eine reguläre Matrix, wir wollen zeigen, dass die Matrix $\tilde{A}$ nach einem Eliminationsschritt eine diagonaldominante Untermatrix $\tilde{A}_{ij>1}$ hat. Für deren Einträge $\tilde{a}_{ij}$ gilt:

$$\tilde{a}_{ij} = a_{ij} - \frac{a_{i1}a_{1j}}{a_{11}}, \quad i, j = 1, \ldots, n.$$

Also gilt für die Untermatrix:

$$\begin{aligned} i = 2, \ldots, n: \quad \sum_{j=2,\, j\neq i}^{n} |\tilde{a}_{ij}| &\leq \underbrace{\sum_{j=1,\, j\neq i}^{n} |a_{ij}|}_{\leq |a_{ii}|} - |a_{i1}| + \frac{|a_{i1}|}{|a_{11}|} \underbrace{\sum_{j=2}^{n} |a_{1j}|}_{\leq |a_{11}|} - \frac{|a_{i1}|}{|a_{11}|}|a_{1i}| \\ &\leq |a_{ii}| - \frac{|a_{i1}|}{|a_{11}|}|a_{1i}| \leq |a_{ii} - \frac{a_{i1}}{a_{11}}a_{1i}| = |\tilde{a}_{ii}|. \end{aligned}$$

Die resultierende Matrix ist wieder diagonaldominant. □

Die Definition der Diagonaldominanz scheint zunächst willkürlich. Es zeigt sich aber, dass viele Matrizen, die in Anwendungen, zum Beispiel bei der Diskretisierung von partiellen Differentialgleichungen auftreten, diese Eigenschaft erfüllen. Zudem ist die Diagonaldominanz einer Matrix sehr einfach zu überprüfen und daher ein gutes Kriterium um die Notwendigkeit der Pivotierung abzuschätzen.

4.2.4 Die Cholesky-Zerlegung für positiv definite Matrizen

Eine wichtige Klasse von Matrizen sind die positiv definiten Matrizen, siehe Definition 4.11 sowie Satz 4.12. Es zeigt sich, dass für symmetrisch positiv definite Matrizen $A \in \mathbb{R}^{n\times n}$ eine symmetrische Zerlegung $A = \tilde{L}\tilde{L}^T$ in eine untere Dreiecksmatrix $\tilde{L}$ erstellt werden kann. Diese ist immer ohne Pivotisierung durchführbar und wird *Cholesky-Zerlegung* genannt. Der Aufwand zum Erstellen der Cholesky-Zerlegung ist erwartungsgemäß nur halb so groß wie der Aufwand zum Erstellen der LR-Zerlegung.

Satz 4.33 (LR-Zerlegung einer positiv definiten Matrix)**.** *Die Matrix $A \in \mathbb{R}^{n\times n}$ sei symmetrisch, positiv definit. Dann existiert eine eindeutige LR-Zerlegung ohne Pivotierung.*

Beweis: Wir gehen ähnlich vor wie bei diagonaldominanten Matrizen und führen den Beweis per Induktion. Da zu sei A eine positiv definite, symmetrische Matrix. Ein Schritt der LR-Zerlegung ist durchführbar, da laut Satz 4.12 gilt $a_{11} > 0$. Wir zeigen, dass die Teilmatrix $\tilde{A}_{ij>1}$ nach einem Eliminationsschritt wieder symmetrisch positiv definit ist. Es gilt aufgrund der Symmetrie von A:

$$\tilde{a}_{ij} = a_{ij} - \frac{a_{1j}a_{i1}}{a_{11}} = a_{ji} - \frac{a_{1i}a_{j1}}{a_{11}} = \tilde{a}_{ji},$$

d.h., $\tilde{A}_{ij>1}$ ist symmetrisch.

Nun sei $x \in \mathbb{R}^n$ ein Vektor $x = (x_1, \ldots, x_n)$ mit $x_2, \ldots, x_n$ beliebig. Den Eintrag x_1 werden wir im Laufe des Beweises spezifizieren. Es gilt wegen der positiven Definitheit von A:

$$\begin{aligned}
0 < (Ax, x) &= \sum_{ij} a_{ij}x_ix_j = a_{11}x_1^2 + 2x_1\sum_{j=2}^{n} a_{1j}x_j + \sum_{ij=2}^{n} a_{ij}x_ix_j \\
&= a_{11}x_1^2 + 2x_1\sum_{j=2}^{n} a_{1j}x_j + \sum_{ij=2}^{n} \underbrace{\left(a_{ij} - \frac{a_{1j}a_{i1}}{a_{11}}\right)}_{=\tilde{a}_{ij}} x_ix_j + \sum_{ij=2}^{n} \frac{a_{1j}a_{i1}}{a_{11}}x_ix_j \\
&= a_{11}\left(x_1^2 + 2x_1\frac{1}{a_{11}}\sum_{j=2}^{n} a_{1j}x_j + \frac{1}{a_{11}^2}\sum_{ij=2}^{n} a_{1j}a_{i1}x_ix_j\right) + \sum_{ij=2}^{n} \tilde{a}_{ij}x_ix_j \\
&= a_{11}\left(x_1 + \frac{1}{a_{11}}\sum_{j=2}^{n} a_{1j}x_j\right)^2 + \sum_{ij=2}^{n} \tilde{a}_{ij}x_ix_j.
\end{aligned}$$

Die positive Definitheit von $\tilde{A}_{ij>1}$ folgt bei der Wahl

$$x_1 = -\frac{1}{a_{11}}\sum_{j=2}^{n} a_{1j}x_j.$$

□

Für eine symmetrisch positiv definite Matrix A ist die LR-Zerlegung immer ohne Pivotisierung durchführbar. Dabei treten nur positive Pivot-Elemente $a_{ii}^{(i-1)}$ auf. Das heißt, die Matrix R hat nur positive Diagonalelemente $r_{ii} > 0$. Es sei $D \in \mathbb{R}^{n\times n}$ die Diagonalmatrix mit $d_{ii} = r_{ii} > 0$. Dann gilt:

$$A = LR = LD\tilde{R},$$

mit einer rechten oberen Dreiecksmatrix $\tilde{R}$, welche nur Einsen auf der Diagonalen hat. Da A symmetrisch ist folgt:

$$A = LR = LD\tilde{R} = \tilde{R}^T DL^T = A^T.$$

Aufgrund der Eindeutigkeit der LR-Zerlegung gilt $L = \tilde{R}^T$ und $R = DL^T$. Da D nur positive Diagonaleinträge hat existiert die Matrix $\sqrt{D}$ und wir schreiben:

$$A = LR = \underbrace{LD^{\frac{1}{2}}}_{=:\tilde{L}} \underbrace{D^{-\frac{1}{2}}R}_{=\tilde{L}^T}.$$

Wir fassen zusammen:

Satz 4.34 (Cholesky-Zerlegung)**.** *Es sei $A \in \mathbb{R}^{n\times n}$ eine symmetrisch, positiv definite Matrix. Dann existiert die Cholesky-Zerlegung:*

$$A = \tilde{L}\tilde{L}^T,$$

in eine untere linke Dreiecksmatrix $\tilde{L}$. Sie kann ohne Pivotierung in

$$\frac{n^3}{6} + O(n^2)$$

arithmetische Operationen durchgeführt werden.

Anstelle eines Beweises geben wir einen effizienten Algorithmus zur direkten Berechnung der Cholesky-Zerlegung an. Hier kann der notwendige Aufwand leicht abgelesen werden:

Algorithmus 4.35 (Direkte Berechnung der Cholesky-Zerlegung)**.** *Gegeben sei eine symmetrisch, positiv definite Matrix $A \in \mathbb{R}^{n\times n}$. Dann sind die Einträge l_{ij}, $j \le i$ der Cholesky-Zerlegung bestimmt durch die Vorschrift:*

$$j = 1, \dots, n: \quad \begin{aligned} &(i) \quad l_{11} = \sqrt{a_{11}}, \qquad \text{bzw. } l_{jj} = \sqrt{a_{jj} - \sum_{k=1}^{j-1} l_{jk}^2} \\ &(ii) \quad l_{ij} = l_{jj}^{-1}\left(a_{ij} - \sum_{k=1}^{j-1} l_{ik}l_{jk}\right), \quad i = j+1, \dots, n. \end{aligned}$$

Der Algorithmus kann iterativ aus der Beziehung $\tilde{L}\tilde{L}^T = A$ hergeleitet werden. Es gilt:

$$a_{ij} = \sum_{k=1}^{\min\{i,j\}} l_{ik}l_{jk}.$$

Wir gehen Spaltenweise $j = 1, 2, \ldots$ vor. Das heißt, L sei für alle Spalten bis $j-1$ bekannt. Dann gilt in Spalte j zunächst für das Diagonalelement:

$$a_{jj} = \sum_{k=1}^{j} l_{kk}^2 \quad \Rightarrow \quad l_{jj} = \sqrt{a_{jj} - \sum_{k=1}^{j-1} l_{jk}^2}.$$

Ist das j-te Diagonalelement l_{jj} bekannt, so gilt für $i > j$:

$$a_{ij} = \sum_{k=1}^{j} l_{ik} l_{jk} \quad \Rightarrow \quad l_{ij} l_{jj} = a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{jk} \quad \Rightarrow \quad l_{ij} = l_{jj}^{-1} \left(a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{jk} \right).$$

4.2.5 Dünn besetzte Matrizen und Bandmatrizen

Der Aufwand zum Erstellen der LR-Zerlegung wächst sehr schnell mit der Größe der Matrix an. In vielen Anwendungsproblemen treten *dünn besetzte Matrizen* auf:

Definition 4.36 (Dünn besetzte Matrix)**.** *Eine Matrix $A \in \mathbb{R}^{n \times n}$ heißt* dünn besetzt*, falls die Matrix A nur $O(n)$ von Null verschiedene Einträge besitzt. Das* Besetzungsmuster *(Sparsity Pattern) $B \subset \{1, \ldots, n\}^2$ von A ist die Menge aller Indexpaare (i, j) mit $a_{ij} \neq 0$.*

Andere, weniger strenge Definitionen von dünn besetzten Matrizen verlangen, dass die Matrix $O(n \log(n))$ oder auch $O(n\sqrt{n})$ beziehungsweise einfach $o(n^2)$ von Null verschiedene Einträge besitzt.

Ein Beispiel für dünn besetzte Matrizen sind *Tridiagonalmatrizen* der Form

$$A \in \mathbb{R}^{n \times n}, \quad a_{ij} = 0 \quad \forall |i - j| > 1.$$

Für Tridiagonalmatrizen kann die LR-Zerlegung sehr effizient durchgeführt werden:

Satz 4.37 (Thomas-Algorithmus)**.** *Es sei $A \in \mathbb{R}^{n \times n}$ eine reguläre Tridiagonalmatrix. Die LR-Zerlegung ist wieder eine Tridiagonalmatrix und kann in $O(n)$ Operationen durchgeführt werden.*

Beweis: Übung. □

Eine Verallgemeinerung von Tridiagonalsystemen sind die Bandmatrizen:

Definition 4.38 (Bandmatrix)**.** *Eine Matrix $A \in \mathbb{R}^{n \times n}$ heißt* Bandmatrix *mit* Bandbreite *$m \in \mathbb{N}$, falls:*

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

Eine Bandmatrix hat höchstens $n(2m + 1)$ von Null verschiedene Einträge.

Es zeigt sich, dass die LR-Zerlegung einer Bandmatrix wieder eine Bandmatrix ist und daher effizient durchgeführt werden kann.

Satz 4.39 (LR-Zerlegung einer Bandmatrix)**.** *Es sei $A \in \mathbb{R}^{n\times n}$ eine Bandmatrix mit Bandbreite m. Die LR-Zerlegung (ohne Permutierung):*

$$LR = A,$$

ist wieder eine Bandmatrix, d.h.:

$$L_{ij} = R_{ij} = 0 \quad \forall |i-j| > m,$$

und kann in

$$O(nm^2)$$

Operationen durchgeführt werden.

Beweis: Wir zeigen induktiv, dass die entstehenden Eliminationsmatrizen $\tilde{A}_{ij>1}$ wieder Bandmatrizen sind. Es gilt:

$$\tilde{a}_{ij} = a_{ij} - \frac{a_{1j}a_{i1}}{a_{11}}, \quad i,j = 2,\dots,n.$$

Es sei nun $|i-j| > m$. Dann ist $a_{ij} = 0$. Ebenso müssen $a_{1j} = a_{i1} = 0$ sein, da $1 \leq i,j$.

Zur Aufwandsberechnung vergleiche den Beweis zu Satz 4.23. Im Fall einer Bandmatrix müssen in Schritt i der Elimination nicht mehr $(n-i-1)^2$ arithmetische Operationen sondern höchstens m^2 arithmetische Operationen durchgeführt werden. Ebenso müssen nur m Elemente der Frobeniusmatrix zur Reduktion bestimmt werden. Insgesamt ergibt sich:

$$\sum_{i=1}^{n}(m+m^2) = nm^2 + O(nm).$$

□

Der Unterschied, eine LR-Zerlegung für eine voll besetzte Matrix und eine Bandmatrix durchzuführen ist enorm. Zur Diskretisierung der Laplace-Gleichung mit Finiten Differenzen müssen lineare Gleichungssysteme mit der sogenannten Modellmatrix gelöst werden:

$$A = \begin{pmatrix} A_m & -I_m & 0 & \cdots & 0 \\ -I_m & A_m & -I_m & & \vdots \\ 0 & -I_m & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & -I_m \\ 0 & \cdots & 0 & -I_m & A_m \end{pmatrix}, \quad A_m = \left.\begin{pmatrix} 4 & -1 & 0 & \cdots & 0 \\ -1 & 4 & -1 & & \vdots \\ 0 & -1 & \ddots & \ddots & 0 \\ \vdots & & \ddots & \ddots & -1 \\ 0 & \cdots & 0 & -1 & 4 \end{pmatrix}\right\} m,$$

Die Matrix $A \in \mathbb{R}^{n\times n}$ hat Bandbreite $m = \sqrt{n}$. In Tabelle 4.2 geben wie die notwendigen Rechenzeiten zum Erstellen der LR-Zerlegung auf aktueller Hardware an. Man vergleiche mit Tabelle 4.1.

n	Operationen	Zeit
100	10 000	1 μs
1 000	10^6	100 μs
10 000	10^8	10 ms
100 000	10^{10}	1 s
1 000 000	10^{12}	2 min

Tabelle 4.2: Rechenzeit zum Erstellen der LR-Zerlegung einer Bandmatrix $A \in \mathbb{R}^{n\times n}$ mit Bandbreite $m = \sqrt{n}$ auf einem Rechner mit 10 GigaFLOPS.

Die Modellmatrix ist eine Bandmatrix mit Bandbreite $m = \sqrt{n}$, hat aber in jeder Zeile neben dem Diagonalelement höchsten 4 von Null verschiedene Einträge $a_{i,i\pm i}$ und $a_{i,i\pm m}$. Bei so dünn besetzten Matrizen stellt sich die Frage, ob die LR-Zerlegung, also die Matrizen L und R das gleiche dünne Besetzungsmuster haben. Es zeigt sich jedoch, dass die LR-Zerlegung einer dünn besetzten Bandmatrix im Allgemeinen selbst eine dicht besetzte Bandmatrix ist. Aus der dünnen Besetzungsstruktur kann kein Nutzen gezogen werden.

Der Aufwand zur Berechnung der LR-Zerlegung einer dünn besetzten Matrix hängt wesentlich von der Sortierung, also der Pivotierung, der Matrix ab. Wir betrachten hierzu ein einfaches Beispiel:

Beispiel 4.40 (LR-Zerlegung dünn besetzter Matrix)**.** *Wir betrachten die beiden Matrizen:*

$$A_1 := \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 1 & 0 & 0 \\ 3 & 0 & 1 & 0 \\ 4 & 0 & 0 & 1 \end{pmatrix}, \quad A_2 := \begin{pmatrix} 1 & 0 & 0 & 4 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & 2 \\ 4 & 3 & 2 & 1 \end{pmatrix}$$

Die beiden Matrizen gehen durch simultane Vertauschen der ersten und vierten, sowie zweiten und dritten Zeile und Spalte auseinander hervor. Die LR-Zerlegung der Matrix A_1 (ohne Permutierung) lautet:

$$L_1 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ 3 & 2 & 1 & 0 \\ 4 & \frac{8}{3} & 1 & 1 \end{pmatrix} \; R_1 = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 0 & -3 & -6 & -8 \\ 0 & 0 & 4 & 4 \\ 0 & 0 & 0 & \frac{7}{3} \end{pmatrix},$$

und für die Matrix A_2 erhalten wie (wieder ohne Pivotierung):

$$L_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 4 & 3 & 2 & 1 \end{pmatrix} \; R_2 = \begin{pmatrix} 1 & 0 & 0 & 4 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & -28 \end{pmatrix}.$$

Obwohl beide Matrizen bis auf Zeilen und Spaltentausch das gleiche lineare Gleichungssystem beschreiben, haben die LR-Zerlegungen ein gänzlich unterschiedliches Besetzungsmuster: die LR-Zerlegung von Matrix A_1 ist voll besetzt, während die LR-Zerlegung zu Matrix A_2 mit dem gleichen dünnen Besetzungsmuster auskommt wie die Matrix A_2 selbst.

Dieses Beispiel lässt sich auf die entsprechende $n \times n$-Matrix verallgemeinern. In Fall 1 sind n^2 Einträge zum Speichern der LR-Zerlegung notwendig, in Fall 2 nur $3n$. Für $n \gg 1\,000$ ist dieser Unterschied entscheidend.

Dünn besetzte Matrizen treten in der Anwendung oft auf, etwa bei der Diskretisierung von Differentialgleichungen, aber auch bei der Berechnung von kubischen Splines. Damit die LR-Zerlegung aus der dünnen Besetzungsstruktur Nutzen ziehen kann, muss die Matrix entsprechend permutiert (man sagt hier sortiert) sein. Werden Nulleinträge in A in der LR-Zerlegung überschrieben, so spricht man von *fill-in*. Die aktuelle Forschung zur Weiterentwicklung der LR-Zerlegung befasst sich weniger mit der Berechnung der LR-Zerlegung selbst, als mit effizienten Sortierverfahren zur Reduktion der *fill-ins*. Wir wissen, dass die LR-Zerlegung einer Bandmatrix mit Bandbreite m wieder eine (voll besetzte) Bandmatrix mit Bandbreite m ist und in $O(nm^2)$ Operationen durchgeführt werden kann. Eine Idee zur Sortierung der Matrix A besteht nun darin, die Einträge so anzuordnen, dass die sortierte Matrix eine Bandmatrix mit möglichst dünner Bandbreite ist.

Ein bekanntes Verfahren ist der *Cuthill-McKee-Algorithmus*. Dieser erstellt eine Sortierung $\{i_1, i_2, \ldots, i_n\}$ der n Indizes, so dass miteinander verbundene Indizes nahe beieinander stehen. Zur Herleitung des Verfahrens benötigen wie einige Begriffe. Es sei $A \in \mathbb{R}^{n\times n}$ eine dünn besetzte Matrix mit Besetzungsmuster $B \subset \{1, \ldots, n\}^2$. Dabei gehen wir der Einfachheit davon aus, dass B symmetrisch ist. Aus $(i, j) \in B$ folgt $(j, i) \in B$. Zu einem Index $i \in \{1, \ldots, n\}$ sei $\mathcal{N}(i)$ die Menge aller mit i verbundenen Indizes:

$$\mathcal{N}(i) := \{j \in \{1, \ldots n\} : \ (i, j) \in B\},$$

Und für eine beliebige Menge $N \subset \{1, \ldots, n\}$ bezeichnen wir mit $\#N$ die Menge der Elemente in N, d.h. mit $\#\mathcal{N}(i)$ die Anzahl der Nachbarn von i.

Der Algorithmus füllt schrittweise eine Indexliste $I = (i_1, i_2, \ldots)$ bis alle Indizes $1, \ldots, n$ einsortiert sind. Wir starten mit dem Index $I = (i_1)$, welcher die geringste Zahl von Nachbarn $\#\mathcal{N}(i_1) \leq \#\mathcal{N}(j), \forall j$ besitzt. Im Anschluss fügen wir die Nachbarn $j \in \mathcal{N}(i_1)$ von i_1 hinzu, in der Reihenfolge der jeweiligen Zahl von Nachbarn. Auf diese Weise hangelt sich der Algorithmus von Nachbar zu Nachbar und arbeitet die Besetzungsstruktur der Matrix ab bis alle Indizes zur Liste hinzugefügt wurden.

Bei der genauen Definition des Algorithmus sind noch einige Sonderfälle zu betrachten, so dass der Algorithmus auch wirklich terminiert und alle Indizes findet:

Algorithmus 4.41 (Cuthill-McKee). *Es sei $A \in \mathbb{R}^{n\times n}$ eine dünn besetzte Matrix mit symmetrischem Besetzungsmuster $B \in (i,j)$. Starte mit leerer Indexmenge $I = (\cdot)$ und iteriere für $k = 1,\dots,n$:*

(0) Stopp, falls die Indexmenge I bereits n Elemente besitzt.

(1) Falls Indexmenge I nur $k-1$ Elemente besitzt, bestimme Indexelement i_k als (noch nicht verwendeter) Index mit minimaler Nachbarzahl:

$$i_k = \arg \min_{j\in\{1,\dots,n\}\setminus I} \#\mathcal{N}(j).$$

(2) Erstelle Liste aller Nachbarn von i_k, welche noch nicht in I enthalten sind:

$$N_k := \mathcal{N}(i_k) \setminus I.$$

Falls $N_k = \emptyset$, Neustart mit $k+1$.

(3) Sortiere Liste N_k aufsteigend nach Anzahl von Nachbarn:

$$N_k = \{s_1^k, s_2^k, \dots\}, \quad \#\mathcal{N}(s_i^k) \le \#\mathcal{N}(s_{i+1}^k).$$

Füge Indizes $\{s_1^k, s_2^k, \dots\}$ in dieser Reihenfolge zu Liste I hinzu.

Wir betrachten zur Veranschaulichung ein Beispiel:

Beispiel 4.42 (Cuthill-McKee). *Es sei eine Matrix $A \in \mathbb{R}^{8\times 8}$ mit folgendem Besetzungsmuster gegeben:*

$$A := \begin{pmatrix} * & * & & & & & & * \\ * & * & & & * & * & & * \\ & & * & & * & & * & * \\ & & & * & & & & \\ & * & * & & * & & * & \\ & * & & & & * & & \\ & & * & & * & & * & \\ * & * & * & & & & & * \end{pmatrix}$$

Index	$N(i)$	$\#N(i)$
1	*1,2,8*	*3*
2	*1,2,5,6,8*	*5*
3	*3,5,7,8*	*4*
4	*4*	*1*
5	*2,3,5,7*	*4*
6	*2,6*	*2*
7	*3,5,7*	*3*
8	*1,2,3,8*	*4*

Schritt $k = 1$: (1) Die Indexliste I ist leer. Der Index 4 hat nur einen Nachbarn (sich selbst), d.h.

$$i_1 = 4, \quad I = (4).$$

(2) Für den Index 4 gilt $N_1 = \mathcal{N}(4) \setminus I = \emptyset$, d.h. weiter mit $k = 2$.

Schritt $k = 2$: (1) Die Indexliste hat nur $k-1 = 1$ Element. Von den verbliebenden Indizes hat Index 6 die minimale Zahl von zwei Nachbarn, d.h.

$$i_2 = 6, \quad I = (4,6).$$

(2) Es gilt $N_2 = \mathcal{N}(6) \setminus I = \{2,6\} \setminus \{4,6\} = \{2\}$.

(3) Ein einzelnes Element ist natürlich sortiert, d.h.

$$i_3 = 2, \quad I = (4,6,2)$$

Schritt $k = 3$*: (1) Diese Schritt greift nicht, da* I *bereits 3 Elemente besitzt, es ist* $i_3 = 2$.

(2) Es gilt $N_3 = \mathcal{N}(2) \setminus I = \{1,2,5,6,8\} \setminus \{4,6,2\} = \{1,5,8\}$.

(3) Es gilt $\#N(1) = 3$, $\#N(5) = 4$ *und* $\#N(8) = 4$, *d.h. wir fügen sortiert hinzu:*

$$i_4 = 1,\ i_5 = 5,\ i_6 = 8, \quad I = (4,6,2,1,5,8).$$

Schritt $k = 4$*: (1)* I *hat mehr als* 4 *Elemente, d.h.* $i_4 = 1$.

(2) Es ist $N_4 = \mathcal{N}(1) \setminus I = \{1,2,8\} \setminus \{4,6,2,1,5,8\} = \emptyset$. *Daher weiter mit* $k = 5$

Schritt $k = 5$*: (1)* I *hat genug Elemente,* $i_5 = 5$.

(2) Es ist $N_5 = \mathcal{N}(5) \setminus I = \{2,3,5,7\} \setminus \{4,6,2,1,5,8\} = \{3,7\}$

(3) Es gilt $\#\mathcal{N}(3) = 4$ *und* $\#\mathcal{N}(7) = 3$, *d.h. Index* 7 *wird zuerst angefügt*

$$i_7 = 7,\ i_8 = 3, \quad I = (4,6,2,1,5,8,7,3).$$

Wir erhalten die Sortierung:

$$1 \to 4,\ 2 \to 6,\ 3 \to 2,\ 4 \to 1,\ 5 \to 5,\ 6 \to 8,\ 7 \to 7,\ 8 \to 3.$$

Wir erstellen die sortiere Matrix $\tilde{A}$ *gemäß* $\tilde{a}_{kj} = a_{i_k i_l}$*:*

$$A := \begin{array}{c} \begin{array}{cccccccc} 4 & 6 & 2 & 1 & 5 & 8 & 3 & 7 \end{array} \\ \begin{pmatrix} * & & & & & & & \\ & * & * & & & & & \\ & * & * & * & * & * & & \\ & & * & * & & * & & \\ & & * & & * & & * & * \\ & & * & * & & * & & * \\ & & & & * & & * & * \\ & & & & * & * & * & * \end{pmatrix} \end{array} \begin{array}{c} 4 \\ 6 \\ 2 \\ 1 \\ 5 \\ 8 \\ 7 \\ 3 \end{array}$$

Die so sortierte Matrix ist eine Bandmatrix mit Bandbreite $m = 4$.

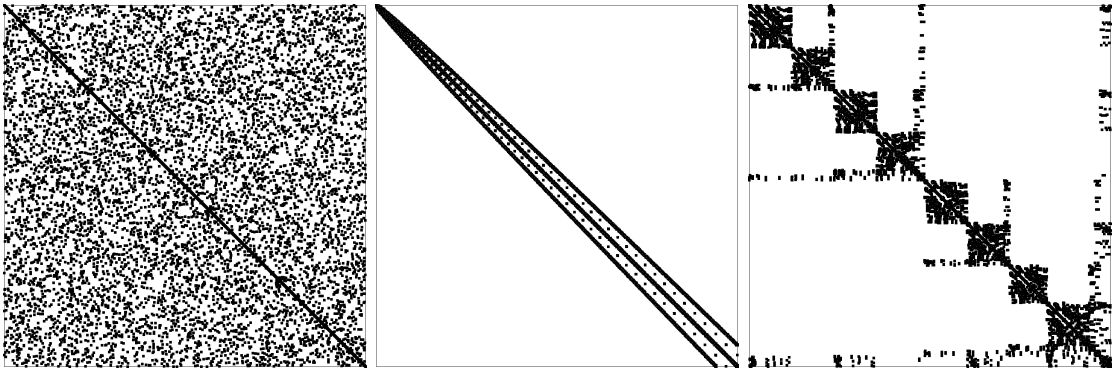

Abbildung 4.1: Besetzungsstruktur einer dünn besetzten Matrix $A \in \mathbb{R}^{1089\times 1089}$ mit insgesamt 9 409 von Null verschiedenen Einträgen. Links: vor Sortierung, Mitte: Cuthill-McKee Algorithmus und rechts: nach Sortierung mit einem Multi-Fronten-Verfahren aus [4].

Die meisten Verfahren basieren auf Methoden der Graphentheorie. Der Cuthill-McKee Algorithmus sucht eine Permutierung des Besetzungsmusters, so dass eng benachbarte Indizes auch in der Reihenfolge nahe beieinander stehen. Andere Methoden versuchen den durch das Besetzungsmuster aufgespannten Graphen möglichst in Teilgraphen zu zerlegen. Diese Teilgraphen entsprechen Blöcken in der Matrix A. Die anschließende LR-Zerlegung erzeugt dann voll besetzte, dafür kleine Blöcke. Die einzelnen Blöcke sind nur schwach gekoppelt. Ein Vorteil dieser Sortierung ist die Möglichkeit, effiziente Parallelisierungsverfahren für die einzelnen Blöcke zu verwenden.

In Abbildung 4.1 zeigen wir die Besetzungsstruktur einer Matrix A vor und nach entsprechender Sortierung. Die Matrix hat eine symmetrische Besetzungsstruktur und 9 409 Einträge ungleich Null (das sind weniger als 1% aller Einträge). Die LR-Zerlegung der unsortierten Matrix ist nahezu voll belegt mit etwa 1 000 000 Einträgen und deren Berechnung erfordert etwa $400 \cdot 10^6$ Operationen. Der Cuthill-McKee Algorithmus erzeugt eine Bandmatrix mit sehr Bandbreite $m \approx 70$. Zur Speicherung der LR-Zerlegung sind etwa 150 000 Einträge notwendig und die Berechnung erfordert etwa $5 \cdot 10^6$ Operationen. Schließlich zeigen wir zum Vergleich die Sortierung mit einem sogenannten *Multi-Fronten-Verfahren*. Hier wird die Matrix in einzelne Blöcke geteilt. Zur Berechnung der LR-Zerlegung sind hier $3 \cdot 10^6$ Operationen notwendig. Details hierzu finden sich in [4].

4.3 Nachiteration

Die numerisch durchgeführte LR-Zerlegung stellt aufgrund von Rundungsfehlern nur eine Näherung dar $A \approx LR$ und die so approximierte Lösung $LR\tilde{x} = b$ ist mit einem Fehler $x - \tilde{x}$ behaftet. Es stellt sich nun die Frage nach einer auswertbaren Abschätzung für diesen Fehler. Ein erster Anhaltspunkt wird durch den *Defekt* gegeben, siehe auch Definition 2.19:

Definition 4.43 (Defekt (LGS))**.** *Es sei $\tilde{x} \in \mathbb{R}^n$ die Näherung zur Lösung $x \in \mathbb{R}^n$ von $Ax = b$. Die Größe*

$$d(\tilde{x}) := b - A\tilde{x},$$

bezeichnet den Defekt*.*

Für die exakte Lösung $x \in \mathbb{R}^n$ von $Ax = b$ gilt $d(x) = 0$. Für allgemeine Approximationen erhalten wir die folgende *a posteriori* Fehlerabschätzung:

Satz 4.44 (Fehlerabschätzung für lineare Gleichungssysteme)**.** *Es sei $\tilde{x} \in \mathbb{R}^n$ die Approximation zur Lösung $x \in \mathbb{R}^n$ von $Ax = b$. Dann gilt:*

$$\frac{\|x - \tilde{x}\|}{\|x\|} \leq \operatorname{cond}(A) \frac{\|d(\tilde{x})\|}{\|b\|}.$$

Beweis: Es gilt:

$$x - \tilde{x} = A^{-1}(b - A\tilde{x}) = A^{-1}d(\tilde{x}) \quad \Rightarrow \quad \|x - \tilde{x}\| \leq \|A^{-1}\| \, \|d_{\tilde{x}}\|.$$

Teilen durch $\|b\| = \|Ax\| \leq \|A\| \, \|x\|$ liefert das gewünschte Ergebnis

$$\frac{\|x - \tilde{x}\|}{\|x\|} \leq \|A\| \, \|A^{-1}\| \, \frac{\|d(\tilde{x})\|}{\|b\|}.$$

□

Zur Veranschaulichung betrachten wir Beispiele 4.29 und 4.30 mit:

$$x \approx \begin{pmatrix} 0.34995 \\ -0.98024 \\ 2.1595 \end{pmatrix}, \quad \tilde{x}_1 = \begin{pmatrix} 2.37 \\ -3.55 \\ 2.15 \end{pmatrix}, \quad \tilde{x}_2 = \begin{pmatrix} 0.35 \\ -0.98 \\ 2.16 \end{pmatrix}$$

mit den relativen Fehlern

$$\frac{\|x - \tilde{x}_1\|_2}{\|x\|_2} \approx 1.5, \quad \frac{\|x - \tilde{x}_2\|_2}{\|x\|_2} \approx 0.00024,$$

sowie Defekten:

$$\frac{\|d(x_1)\|_2}{\|b\|_2} \approx 3.8, \quad \frac{\|d(x_2)\|_2}{\|b\|_2} \approx 0.0009.$$

Der Spektralkondition der Matrix A ist $\operatorname{cond}_2(A) \approx 6$, d.h., es ergeben sich die Fehlerschranken:

$$\frac{\|x - \tilde{x}_1\|_2}{\|x\|_2} \leq 6 \cdot 3.8 \approx 20, \quad \frac{\|x - \tilde{x}_2\|_2}{\|x\|_2} \leq 6 \cdot 0.0009 = 0.005.$$

In beiden Fällen wir der Fehler um einen Faktor $10 \sim 20$ überschätzt. Zur praktischen Auswertung dieser Fehlerabschätzung wird die Konditionszahl der Matrix A benötigt. Diese ist im Allgemeinen jedoch nicht verfügbar, da A^{-1} weder bekannt, noch einfach zu berechnen ist.

Neben seiner Rolle zur Fehlerabschätzung kommt dem Defekt eine weitere wichtige Bedeutung zu. Wir gehen davon aus, dass wir den Defekt $d(\tilde{x})$ ohne Rundungsfehler berechnen können. Weiter nehmen wir an, dass wir auch die *Defekt-Gleichung*

$$Aw = d(\tilde{x}) = b - A\tilde{x},$$

exakt nach $w \in \mathbb{R}^n$ lösen können. Dann gilt für $\tilde{x} + w$:

$$\tilde{x} + w = \tilde{x} + A^{-1}(b - A\tilde{x}) = \tilde{x} + x - \tilde{x} = x.$$

Dieser Vorgang wird *Defektkorrektur* oder *Nachiteration* genannt. Die Annahme, dass Defekt und Defektgleichung ohne Rundungsfehler gelöst werden können ist natürlich nicht realistisch (dann könnte ja auch das Original-System exakt gelöst werden). Dennoch erhalten wir als Grundlage der Nachiteration das folgende Ergebnis:

Satz 4.45 (Nachiteration)**.** *Es sei ϵ (klein genug) die Fehlertoleranz. Durch $\tilde{x} \in \mathbb{R}^n$ sei eine Approximation zu $Ax = b$ gegeben. Weiter sei $\tilde{d}$ eine Approximation zu $d(\tilde{x})$ mit doppelter Genauigkeit:*

$$\frac{\|d(\tilde{x}) - \tilde{d}\|}{\|d(\tilde{x})\|} \leq \operatorname{cond}(A)\epsilon^2. \tag{4.3}$$

Es sei $\tilde{w}$ eine Approximation der Defektgleichung $Aw = \tilde{d}$ mit einfacher Genauigkeit

$$\frac{\|w - \tilde{w}\|}{\|w\|} \leq \operatorname{cond}(A)\epsilon. \tag{4.4}$$

Dann gilt für die Korrektur $\tilde{x} + \tilde{w}$ die Abschätzung

$$\frac{\|x - (\tilde{x} + \tilde{w})\|}{\|x\|} \leq \epsilon c(A) \frac{\|x - \tilde{x}\|}{\|x\|},$$

mit einer Konstante $c(A)$, die von der Konditionszahl $\operatorname{cond}(A)$ *abhängt.*

Beweis: Wir definieren zunächst eine Hilfsgröße: es sei $\hat{w}$ die exakte Lösung der exakten Defektgleichung $A\hat{w} = d(\tilde{x})$:

$$A\hat{w} = b - A\tilde{x} \quad \Rightarrow \quad \hat{w} = A^{-1}b - A^{-1}A\tilde{x} = x - \tilde{x}. \tag{4.5}$$

Für den Fehler $\hat{w} - w$ zwischen den exakten Lösungen von $A\hat{w} = d(\tilde{x})$ und $Aw = \tilde{d}$ gilt laut Störungssatz 4.19:

$$\frac{\|\hat{w} - w\|}{\|\hat{w}\|} \leq \operatorname{cond}(A) \underbrace{\frac{\|d(\tilde{x}) - \tilde{d}\|}{\|d(\tilde{x})\|}}_{(4.3)} \leq \epsilon^2 \operatorname{cond}(A)^2 \tag{4.6}$$

Jetzt folgt durch Einschieben von $\pm\hat{w}$ sowie $\pm w$ in den Fehler $\|x-(\tilde{x}+\tilde{w})\|$:

$$\begin{aligned}\|x-(\tilde{x}+\tilde{w})\| &\leq \underbrace{\|x-(\tilde{x}+\hat{w})\|}_{=0 \text{ wegen } (4.5)} + \underbrace{\|\hat{w}-w\|}_{(4.6)} + \underbrace{\|w-\tilde{w}\|}_{(4.4)} \\ &\leq \epsilon^2 \operatorname{cond}(A)^2\|\hat{w}\| + \epsilon \operatorname{cond}(A)\|w\| \\ &\leq \epsilon \operatorname{cond}(A)\big(\epsilon \operatorname{cond}(A)\|\hat{w}\| + \|\hat{w}\| + \underbrace{\|w-\hat{w}\|}_{(4.6)}\big) \\ &\leq \epsilon \operatorname{cond}(A)\big(1+\epsilon \operatorname{cond}(A) + \epsilon^2 \operatorname{cond}(A)^2\big)\|\hat{w}\| \\ &\leq \epsilon c(A)\|x-\tilde{x}\|,\end{aligned}$$

mit $c(A) := \operatorname{cond}(A)(1+\epsilon \operatorname{cond}(A) + \epsilon^2 \operatorname{cond}(A)^2)$. Das Ergebnis mit Teilen durch $\|x\|$. □

Durch einen Nachiterationsschritt kann der Fehler um den Faktor $\epsilon c(A)$ reduziert werden. Die Konstante $c(A)$ hängt dabei allerdings sehr ungünstig von der oft sehr großen Konditionszahl der Matrix ab. Die Nachiteration ist ein universelles Prinzip und nicht auf die LR-Zerlegung beschränkt. Dennoch definieren wir für diese:

Algorithmus 4.46 (Nachiteration)**.** *Es sei $A \in \mathbb{R}^{n\times n}$ eine reguläre Matrix und $b \in \mathbb{R}^n$. Zur Lösung von $Ax = b$:*

1. *Erstelle LR-Zerlegung von A mit einfacher Genauigkeit*

$$LR = PA.$$

2. *Setze $d^{(1)} = b$ sowie $x^{(1)} = 0$ und iteriere für $i = 1, 2, \ldots$:*

$$\begin{aligned}(i)\quad & Ly^{(i)} = Pd^{(i)} && \text{mit einfacher Genauigkeit}\\ (ii)\quad & Rw^{(i)} = y^{(i)} && \text{mit einfacher Genauigkeit}\\ (iii)\quad & x^{(i+1)} = x^{(i)} + w^{(i)} && \text{mit doppelter Genauigkeit}\\ (iv)\quad & d^{(i+1)} = b - Ax^{(i+1)} && \text{mit doppelter Genauigkeit}\end{aligned}$$

Der Vorteil der Nachiteration liegt in der mehrfachen Verwendung der erstellten LR-Zerlegung. Zur Berechnung der LR-Zerlegung sind $O(n^3)$ Operationen notwendig, während zum Vorwärts- und Rückwärtseinsetzen, sowie zur Berechnung des Defektes nur $O(n^2)$ Operationen benötigt werden. D.h., selbst bei Verwenden höherer Genauigkeit ist der Aufwand in Schritt *2.(iii)* des Verfahrens klein im Vergleich zu Schritt *1.*

Die Annahme, dass zum Erstellen der LR-Zerlegung mit geringerer Genauigkeit gerechnet wird, also zur Defektberechnung ist nicht unrealistisch. Der Speichertyp `float` von einfacher Genauigkeit benötigt zur Speicherung einer Zahl nur den halben Speicher verglichen mit `double`. Gerade bei sehr großen Matrizen $n \gg 1\,000\,000$ spielt der Speicherbedarf eine wesentliche Rolle. Darüber hinaus unterscheidet moderne Hardware (z.B. GPU's) zwischen der Rechnung mit doppelter und einfacher Genauigkeit, deren Operationen oft weit schneller durchgeführt werden können.

Beispiel 4.47 (Nachiteration)**.** *Wir führen Beispiel 4.30 fort. Zur Approximation $\tilde{x}_2$ berechnen wir den Defekt $d(\tilde{x}_2)$ mit doppelter Genauigkeit (hier sogar exakt):*

$$\tilde{x}_2 = \begin{pmatrix} 0.35 \\ -0.98 \\ 2.16 \end{pmatrix}, \quad d(\tilde{x}_2) = b - A\tilde{x}_2 = \begin{pmatrix} 0.001 \\ 0 \\ -0.002 \end{pmatrix}.$$

Mit dreistelliger (also hier einfacher) Genauigkeit lösen wir zunächst $\tilde{y} = Pd(\tilde{x})$ mit $Pd(\tilde{x}) = (-0.001, -0.002, 0)^T$ und erhalten:

$$\tilde{y} = \begin{pmatrix} 0.001 \\ -0.00165 \\ 0.000611 \end{pmatrix}.$$

Rückwärtseinsetzen $R\tilde{w} = \tilde{y}$ mit dreistelliger Genauigkeit ergibt:

$$\tilde{w} = \begin{pmatrix} -0.0000545 \\ -0.000227 \\ -0.000466 \end{pmatrix}.$$

Wir berechnen die korrigierte Lösung $\hat{x} := \tilde{x} + \tilde{w}$ mit sechsstelliger (also doppelter) Genauigkeit zu:

$$\hat{x} = \begin{pmatrix} 0.349946 \\ -0.980227 \\ 2.15953 \end{pmatrix}.$$

Diese verbesserte Lösung ist mit dem relativen Fehler

$$\frac{\|\hat{x} - x\|_2}{\|x\|_2} \approx 0.000017 \cdot 10^{-5}$$

versehen. Der Fehler konnte durch einen Nachiterationsschritt um zwei Größenordnungen verbessert werden!

Mit der gestörten LR-Zerlegung $\tilde{L}\tilde{R} \approx A$ lässt sich ein Nachiterationsschritt kompakt schreiben als:

$$x^{(i+1)} = x^{(i)} + Cd(\tilde{x}^{(i)}), \quad C := \tilde{R}^{-1}\tilde{L}^{-1}.$$

Dabei ist die Matrix C eine Approximation zur Inversen von A. Es stellt sich die Frage, ob die Nachiteration auch dann ein konvergentes Verfahren bildet, wenn $\tilde{C}$ eine noch gröbere Approximation der Inversen $\tilde{C} \approx A^{-1}$ ist. Dabei könnte man an Approximationen denken, die auf der einen Seite weiter von A^{-1} entfernt sind, dafür aber wesentlich einfacher, z.B. in $O(n^2)$ Operationen zu erstellen sind. Dieser Ansatz ist Ausgangspunkt von allgemeinen *Defektkorrektur-Verfahren*, wie wir sie in Kapitel 5 untersuchen werden.

4.4 Orthogonalisierungsverfahren und die QR-Zerlegung

Die Zerlegung einer regulären Matrix $A \in \mathbb{R}^{n\times n}$ in die beiden Dreiecksmatrizen L und R basiert auf der Elimination mit Frobeniusmatrizen, d.h. $R = FA$, mit $L := F^{-1}$. Es gilt:

$$\operatorname{cond}(R) = \operatorname{cond}(FA) \leq \operatorname{cond}(F)\operatorname{cond}(A) = \|F\|\,\|L\|\operatorname{cond}(A).$$

Bei entsprechender Pivotisierung gilt für die Einträge der Matrizen F sowie L $|f_{ij}| \leq 1$ und $|l_{ij}| \leq 1$, siehe Satz 4.28. Dennoch können die Normen von L und F im Allgemeinen nicht günstiger als $\|F\|_\infty \leq n$ und $\|L\|_\infty \leq n$ abgeschätzt werden. Es gilt dann:

$$\operatorname{cond}_\infty(R) \leq n^2 \operatorname{cond}_\infty(A).$$

Die Matrix R, welche zur Rückwärtselimination gelöst werden muss, hat eine unter Umständen weit schlechtere Konditionierung als die Matrix A selbst (welche auch schon sehr schlecht konditioniert sein kann). Wir suchen im Folgenden einen Zerlegungsprozess in eine Dreiecksmatrix, welche numerisch stabil ist, indem die Zerlegung nur mit Hilfe von orthogonalen Matrizen $Q \in \mathbb{R}^{n\times n}$ durchgeführt wird, für welche $\operatorname{cond}_2(Q) = 1$ gilt:

Definition 4.48 (Orthogonale Matrix)**.** *Eine Matrix $Q \in \mathbb{R}^{n\times n}$ heißt* orthogonal*, falls ihre Zeilen und Spaltenvektoren eine Orthonormalbasis des $\mathbb{R}^n$ bilden.*

Es gilt:

Satz 4.49 (Orthogonale Matrix)**.** *Es sei $Q \in \mathbb{R}^{n\times n}$ eine orthogonale Matrix. Dann ist Q regulär und es gilt:*

$$Q^{-1} = Q^T, \quad Q^TQ = I, \quad \|Q\|_2 = 1, \quad \operatorname{cond}_2(Q) = I.$$

Es gilt $\det(Q) = 1$ *oder* $\det(Q) = -1$*. Für zwei orthogonale Matrizen $Q_1, Q_2 \in \mathbb{R}^{n\times n}$ ist auch das Produkt Q_1Q_2 eine orthogonale Matrix. Für eine beliebige Matrix $A \in \mathbb{R}^{n\times n}$ gilt $\|QA\|_2 = \|A\|_2$. Für beliebige Vektoren $x, y \in \mathbb{R}^n$ gilt:*

$$\|Qx\|_2 = \|x\|_2, \quad (Qx, Qy)_2 = (x, y)_2.$$

Beweis: Wir beweisen hier nur die im Kontext der numerischen Stabilität wesentlich Eigenschaft, dass die Multiplikation mit orthogonalen Matrizen die Kondition einer Matrix (d.h. die 2-Norm) nicht verändert, die weiteren Teile des Beweises belassen wir als Übung. Es gilt:

$$\|QAx\|_2^2 = (QAx, QAx)_2 = (Q^TQAx, Ax)_2 = (Ax, Ax)_2 = \|Ax\|_2^2.$$

□

Wir definieren:

Definition 4.50 (QR-Zerlegung)**.** *Die Zerlegung einer Matrix $A \in \mathbb{R}^{n\times n}$ in eine orthogonale Matrix $Q \in \mathbb{R}^{n\times n}$ sowie eine rechte obere Dreiecksmatrix $R \in \mathbb{R}^{n\times n}$ gemäß*

$$A = QR,$$

heißt QR-Zerlegung.

Die QR-Zerlegung hat den Vorteil, dass die Matrix $R = Q^T A$ die gleiche Konditionszahl hat wie die Matrix A selbst. Einmal erstellt, kann die QR-Zerlegung genutzt werden, um lineare Gleichungssysteme mit der Matrix A effizient zu lösen:

$$Ax = b \quad \Leftrightarrow \quad Q^T Ax = Q^T b \quad \Leftrightarrow \quad Rx = Q^T b.$$

Zur Realisierung der QR-Zerlegung stellt sich die Frage nach der Konstruktion orthogonaler Matrizen Q zur Reduktion von A auf Dreiecksgestalt.

4.4.1 Das Gram-Schmidt Verfahren

Der wichtigste Algorithmus zum Erstellen einer Orthonormalbasis ist das Orthogonalisierungsverfahren nach Gram-Schmidt:

Satz 4.51 (Gram-Schmidt Orthonormalisierungsverfahren)**.** *Es sei durch $\{a_1, \ldots, a_n\}$ eine Basis des $\mathbb{R}^n$ gegeben, durch $(\cdot,\cdot)$ ein Skalarprodukt mit induzierter Norm $\|\cdot\|$. Die Vorschrift:*

$$(i) \quad q_1 := \frac{a_1}{\|a_1\|},$$

$$i = 2, \ldots, n: \quad (ii) \quad \tilde{q}_i := a_i - \sum_{j=1}^{i-1} (a_i, q_j) q_j, \quad q_i := \frac{\tilde{q}_i}{\|\tilde{q}_i\|},$$

erzeugt eine Orthonormalbasis $\{q_1, \ldots, q_n\}$ des $\mathbb{R}^n$. Es gilt ferner:

$$(q_i, a_j) = 0 \quad \forall 1 \le j < i \le n.$$

Beweis: Wir führen den Beweis per Induktion. Für $i = 1$ ist $a_1 \neq 0$, da durch $a_1, \ldots, a_n$ der ganze $\mathbb{R}^n$ aufgespannt wird. Für $i = 2$ gilt:

$$(\tilde{q}_2, q_1) = (a_2, q_1) - (a_2, q_1) \underbrace{(q_1, q_1)}_{=1} = 0.$$

Aus der linearen Unabhängigkeit von a_2 und q_1 folgt, dass $\tilde{q}_2 \neq 0$. Es sei nun also $(q_k, q_l) = \delta_{kl}$ für $k, l > i$. Dann gilt für $k < i$ beliebig

$$(\tilde{q}_i, q_k) = (a_i, q_k) - \sum_{j=1}^{i-1} (a_i, q_j) \underbrace{(q_j, q_k)}_{=\delta_{jk}} = (a_i, q_k) - (a_i, q_k) = 0.$$

Da $\text{span}\{q_1, \ldots, q_j\} = \text{span}\{a_1, \ldots, a_j\}$ folgt auch $(q_i, a_j) = 0$ für $j < i$. □

Mit Hilfe des Gram-Schmidt Verfahrens lässt sich die QR-Zerlegung einer Matrix $A \in \mathbb{R}^{n\times n}$ unmittelbar konstruieren:

Satz 4.52 (QR-Zerlegung)**.** *Es sei $A \in \mathbb{R}^{n\times n}$ eine reguläre Matrix. Dann existiert eine QR-Zerlegung $A = QR$ in eine orthogonale Matrix $Q \in \mathbb{R}^{n\times n}$ und eine rechte obere Dreiecksmatrix $R \in \mathbb{R}^{n\times n}$.*

Beweis: Es seien $A = (a_1, \ldots, a_n)$ die Spaltenvektoren der Matrix A. Da A regulär ist, sind die Vektoren a_i linear unabhängig und bilden eine Basis des $\mathbb{R}^n$. Es sei $\{q_1, \ldots, q_n\}$ die durch das Gram-Schmidt Verfahren aus $\{a_1, \ldots, a_n\}$ konstruierte Orthonormalbasis. Dann ist durch $Q = (q_1, \ldots, q_n) \in \mathbb{R}^{n\times n}$ eine orthogonale Matrix gegeben. Die Matrix $R := Q^T A$ ist regulär und aufgrund von Satz 4.51 gilt für ihre Einträge:

$$r_{ij} = (q_i, a_j) = 0 \quad \forall j < i.$$

Das heißt: R ist eine rechte obere Dreiecksmatrix. □

Die QR-Zerlegung einer Matrix kann ohne einer weiteren Normierungsbedingung nicht eindeutig sein. Angenommen, wir modifizieren den Normierungsschritt *(ii)* im Gram-Schmidt Verfahren Satz 4.51) zu:

$$q_i' := -\frac{\tilde{q}_i}{\|\tilde{q}_i\|}.$$

Dann wäre das resultierende System $\{q_1, \ldots, q_n\}$ wieder orthonormal und mit $Q'R' = QR$ wäre eine zweite (echt verschiedene) QR-Zerlegung gegeben. Wir können das Vorzeichen von q_i in jedem Schritt so wählen, dass $r_{ii} = (q_i, a_i) > 0$, dass also R nur positive Diagonalelemente besitzt. Dann gilt:

Satz 4.53 (Eindeutigkeit der QR-Zerlegung)**.** *Die QR-Zerlegung $A = QR$ einer regulären Matrix $A \in \mathbb{R}^{n\times n}$ mit $r_{ii} > 0$ ist eindeutig.*

Beweis: Es seien $A = Q_1R_1 = Q_2R_2$ zwei QR-Zerlegungen von A. Dann gilt:

$$Q_2^T Q_1 = R_2 R_1^{-1}, \quad Q_1^T Q_2 = R_1 R_2^{-1}.$$

Die Produkte $R_2R_1^{-1}$ sowie $R_1R_2^{-1}$ sind rechte obere Dreiecksmatrizen. Weiter gilt $Q := Q_2^T Q_1 = (Q_1^T Q_2)^T = Q^T$, d.h. Q ist eine orthogonale Matrix mit $Q^{-1} = Q^T$. Wegen $Q^{-1} = Q^T$ und $Q = R_2R_1^{-1}$ muss Q eine Diagonalmatrix sein. Aus der Beziehung

$$QR_1 = Q_2^T Q_1 R_1 = Q_2^T A = R_2$$

folgt für den j-ten Einheitsvektor e_j:

$$QR_1 e_j = q_{jj} r_{jj}^1 = r_{jj}^2 > 0,$$

also $q_{jj} > 0$ und wegen der Orthogonalität $Q = I$. D.h.:

$$R_1 = R_2 \quad \Rightarrow \quad Q_1 = AR_1^{-1} = AR_2^{-1} = Q_2.$$

□

Die große Schwäche des Gram-Schmidt Verfahrens ist die geringe numerische Stabilität, insbesondere, wenn die Vektoren a_i "fast parallel" sind. In diesem Fall droht Auslöschung. Wir betrachten hierzu ein Beispiel:

Beispiel 4.54 (QR-Zerlegung mit Gram-Schmidt)**.** *Es sei die Matrix*

$$A := \begin{pmatrix} 1 & 1 & 1 \\ 0.01 & 0 & 0.01 \\ 0 & 0.01 & 0.01 \end{pmatrix}$$

gegeben. Wir bestimmen mit dem Gram-Schmidt Verfahren aus den Spaltenvektoren $A = (a_1, a_2, a_3)$ die entsprechende Orthonormalbasis. Bei dreistelliger Genauigkeit erhalten wir:

$$q_1 = \frac{q_1}{\|q_1\|} \approx \begin{pmatrix} 1 \\ 0.01 \\ 0 \end{pmatrix}.$$

Weiter:

$$\tilde{q}_2 = q_2 - (a_2, q_1)q_1 \approx a_2 - q_1 = \begin{pmatrix} 0 \\ -0.01 \\ 0.01 \end{pmatrix}, \quad q_2 = \frac{\tilde{q}_2}{\|\tilde{q}_2\|} \approx \begin{pmatrix} 0 \\ -0.707 \\ 0.707 \end{pmatrix}.$$

Schließlich:

$$\tilde{q}_3 = a_3 - (a_3, q_1)q_1 - (a_3, q_2)q_2 \approx a_3 - q_1 - 0 = \begin{pmatrix} 0 \\ 0 \\ 0.01 \end{pmatrix}, \quad q_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}.$$

Die QR-Zerlegung ergibt sich mit der "orthogonalen Matrix"

$$\tilde{Q} = \begin{pmatrix} 1 & 0 & 0 \\ 0.01 & -0.707 & 0 \\ 0 & 0.707 & 1 \end{pmatrix},$$

sowie der rechten oberen Dreiecksmatrix $\tilde{R}$ mit $\tilde{r}_{ij} = (q_i, a_j)$ für $j \geq i$:

$$\tilde{R} := \begin{pmatrix} (q_1, a_1) & (q_1, a_2) & (q_1, a_3) \\ 0 & (q_2, a_2) & (q_2, a_3) \\ 0 & 0 & (q_3, a_3) \end{pmatrix} \approx \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0.00707 & 0 \\ 0 & 0 & 0.01 \end{pmatrix}.$$

Die Probe $A = \tilde{Q}\tilde{R}$ ergibt:

$$\begin{pmatrix} 1 & 0 & 0 \\ 0.01 & -0.707 & 0 \\ 0 & 0.707 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0.00707 & 0 \\ 0 & 0 & 0.01 \end{pmatrix} \approx \begin{pmatrix} 1 & 1 & 1 \\ 0.01 & 0.005 & 0.01 \\ 0 & 0.005 & 0.01 \end{pmatrix} =: \tilde{A}$$

Auf den ersten Blick sieht $\tilde{A}$ nach einer brauchbaren Näherung für A aus. Der relative Fehler $\|\tilde{Q}\tilde{R} - A\|_2/\|A\|_2 \approx 0.004$ ist (beachte dreistellige Arithmetik) nicht sonderlich groß. $\tilde{A}$ ist jedoch nicht einmal regulär! Dieser wesentliche Fehler liegt an der Instabilität des Gram-Schmidt-Verfahrens. Für das berechnete $\tilde{Q}$ gilt:

$$I \stackrel{!}{=} \tilde{Q}^T\tilde{Q} \approx \begin{pmatrix} 1 & -0.00707 & 0 \\ -0.00707 & 1 & 0.707 \\ 0 & 0.707 & 1 \end{pmatrix},$$

mit einem relativen Fehler $\|\tilde{Q}^T\tilde{Q} - I\|_2 \approx 0.7$!

Die QR-Zerlegung hat prinzipiell bessere Stabilitätseigenschaften als die LR-Zerlegung. Das Gram-Schmidt-Verfahren eignet sich jedoch nicht, um die orthogonale Matrix Q zu erstellen. Im Folgenden entwickeln wir eine Transformationsmethode zum Erstellen der QR-Zerlegung welche selbst auf orthogonalen, und somit optimal konditionierten, Transformationen aufbaut.

4.4.2 Householder-Transformationen

Das geometrische Prinzip hinter dem Gram-Schmidt Verfahren ist die Projektion der Vektoren a_i auf die bereits erstellte Orthogonalbasis $q_1, \ldots, q_{i-1}$. Diese Projektion ist schlecht konditioniert, falls a_i fast parallel zu den q_j mit $j < i$, etwa $a_i \approx q_j$ ist. Dann droht Auslöschung. Wir können einen Schritt des Gram-Schmidt Verfahrens kompakt mit Hilfe eines Matrix-Vektor Produktes schreiben (komponentenweise nachrechnen!)

$$\tilde{q}_i = [I - G^{(i)}]a_i, \quad G^{(i)} = \sum_{l=1}^{i-1} q_l q_l^T,$$

mit dem *dyadischen Produkt* zweier Vektoren $vv^T \in \mathbb{R}^{n\times n}$. Die Matrix $[I - G^i]$ stellt eine Projektion dar:

$$[I - G^i]^2 = I - 2\sum_{l=1}^{i-1} q_l q_l^T + \sum_{k,l=1}^{i-1} q_l \underbrace{q_l^T q_k}_{=\delta_{lk}} q_k^T = [I - G^i].$$

Weiter gilt:

$$[I - G^i]q_k = q_k - \sum_{l=1}^{i-1} q_l \underbrace{q_l^T q_k}_{=\delta_{lk}} = 0.$$

Die Matrix $I - G^i$ ist also nicht regulär. Falls in Schritt i der Vektor a_i fast parallel zu den bereits orthogonalen Vektoren ist, also $\tilde{a}_i \in \delta a_i + \text{span}\{q_1, \ldots, q_{i-1}\}$, so gilt

$$\tilde{q}_i = [I - G^i]\tilde{a}_i = [I - G^i]\delta a_i \quad \Rightarrow \quad \frac{\|\delta q_i\|}{\|\tilde{q}_i\|} = \frac{\|\delta q_i\|}{\|[I - G^i]\delta a_i\|} \sim \frac{\|\delta a_i\|}{\|a_i\|}.$$

Bei $\delta a_i \to 0$ ist also eine beliebig große Fehlerverstärkung möglich. Man vergleiche Bemerkung 4.16 zur Konditionierung der Matrix-Vektor Multiplikation (bei regulärer Matrix).

Im Folgenden suchen wir eine Transformation von A zu einer Dreiecksmatrix R, die selbst auf orthogonalen Operationen aufbaut. Eine orthogonale Matrix $Q \in \mathbb{R}^{n\times n}$ mit $\det(Q) = 1$ stellt eine Drehung dar und bei $\det(Q) = -1$ eine Spiegelung (oder eine Drehspiegelung). Die Householder-Transformationen nutzen das Prinzip der Spiegelung, um eine Matrix $A \in \mathbb{R}^{n\times n}$ in einer rechte obere Dreiecksmatrix zu transformieren.

Definition 4.55 (Householder-Transformation)**.** *Für einen Vektor $v \in \mathbb{R}^n$ mit $\|v\|_2 = 1$ ist durch vv^T das* dyadische Produkt *definiert und die Matrix*

$$S := I - 2vv^T \in \mathbb{R}^{n\times n},$$

heißt Householder-Transformation*.*

Es gilt:

Satz 4.56 (Householder-Transformation)**.** *Jede Householder-Transformation $S = I-2vv^T$ ist symmetrisch und orthogonal. Das Produkt zweier Householder-Transformationen S_1S_2 ist wieder einer symmetrische orthogonale Matrix.*

Beweis: Es gilt:

$$S^T = [I - 2vv^T]^T = I - 2(vv^T)^T = I - 2vv^T = S.$$

Weiter gilt:

$$S^TS = I - 4vv^T + 4v\underbrace{v^Tv}_{=1}v^T = I,$$

d.h. $S^{-1} = S^T$. Mit zwei symmetrischen orthogonalen Matrizen S_1 sowie S_2 gilt:

$$(S_1S_2)^T = S_2^TS_1^T = S_2S_1, \quad (S_1S_2)^{-1} = S_2^{-1}S_1^{-1} = S_2^TS_1^T = (S_1S_2)^T.$$

□

Wir schließen die grundlegende Untersuchung der Householder-Transformationen mit einer geometrischen Charakterisierung ab:

Bemerkung 4.57 (Householder-Transformation als Spiegelung)**.** *Es sei $v \in \mathbb{R}^n$ ein beliebiger normierter Vektor mit $\|v\|_2 = 1$. Weiter sei $x \in \mathbb{R}^n$ gegeben mit $x = \alpha v + w^\perp$, wobei $w^\perp \in \mathbb{R}^n$ ein Vektor mit $v^Tw^\perp = 0$ im orthogonalen Komplement zu v ist. Dann gilt für die Householder-Transformation $S := I - 2vv^T$:*

$$S(\alpha v + w^\perp) = [I - 2vv^T](\alpha v + w^\perp) = \alpha(v - 2v\underbrace{v^Tv}_{=1}) + w^\perp - 2v\underbrace{v^Tw^\perp}_{=0} = -\alpha v + w^\perp.$$

D.h., die Householder-Transformation beschreibt eine Spiegelung an der auf v senkrecht stehenden Ebene.

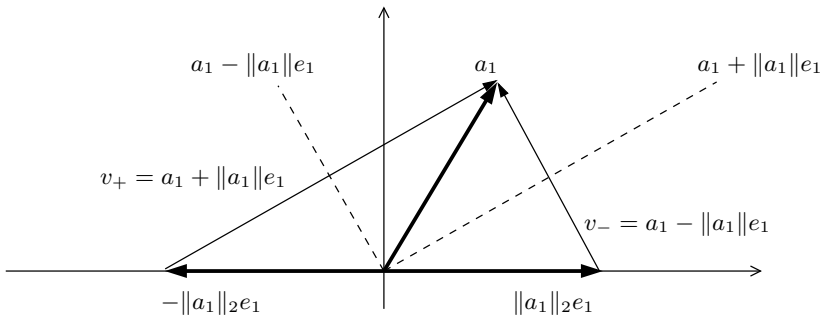

Abbildung 4.2: Spiegelungsachsen (gestrichelt) und Normalen zur Spiegelung v_+ und v_- zur Spiegelung von a_1 auf $\text{span}\{e_1\}$.

Mit Hilfe der Householder-Transformationen soll eine Matrix $A \in \mathbb{R}^{n\times n}$ nun Schritt für Schritt in eine rechte obere Dreiecksmatrix transformiert werden:

$$A^{(0)} := A, \quad A^{(i)} = S^{(i)}A^{(i-1)}, \quad S^{(i)} = I - 2v^{(i)}(v^{(i)})^T,$$

mit $R := A^{(n-1)}$. Wir beschreiben den ersten Schritt des Verfahrens: die reguläre Matrix $A \in \mathbb{R}^{n\times n}$ soll durch Multiplikation mit einer orthogonalen Householder-Transformation so transformiert werden $A^{(1)} = S^{(1)}A$, dass $a_{i1}^{(1)} = 0$ für alle $i > 1$ unterhalb der ersten Diagonale. Hierzu schreiben wir die Matrix $A = (a_1, a_2, \ldots, a_n)$ mit ihren Spaltenvektoren. Dann ist $A^{(1)} = (a_1^{(1)}, \ldots, a_n^{(1)})$ mit $a_i^{(1)} = S^{(1)}a_i$. Wir suchen die Householder-Transformation $S^{(1)}$, so dass:

$$e_1 \stackrel{!}{=} a_1^{(1)} = S^{(1)}a_1.$$

Hierzu müssen wir auf der Ebene senkrecht zu $a_1 \pm \|a_1\|e_1$ spiegeln, siehe Abbildung 4.2. Wir wählen

$$v := \frac{a_1 + \text{sign}(a_{11})\|a_1\|e_1}{\|a_1 + \text{sign}(a_{11})\|a_1\|e_1\|}, \tag{4.7}$$

um durch optimale Wahl des Vorzeichens die Gefahr von Auslöschung zu vermeiden. Mit dieser Wahl gilt:

$$a_i^{(1)} = S^{(1)}a_i = a_i - 2(v, a_i)v, \quad i = 2, \ldots, n, \quad a_1^{(1)} = -\,\text{sign}(a_{11})\|a_1\|e_1. \tag{4.8}$$

Die resultierende Matrix $\tilde{A}^{(1)}$ ist wieder regulär und es gilt $\tilde{a}_1^{(1)} \in \text{span}(e_1)$:

$$A := \begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & \ddots & & \vdots \\ a_{31} & a_{32} & \ddots & \ddots & \vdots \\ \vdots & & & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & \cdots & a_{nn} \end{pmatrix} \quad \to \quad A^{(1)} := S^{(1)}A = \left(\begin{array}{c|cccc} a_{11}^{(1)} & a_{12}^{(1)} & a_{13}^{(1)} & \cdots & a_{1n}^{(1)} \\ \hline 0 & a_{22}^{(1)} & \ddots & & \vdots \\ 0 & a_{32}^{(1)} & \ddots & \ddots & \vdots \\ \vdots & & & \ddots & \vdots \\ 0 & a_{n2}^{(1)} & \cdots & \cdots & a_{nn}^{(1)} \end{array}\right).$$

Wir setzen das Verfahren mit der Teilmatrix $A^{(1)}_{kl>1}$ fort. Hierzu sei der verkürzte zweite Spaltenvektor definiert als $\tilde{a}^{(1)} = (a^{(1)}_{22}, a^{(1)}_{32}, \ldots, a^{(1)}_{n2})^T \in \mathbb{R}^{n-1}$. Dann wählen wir:

$$\mathbb{R}^{n-1} \ni \tilde{v}^{(2)} := \frac{\tilde{a}^{(1)} + \operatorname{sign}(a^{(1)}_{22})\|\tilde{a}^{(1)}\|e_1}{\left\|\tilde{a}^{(1)} + \operatorname{sign}(a^{(1)}_{22})\|\tilde{a}^{(1)}\|e_1\right\|}, \quad S^{(2)} = \left(\begin{array}{c|ccc} 1 & 0 & \cdots & 0 \\ \hline 0 & & & \\ 0 & & I - 2\tilde{v}^{(2)}(v^{(2)})^T & \\ \vdots & & & \\ 0 & & & \end{array}\right) \in \mathbb{R}^{n\times n}.$$

Multiplikation mit $S^{(2)}$ von links, also $A^{(2)} := S^{(2)}A^{(1)}$ lässt die erste Zeile unverändert. Eliminiert wird der Block $A^{(1)}_{kl>1}$. Für die resultierende Matrix $A^{(2)}$ gilt $a^{(2)}_{kl} = 0$ für $l = 1, 2$ und $k > l$. Nach $n - 1$ Schritten gilt:

$$R := \underbrace{S^{(n-1)}S^{(n-2)}\cdots S^{(1)}}_{=:Q^T} A.$$

Alle Transformationen sind orthogonal, somit ist auch $Q \in \mathbb{R}^{n\times n}$ eine orthogonale (und natürlich reguläre) Matrix.

Wir fassen zusammen:

Satz 4.58 (QR-Zerlegung mit Householder-Transformationen)**.** *Es sei $A \in \mathbb{R}^{n\times n}$ eine reguläre Matrix. Dann lässt sich die QR-Zerlegung von A nach Householder in*

$$\frac{2n^3}{3} + O(n^2)$$

arithmetische Operationen numerisch stabil durchführen.

Beweis: Die Durchführbarkeit der QR-Zerlegung geht aus dem Konstruktionsprinzip hervor. Aus der Regularität der Matrizen $A^{(i)}$ folgt, dass der Teilvektor $\tilde{a}^{(i)} \in \mathbb{R}^{n-i}$ ungleich Null sein muss. Dann ist $\tilde{v}^{(i)}$ gemäß (4.7) wohl definiert. Die folgende Elimination wird gemäß (4.8) spaltenweise durchgeführt:

$$\tilde{a}^{(i+1)}_k = \underbrace{[I - 2\tilde{v}^{(i)}(\tilde{v}^{(i)})^T]}_{=\tilde{S}^{(i)}} \tilde{a}^{(i)}_k = \tilde{a}^{(i)}_k - 2(\tilde{v}^{(i)}, \tilde{a}^{(i)}_k)\tilde{v}^{(i)}.$$

Die numerische Stabilität folgt aus $\operatorname{cond}_2(S^{(i)}) = 1$, siehe Bemerkung 4.16.

Wir kommen nun zur Abschätzung des Aufwands. Im Schritt $A^{(i)} \to A^{(i+1)}$ muss zunächst der Vektor $\tilde{v}^{(i)} \in \mathbb{R}^{n-i}$ berechnet werden. Hierzu sind $2(n - i)$ arithmetische Operationen notwendig. Im Anschluss erfolgt die spaltenweise Elimination. (Es wird natürlich nicht die Matrix $S^{(i)}$ aufgestellt und die Matrix-Matrix Multiplikation durchgeführt!) Für jeden der

$n-i$ Spaltenvektoren ist ein Skalarprodukt $(\tilde{v}^{(i)}, \tilde{a}^{(i)})$ (das sind $n-i$ arithmetische Operationen) sowie eine Vektoraddition (weitere $n-i$ Operationen) durchzuführen. Insgesamt ergibt sich so ein Aufwand:

$$N_{QR} = 2\sum_{i=1}^{n-1}(n-i) + (n-i)^2 = n(n-1) + \frac{n(n-1)(2n-1)}{3} = \frac{2n^3}{3} + O(n^2).$$

□

Bemerkung 4.59 (QR-Zerlegung mit Householder-Transformationen)**.** *Die Householder-Matrizen $\tilde{S}^{(i)} = I - 2\tilde{v}^{(i)}(\tilde{v}^{(i)})^T$ werden nicht explizit aufgestellt. Auch kann das Produkt*

$$Q := (S^{(1)})^T \cdots (S^{(n-1)})^T,$$

aus Effizienzgründen nicht explizit berechnet werden. Die Matrix Q steht nur implizit zur Verfügung durch Speichern der Vektoren $\tilde{v}^{(i)}$. Mit implizit *meint man, dass etwa zur Berechnung des Produktes $\tilde{b} := Q^T b$ (ist notwendig zum Lösen der Gleichungssysteme) die Householder-Transformationen erneut Schritt für Schritt angewendet werden müssen:*

$$\tilde{b} = Q^T b = S^{(n-1)} \cdots S^{(1)} b.$$

Jedes Produkt wird mittels der Vorschrift (4.7) berechnet, ohne dass die Matrizen $S^{(i)}$ explizit aufgestellt werden:

$$b^{(i+1)} = b^{(i)} - 2(v^{(i)}, b^{(i)})v^{(i)}.$$

Neben der oberen Dreiecksmatrix R müssen die Vektoren $\tilde{v}^{(i)} \in \mathbb{R}^{n-i}$ gespeichert werden. Im Gegensatz zur LR-Zerlegung kann dies nicht alleine im Speicherplatz der Matrix A geschehen, da sowohl R als auch die $\tilde{v}^{(i)}$ die Diagonale besetzen. Bei der praktischen Realisierung muss ein weiterer Diagonalvektor vorgehalten werden.

Abschließend berechnen wir die QR-Zerlegung zu Beispiel 4.54 mit Hilfe der Householder-Transformationen:

Beispiel 4.60 (QR-Zerlegung mit Householder-Transformationen)**.** *Wir betrachten wieder die Matrix:*

$$A := \begin{pmatrix} 1 & 1 & 1 \\ 0.01 & 0 & 0.01 \\ 0 & 0.01 & 0.01 \end{pmatrix},$$

und führen alle Rechnungen mit dreistelliger Genauigkeit durch. Wir wählen mit $a_1 = (1, 0.01, 0)^T$ und $\|a_1\| \approx 1$ den ersten Vektor zur Spiegelung als:

$$v^{(1)} = \frac{a_1 + \|a_1\|e_1}{\|a_1 + \|a_1\|e_1\|} \approx \begin{pmatrix} 1 \\ 0.005 \\ 0 \end{pmatrix}.$$

Hiermit ergibt sich (wird eigentlich nicht benötigt!)

$$S^{(1)} = I - 2v^{(1)}(v^{(1)})^T \approx \begin{pmatrix} -1 & -0.01 & 0 \\ -0.01 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

und die neuen Spaltenvektoren ergeben sich zu:

$$\begin{aligned} a_1^{(1)} &= -\|a_1\|e_1 \approx -e_1, \\ a_2^{(1)} &= a_2 - 2(v^{(1)}, a_2)v^{(1)} \approx a_2 - 2v^{(1)}, \\ a_3^{(1)} &= a_3 - 2(v^{(1)}, a_3)v^{(1)} \approx a_3 - 2v^{(1)}, \end{aligned} \quad \Rightarrow \quad A^{(1)} = \begin{pmatrix} -1 & -1 & -1 \\ 0 & -0.01 & 0 \\ 0 & 0.01 & 0.01 \end{pmatrix}.$$

Wir fahren mit der Teilmatrix $A^{(1)}_{kl>1}$ fort und wählen mit $\tilde{a}_2^{(1)} = (-0.01, 0.01)^T$ (man beachte das Vorzeichen $\operatorname{sign}(a_{22}^{(1)}) =$ "-")

$$\tilde{v}^{(2)} = \frac{\tilde{a}_2^{(1)} - \|\tilde{a}_2^{(1)}\|\tilde{e}_2}{\|\tilde{a}_2^{(1)} - \|\tilde{a}_2^{(1)}\|\tilde{e}_2\|} \approx \begin{pmatrix} -0.924 \\ 0.383 \end{pmatrix}.$$

Damit ergibt sich als Householder-Transformation

$$I - 2\tilde{v}^{(2)}(\tilde{v}^{(2)})^T \approx \tilde{S}^{(2)} = \begin{pmatrix} -0.708 & 0.708 \\ 0.708 & 0.706 \end{pmatrix}, \quad S^{(2)} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -0.708 & 0.708 \\ 0 & 0.708 & 0.706 \end{pmatrix}.$$

Wir erhalten:

$$\begin{aligned} \tilde{a}_2^{(2)} &= \|\tilde{a}_1^{(1)}\|\tilde{e}_2, \\ \tilde{a}_3^{(2)} &= \tilde{a}_3^{(1)} - 2(\tilde{a}_3^{(1)}, \tilde{v}^{(2)})\tilde{v}^{(2)} \approx \tilde{a}_3^{(1)} - 2 \cdot 0.00383\tilde{v}^{(2)}, \end{aligned} \quad \Rightarrow \quad A^{(2)} = \begin{pmatrix} -1 & -1 & -1 \\ 0 & 0.0141 & 0.00708 \\ 0 & 0 & 0.00707 \end{pmatrix}.$$

Zur Probe berechnen wir $Q^T = S^{(2)}S^{(1)}$ (mit dreistelliger Genauigkeit):

$$Q^T \approx \begin{pmatrix} -1 & -0.01 & 0 \\ 0.00708 & -0.708 & 0.708 \\ -0.00708 & 0.708 & 0.707 \end{pmatrix}.$$

Bei dreistelliger Genauigkeit gilt für das Produkt $Q^TQ \approx I$:

$$Q^TQ \approx \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & -7.58 \cdot 10^{-4} \\ 0 & -0.758 \cdot 10^{-4} & 1 \end{pmatrix}.$$

Dies entspricht einem relativen Fehler $\|Q^TQ - I\|_2 \leq 10^{-3}$ im Rahmen der Rechengenauigkeit. Weiter gilt für die Zerlegung $A \approx QR$:

$$A \approx QR = \begin{pmatrix} 1 & 1 & 1 \\ 0.01 & 1.72 \cdot 10^{-5} & 0.01 \\ 0 & 0.00998 & 0.01 \end{pmatrix}.$$

Auch hier gilt $\|A - QR\|_2 / \|A\|_2 \leq 10^{-4}$.

Man vergleiche abschließend das Resultat mit dem Ergebnis von Beispiel 4.54. Dort wurden relative Fehler $\|Q^T Q - I\|_2 \approx 0.7$ *sowie* $\|QR - A\|_2 / \|A\|_2 \approx 0.004$ *erreicht.*

4.4.3 Givens-Rotationen

Das Gram-Schmidt Verfahren basiert auf Projektionen, die Householder-Transformationen sind Spiegelungen. Schließlich stellen wir kurz eine Methode vor, die auf Rotationen beruht. Wir definieren:

Definition 4.61 (Givens-Rotation)**.** *Unter einer* Givens-Rotation *im* $\mathbb{R}^n$ *versteht man die Drehung in der durch zwei Einheitsvektoren* e_i *und* e_j *aufgespannten Ebene. Die Transformationsmatrix ist gegeben durch:*

$$G(i,j,\theta) = \begin{pmatrix} 1 & & & & & & & & \\ & \ddots & & & & & & & \\ & & 1 & & & & & & \\ & & & c & & -s & & & \\ & & & & \ddots & & & & \\ & & & s & & c & & & \\ & & & & & & 1 & & \\ & & & & & & & \ddots & \\ & & & & & & & & 1 \end{pmatrix}, \quad c := \cos(\theta), \quad s := \sin(\theta).$$

Es gilt:

Satz 4.62 (Givens-Rotation)**.** *Die Givens-Rotation* $G(i,j,\theta)$ *ist eine orthogonale Matrix mit* $\det(G) = 1$*. Es ist* $G(i,j,\theta)^{-1} = G(i,j,-\theta)$.

Wie die Householder-Transformationen sind die Givens-Rotationen orthogonale Matrizen. Die Multiplikation von links an eine Matrix, also GA oder einen Vektor, also Gx ändert

nur die i-te und j-te Zeile von A, bzw. von x:

$$G(i,j)A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{i-1,1} & \cdots & a_{i-1,n} \\ ca_{i1} - sa_{j1} & \cdots & ca_{in} - sa_{jn} \\ a_{i+1,1} & \cdots & a_{i+1,n} \\ \vdots & \ddots & \vdots \\ a_{j-1,1} & \cdots & a_{j-1,n} \\ sa_{i1} + ca_{j1} & \cdots & sa_{in} + ca_{jn} \\ a_{j+1,1} & \cdots & a_{j+1,n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{pmatrix}.$$

Die QR-Zerlegung auf der Basis von Givens-Rotationen transformiert die Matrix A wieder schrittweise in eine obere rechte Dreiecksmatrix R. Durch Anwenden einer Givens-Rotation kann jedoch nur ein einzelnes Unterdiagonalelement eliminiert werden und nicht eine ganze Spalte:

$$\begin{pmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{pmatrix} \to \begin{pmatrix} * & * & * & * \\ 0 & * & * & * \\ * & * & * & * \\ * & * & * & * \end{pmatrix} \to \begin{pmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & * & * & * \\ * & * & * & * \end{pmatrix} \to \begin{pmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & * & * & * \\ 0 & * & * & * \end{pmatrix}$$
$$\to \begin{pmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \\ 0 & * & * & * \end{pmatrix} \to \cdots \to \begin{pmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \\ 0 & 0 & 0 & * \end{pmatrix}.$$

Wir betrachten einen Schritt des Verfahrens. Dazu sei die Matrix A gegeben in der Form:

$$A = \begin{pmatrix} a_{11} & \cdots & \cdots & \cdots & \cdots & a_{1n} \\ 0 & \ddots & & \ddots & & \vdots \\ \vdots & \ddots & a_{ii} & & & \vdots \\ \vdots & & 0 & a_{i+1,i+1} & & \vdots \\ \vdots & & \vdots & \vdots & & \vdots \\ \vdots & & 0 & \vdots & & \vdots \\ \vdots & & a_{ji} & \vdots & \ddots & \vdots \\ \vdots & & \vdots & \vdots & & \vdots \\ 0 & \cdots & a_{ni} & a_{n,i+1} & \cdots & a_{nn}. \end{pmatrix}$$

Wir suchen die Givens-Rotation $G(i,j,\theta)$ zur Elimination von a_{ji}. Für GA gilt:

$$(G(i,j,\theta)A)_{ji} = sa_{ii} + ca_{ji}, \quad c := \cos(\theta), \quad s := \sin(\theta).$$

Anstelle den Winkel θ zu finden bestimmen wir gleich die Werte c und s mit dem Ansatz

$$sa_{ii} + ca_{ji} = 0, \quad c^2 + s^2 = 1 \quad \Rightarrow \quad c := \frac{a_{ii}}{\sqrt{a_{ii}^2 + a_{ji}^2}}, \quad s := -\frac{a_{ji}}{\sqrt{a_{ii}^2 + a_{ji}^2}}.$$

Anwendung von $G(i, j, \theta)$ auf A ergibt:

$$(G(i,j,\theta)A)_{ji} = \frac{-a_{ji}a_{ii} + a_{ii}a_{ji}}{\sqrt{a_{ii}^2 + a_{ji}^2}} = 0, \quad (G(i,j,\theta)A)_{ii} = \frac{a_{ii}^2 + a_{ji}^2}{\sqrt{a_{ii}^2 + a_{ji}^2}} = \sqrt{a_{ii}^2 + a_{ji}^2}.$$

Das Element a_{ji} wird eliminiert. Zur Elimination der i-ten Spalte (unterhalb der Diagonale) sind $(n - i - 1)$ Givens-Rotationen notwendig. Hieraus lässt sich leicht abschätzen, dass der Aufwand zum Erstellen der QR-Zerlegung nach Givens großer ist als der Aufwand bei Verwenden der Householder-Transformationen. Genaue Analyse und effiziente Durchführung führt zu $\frac{4n^3}{3} + O(n^2)$ arithmetische Operationen, also dem doppelten Aufwand verglichen mit der Householder-Methode.

Die QR-Zerlegung nach Givens gewinnt aber an Bedeutung, wenn die Matrix A bereits dünn besetzt ist. Nur Unterdiagonalelemente $a_{ji} \neq 0$ müssen gezielt eliminiert werden. Bei sogenannten *Hessenberg-Matrizen* (das sind rechte obere Dreiecksmatrizen, die zusätzlich noch eine linke Nebendiagonale besitzen) kann die QR-Zerlegung mit Givens-Rotationen in nur $O(n^2)$ Operationen durchgeführt werden. Die QR-Zerlegung von Hessenberg-Matrizen spielt die entscheidende Rolle bei dem wichtigsten Verfahren zur Berechnung von Eigenwerten einer Matrix, siehe Abschnitt 4.6.

Beispiel 4.63 (QR-Zerlegung nach Givens)**.** *Wie in Beispielen 4.54 und 4.60 sei wieder die folgende Matrix gegeben:*

$$A := \begin{pmatrix} 1 & 1 & 1 \\ 0.01 & 0 & 0.01 \\ 0 & 0.01 & 0.01 \end{pmatrix}$$

Wir führen alle Rechnungen mit dreistelliger Genauigkeit durch. Zur Elimination von $a_{21} = 0.01$ *ist:*

$$c^{(1)} = \frac{1}{\sqrt{1 + 0.01^2}} \approx 1, \quad s^{(1)} = -\frac{0.01}{\sqrt{1 + 0.01^2}} \approx -0.01,$$

d.h.

$$G^{(1)} = \begin{pmatrix} 1 & 0.01 & 0 \\ -0.01 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}, \quad A^{(1)} = G^{(1)}A \approx \begin{pmatrix} 1 & 1 & 1 \\ 0 & -0.01 & 0 \\ 0 & 0.01 & 0.01 \end{pmatrix}$$

Im zweiten Schritt wählen wir zur Elimination von $a_{32}^{(1)} = 0.01$

$$c^{(2)} = \frac{-0.01}{\sqrt{0.01^2 + 0.01^2}} \approx -0.707, \quad s^{(2)} = -\frac{0.01}{\sqrt{0.01^2 + 0.01^2}} \approx -0.707,$$

also

$$G^{(2)} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -0.707 & 0.707 \\ 0 & -0.707 & -0.707 \end{pmatrix}, \quad A^{(2)} = G^{(2)} A^{(1)} \approx \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0.0141 & 0.00707 \\ 0 & 0 & -0.00707 \end{pmatrix} = \tilde{R}.$$

Zur Probe berechnen wir zunächst $\tilde{Q} := (G^{(1)})^T (G^{(2)})^T$*:*

$$\tilde{Q} = \begin{pmatrix} 1 & 0.00707 & 0.00707 \\ 0.01 & -0.707 & -0.707 \\ 0 & 0.707 & -0.707 \end{pmatrix}.$$

Die Matrizen $\tilde{Q}$ *sowie* $\tilde{R}$ *sind nahezu identisch zu denen der Householder-Transformation in Beispiel 4.60. Daher ist auch die Genauigkeit der Approximation entsprechend gut:*

$$\tilde{Q}^T \tilde{Q} = \begin{pmatrix} 1.0001 & 0 & 0 \\ 0 & 0.99975 & -5 \cdot 10^{-5} \\ 0 & -5 \cdot 10^{-5} & 0.99975 \end{pmatrix}, \quad \tilde{Q}\tilde{R} \approx \begin{pmatrix} 1 & 1 & 1 \\ 0.01 & 3 \cdot 10^{-5} & 0.01 \\ 0 & 0.00997 & 0.009997 \end{pmatrix},$$

mit relativen Fehlern $\|\tilde{Q}\tilde{R} - A\|_2 / \|A\|_2 \approx 0.00005$ *sowie* $\|\tilde{Q}^T\tilde{Q} - I\|_2 \approx 0.0003$.

4.5 Überbestimmte Gleichungssysteme, Gauß'sche Ausgleichrechnung

In vielen Anwendungen treten lineare Gleichungssysteme auf, die eine unterschiedliche Anzahl von Gleichungen und Unbekannten besitzen:

$$Ax = b, \quad A \in \mathbb{R}^{n \times m}, \quad x \in \mathbb{R}^m, \quad b \in \mathbb{R}^n, \quad n \neq m.$$

Im Fall $n > m$ sprechen wir von *überbestimmten* Gleichungssystemen, im Fall $n < m$ von *unterbestimmten* Gleichungssystemen. Die Untersuchung der eindeutigen Lösbarkeit von allgemeinen Gleichungssystemen mit rechteckiger Matrix A kann nicht mehr an deren Regularität ausgemacht werden. Stattdessen wissen wir, dass ein Gleichungssystem genau dann lösbar ist, falls der Rang der Matrix A gleich dem Rang der erweiterten Matrix $(A|b)$ ist. Gilt zusätzlich $\operatorname{rang}(A) = m$, so ist die Lösung eindeutig. Ein unterbestimmtes lineares Gleichungssystem kann daher nie eindeutig lösbar sein. Wir betrachten in diesem Abschnitt ausschließlich überbestimmte Gleichungssysteme, d.h. den Fall $n > m$. Ein solches Gleichungssystem ist im allgemeinen Fall nicht lösbar:

Beispiel 4.64 (Überbestimmtes Gleichungssystem)**.** *Wir suchen das quadratische Interpolationspolynom*

$$p(x) = a_0 + a_1 x + a_2 x^2,$$

gegeben durch die Vorschrift:

$$p(-1/4) = 0, \quad p(1/2) = 1, \quad p(2) = 0, \quad p(5/2) = 1.$$

Dies ergibt die vier Gleichungen:

$$\begin{array}{rcrcrl} a_0 & - & \frac{1}{4}a_1 & + & \frac{1}{16}a_2 & = 0 \\ a_0 & + & \frac{1}{2}a_1 & + & \frac{1}{4}a_2 & = 1 \\ a_0 & + & 2a_1 & + & 4a_2 & = 0 \\ a_0 & + & \frac{5}{2}a_1 & + & \frac{25}{4}a_2 & = 1 \end{array}$$

Wir versuchen, das lineare Gleichungssystem mit Gauß-Elimination zu lösen:

$$\left(\begin{array}{ccc|c} 1 & -\frac{1}{4} & \frac{1}{16} & 0 \\ 1 & \frac{1}{2} & \frac{1}{4} & 1 \\ 1 & 2 & 4 & 0 \\ 1 & \frac{5}{2} & \frac{25}{4} & 1 \end{array}\right) \rightarrow \left(\begin{array}{ccc|c} 1 & -\frac{1}{4} & \frac{1}{16} & 0 \\ 0 & 3 & 3 & 4 \\ 0 & 9 & \frac{63}{16} & 0 \\ 0 & 11 & \frac{99}{4} & 4 \end{array}\right) \rightarrow \left(\begin{array}{ccc|c} 1 & -\frac{1}{4} & \frac{1}{16} & 0 \\ 0 & 3 & 3 & 4 \\ 0 & 0 & -\frac{81}{16} & -12 \\ 0 & 0 & \frac{319}{16} & 4 \end{array}\right).$$

D.h., es müsste gelten $a_2 = 64/27 \approx 2.37$ *sowie* $a_2 = 64/319 \approx 0.2$.

In Anbetracht von Satz 3.6 ist dieses Ergebnis für die Lagrange-Interpolation zu erwarten. Bei allgemeinen überbestimmten Gleichungssystemen muss daher die Zielstellung geändert werden: gesucht wird nicht *die* Lösung des Gleichungssystems, sondern ein Vektor $x \in \mathbb{R}^m$, welcher in gewissem Sinne die beste Approximation ist. Entsprechend der Bestapproximation von Funktionen aus Abschnitt 3.6.1 definieren wir:

Definition 4.65 (Methode der kleinsten Fehlerquadrate, Least-Squares)**.** *Es sei* $Ax = b$ *mit* $A \in \mathbb{R}^{n\times m}$ *und* $b \in \mathbb{R}^n$*. Dann ist die* Least-Squares Lösung $x \in \mathbb{R}^m$ *als die Näherung bestimmt, deren Defekt die kleinste euklidische Norm annimmt:*

$$\|b - Ax\|_2 = \min_{y\in\mathbb{R}^m} \|b - Ay\|_2 \tag{4.9}$$

Der wesentliche Unterschied zwischen dieser Aufgabenstellung und Satz 3.79 zur Gauß-Approximation ist die Wahl der Norm: hier betrachten wir die euklidische Vektor-Norm, bei der Gauß-Approximation von Funktionen die L^2-Norm. Beiden Normen ist gemein, dass sie durch ein Skalarprodukt gegeben sind. Es gilt:

Satz 4.66 (Kleinste Fehlerquadrate)**.** *Angenommen, für die Matrix* $A \in \mathbb{R}^{n\times m}$ *mit* $n > m$ *gilt* $m = \operatorname{rang}(A)$*. Dann ist die Matrix* $A^TA \in \mathbb{R}^{m\times m}$ *positiv definit und die Least-Squares-Lösung* $x \in \mathbb{R}^m$ *ist eindeutig bestimmt als Lösung des* Normalgleichungssystems*:*

$$A^TAx = A^Tb.$$

Beweis: *(i)* Es gilt $\operatorname{rang}(A) = m$. D.h. $Ax = 0$ gilt nur dann, wenn $x = 0$. Hieraus folgt die positive Definitheit der Matrix A^TA:

$$(A^TAx, x)_2 = (Ax, Ax)_2 = \|Ax\|_2^2 > 0 \quad \forall x \neq 0,$$

und das Gleichungssystem $A^TAx = A^Tb$ ist für jede rechte Seite $b \in \mathbb{R}^n$ eindeutig lösbar.

(ii) Es sei $x \in \mathbb{R}^m$ die Lösung des Normalgleichungssystems. Dann gilt für beliebiges $y \in \mathbb{R}^m$:

$$\begin{aligned}\|b - A(x+y)\|_2^2 &= \|b - Ax\|_2^2 + \|Ay\|_2^2 - 2(b - Ax, Ay)_2 \\ &= \|b - Ax\|_2^2 + \|Ay\|_2^2 - 2(\underbrace{A^T b - A^T Ax}_{=0}, y)_2 \\ &\geq \|b - Ax\|_2^2.\end{aligned}$$

(iii) Nun sei x das Minimum von (4.9). D.h., es gilt für beliebigen Vektor y:

$$\|b - Ax\|_2^2 \leq \|b - A(x+y)\|_2^2 = \|b - Ax\|_2^2 + \|Ay\|_2^2 - 2(b - Ax, Ay)_2 \quad \forall y \in \mathbb{R}^m.$$

Hieraus folgt:

$$-\|A\|_2^2\|y\|_2^2 \leq -\|Ay\|_2^2 \leq 2(A^T Ax - A^T b, y)_2 \leq \|Ay\|_2^2 \leq \|A\|_2^2\|y\|_2^2.$$

Für $y = se_i$, wobei $s \in \mathbb{R}$ und e_i der i-te Einheitsvektor ist gilt:

$$-s\|A\|_2^2 \leq 2[A^T Ax - A^T b]_i \leq s\|A\|_2^2 \quad \forall s \in \mathbb{R},\ i = 1, \ldots, n.$$

Bei $s \to 0$ folgt $[A^T Ax]_i = [A^T b]_i$. □

Die beste Approximation eines überbestimmten Gleichungssystems kann durch Lösen des Normalgleichungssystems gefunden werden. Der naive Ansatz, die Matrix $A^T A$ zu bestimmen und dann das Normalgleichungsystem etwa mit dem Cholesky-Verfahren zu lösen ist numerisch nicht ratsam. Zunächst ist der Aufwand zur Berechnung von $A^T A$ sehr groß und die Berechnung der Matrix-Matrix Multiplikation ist schlecht konditioniert. Weiter gilt die Abschätzung:

$$\operatorname{cond}(A^T A) \approx \operatorname{cond}(A)^2.$$

Wir lösen mit dieser Methode das überbestimmte Gleichungssystem aus Beispiel 4.64:

Beispiel 4.67 (Lösen der Normalgleichung)**.** *Die exakte Lösung des Normalgleichungssystems $A^T Ax = A^T b$ ist gegeben durch:*

$$x \approx \begin{pmatrix} 0.3425 \\ 0.3840 \\ -0.1131 \end{pmatrix}, \quad p(x) = 0.3425 + 0.3740x - 0.1131x^2.$$

Es gilt:

$$\|b - Ax\|_2 \approx 0.947.$$

Wir stellen das Normalgleichungssystem mit dreistelliger Rechnung auf:

$$A^T A = \begin{pmatrix} 4 & 4.75 & 10.6 \\ 4.75 & 10.6 & 23.7 \\ 10.6 & 23.7 & 55.1 \end{pmatrix}, \quad A^T b = \begin{pmatrix} 2 \\ 3 \\ 6.5 \end{pmatrix}.$$

Die exakte Lösung ist bestimmt durch die Polynomkoeffizienten $x = (a_0, a_1, a_2)^T$ sowie das Polynom $p(x) = a_0 + a_1 x + a_2 x^2$:

Wir bestimmen die Cholesky-Zerlegung mit dem direkten Verfahren:

$$\begin{aligned} l_{11} &= \sqrt{4} = 2 \\ l_{21} &= 4.75/2 \approx 2.38 \\ l_{31} &= 10.6/2 = 5.3 \\ l_{22} &= \sqrt{10.6 - 2.38^2} \approx 2.22 \\ l_{32} &= (23.7 - 2.38 \cdot 5.3)/2.22 \approx 5 \\ l_{33} &= \sqrt{55.1 - 5.3^2 - 5^2} \approx 1.42. \end{aligned} \quad , \quad L := \begin{pmatrix} 2 & 0 & 0 \\ 2.38 & 2.22 & 0 \\ 5.3 & 5 & 1.42 \end{pmatrix}.$$

Wir lösen:

$$L \underbrace{L^T x}_{=y} = A^T b \quad \Rightarrow \quad y \approx \begin{pmatrix} 1 \\ 0.279 \\ -0.137 \end{pmatrix} \quad \Rightarrow \quad \tilde{x} \approx \begin{pmatrix} 0.424 \\ 0.343 \\ -0.0965 \end{pmatrix},$$

mit dem Polynom

$$p(x) = 0.424 + 0.343x - 0.0965x^2,$$

und dem Defekt $\|b - A\tilde{x}\| \approx 0.96$ sowie dem relativen Fehler zur exakten Lösung:

$$\frac{\|\tilde{x} - x\|}{\|x\|} \approx 0.18.$$

Wir entwickeln ein alternatives Verfahren, welches das Aufstellen des Normalgleichungssystems $A^T Ax = A^T b$ umgeht und nutzen hierfür eine Erweiterung der bereits vorgestellten QR-Zerlegung auf allgemeine rechteckige Matrizen:

Satz 4.68 (QR-Zerlegung rechteckiger Matrizen)**.** *Es sei $A \in \mathbb{R}^{n \times m}$ mit $n > m$ und* $\operatorname{rang}(A) = m$*. Dann existiert eine orthogonale Matrix $Q \in \mathbb{R}^{n \times n}$, sowie eine "rechteckige obere Dreiecksmatrix" $\tilde{R} \in \mathbb{R}^{n \times m}$ so dass gilt $A = Q\tilde{R}$ mit:*

$$\tilde{R} = \left(\begin{array}{c} R \\ \hline 0 \end{array}\right) \begin{matrix} \}m \\ \\ \}n-m \end{matrix} \quad , \quad R = \begin{pmatrix} * & * & \cdots & * \\ 0 & * & \ddots & * \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & * \end{pmatrix} \in \mathbb{R}^{m \times m}.$$

Beweis: Der Beweis folgt durch Anwenden von m Schritten der QR-Zerlegung mit Householder-Transformationen auf die Matrix A. Die Spaltenvektoren $A = (a_1^{(0)}, \ldots, a_m^{(0)})$ sind linear

unabhängig. Diese Eigenschaft bleibt durch Anwenden von orthogonalen Householder-Transformationen $S^{(1)}, \ldots, S^{(m)}$ erhalten, d.h.:

$$\dim\big(\operatorname{span}\{a_1^{(i)}, \ldots, a_m^{(i)}\}\big) = m, \quad i = 1, \ldots, m, \quad a_j^{(i)} = S^{(i)} a_j^{(i-1)}.$$

Die ersten m Schritte der Householder-Transformation sind durchführbar und es gilt $\operatorname{rang}(A) = \operatorname{rang}(Q^T A)$ mit

$$Q^T = S^{(m)} \cdots S^{(1)}.$$

Die Matrix A wird dabei schrittweise auf "Dreiecksgestalt" gebracht:

$$\begin{pmatrix} * & * & \cdots & * \\ * & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ * & \cdots & \cdots & * \end{pmatrix} \rightarrow \begin{pmatrix} * & * & \cdots & * \\ 0 & * & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ 0 & * & \cdots & * \end{pmatrix} \rightarrow \left(\begin{array}{cccc} * & * & \cdots & * \\ 0 & * & \ddots & \vdots \\ \vdots & \ddots & \ddots & \\ \vdots & \ddots & 0 & * \\ \hline \vdots & & \ddots & 0 \\ \vdots & & & \vdots \\ 0 & \cdots & \cdots & 0 \end{array}\right) =: \tilde{R} \in \mathbb{R}^{n \times m}.$$

Im Gegensatz zur Householder-Transformation bei quadratischen Matrizen müssen m anstelle von $m-1$ Schritte durchgeführt werden, um den Block unterhalb der m-ten Zeile zu eliminieren.

Die resultierende Matrix $\tilde{R}$ hat Rang m für ihre Einträge $\tilde{r}_{ij}$ mit $i > m$ gilt $\tilde{r}_{ij} = 0$. Daher ist der obere Matrixblock $R \in \mathbb{R}^{m \times m}$ mit $r_{ij} = \tilde{r}_{ij}$ für $i, j \leq m$ regulär mit $\operatorname{rang}(R) = \operatorname{rang}(\tilde{R}) = m$. □

Mit dieser verallgemeinerten QR-Zerlegung gilt nun:

$$A^T A x = A^T b \quad \Leftrightarrow \quad (Q\tilde{R})^T Q\tilde{R} x = (Q\tilde{R})^T b \quad \Leftrightarrow \quad \tilde{R}^T \tilde{R} x = \tilde{R}^T Q^T b.$$

Da alle Einträge von $\tilde{R}$ im unteren Block Null sind gilt $\tilde{R}^T \tilde{R} = R^T R$ und weiter mit

$$\tilde{b}_i := (Q^T b)_{i \leq m},$$

ist das Normalgleichungssystem äquivalent zum einfachen Dreieckssystem

$$A^T A x = A^T b \quad \Leftrightarrow \quad RX = \tilde{b},$$

mit einer $n \times n$-Matrix R.

Beispiel 4.69 ("Lösen" eines überbestimmten Gleichungssystems mit erweiterter QR–Zerlegung)**.** *Für das überbestimmte lineare Gleichungssystem aus Beispiel 4.64 gilt:*

$$\begin{pmatrix} 1 & -\frac{1}{4} & \frac{1}{16} \\ 1 & \frac{1}{2} & \frac{1}{4} \\ 1 & 2 & 4 \\ 1 & \frac{5}{2} & \frac{25}{4} \end{pmatrix}, \quad b = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}.$$

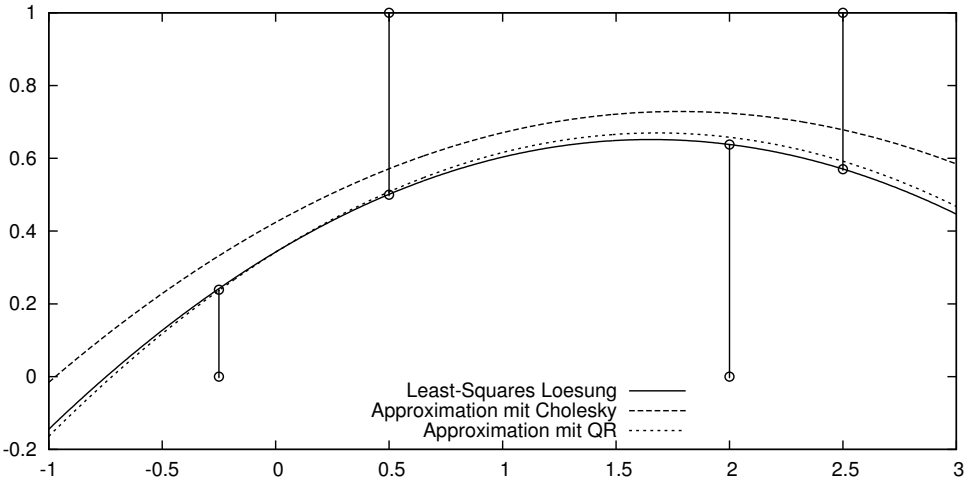

Abbildung 4.3: Lösen eines überbestimmten Gleichungssystems zum Finden der besten Approximation an Messdaten. Siehe Beispiele 4.64, 4.67 sowie 4.69.

Wir wählen im ersten Schritt der Householder-Transformation (dreistellige Rechnung)

$$v^{(1)} = \frac{a_1 + \|a_1\| e_1}{\| a_1 + \|a_1\| e_1 \|} \approx \begin{pmatrix} 0.866 \\ 0.289 \\ 0.289 \\ 0.289 \end{pmatrix}.$$

Dann gilt mit $a_i^{(1)} = a_i - 2(a_i, v^{(1)})v^{(1)}$*:*

$$\tilde{A}^{(1)} \approx \begin{pmatrix} -2 & -2.38 & -5.29 \\ 0 & -0.210 & -1.54 \\ 0 & 1.29 & 2.21 \\ 0 & 1.79 & 4.46 \end{pmatrix} =: (a_1^{(1)}, a_2^{(1)}, a_3^{(1)}).$$

Mit dem reduzierten Vektor $\tilde{a}_2^{(1)} = (-0.21, 1.29, 1.79)^T$ *gilt weiter*

$$\tilde{v}^{(2)} = \frac{\tilde{a}_2^{(1)} - \|\tilde{a}_2^{(1)}\| \tilde{e}_2}{\| \tilde{a}_2^{(1)} + \|\tilde{a}_2^{(1)}\| \tilde{e}_2 \|} \approx \begin{pmatrix} -0.740 \\ 0.393 \\ 0.546 \end{pmatrix}.$$

Und hiermit:

$$\tilde{A}^{(2)} \approx \begin{pmatrix} -2 & -2.38 & -5.29 \\ 0 & 2.22 & 5.04 \\ 0 & 0 & -1.28 \\ 0 & 0 & -0.392 \end{pmatrix} =: (a_1^{(2)}, a_2^{(2)}, a_3^{(2)}).$$

Wir müssen einen dritten Schritt durchführen und mit $\tilde{a}_3^{(2)} = (-1.28, -0.392)$ *ergibt sich nach gleichem Prinzip:*

$$\tilde{v}^{(3)} = \begin{pmatrix} -0.989 \\ -0.148 \end{pmatrix}.$$

Schließlich erhalten wir:

$$\tilde{R} = \tilde{A}^{(3)} \approx \begin{pmatrix} -2 & -2.38 & -5.29 \\ 0 & 2.22 & 5.04 \\ 0 & 0 & 1.34 \\ 0 & 0 & 0 \end{pmatrix}.$$

Zum Lösen des Ausgleichsystems:

$$A^T Ax = A^T b \quad \Leftrightarrow \quad Rx = \tilde{b},$$

bestimmen wir zunächst die rechte Seite nach der Vorschrift $b^{(i)} = b^{(i-1)} - 2(b^{(i-1)}, v^{(i)})v^{(i)}$*:*

$$\tilde{b} = \begin{pmatrix} -1 \\ 0.28 \\ -0.155 \end{pmatrix}.$$

Abschließend lösen wir durch Rückwärtseinsetzen $Rx = \tilde{b}^{(3)}$*:*

$$\tilde{x} \approx \begin{pmatrix} 0.342 \\ 0.390 \\ -0.116 \end{pmatrix}$$

und erhalten das Interpolationspolynom:

$$p(x) = 0.342 + 0.39x - 0.116x^2,$$

mit Defekt

$$\|b - A\tilde{x}\| \approx 0.947,$$

und relativem Fehler zur exakten Least-Squares-Lösung:

$$\frac{\|\tilde{x} - x\|}{\|x\|} \approx 0.01,$$

d.h., ein Fehler von etwa einem 1% *anstelle von fast 20% beim direkten Lösen des Normalsystems. In Abbildung 4.3 zeigen wir die exakte Lösung sowie die beiden Approximierten Lösungen zu diesem Beispiel.*

Die in Abschnitt 3.6.1 betrachtete diskrete Gauss-Approximation ist eine Anwendung der Methode der kleinsten Fehlerquadrate. Die lineare Ausgleichsrechnung ist ein Spezialfall mit Matrizen $A \in \mathbb{R}^{n\times 2}$.

4.6 Berechnung von Eigenwerten

In diesem Abschnitt befassen wir uns mit dem Eigenwertproblem: zu gegebener Matrix $A \in \mathbb{R}^{n\times n}$ sind die Eigenwerte (und gegebenenfalls Eigenvektoren) gesucht. Wir erinnern an Definition 4.9 und formulieren

Satz 4.70 (Eigenwert)**.** *Es sei $A \in \mathbb{R}^{n\times n}$. Dann gilt:*

- *Die Matrix A hat genau n Eigenwerte, ihrer Vielfachheit nach gezählt.*
- *Die Eigenwerte sind Nullstellen des* charakteristischen Polynoms:

$$\det(A - \lambda I) = 0.$$

- *Reelle Matrizen können komplexe Eigenwerte haben, komplexe Eigenwerte treten stets als konjugierte Paare $\lambda, \bar{\lambda}$ auf.*
- *Eigenvektoren zu unterschiedlichen Eigenwerten sind linear unabhängig.*
- *Falls n linear unabhängige Eigenvektoren existieren, so existiert eine reguläre Matrix $S \in \mathbb{R}^{n\times n}$, so dass $S^{-1}AS = D$ eine Diagonalmatrix ist. Die Spaltenvektoren von S sind die Eigenvektoren und die Diagonaleinträge von D die Eigenwerte.*
- *Symmetrische Matrizen $A = A^T$ haben nur reelle Eigenwerte. Es existiert eine Orthonormalbasis von Eigenvektoren und eine Diagonalisierung $Q^TAQ = D$ mit einer orthogonalen Matrix.*
- *Bei Dreiecksmatrizen stehen die Eigenwerte auf der Diagonalen.*

Zu einem Eigenwert λ sind die Eigenvektoren als Lösung des homogenen linearen Gleichungssystems bestimmt:

$$(A - \lambda I)w = 0.$$

Umgekehrt gilt:

Definition 4.71 (Rayleigh-Quotient)**.** *Sei $A \in \mathbb{R}^{n\times n}$ sowie $w \in \mathbb{R}^n$ ein Eigenvektor. Dann ist durch den* Rayleigh-Quotienten *der zugehörige Eigenwert gegeben:*

$$\lambda = \frac{(Aw, w)_2}{\|w\|_2^2}.$$

Mit Hilfe dieser Definition folgt eine einfache Schranke für die Eigenwerte:

$$|\lambda| \leq \sup_{w\neq 0} \frac{(Aw, w)_2}{\|w\|_2^2} \leq \frac{\|A\|_2 \|w\|_2^2}{\|w\|_2^2} = \|A\|_2.$$

Wir haben in Abschnitt 4.1 bereits gesehen, dass diese Schranke für die Beträge der Eigenwerte in jeder Matrixnorm mit verträglicher Vektornorm gilt.

4.6.1 Konditionierung der Eigenwertaufgabe

Bevor wir auf konkrete Verfahren zur Eigenwertberechnung eingehen, analysieren wir die Kondition der Aufgabe, d.h. die Abhängigkeit der Eigenwerte von der Störung der Matrix. Hierfür benötigen wir zunächst einen Hilfsatz:

Hilfsatz 4.72. *Es seien $A, B \in \mathbb{R}^{n\times n}$ beliebige Matrizen. Dann gilt für jeden Eigenwert λ von A, der nicht zugleich Eigenwert von B für jede natürliche Matrizennorm die Abschätzung:*

$$\|(\lambda I - B)^{-1}(A - B)\| \geq 1.$$

Beweis: Es sei $w \neq 0$ ein Eigenvektor zu λ. Dann gilt:

$$(A - B)w = (\lambda I - B)w.$$

Wenn λ kein Eigenwert von B ist, so ist die Matrix $(\lambda I - B)$ regulär, d.h. es folgt:

$$(\lambda I - B)^{-1}(A - B)w = w.$$

Und schließlich durch Normbilden:

$$1 = \frac{\|(\lambda I - B)^{-1}(A - B)w\|}{\|w\|} \leq \sup_x \frac{\|(\lambda I - B)^{-1}(A - B)x\|}{\|x\|} = \|(\lambda I - B)^{-1}(A - B)\|.$$

□

Auf dieser Basis erhalten wir ein einfaches Kriterium zur Eingrenzung der Eigenwerte einer Matrix:

Satz 4.73 (Gerschgorin-Kreise)**.** *Es sei $A \in \mathbb{R}^{n\times n}$. Alle Eigenwerte λ von A liegen in der Vereinigung der* Gerschgorin-Kreise*:*

$$K_i = \{z \in \mathbb{C} : \ |z - a_{ii}| \leq \sum_{k=1,k\neq i}^{n} |a_{ik}|\}, \quad i = 1, \ldots, n.$$

Angenommen zu den Indexmengen $I_m = \{i_1, \ldots, i_m\}$ und $I'_m = \{1, \ldots, n\} \setminus I_m$ seien die Vereinigungen $U_m = \bigcup_{i\in I_m} K_i$ und $U'_m = \bigcup_{i\in I'_m} K_i$ disjunkt. Dann liegen genau m Eigenwerte (ihrer algebraischen Vielfachheit nach gezählt) in U_m und $n - m$ Eigenwerte in U'_m.

Beweis: *(i)* Es sei $D \in \mathbb{R}^{n\times n}$ die Diagonalmatrix $D = \mathrm{diag}(a_{ii})$. Weiter sei λ ein Eigenwert von A mit $\lambda \neq a_{ii}$. (In diesem Fall wäre die Aussage des Satzes trivial erfüllt). Hilfsatz 4.72 besagt bei Wahl der maximalen Zeilensummennorm:

$$1 \leq \|(\lambda I - D)^{-1}(A - D)\|_\infty = \max_i \left\{ \sum_{j=1,j\neq i}^{n} |(\lambda - a_{ii})^{-1} a_{ij}| \right\} = \max_i \left\{ |\lambda - a_{ii}|^{-1} \sum_{j=1,j\neq i}^{n} |a_{ij}| \right\}.$$

D.h. es existiert zu jedem λ ein Index $i \in \{1, \ldots, n\}$ so dass gilt:

$$|\lambda - a_{ii}| \leq \sum_{j=1,j\neq i}^{n} |a_{ij}|.$$

(ii) Es seien durch I_m und I'_m Indexmengen mit oben genannter Eigenschaft gegeben. Wir definieren die Matrix

$$A_s := D + s(A - D),$$

mit $A_0 = D$ und $A_1 = A$. Entsprechend definieren wir die Vereinigungen:

$$U_{m,s} := \bigcup_{i \in I_m} K_i(A_s), \quad U'_{m,s} := \bigcup_{i \in I'_m} K_i(A_s), \quad K_i(A_s) = \{z \in \mathbb{C},\ |z - a_{ii}| \leq s \sum_{j \neq i} |a_{ij}|\}.$$

Aufgrund der stetigen Abhängigkeit der Kreisradien von s gilt $U_{m,s} \cap U'_{m,s} = \emptyset$ für $s \in [0, 1]$. Im Fall $s = 0$ gilt $A_0 = D$ und jeder Eigenwert von A_0 liegt im Mittelpunkt (also $\lambda_i = a_{ii}$) des trivialen Kreises mit Radius Null. Das Ergebnis folgt nun durch die stetige Abhängigkeit der Eigenwerte von s. □

Wir betrachten hierzu ein Beispiel:

Beispiel 4.74 (Gerschgorin-Kreise)**.** *Wir betrachten die Matrix*

$$A = \begin{pmatrix} 2 & 0.1 & -0.5 \\ 0.2 & 3 & 0.5 \\ -0.4 & 0.1 & 5 \end{pmatrix},$$

mit den Eigenwerten $\mathrm{spr}(A) \approx \{1.91, 3.01, 5.08\}$*. Eine erste Schranke liefern verschiedene mit Vektornormen verträgliche Matrixnormen von A:*

$$\|A\|_\infty = 5.5, \quad \|A\|_1 = 6, \quad \|A\|_2 \approx 5.1.$$

Die Gerschgorin-Kreise von A sind:

$$K_1 = \{z \in \mathbb{C},\ |z-2| \leq 0.6\}, \quad K_2 = \{z \in \mathbb{C},\ |z-3| \leq 0.7\}, \quad K_3 = \{z \in \mathbb{C},\ |z-5| \leq 0.5\}.$$

Diese Abschätzung kann verfeinert werden, da A und A^T *die gleichen Eigenwerte besitzen. Die Radien der Gerschgorin-Kreise können auch als Summe der Spaltenbeträge berechnet werden. Zusammen ergibt sich:*

$$K_1 = \{z \in \mathbb{C},\ |z-2| \leq 0.6\}, \quad K_2 = \{z \in \mathbb{C},\ |z-3| \leq 0.2\}, \quad K_3 = \{z \in \mathbb{C},\ |z-5| \leq 0.5\}.$$

Alle drei Kreise sind disjunkt.

Wir können nun den allgemeinen Stabilitätssatz für das Eigenwertproblem beweisen:

Satz 4.75 (Stabilität des Eigenwertproblems)**.** *Es sei* $A \in \mathbb{R}^{n \times n}$ *eine Matrix mit n linear unabhängigen Eigenvektoren* $w_1, \ldots, w_n$*. Durch* $\tilde{A} = A + \delta A$ *sei eine beliebig gestörte Matrix gegeben. Dann existiert zu jedem Eigenwert* $\lambda(\tilde{A})$ *von* $\tilde{A} = A + \delta A$ *ein Eigenwert* $\lambda(A)$ *von A so dass mit der Matrix* $W := (w_1, \ldots, w_n)$ *gilt:*

$$|\lambda(A) - \lambda(\tilde{A})| \leq \mathrm{cond}_2(W) \|\delta A\|.$$

BEWEIS: Es gilt für $i = 1, \ldots, n$ die Beziehung $Aw_i = \lambda_i(A)w_i$, oder in Matrixschreibweise $AW = W \operatorname{diag}(\lambda_i(A))$, also

$$W^{-1}AW = \operatorname{diag}(\lambda_i(A)).$$

Wir betrachten nun einen Eigenwert $\tilde{\lambda} = \lambda(\tilde{A})$. Falls $\tilde{\lambda}$ auch Eigenwert von A ist, so gilt die Behauptung. Also sei $\tilde{\lambda}$ nun kein Eigenwert von A. Dann folgt:

$$\|(\tilde{\lambda}I - A)^{-1}\|_2 = \|W^{-1}[\tilde{\lambda}I - \operatorname{diag}(\tilde{\lambda}_i(A))]^{-1}W\|_2 \leq \operatorname{cond}_2(W)\|[\tilde{\lambda}I - \operatorname{diag}(\lambda_i(A))]^{-1}\|_2.$$

Für die (symmetrische) Diagonalmatrix $\tilde{\lambda}I - \operatorname{diag}(\lambda_i(A))$ gilt

$$\|[\tilde{\lambda}I - \operatorname{diag}(\lambda_i(A))]^{-1}\|_2 = \max_{i=1,\ldots,n} |\tilde{\lambda} - \lambda_i(A)|^{-1}.$$

Mit Hilfsatz 4.72 folgt das gewünschte Ergebnis:

$$1 \leq \|[\tilde{\lambda}I - A)^{-1}\delta A\|_2 \leq \|[\tilde{\lambda}I - A)^{-1}\| \, \|\delta A\|_2 \leq \operatorname{cond}_2(W) \max_{i=1,\ldots,n} |\tilde{\lambda} - \lambda_i(A)|_2^{-1} \, \|\delta A\|_2.$$

□

Die Konditionierung des Eigenwertproblems einer Matrix A hängt von der Konditionszahl der Matrix der Eigenvektoren ab. Für symmetrische (hermitesche) Matrizen existiert eine Orthonormalbasis von Eigenvektoren mit $\operatorname{cond}_2(W) = 1$. Für solche Matrizen ist das Eigenwertproblem gut konditioniert. Für allgemeine Matrizen kann das Eigenwertproblem beliebig schlecht konditioniert sein.

4.6.2 Direkte Methode zur Eigenwertberechnung

Die Eigenwerte einer Matrix A können prinzipiell als Nullstellen des charakteristischen Polynoms $\chi_A(z) = \det(zI - A)$ berechnet werden. Die Berechnung der Nullstellen kann zum Beispiel mit dem Newton-Verfahren geschehen, Startwerte können mit Hilfe der Gerschgorin-Kreise bestimmt werden.

In Kapitel 2 haben die Berechnung von Nullstellen eines Polynoms jedoch als ein sehr schlecht konditioniertes Problem kennengelernt. Wir betrachten hierzu ein Beispiel.

Beispiel 4.76 (Direkte Berechnung von Eigenwerten)**.** *Es sei $A \in \mathbb{R}^{5\times 5}$ eine Matrix mit den Eigenwerten $\lambda_i = 1$, $i = 1, \ldots, 5$. Dann gilt:*

$$\chi_A(z) = \prod_{i=1}^{5}(z - i) = z^5 - 15z^4 + 85x^3 - 225x^2 + 274z - 120.$$

Der Koeffizient -55 vor z^9 sei mit einem relativen Fehler von 0.1% gestört:

$$\tilde{\chi}_A(z) = z^5 - 0.999 \cdot 15z^4 + 85x^3 - 225x^2 + 274z - 120.$$

Dieses gestörte Polynom hat die Nullstellen (d.h. Eigenwerte):

$$\lambda_1 \approx 0.999, \quad \lambda_2 \approx 2.05, \quad \lambda_3 \approx 2.76, \quad \lambda_{4/5} \approx 4.59 \pm 0.430i.$$

Die Eigenwerte können also nur mit einem (ab λ_3) wesentlichen Fehler bestimmt werden. Es gilt etwa:

$$\frac{|4.59 + 0.43i - 5|}{5} \approx 0.1,$$

d.h. der Fehler in den Eigenwerten beträgt 10%*, eine Fehlerverstärkung von* 100*.*

Das Aufstellen des charakteristischen Polynoms erfordert eine Vielzahl schlecht konditionierter Additionen sowie Multiplikationen. Für allgemeine Matrizen verbietet sich dieses direkte Verfahren. Lediglich für spezielle Matrizen wie Tridiagonalsysteme lassen die Eigenwerte bestimmen, ohne das zunächst die Koeffizienten des charakteristischen Polynoms explizit berechnet werden müssen.

4.6.3 Iterative Verfahren zur Eigenwertberechnung

Das vorangegangene Beispiel widerspricht scheinbar zunächst der (auf jeden bei hermiteschen Matrizen) guten Konditionierung der Eigenwertsuche. Wir leiten nun stabile numerische Verfahren her, die die Eigenwerte nicht mehr direkt berechnen, sondern sie iterativ approximieren.

Zur Herleitung eines ersten einfachen Iterationsverfahren machen wir die folgende Beobachtung: angenommen, zu gegebener Matrix $A \in \mathbb{R}^{n\times n}$ existiere eine Basis aus Eigenvektoren $\{w_1, \dots, w_n\}$, d.h. die Matrix sei diagonalisierbar. Weiter gelte $|\lambda_n| > |\lambda_{n-1}| \geq \cdots \geq |\lambda_1| > 0$. Dann sei $x \in \mathbb{R}^n$ in Basisdarstellung:

$$x^{(0)} = \sum_{j=1}^{n} \alpha_j w_j.$$

Wir nehmen an, dass $\alpha_n \neq 0$, dass also die Komponente zum Eigenvektor des betragsmäßig größten Eigenwertes nicht trivial ist. Wir definieren die Iteration

$$x^{(i+1)} = \frac{Ax^{(i)}}{\|Ax^{(i)}\|}.$$

Dann gilt

$$x^{(i+1)} = \frac{A\frac{Ax^{(i-1)}}{\|Ax^{(i-1)}\|}}{\|A\frac{Ax^{(i-1)}}{\|Ax^{(i-1)}\|}\|} = \frac{A^2x^{(i-1)}}{\|A^2x^{(i-1)}\|}\frac{\|Ax^{(i-1)}\|}{\|Ax^{(i-1)}\|} = \cdots = \frac{A^{i+1}x^{(0)}}{\|A^{i+1}x^{(0)}\|}. \tag{4.10}$$

Weiter gilt mit der Basisdarstellung von $x^{(0)}$:

$$A^i x^{(0)} = \sum_{j=1}^{n} \alpha_j \lambda_j^i w_j = \alpha_n \lambda_n^i \left(w_n + \sum_{j=1}^{n-1} \frac{\alpha_j}{\alpha_n} \frac{\lambda_j^i}{\lambda_n^i} \right). \tag{4.11}$$

Es gilt $|\lambda_j/\lambda_n| < 1$ für $j < n$, daher folgt:

$$A^i x^{(0)} = \sum_{j=1}^{n} \alpha_j \lambda_j^i w_j = \alpha_n \lambda_n^i \left(w_n + o(1)\right),$$

Hieraus folgt durch Normierung:

$$\frac{A^i x^{(0)}}{\|A^i x^{(0)}\|} = \left(\frac{\alpha_n \lambda_n^i}{|\alpha_n \lambda_n^i|}\right) \frac{w_n}{\|w_n\|} + o(1) \to \operatorname{span}\{w_n\} \quad (i \to \infty).$$

Die Iteration läuft in den Raum, der durch w_n aufgespannt wird. Für einen Vektor w, der Vielfaches eines Eigenvektors ist $w = sw_n$ gilt:

$$Aw = sAw_n = s\lambda_n w_n = \lambda_n w.$$

Diese vektorwertige Gleichung gilt in jeder Komponente, kann daher nach dem Eigenwert aufgelöst werden:

$$\lambda_n = \frac{[Aw]_k}{w_k}.$$

Wir fassen zusammen:

Satz 4.77 (Potenzmethode nach von Mises)**.** *Es sei $A \in \mathbb{R}^{n\times n}$ eine Matrix n linear unabhängigen Eigenvektoren $\{w_1, \ldots, w_n\}$. Der betragsmäßig größte Eigenwert sei separiert $|\lambda_n| > |\lambda_{n-1}| \geq \cdots \geq |\lambda_1|$. Es sei $x^{(0)} \in \mathbb{R}^n$ ein Startwert mit nichttrivialer Komponente in Bezug auf w_n. Für einen beliebigen Index $k \in \{1, \ldots, n\}$ konvergiert die Iteration*

$$\tilde{x}^{(i)} = Ax^{(i-1)}, \quad x^{(i)} := \frac{\tilde{x}^{(i)}}{\|\tilde{x}^{(i)}\|}, \quad \lambda^{(i)} := \frac{\tilde{x}_k^{(i)}}{x_k^{(i-1)}},$$

gegen den betragsmäßig größten Eigenwert:

$$|\lambda^{(i+1)} - \lambda_n| = O\left(\left|\frac{\lambda_{n-1}}{\lambda_n}\right|^i\right), \quad i \to \infty.$$

Beweis: Wir knüpfen an der Vorbereitung des Beweises (4.10) an. Es gilt:

$$\lambda^{(i)} = \frac{\tilde{x}_k^{(i)}}{x_k^{(i-1)}} = \frac{[Ax^{(i-1)}]_k}{x_k^{(i-1)}} = \frac{[A^i x^{(0)}]_k}{[A^{i-1} x^{(0)}]_k}.$$

Weiter, mit (4.11) gilt:

$$\begin{aligned}
\lambda^{(i)} &= \frac{a_n \lambda_n^i \left([w_n]_k + \sum_{j=1}^{n-1} \frac{\alpha_j}{\alpha_n} \frac{\lambda_j^i}{\lambda_n^i} [w_j]_k\right)}{a_n \lambda_n^{i-1} \left([w_n]_k + \sum_{j=1}^{n-1} \frac{\alpha_j}{\alpha_n} \frac{\lambda_j^{i-1}}{\lambda_n^{i-1}} [w_j]_k\right)} = \lambda_n \frac{[w_n]_k + \sum_{j=1}^{n-1} \frac{\alpha_j}{\alpha_n} \frac{\lambda_j^i}{\lambda_n^i} [w_j]_k}{[w_n]_k + \sum_{j=1}^{n-1} \frac{\alpha_k}{\alpha_n} \frac{\lambda_j^{i-1}}{\lambda_n^{i-1}} [w_j]_k} \\
&= \lambda_n \frac{[w_n]_k + \left(\frac{\alpha_{n-1}}{\alpha_n} + o(1)\right) \left|\frac{\lambda_{n-1}}{\lambda_n}\right|^i [w_n]_k}{[w_n]_k + \left(\frac{\alpha_{n-1}}{\alpha_n} + o(1)\right) \left|\frac{\lambda_{n-1}}{\lambda_n}\right|^{i-1} [w_n]_k} = \lambda_n \left(1 + O\left(\left|\frac{\lambda_{n-1}}{\lambda_n}\right|^{i-1}\right)\right).
\end{aligned}$$

Die letzte Abschätzung nutzt die Reihenentwicklung:

$$\frac{1+x^{n+1}}{1+x^n} = 1 + O(|x|^n).$$

□

Die Potenzmethode ist eine sehr einfache und numerisch stabile Iteration. Sie kann allerdings nur den betragsmäßig größten Eigenwert ermitteln. Der Konvergenzbeweis kann verallgemeinert werden auf Matrizen, deren größter Eigenwert mehrfach vorkommt. Konvergenz gilt jedoch nur dann, wenn aus $|\lambda_n| = |\lambda_i|$ auch $\lambda_n = \lambda_i$ folgt. Es darf also nicht zwei verschiedene Eigenwerte geben, die beide den gleichen, größten Betrag annehmen. Dies schließt zum Beispiel den Fall zweier komplex konjugierter Eigenwerten λ und $\bar{\lambda}$ als beträgsgrößte aus.

Bemerkung 4.78 (Potenzmethode bei symmetrischen Matrizen)**.** *Die Potenzmethode kann im Fall symmetrischer Matrizen durch Verwenden des Rayleigh-Quotienten verbessert werden. Die Iteration*

$$\lambda^{(i)} = \frac{(Ax^{(i-1)}, x^{(i-1)})_2}{(x^{(i-1)}, x^{(i-1)})_2} = (\tilde{x}^{(i)}, x^{(i-1)})_2,$$

liefert die Fehlerabschätzung:

$$\lambda^{(i)} = \lambda_n + O\left(\left|\frac{\lambda_{n-1}}{\lambda_n}\right|^{2i}\right).$$

Beispiel 4.79 (Potenzmethode nach von Mises)**.** *Es sei:*

$$A = \begin{pmatrix} 2 & 1 & 2 \\ -1 & 2 & 1 \\ 1 & 2 & 4 \end{pmatrix},$$

mit den Eigenwerten:

$$\lambda_1 \approx 0.80131, \quad \lambda_2 = 2.2865, \quad \lambda_3 = 4.9122.$$

Wir starten die Potenzmethode mit $x^{(0)} = (1,1,1)^T$, und wählen zur Normierung die

Maximumsnorm. Weiter wählen wir als Index $k = 1$:

$$i = 1: \quad \tilde{x}^{(1)} = Ax^{(0)} = \begin{pmatrix} 5 \\ 2 \\ 7 \end{pmatrix}, \qquad x^{(1)} = \frac{\tilde{x}^{(1)}}{\|\tilde{x}^{(1)}\|_\infty} \approx \begin{pmatrix} 0.714 \\ 0.286 \\ 1 \end{pmatrix}, \qquad \lambda^{(1)} = \frac{\tilde{x}_1^{(1)}}{x_1^{(0)}} \approx 5,$$

$$i = 2: \quad \tilde{x}^{(2)} = Ax^{(1)} = \begin{pmatrix} 3.71 \\ 0.857 \\ 5.29 \end{pmatrix}, \qquad x^{(2)} = \frac{\tilde{x}^{(2)}}{\|\tilde{x}^{(2)}\|_\infty} \approx \begin{pmatrix} 0.702 \\ 0.162 \\ 1 \end{pmatrix}, \qquad \lambda^{(2)} = \frac{\tilde{x}_1^{(2)}}{x_1^{(1)}} \approx 5.19,$$

$$i = 3: \quad \tilde{x}^{(3)} = Ax^{(2)} = \begin{pmatrix} 3.57 \\ 0.124 \\ 1 \end{pmatrix}, \qquad x^{(3)} = \frac{\tilde{x}^{(3)}}{\|\tilde{x}^{(3)}\|_\infty} \approx \begin{pmatrix} 0.710 \\ 0.124 \\ 1 \end{pmatrix}, \qquad \lambda^{(3)} = \frac{\tilde{x}_1^{(3)}}{x_1^{(2)}} \approx 5.077,$$

$$i = 4: \quad \tilde{x}^{(4)} = Ax^{(3)} = \begin{pmatrix} 3.54 \\ 0.538 \\ 4.96 \end{pmatrix}, \qquad x^{(4)} = \frac{\tilde{x}^{(4)}}{\|\tilde{x}^{(4)}\|_\infty} \approx \begin{pmatrix} 0.715 \\ 0.108 \\ 1 \end{pmatrix}, \qquad \lambda^{(4)} = \frac{\tilde{x}_1^{(4)}}{x_1^{(3)}} \approx 4.999,$$

$$i = 5: \quad \tilde{x}^{(5)} = Ax^{(4)} = \begin{pmatrix} 3.54 \\ 0.502 \\ 4.93 \end{pmatrix}, \qquad x^{(5)} = \frac{\tilde{x}^{(5)}}{\|\tilde{x}^{(5)}\|_\infty} \approx \begin{pmatrix} 0.717 \\ 0.102 \\ 1 \end{pmatrix}, \qquad \lambda^{(5)} = \frac{\tilde{x}_1^{(5)}}{x_1^{(4)}} \approx 4.950,$$

$$i = 6: \quad \tilde{x}^{(6)} = Ax^{(5)} = \begin{pmatrix} 3.54 \\ 0.486 \\ 4.92 \end{pmatrix}, \qquad x^{(6)} = \frac{\tilde{x}^{(6)}}{\|\tilde{x}^{(6)}\|_\infty} \approx \begin{pmatrix} 0.719 \\ 0.0988 \\ 1 \end{pmatrix}, \qquad \lambda^{(6)} = \frac{\tilde{x}_1^{(6)}}{x_1^{(5)}} \approx 4.930.$$

Neben der Festlegung auf den betragsgrößten Eigenwert hat die Potenzmethode den Nachteil sehr langsamer Konvergenz, falls die Eigenwerte nicht hinreichend separiert sind. Eine einfache Erweiterung der Potenzmethode ist die *Inverse Iteration nach Wieland* zur Berechnung des kleinsten Eigenwerts einer Matrix. Zur Herleitung verwenden wir die Tatsache, dass zu einem Eigenwert λ einer regulären Matrix $A \in \mathbb{R}^{n\times n}$ durch λ^{-1} ein Eigenwert der inversen Matrix $A^{-1} \in \mathbb{R}^{n\times n}$ gegeben ist:

$$Aw = \lambda(A)w \quad \Leftrightarrow \quad A^{-1}w = \lambda(A)^{-1}w =: \lambda(A^{-1})w.$$

Die Potenzmethode, angewendet auf die inverse Matrix liefert den betragsgrößten Eigenwert $\lambda_{\max}(A^{-1})$ von A^{-1}. Der Kehrwert dieses Eigenwertes ist der betragskleinste Eigenwert der Matrix A selbst $\lambda_{\min}(A) = \lambda_{\max}(A^{-1})^{-1}$. Dieses Prinzip kann weiter verallgemeinert werden. Dazu sei $\lambda(A)$ ein Eigenwert mit zugehörigem Eigenvektor $w \in \mathbb{R}^n$ von A und $\sigma \in \mathbb{C}$ eine beliebige komplexe Zahl (jedoch kein Eigenwert von A). Dann gilt:

$$(A - \sigma I)w = (\lambda(A) - \sigma)w = \lambda(A - \sigma I)w \quad \Leftrightarrow \quad [A - \sigma I]^{-1}w = \lambda([A - \sigma I]^{-1})w.$$

Die Anwendung der Potenzmethode auf die Matrix $[A - \sigma I]^{-1}$ liefert nach vorangestellter Überlegung den betragskleinsten Eigenwert der Matrix $[A-\sigma I]$, d.h. den Eigenwert von A, der σ am nächsten liegt. Liegen nun Schätzungen für die Eigenwerte der Matrix A vor, so können die genauen Werte mit der Inversen Iteration bestimmt werden. Die Gerschgorin-Kreise liefern oft einen guten Anhaltspunkt für σ.

Satz 4.80 (Inverse Iteration mit Shift)**.** *Es sei $A \in \mathbb{R}^{n\times n}$ eine reguläre Matrix. Es sei $\sigma \in \mathbb{C}$. Für die Eigenwerte λ_i von A gelte:*

$$0 < |\lambda_1 - \sigma| < |\lambda_2 - \sigma| \leq \cdots \leq |\lambda_n - \sigma|.$$

Es sei $x^{(0)} \in \mathbb{R}^n$ ein geeigneter normierter Startwert. Für $i = 1, 2, \ldots$ konvergiert die Iteration

$$[A - \sigma I]\tilde{x}^{(i)} = x^{(i-1)}, \quad x^{(i)} := \frac{\tilde{x}^{(i)}}{\|\tilde{x}^{(i)}\|}, \quad \mu^{(i)} := \frac{\tilde{x}_k^{(i)}}{x_k^{(i-1)}}, \quad \lambda^{(i)} := \sigma + (\mu^{(i)})^{-1},$$

für jeden Index $k \in \{1, \ldots, n\}$ gegen den Eigenwert λ_1 von A:

$$|\lambda_1 - \lambda^{(i)}| = O\left(\left|\frac{\lambda_1 - \sigma}{\lambda_2 - \sigma}\right|^i\right).$$

Beweis: Der Beweis ist eine einfache Folgerung aus Satz 4.77. Falls σ kein Eigenwert von A ist, so ist $B := A - \sigma I$ invertierbar. Die Matrix B^{-1} hat die Eigenwerte $\mu_1, \ldots, \mu_n$, mit

$$\mu_i = (\lambda_i - \sigma)^{-1} \quad \Leftrightarrow \quad \lambda_i = \mu_i^{-1} + \sigma. \tag{4.12}$$

Für diese gilt nach Voraussetzung:

$$|\mu_1| > |\mu_2| \geq \cdots \geq |\mu_n| > 0.$$

Die Potenzmethode, Satz 4.77, angewandt auf die Matrix B^{-1} liefert eine Approximation für μ_1:

$$|\mu^{(i)} - \mu_1| = O\left(\left|\frac{\mu_2}{\mu_1}\right|^i\right).$$

Die gewünschte Aussage folgt mit (4.12). □

In jedem Schritt der Iteration muss ein Lineares Gleichungssystem $[A - \sigma I]\tilde{x} = x$ gelöst werden. Dies geschieht am besten mit einer Zerlegungsmethode, etwa der LR-Zerlegung der Matrix. Die Zerlegung kann einmal in $O(n^3)$ Operationen erstellt werden, anschließend sind in jedem Schritt der Iteration weitere $O(n^2)$ Operationen für das Vorwärts- und Rückwärtseinsetzen notwendig. Die Konvergenzgeschwindigkeit der Inversen Iteration mit Shift kann durch eine gute Schätzung σ gesteigert werden. Eine weitere Verbesserung der Konvergenzgeschwindigkeit kann durch ständiges Anpassen der Schätzung σ erreicht werden. Wird in jedem Schritt die beste Approximation $\sigma + 1/\mu^{(i)}$ als neue Schätzung verwendet, so kann mindestens superlineare Konvergenz erreicht werden. Hierzu wählen wir: $\sigma^{(0)} = \sigma$ und iterieren

$$\sigma^{(i)} = \sigma^{(i-1)} + \frac{1}{\mu^{(i)}}.$$

Jede Modifikation des Shifts ändert jedoch das lineare Gleichungssystem und erfordert die erneute (teure) Erstellung der LR-Zerlegung.

Beispiel 4.81 (Inverse Iteration nach Wieland)**.** *Es sei:*

$$A = \begin{pmatrix} 2 & -0.1 & 0.4 \\ 0.3 & -1 & 0.4 \\ 0.2 & -0.1 & 4 \end{pmatrix}$$

mit den Eigenwerten:

$$\lambda_1 \approx -0.983, \quad \lambda_2 = 1.954, \quad \lambda_3 = 4.029.$$

Wir wollen alle Eigenwerte mit der inversen Iteration bestimmen. Startwerte erhalten wir durch Analyse der Gerschgorin-Kreise:

$$K_1 = K_{0.5}(2), \quad K_2 = K_{0.2}(-1), \quad K_3 := K_{0.3}(4).$$

Die drei Kreise sind disjunkt und wir wählen als Shift in der Inversen Iteration $\sigma_1 = 2$*,* $\sigma_2 = -1$ *sowie* $\sigma_3 = 4$*. Die Iteration wird stets mit* $v = (1, 1, 1)^T$ *und Normierung bezüglich der Maximumsnorm gestartet. Wir erhalten die Näherungen:*

$$\begin{array}{lllll} \sigma_1 = 2: & \mu_1^{(1)} = -16.571, & \mu_1^{(2)} = -21.744, & \mu_1^{(3)} = -21.619, & \mu_1^{(4)} = -21.622, \\ \sigma_2 = -1: & \mu_2^{(1)} = 1.840, & \mu_2^{(2)} = 49.743, & \mu_2^{(3)} = 59.360, & \mu_2^{(4)} = 59.422, \\ \sigma_3 = 4: & \mu_3^{(1)} = 6.533, & \mu_3^{(2)} = 36.004, & \mu_3^{(3)} = 33.962, & \mu_3^{(4)} = 33.990. \end{array}$$

Diese Approximationen ergeben die folgenden Eigenwert-Näherungen:

$$\lambda_1 = \sigma_1 + 1/\mu_1^{(4)} \approx 1.954, \quad \lambda_2 = \sigma_2 + 1/\mu_2^{(4)} \approx -0.983, \quad \lambda_3 = \sigma_3 + 1/\mu_3^{(4)} \approx 4.029.$$

Alle drei Näherungen sind in den ersten wesentlichen Stellen exakt.

Das Beispiel demonstriert, dass die inverse Iteration mit Shift zu einer wesentlichen Beschleunigung der Konvergenz führt, falls gute Schätzungen der Eigenwerte vorliegen.

4.6.4 Das QR-Verfahren zur Eigenwertberechnung

Wir haben bereits die QR-Zerlegung einer Matrix A in eine orthogonale Matrix $Q \in \mathbb{R}^{n\times n}$ sowie eine rechte obere Dreiecksmatrix $R \in \mathbb{R}^{n\times n}$ kennengelernt. Das QR-Verfahren zur Berechnung der Eigenwerte von A beruht auf der folgenden Beobachtung:

$$A = QR = QR(QQ^T) = Q(RQ)Q^T, \tag{4.13}$$

d.h., die Matrix A ist orthogonal ähnlich zur Matrix RQ, hat also die gleichen Eigenwerte wie diese. Wir definieren:

Algorithmus 4.82 (QR-Verfahren). *Es sei $A \in \mathbb{R}^{n\times n}$. Ausgehend von $A^{(1)} := A$ iteriere für $i = 1, \ldots$*

1. *Erstelle die QR-Zerlegung*
$$A^{(i)} =: Q^{(i)} R^{(i)},$$
2. *Berechne*
$$A^{(i+1)} := R^{(i)} Q^{(i)}.$$

Das QR-Verfahren erzeugt eine Folge $A^{(i)}, i \geq 1$ von Matrizen, die gemäß (4.13) alle ähnlich zur Matrix A sind. Wir werden sehen, dass die Diagonaleinträge der Folgenglieder $A^{(i)}$ gegen die Eigenwerte der Matrix A laufen.

Satz 4.83 (QR-Verfahren zur Eigenwertberechnung). *Es sei $A \in \mathbb{R}^{n\times n}$ eine Matrix mit separierten Eigenwerten*
$$|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n|.$$
Dann gilt für die Diagonalelemente $a_{ii}^{(t)}$ der durch das QR-Verfahren erzeugten Matrizen $A^{(t)}$:
$$\{d_{11}^{(t)}, \ldots, d_{nn}^{(t)}\} \to \{\lambda_1, \ldots, \lambda_n\} \quad (t \to \infty).$$

Beweis: Für den technisch aufwändigen Beweis verweisen wir auf [9] oder [6]. □

Im Allgemeinen konvergieren die Diagonalelemente der Folgenglieder $A^{(i)}$ mindestens linear gegen die Eigenwerte. In speziellen Fällen wird jedoch sogar kubische Konvergenz erreicht. Verglichen mit der Potenzmethode und der inversen Iteration weist das QR-Verfahren daher zum einen bessere Konvergenzeigenschaften auf, gleichzeitig werden alle Eigenwerte der Matrix A bestimmt. In jedem Schritt der Iteration muss jedoch eine QR-Zerlegung der Iterationsmatrix $A^{(i)}$ erstellt werden. Bei allgemeiner Matrix $A \in \mathbb{R}^{n\times n}$ sind hierzu $O(n^3)$ arithmetische Operationen notwendig. Um die Eigenwerte mit hinreichender Genauigkeit zu approximieren sind oft sehr viele, > 100 Schritte notwendig. In der praktischen Anwendung wird das QR-Verfahren daher immer in Verbindung mit einer *Reduktionsmethode* (siehe folgendes Kapitel) eingesetzt, bei der die Matrix A zunächst in eine "einfache" ähnliche Form transformiert wird.

Bemerkung 4.84 (LR-Verfahren). *Wie das QR-Verfahren liefert auch das* LR-Verfahren*:*
$$A^{(i)} =: L^{(i)} R^{(i)}, \quad A^{(i+1)} := R^{(i+1)} L^{(i+1)},$$
eine Folge von Matrizen $A^{(i)}$, deren Diagonalelemente gegen die Eigenwerte der Matrix A konvergieren. Das LR-Verfahren zur Eigenwertberechnung konvergiert nur dann, wenn die LR-Zerlegung ohne Pivotisierung durchgeführt werden kann. Daher wird bei allgemeinen Matrizen üblicherweise das QR-Verfahren bevorzugt.

4.6.5 Reduktionsmethoden zur Eigenwertbestimmung

Das Erstellen der QR-Zerlegung einer Matrix $A \in \mathbb{R}^{n\times n}$ mit Hilfe von Householder-Matrizen bedarf $O(n^3)$ arithmetischer Operationen, siehe Satz 4.58. Da in jedem Schritt des QR-Verfahrens diese Zerlegung neu erstellt werden muss, ist dieser Aufwand zu groß. Hat die Matrix A jedoch eine spezielle Struktur, ist sie z.B. eine Bandmatrix, so kann auch die QR-Zerlegung mit weit geringerem Aufwand erstellt werden. Es gilt:

Satz 4.85 (Ähnliche Matrizen)**.** *Zwei Matrizen $A, B \in \mathbb{C}^{n\times n}$ heißen* ähnlich*, falls es eine reguläre Matrix $S \in \mathbb{C}^{n\times n}$ gibt, so dass gilt:*

$$A = S^{-1}BS.$$

Ähnliche Matrizen haben das gleiche charakteristische Polynom und die gleichen Eigenwerte. Zu einem Eigenwert λ sowie Eigenvektor w von A gilt:

$$B(Sw) = S(Aw) = \lambda Sw.$$

Ziel dieses Kapitels ist es durch Ähnlichkeitstransformationen

$$A = A^{(0)} \quad \to \quad A^{(i)} = (S^{(i)})^{-1}A^{(i)}S^{(i)},$$

die Matrix A Schritt für Schritt in eine ähnliche Matrix (also mit den gleichen Eigenwerten) zu transformieren, die eine einfache Struktur hat, so dass die Eigenwerte leichter zu bestimmen, oder sogar direkt ablesbar sind. Mögliche *Normalformen*, bei denen die Eigenwerte unmittelbar ablesbar sind, sind die *Jordan'sche Normalform*, die Diagonalisierung von A, oder die *Schur'sche Normalform*:

Definition 4.86 (Schur'sche Normalform)**.** *Die Matrix $A \in \mathbb{C}^{n\times n}$ habe die Eigenwerte $\lambda_1, \ldots, \lambda_n$ (ihrer Vielfachheit gezählt). Dann existiert eine unitäre Matrix $U \in \mathbb{C}^{n\times n}$, so dass*

$$\bar{U}^T AU = \begin{pmatrix} \lambda_1 & * & \cdots & * \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & * \\ 0 & \cdots & 0 & \lambda_n \end{pmatrix}.$$

Falls $\bar{A}^T = A$ hermitesch ist, so ist $\bar{U}^T AU$ auch hermitesch, also eine Diagonalmatrix.

Die Aufgabe, eine Matrix A in eine Normalform zu transformieren, ist üblicherweise nur bei Kenntnis der Eigenwerte möglich. Dieser Weg eignet sich somit nicht zur Eigenwertberechnung. Daher werden wir im Folgenden die Reduktion der Matrix A auf eine Normalform kennenlernen, bei der die Eigenwerte zwar nicht unmittelbar abgelesen werden, die QR-Zerlegung jedoch mit sehr geringem Aufwand erstellt werden kann. Diese reduzierte Normalform dient dann als Grundlage für das QR-Verfahren zur Eigenwertberechnung. Wir definieren:

Satz 4.87 (Hessenberg-Normalform)**.** *Zu jeder Matrix $A \in \mathbb{R}^{n\times n}$ existiert eine orthogonale Matrix $Q \in \mathbb{R}^{n\times n}$, so dass*

$$Q^T AQ = \begin{pmatrix} * & \cdots & \cdots & \cdots & * \\ * & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & * & * \end{pmatrix},$$

eine Hessenberg-Matrix *ist, also eine rechte obere Dreiecksmatrix, die zusätzlich eine untere Nebendiagonale besitzt. Falls $A = A^T$ symmetrisch ist, so ist $Q^T AQ$ eine Tridiagonalmatrix.*

Beweis: Die Konstruktion der Hessenberg-Matrix erfolgt ähnlich der QR-Zerlegung mit Householder-Transformationen. Um Ähnlichkeitstransformationen sicherzustellen müssen wir die orthogonalen Householder-Matrizen $S^{(i)}$ jedoch von links und rechts an die Matrix A multiplizieren.

Wir beschreiben den ersten Schritt. Es seien $A = (a_1, \ldots, a_n)$ die Spaltenvektoren von A. Wir bestimmen den Vektor $v^{(1)} = (0, v_2^{(1)}, \ldots, v_n^{(1)})^T$ so, dass mit $S^{(1)} = I - 2v^{(1)}(v^{(1)})^T$ gilt

$$S^{(1)} a_1 \in \operatorname{span}(e_1, e_2).$$

Hierzu wählen wir eine Householder-Transformation mit Vektor

$$v^{(1)} = \frac{\tilde{a}_1 + \|\tilde{a}_1\| e_2}{\| \tilde{a}_1 + \|\tilde{a}_1\| e_2 \|},$$

wobei $\tilde{a}_1 = (0, a_{21}, \ldots, a_{n1})$ der reduzierte erste Spaltenvektor ist. Dann gilt mit $S^{(1)} = I - 2v^{(1)}(v^{(1)})^T$ mit $S^{(1)} = (S^{(1)})^T$:

$$A^{(1)} = S^{(1)} A (S^{(1)})^T = \left(\begin{array}{c|ccc} a_{11} & a_{12} & \cdots & a_{1n} \\ \hline * & * & \cdots & * \\ 0 & \vdots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ 0 & * & \cdots & * \end{array}\right) \cdot S^{(1)} =: \left(\begin{array}{c|ccc} a_{11} & * & \cdots & * \\ \hline * & & & \\ 0 & & \tilde{A}^{(1)} & \\ \vdots & & & \\ 0 & & & \end{array}\right), \quad \tilde{A}^{(1)} \in \mathbb{R}^{n-1\times n-1}.$$

Im zweiten Schritt wird das entsprechende Verfahren auf die Matrix $\tilde{A}^{(1)}$ angewendet. Nach $n-2$ Schritten erhalten wir mit Matrix $A^{(n-2)}$, welche Hessenberg-Gestalt hat.

$$Q^T AQ := \underbrace{S^{(n-2)} \cdots S^{(1)}}_{=:Q^T} A \underbrace{S^{(1)} \cdots S^{(n-2)}}_{=:Q}$$

Im Falle $A = A^T$ gilt:

$$(Q^T AQ)^T = Q^T A^T Q = Q^T AQ.$$

D.h., auch $A^{(n-2)}$ ist wieder symmetrisch. Symmetrische Hessenberg-Matrizen sind Tridiagonalmatrizen. □

Bemerkung 4.88 (Hessenberg-Normalform)**.** *Die Transformation einer Matrix $A \in \mathbb{R}^{n\times n}$ in Hessenberg-Form erfordert bei Verwendung der Householder-Transformationen $\frac{5}{3}n^3 + O(n^2)$ arithmetische Operationen. Im Fall symmetrischer Matrizen erfordert die Transformation in eine Tridiagonalmatrix $\frac{2}{3}n^3 + O(n^2)$ Operationen.*

Die Transformation in Hessenberg-Form benötigt demnach etwas mehr arithmetische Operationen als *eine* QR-Zerlegung. Im Anschluss können die QR-Zerlegungen einer Hessenberg-Matrix mit weitaus geringerem Aufwand erstellt werden. Weiter gilt, dass für die QR-Zerlegung einer Hessenberg-Matrix A gilt, dass die Matrix RQ wieder Hessenberg-Gestalt hat. Beides fasst der folgende Satz zusammen:

Satz 4.89 (QR-Zerlegung von Hessenberg-Matrizen)**.** *Es sei $A \in \mathbb{R}^{n\times n}$ eine Hessenberg-Matrix. Dann kann die QR-Zerlegung $A = QR$ mit Householder-Transformationen in $2n^2 + O(n)$ arithmetische Operationen durchgeführt werden. Die Matrix RQ hat wieder Hessenberg-Gestalt.*

Beweis: Es gilt $a_{ij} = 0$ für $i > j+1$. Wir zeigen, dass induktiv, dass diese Eigenschaft für alle Matrizen $A^{(i)}$ gilt, die im Laufe der QR-Zerlegung entstehen. Zunächst folgt im ersten Schritt für $v^{(1)} = a_1 + \|a_1\|e_1$ hieraus $v_k^{(1)} = 0$ für alle $k > 2$. Die neuen Spaltenvektoren berechnen sich zu:

$$a_i^{(1)} = a_i - (a_i, v^{(1)})v^{(1)}. \tag{4.14}$$

Da $v_k^{(1)} = 0$ für $k < 2$ gilt $(a_i^{(1)})_k = 0$ für $k > i+1$, d.h. $A^{(1)}$ hat wieder Hessenberg-Form. Diese Eigenschaft gilt induktiv für $i = 2, \ldots, n-1$.

Da der (reduzierte) Vektor $\tilde{v}^{(i)}$ in jedem Schritt nur zwei von Null verschiedene Einträge hat, kann dieser in 4 Operationen erstellt werden. Die Berechnung der $n - i$ neuen Spaltenvektoren gemäß (4.14) bedarf je 4 arithmetischer Operationen. Insgesamt ergibt sich ein Aufwand von

$$\sum_{i=1}^{n-1} 4 + 4(n-i) = 2n^2 + O(n).$$

Es bleibt (als Übung) die Hessenberg-Gestalt der Matrix $A' := RQ$ nachzuweisen. □

In Verbindung mit der Reduktion auf Hessenberg, bzw. auf Tridiagonalgestalt ist das QR-Verfahren eines der effizientesten Verfahren zur Berechnung von Eigenwerten einer Matrix $A \in \mathbb{R}^{n\times n}$. Die QR-Zerlegung kann in $O(n^2)$ Operationen durchgeführt werden, und im Laufe des QR-Verfahrens entstehen ausschließlich Hessenberg-Matrizen.

Die Konvergenz des QR-Verfahrens hängt von der Separation der Eigenwerte $|\lambda_i|/|\lambda_{i+1}|$ ab. Je weiter die Eigenwerte voneinander entfernt sind, umso besser konvergiert das Verfahren.

Im allgemeinen kann lineare Konvergenz gezeigt werden. In speziellen Fällen kann jedoch sogar kubische Konvergenz der Diagonalelemente $A_{ii}^{(t)}$ gegen die Eigenwerte gezeigt werden.

Wie die Inverse Iteration kann das QR-Verfahren durch Einführen eines *Shifts* beschleunigt werden. Mit Koeffizienten μ_i wird die Iteration ersetzt durch die Vorschrift:

$$A^{(i-1)} - \mu_i I = Q^{(i)} R^{(i)}, \quad A^{(i)} := R^{(i)} Q^{(i)} + \mu_i I.$$

Abschließend betrachten wir hierzu ein Beispiel:

Beispiel 4.90 (Eigenwert-Berechnung mit Reduktion und QR-Verfahren)**.** *Wir betrachten die Matrix*

$$A = \begin{pmatrix} 338 & -20 & -90 & 32 \\ -20 & 17 & 117 & 70 \\ -90 & 117 & 324 & -252 \\ 32 & 70 & -252 & 131 \end{pmatrix}.$$

Die Matrix A ist symmetrisch und hat die Eigenwerte:

$$\lambda_1 \approx 547.407, \quad \lambda_2 \approx 297.255, \quad \lambda_3 \approx -142.407, \quad \lambda_4 \approx 107.745.$$

Schritt 1: Reduktion auf Hessenberg- (Tridiagonal)-Gestalt.

Im ersten Reduktionsschritt wählen wir mit $\tilde{a}_1 = (0, -20, -90, 32)$ *den Spiegelungsvektor* $v^{(1)}$ *als:*

$$v^{(1)} = \frac{\tilde{a}_1 + \|\tilde{a}_1\| e_2}{\| \tilde{a}_1 + \|\tilde{a}_1\| e_2 \|}.$$

Wir erhalten mit $S^{(1)} := I - 2v^{(1)}(v^{(1)})^T$*:*

$$A^{(1)} = S^{(1)} A S^{(1)} = \begin{pmatrix} 338 & 97.591 & 0 & 0 \\ 97.591 & 477.579 & 27.797 & 106.980 \\ 0 & 27.797 & -82.345 & -103.494 \\ 0 & 106.980 & -103.494 & 76.765 \end{pmatrix}.$$

Mit $\tilde{a}_2^{(1)} = (0, 0, 27.797, 106.980)^T$ *und*

$$v^{(2)} = \frac{\tilde{a}_2 + \|\tilde{a}_2\| e_3}{\| \tilde{a}_2 + \|\tilde{a}_2\| e_3 \|}$$

folgt mit $S^{(2)} := I - 2v^{(2)}(v^{(2)})^T$*:*

$$H := A^{(2)} = S^{(2)} A^{(1)} S^{(2)} = \begin{pmatrix} 338 & 97.591 & 0 & 0 \\ 97.591 & 477.579 & 110.532 & 0 \\ 0 & 110.532 & 16.231 & -129.131 \\ 0 & 0 & -129.131 & -21.900 \end{pmatrix}.$$

Die Matrix H hat nun Tridiagonalgestalt. Alle Transformationen waren Ähnlichkeitstransformationen. Daher haben H und A die gleichen Eigenwerte.

Schritt 2: QR-Verfahren

Wir führen nun einige Schritte des QR-Verfahrens durch, verzichten dabei auf die Zwischenschritte zum Erstellen der QR-Zerlegung. Es sei $A^{(1)} := H$*. Dann ist:*

$$A^{(1)} = Q^{(1)}R^{(1)} \qquad A^{(2)} = R^{(1)}Q^{(1)} = \begin{pmatrix} 400.759 & 123.634 & 0 & 0 \\ 123.634 & 441.338 & -32.131 & 0 \\ 0 & -32.131 & -42.082 & -122.498 \\ 0 & 0 & -122.498 & 9.985 \end{pmatrix}$$

$$A^{(2)} = Q^{(2)}R^{(2)} \qquad A^{(3)} = R^{(2)}Q^{(2)} = \begin{pmatrix} 473.938 & 113.971 & 0 & 0 \\ 113.971 & 370.411 & 10.831 & 0 \\ 0 & 10.831 & -74.680 & -111.053 \\ 0 & 0 & -111.053 & 40.331 \end{pmatrix}$$

$$A^{(3)} = Q^{(3)}R^{(3)} \qquad A^{(4)} = R^{(3)}Q^{(3)} = \begin{pmatrix} 520.095 & 78.016 & 0 & 0 \\ 78.016 & 324.515 & -4.350 & 0 \\ 0 & -4.350 & -98.717 & -94.942 \\ 0 & 0 & -94.942 & 64.105 \end{pmatrix}$$

$$A^{(4)} = Q^{(4)}R^{(4)} \qquad A^{(5)} = R^{(4)}Q^{(4)} = \begin{pmatrix} 538.682 & 45.895 & 0 & 0 \\ 45.895 & 305.970 & 1.926 & 0 \\ 0 & 1.926 & -115.400 & -77.620 \\ 0 & 0 & -77.620 & 80.747 \end{pmatrix}$$

$$\vdots$$

$$A^{(9)} = Q^{(9)}R^{(9)} \qquad A^{(10)} = R^{(9)}Q^{(9)} = \begin{pmatrix} 547.387 & 2.245 & 0 & 0 \\ 2.245 & 297.275 & -0.0427 & 0 \\ 0 & -0.0453 & -140.561 & -21.413 \\ 0 & 0 & -21.413 & 105.898 \end{pmatrix}.$$

Für die Diagonalelemente gilt:

$$a_{11}^{(10)} \approx 547.387, \quad a_{22}^{(10)} \approx 297.275, \quad a_{33}^{(10)} \approx -140.561, \quad a_{44}^{(10)} \approx 105.898.$$

Diese Diagonalelemente stellen sehr gute Näherungen an alle *Eigenwerte der Matrix A dar:*

$$\frac{|a_{11}^{(10)} - \lambda_1|}{|\lambda_1|} \approx 0.00005, \quad \frac{|a_{22}^{(10)} - \lambda_2|}{|\lambda_2|} \approx 0.00007, \quad \frac{|a_{33}^{(10)} - \lambda_3|}{|\lambda_3|} \approx 0.01, \quad \frac{|a_{44}^{(10)} - \lambda_4|}{|\lambda_4|} \approx 0.01.$$

5 Numerische Iterationsverfahren

In diesem Kapitel besprechen wir numerische Iterationsverfahren (insbesondere Fixpunktverfahren) als eine weitere Lösungsmethode zur Lösung von linearen Gleichungssystemen (Kapitel 4) sowie zur Lösung von nicht-linearen Gleichungen (siehe Kapitel 2). Das zentrale Hilfsmittel ist der Banachsche Fixpunktsatz, der als Voraussetzung eine Fixpunktformulierung der Aufgabenstellung verlangt. Alle Resultate dieses Kapitels werden direkt für den höherdimensionalen Fall hergeleitet.

5.1 Der Banachsche Fixpunktsatz

In dem ganzen Kapitel betrachten wir Iterationen der Art

$$x^0 \in \mathbb{R}^n, \quad x^{k+1} = g(x^k), \quad k = 0, 1, 2, \ldots,$$

mit einer Abbildung $g(\cdot) : \mathbb{R}^n \to \mathbb{R}^n$. Ein Punkt $x \in \mathbb{R}^n$ heißt Fixpunkt, falls $g(x) = x$.

Beispiel 5.1 (Newton-Verfahren als Fixpunktiteration)**.** *Zur Lösung eines nicht-linearen Gleichungssystems im $\mathbb{R}^n$ sei*

$$f_i(x_1, \ldots, x_n) = 0, \quad i = 1, \ldots, n,$$

oder in kurzer Schreibweise:

$$f(x) = 0$$

mit $f = (f_1, \ldots, f_n)^T$ und $(x_1, \ldots, x_n)^T$. Dann lässt sich das vereinfachte Newton-Verfahren schreiben als

$$x^{k+1} = x^k + C^{-1} f(x^k), \quad k = 0, 1, 2, \ldots$$

mit einer Matrix $C \in \mathbb{R}^{n \times n}$. Das Standard Verfahren von Newton verlangt die erste Ableitung, so dass $C := f'(x^k) \in \mathbb{R}^{n \times n}$. Die Berechnung der höherdimensionalen Ableitung zeigen wir später.

Im Folgenden sei $\|\cdot\|$ eine beliebige Vektornorm auf $\mathbb{R}^n$ und $\|\cdot\|$ die entsprechende natürliche Matrizennorm. Die ausführliche Diskussion der entsprechenden Eigenschaften findet der aufmerksame Leser in Kapitel 4.

Zunächst rekapitulieren wir die bereits aus der Analysis bekannte Lipschitz-Stetigkeit einer Funktion:

Definition 5.2 (Lipschitz-Stetigkeit, Kontraktion)**.** *Es sei $G \subset \mathbb{R}^n$ eine nichtleere abgeschlossene Menge. Eine Abbildung $g : G \to \mathbb{R}^n$ wird Lipschitz-stetig genannt, falls*

$$\|g(x) - g(y)\| \leq q\|x - y\|, \quad x, y \in G,$$

mit $q > 0$. Falls $0 < q < 1$, so nennt man g eine Kontraktion auf G.

Zur Rekapitulation:

Bemerkung 5.3. *Differenzierbarkeit $\Rightarrow$ absolute Stetigkeit $\Rightarrow$ Lipschitz-Stetigkeit $\Rightarrow$ gleichmäßige Stetigkeit $\Rightarrow$ Stetigkeit.*

Zum Beispiel ist die Wurzelfunktion $f(x) = \sqrt{x}$ auf $[0, 1]$ zwar gleichmäßig stetig aber nicht Lipschitz-stetig.

Der Banachsche Fixpunktsatz besagt nun, dass jede Selbstabbildung $g : G \to G$, welche eine Kontraktion ist, einen Fixpunkt besitzt:

Satz 5.4 (Banach'scher Fixpunktsatz)**.** *Es sei $G \subset \mathbb{R}^n$ eine nichtleere und abgeschlossene Punktmenge und $g : G \to G$ eine Lipschitz-stetige Selbstabbildung, mit Lipschitz-Konstante $q < 1$ (also eine Kontraktion).*

- *Dann existiert genau ein Fixpunkt $z \in G$ von g und die Iterationsfolge $(x^k)_k$ konvergiert für jeden Startpunkt $x^0 \in G$, so dass $x^k \to z$ für $k \to \infty$.*
- *Es gilt die a priori Abschätzung:*

$$\|x^k - z\| \leq \frac{q^k}{1 - q}\|x^1 - x^0\|.$$

- *Es gilt die a posteriori Abschätzung:*

$$\|x^k - z\| \leq \frac{q}{1 - q}\|x^k - x^{k-1}\|.$$

Beweis:

(i) Existenz eines Grenzwertes. Da g eine Selbstabbildung in G ist, sind alle Iterierten $x^k = g(x^{k-1}) = \ldots = g^k(x^0)$ bei Wahl eines beliebigen Startpunkts $x^0 \in G$ definiert. Aufgrund der Kontraktionseigenschaft gilt:

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

Wir zeigen, dass $(x^k)_k$ eine Cauchy-Folge ist. Für jedes $l \geq m$ erhalten wir:

$$\begin{aligned} \|x^l - x^m\| &\leq \|x^l - x^{l-1}\| + \ldots + \|x^{m+1} - x^m\| \\ &\leq (q^{l-1} + q^{l-2} + \ldots + q^m)\|x^1 - x^0\| \\ &= q^m \frac{1-q^{l-m}}{1-q}\|x^1 - x^0\| \\ &\leq q^m \frac{1}{1-q}\|x^1 - x^0\| \to 0 \quad (l \geq m \to 0). \end{aligned} \tag{5.1}$$

D.h., $(x^l)_{l\in\mathbb{N}}$ ist eine Cauchy-Folge. Da alle Folgenglieder in G liegen und G als abgeschlossene Teilmenge des $\mathbb{R}^n$ vollständig ist, existiert der Grenzwert $x^l \to z \in G$.

(ii) Fixpunkteigenschaft. Als zweites weisen wir nach, dass z tatsächlich ein Fixpunkt von g ist. Aus der Stetigkeit von g folgt mit $x^k \to z$ auch $g(x^k) \to g(z)$. Dann gilt für die Iteration $x^{k+1} := g(x^k)$ bei Grenzübergang

$$z \leftarrow x^{k+1} = g(x^k) \to g(z) \quad (k \to \infty).$$

(iii) Eindeutigkeit. Die Eindeutigkeit folgt aus der Kontraktionseigenschaft. Es seien z und $\hat{z}$ zwei Fixpunkte von g. Dann ist

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

Dies kann wegen $q < 1$ nur dann gültig sein, wenn $\|z - \hat{z}\| = 0$, d.h. $z = \hat{z}$ ist. Damit ist der Fixpunkt eindeutig.

(iv) A priori Fehlerabschätzung. Es gilt mit (5.1)

$$\|z - x^m\| \xleftarrow[l\to\infty]{} \|x^l - x^m\| \leq q^m \frac{1}{1-q}\|x^1 - x^0\|.$$

(v) A posteriori Fehlerabschätzung. Es gilt wieder mit (5.1):

$$\|x^m - z\| \leq q\frac{1}{1-q}\|x^m - x^{m-1}\|.$$

□

Zur Anwendung des Banachschen Fixpunktsatzes auf eine Abbildung $g : G \subset \mathbb{R}^n \to \mathbb{R}^n$ müssen die beiden Voraussetzungen Selbstabbildung sowie Kontraktionseigenschaft nachgewiesen werden. Angenommen, g sei eine Kontraktion. Dann müssen wir die Existenz einer abgeschlossenen und nichtleeren Teilmenge von G nachweisen, welche von der Abbildung g auf sich selbst abgebildet wird. Angenommen, auf der Kugel

$$K_\rho(c) := \{x \in \mathbb{R}^n | \, \|x - c\| \leq \rho\}, \quad \rho > 0, c \in \mathbb{R}^n,$$

sei g eine Kontraktion mit Lipschitz-Konstante $q < 1$. Dann gilt für $x \in K_\rho(c)$:

$$\|g(x) - c\| \leq \|g(x) - g(c)\| + \|g(c) - c\|,$$

wobei $\|g(x) - g(c)\| \leq q\rho$. Falls zusätzlich gilt:

$$\|g(c) - c\| \leq (1-q)\rho,$$

dann ist

$$\|g(x) - c\| \leq q\rho + (1-q)\rho = \rho$$

und g bildet in sich selbst ab.

Als nächstes rekapitulieren wir den Schrankensatz, der die erste Ableitung (partielle Ableitung) mit der Lipschitz-Stetigkeit verknüpft:

Satz 5.5 (Schrankensatz)**.** *Die Abbildung $g : G \to \mathbb{R}^n$ sei stetig differenzierbar auf der konvexen Menge G. Dann gilt*

$$\|g(x) - g(y)\| \leq \sup_{\xi \in G} \|g'(\xi)\| \, \|x - y\|, \quad x, y \in G,$$

mit der partiellen Ableitung (Jacobi-Matrix, weiter unten ausführlicher)

$$g'(x) = \left(\frac{\partial g_i}{\partial x_j} \right)_{i,j=1,\ldots,n} \in \mathbb{R}^{n \times n}.$$

Falls $\sup_{\xi \in G} \|g'(\xi)\| < 1$, dann ist g eine Kontraktion auf G. Insbesondere gilt in 1D auf dem Intervall $G := [a, b]$:

$$q := \max_{\xi \in [a,b]} |g'(\xi)|.$$

Beweis: Der 1D-Fall ist ein Spezialfall des höher-dimensionalen Falles. Aufgrund seiner Einfachheit beweisen wir in separat.

(i) Der eindimensionale Fall. Es seien $x, y \in [a, b]$. Nach dem reellen Mittelwertsatz gibt es ein $\xi \in [a, b]$, so dass

$$|g(x) - g(y)| = |g'(\xi)(x - y)| = |g'(\xi)| \, |x - y| \leq q|x - y|.$$

(ii) Der n-dimensionale Fall. Es seien $x, y \in G$. Wir setzen aus Normierungsgründen für $i = 1, \ldots, n$:

$$\phi_i(s) := g_i(x + s(y - x)), \quad 0 \leq s \leq 1.$$

Dann gilt

$$g_i(x) - g_i(y) = \phi_i(1) - \phi_i(0) = \int_0^1 \phi'(s) \, ds.$$

Für die Ableitung gilt

$$\phi_i'(s) = \sum_{j=1}^{n} \frac{\partial g_i}{\partial x_j}(x + s(y - x))(y - x)_j.$$

Hiermit und den Stetigkeitseigenschaften der Vektornorm (eine Norm ist immer eine stetige Abbildung!) folgt

$$\begin{aligned}\|g(y) - g(x)\| &= \left\| \int_0^1 g'(x + s(y-x)) \cdot (y-x)\, ds \right\| \\ &\leq \int_0^1 \|g'(x + s(y-x))\|\, ds\, \|y - x\| \\ &\leq \sup_{\xi \in G} \|g'(\xi)\|\, \|y - x\|.\end{aligned}$$

□

Zusammenfassen der Ergebnisse liefert das wichtige:

Korollar 5.6. *Zu jedem Fixpunkt $z \in G$ der Abbildung g mit $\|g'(z)\| < 1$ gibt es eine Umgebung*

$$K_\rho = \{x \in \mathbb{R}^n | \, \|x - z\| \leq \rho\} \subset G,$$

so das g eine Kontraktion von $K_\rho(z)$ in sich selbst ist.

Beispiel 5.7 (Konvergenz zur Lösung nicht-linearer Gleichungen)**.** *Zu $f :\in \mathbb{R}^n \to \mathbb{R}^n$ suchen wir eine Nullstelle $x \in \mathbb{R}^n$ mit $f(x) = 0$. Mit einer Matrix $C \in \mathbb{R}^{n\times n}$ definieren wir die Iteration:*

$$x^0 \in \mathbb{R}^n, \quad x^{k+1} = x^k + C^{-1} f(x^k), \quad k = 0, 1, 2, \ldots$$

Dieses Verfahren konvergiert, falls f auf einer Kugel $K_\rho(c) \subset \mathbb{R}^n$ stetig differenzierbar ist und

$$\sup_{\zeta \in K_\rho(c)} \|I + C^{-1} f'(\zeta)\| =: q < 1, \quad \|C^{-1} f(c)\| \leq (1-q)\rho.$$

Beispiel 5.8 (Lösung linearer Gleichungssysteme)**.** *Es seien $A \in \mathbb{R}^{n\times n}$ und $b \in \mathbb{R}^n$ gegeben. Das lineare Gleichungssystem ist äquivalent zur Nullstellenaufgabe*

$$f(x) := b - Ax = 0.$$

Die iterative Lösung (im Gegensatz zur direkten Lösung) kann mit einer regulären Matrix $C \in \mathbb{R}^{n\times n}$ als Fixpunktaufgabe hergeleitet werden:

$$x = g(x) := x + C^{-1} f(x) = x + C^{-1}(b - Ax) = (I - C^{-1}A)x + C^{-1}b.$$

Die Matrix $B := I - C^{-1}A$ nennen wir die Iterationsmatrix der zugehörigen Fixpunktiteration (auch sukzessive Approximation genannt):

$$x^{k+1} = Bx^k + C^{-1}b, \quad k = 1, 2, \ldots.$$

Die Abbildung g ist wegen

$$\|g(x) - g(y)\| = \|B(x-y)\| \leq \|B\|\|x - y\|$$

für $\|B\| < 1$ mit $B := I - C^{-1}A$ eine Kontraktion auf ganz $\mathbb{R}^n$. Dann konvergiert die Iterationsfolge gegen einen Fixpunkt von g und somit zur Lösung von $Ax = b$.

Bemerkung 5.9. *Später werden wir mit Hilfe des Banachschen Fixpunktsatzes verschiedene Verfahren zur iterativen Lösung von linearen Gleichungssystemen herleiten.*

Bemerkung 5.10. *Die Konvergenzanalyse der Fixpunktverfahren kann mit Hilfe der bereits diskutieren Techniken in Kapitel 2.6 durchgeführt werden.*

5.2 Fixpunkt-Iterationen zum Lösen von nichtlinearen Gleichungen

Wir rekapitulieren aus Kapitel 2, dass das klassische Newton-Verfahren in einer Dimension als Fixpunktiteration aufgefasst werden kann. Die Newton-Iteration gehört zur Klasse der Fixpunktiterationen mit der Iterationsfunktion

$$F(x) := x - \frac{f(x)}{f'(x)}. \tag{5.2}$$

Jeder Fixpunkt $z = F(z)$ ist offenbar eine Nullstelle $f(z) = 0$.

5.2.1 Newton-Verfahren im $\mathbb{R}^n$

Aufgrund seiner herausragenden Bedeutung widmen wir dem Newton-Verfahren für höhere Dimensionen einen eigenen Abschnitt. Die prinzipiellen Aussagen (Existenz, quadratische Konvergenz, gedämpftes Newton-Verfahren, vereinfachtes Newton-Verfahren) sind mit dem $1D$-Fall vergleichbar.

Es sei $f : D \subset \mathbb{R}^n \to \mathbb{R}^n$. Zur Lösung von $f(x) = (f_1(x), \ldots, f_n(x)) = 0$ lautet die Newton-Iteration formal:

$$x^0 \in \mathbb{R}^n, \quad x^{k+1} = x^k - f'(x^k)^{-1} f(x^k), \quad k = 0, 1, 2, \ldots \tag{5.3}$$

Die Ableitung $f'(x) : \mathbb{R}^n \to \mathbb{R}^{n \times n}$ ist die *Jacobi-Matrix* von f:

$$f'(x)_{ij} = \frac{\partial f_i}{\partial x_j}, \quad i, j = 1, \ldots, n.$$

Die Tatsache, dass die Ableitung von f im mehrdimensionalen Fall eine Matrix ist, stellt den wesentlichen Unterschied zum eindimensionalen Newton-Verfahren dar. Anstelle einer Ableitung sind nun n^2 Ableitungen zu berechnen. Und anstelle einer Division durch $f'(x_k)$ ist in jedem Schritt der Newton-Iteration ein lineares Gleichungssystem mit Koeffizientenmatrix $f'(x_k) \in \mathbb{R}^{n \times n}$ zu lösen. Wir multiplizieren in (5.3) mit $f'(x_k)$ und erhalten

$$x^0 \in \mathbb{R}^n, \quad f'(x_k) x^{k+1} = f'(x_k) x^k - f(x^k), \quad k = 0, 1, 2, \ldots$$

Das Newton-Verfahren wird als Defektkorrektur-Verfahren durchgeführt. So kann in jedem Schritt der Aufwand einer Matrix-Vektor Multiplikation gespart werden:

Definition 5.11 (Newton-Verfahren als Defektkorrektur)**.** *Wähle Startwert* $x^0 \in \mathbb{R}^n$ *und iteriere:*

$$f'(x^k)\delta x = d^k, \quad d^k := -f(x^k),$$
$$x^{k+1} = x^k + \delta x, \quad k = 0, 1, 2, \ldots.$$

Im Folgenden diskutieren wir kurz das Aufstellen der Jacobi-Matrix. Die Funktion f besitzt n Komponentenfunktionen f_i und n unterschiedliche Variablen $x_1, \ldots, x_n$. Jede Änderung in einer Komponentenfunktion f_i bezüglich der Variablen x_j wird durch die partielle Ableitung (Analysis 2) beschrieben:

$$\frac{\partial f_i}{\partial x_j}.$$

Letztendlich erhalten wir somit eine $n \times n$ Matrix:

$$f'(x) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_1} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \frac{\partial f_n}{\partial x_2} & \cdots & \frac{\partial f_n}{\partial x_n}. \end{pmatrix}$$

5.2.2 Newton-Kantorovich

Die zentrale Konvergenzaussage des Newton-Verfahrens wird im Satz von Newton-Kantorovich zusammengefasst. Hierzu sei $f : G \subset \mathbb{R}^n \to \mathbb{R}^n$ eine differenzierbare Abbildung. Mit $\|\cdot\|$ bezeichnen wir stets die euklidische Vektornorm und induzierte Matrixnorm, also die Spektralnorm. Wir suchen eine Nullstelle $z \in \mathbb{R}^n$ so dass $f(z) = 0$.

Satz 5.12 (Newton-Kantorovich)**.** *Es sei $D \subset \mathbb{R}^n$ eine offene und konvexe Menge. Weiterhin sei $f : D \subset \mathbb{R}^n \to \mathbb{R}^n$ stetig-differenzierbar.*
(i) *Die Jacobi-Matrix f' sei gleichmäßig Lipschitz-stetig für alle $x, y \in D$:*

$$\|f'(x) - f'(y)\| \leq \gamma \|x - y\|, \quad x, y \in D. \tag{5.4}$$

(ii) *Weiter habe die Jacobi-Matrix auf D eine gleichmäßig beschränkte Inverse*

$$\|f'(x)^{-1}\| \leq \beta, \quad x \in D. \tag{5.5}$$

(iii) *Es gelte für den Startpunkt $x^0 \in D$:*

$$q := \alpha\beta\gamma < \frac{1}{2}, \quad \alpha := \|f'(x^0)^{-1} f(x^0)\|. \tag{5.6}$$

(iv) *Für $r := 2\alpha$ ist die abgeschlossene Kugel*

$$B_r(x_0) := \{x \in \mathbb{R}^n : \ \|x - x_0\| \leq r\}$$

in der Menge D enthalten.
Dann besitzt die Funktion f eine Nullstelle $z \in B_r(x_0)$ und die Newton-Iteration

$$\begin{aligned} f'(x^k)\delta x &= d^k, \quad d^k := -f(x^k), \\ x^{k+1} &= x^k + \delta x, \quad k = 0, 1, 2, \ldots. \end{aligned}$$

konvergiert quadratisch gegen diese Nullstelle z. Darüber hinaus gilt die a priori Fehlerabschätzung

$$\|x^k - z\| \leq 2\alpha q^{2^k - 1}, \quad k = 0, 1, \ldots.$$

Beweis: Der Beweis zum Satz ist weitaus aufwändiger als im eindimensionalen Fall, daher geben wir zunächst eine Skizze an:

(i) Herleitung von Hilfsabschätzungen.

(ii) Alle Iterierten liegen in der Kugel $B_r(x_0)$ und es gilt die a priori Fehlerabschätzung für $\|x^k - x^0\|$ (Beweis über vollständige Induktion).

(iii) Zeige $(x^k)_{k \in \mathbb{N}}$ ist Cauchy-Folge und hat damit in $\mathbb{R}^n$ einen eindeutigen Grenzwert z.

(iv) Existenz einer Nullstelle: Zeige, dass z eine Nullstelle der Funktion f ist.

(v) Eindeutigkeit: Zeige, dass die Nullstelle z eindeutig ist.

Nun zum ausführlichen Beweis:

(i) Herleitung von Hilfsabschätzungen:
Es seien $x, y, z \in D$. Da D konvex ist, gilt für alle $x, y \in D$:

$$f_j(x) - f_j(y) = \int_0^1 \frac{d}{ds} f_j(sx + (1-s)y) \, \mathrm{d}s, \quad j = 1, \ldots, n.$$

Mit der Kettenregel erhalten wir

$$\frac{d}{ds} f_j(sx + (1-s)y) = \sum_{k=1}^{n} \frac{\partial f_j}{\partial x_k}(sx + (1-s))(x_k - y_k)$$

und damit

$$f_j(x) - f_j(y) = \int_0^1 \sum_{k=1}^{n} \frac{\partial f_j}{\partial x_k}(sx + (1-s)y)(x_k - y_k)\, ds.$$

In kompakter Schreibweise bedeutet dies:

$$f(x) - f(y) = \int_0^1 f'(sx + (1-s)y)(x - y)\, ds.$$

Unter Hinzunahme von $f'(z)(y-x)$ folgern wir nun, dass

$$f(x) - f(y) - f'(z)(y-x) = \int_0^1 (f'(sx + (1-s)y) - f'(z))(x - y)\, ds.$$

Mit Hilfe der Lipschitz-Stetigkeit von f', Bedingung (5.4), folgern wir

$$\begin{aligned} \|f(y) - f(x) - f'(z)(y-x)\| &\leq \gamma \|y - x\| \int_0^1 \|s(x-z) + (1-s)(y-z)\|\, ds \\ &\leq \frac{\gamma}{2} \|y - x\| (\|x - z\| + \|y - z\|). \end{aligned}$$

Für die Wahl $z = x$ schließen wir damit auf

$$\|f(y) - f(x) - f'(x)(y-x)\| \leq \frac{\gamma}{2} \|y - x\|^2, \quad \forall x, y \in D. \tag{5.7}$$

Und für die die Wahl $z = x_0$ erhalten wir:

$$\|f(y) - f(x) - f'(x_0)(y-x)\| \leq r\gamma \|y - x\|, \quad \forall x, y \in B_r(x_0). \tag{5.8}$$

(ii) Wir zeigen: alle Iterierten liegen in $B_r(x_0)$ und es gilt die a priori Fehlerabschätzung für $\|x^k - x^0\|$. Wir führen den Beweis über vollständige Induktion und zeigen:

$$\|x^k - x^0\| \leq r, \quad \|x^k - x^{k-1}\| \leq \alpha q^{2^k - 1}, \quad k = 1, 2, \ldots. \tag{5.9}$$

Wir starten mit $k = 1$. Es gilt mit Hilfe der Bedingungen (5.5) und (5.6):

$$\|x_1 - x_0\| = \|f'(x_0)^{-1} f(x_0)\| = \alpha = \frac{r}{2} < r.$$

Die Aussage ist also wahr.

Induktionsschritt $k \to k+1$. Nach Induktionsvoraussetzung seien die beiden Gleichungen (5.9) wahr für $k \geq 1$. Das ist aufgrund der Bedingung (5.5) und $x^k \in B_r(x_0) \subset D$, die Iterierte x^{k+1} wohl-definiert. Dann gilt unter Ausnutzung von Bedingung (5.5), der

Newton-Iteration für x^k und Abschätzung (5.7), der Induktionsvoraussetzung (5.9) und der Definition von q folgende Abschätzungskette:

$$\begin{aligned}
\|x^{k+1} - x^k\| &= \|f'(x^k)^{-1} f(x^k)\| \\
&\leq \beta \, \|f(x^k)\| \\
&= \beta \|f(x^k) - f(x^{k-1}) - f'(x^{k-1})(x^k - x^{k-1})\| \\
&\leq \frac{\beta\gamma}{2} \|x^k - x^{k-1}\|^2 \\
&\leq \frac{\beta\gamma}{2} \left[\alpha q^{2^{k-1}-1}\right]^2 \\
&= \frac{\alpha}{2} q^{2^k-1} \\
&< \alpha q^{2^k-1}.
\end{aligned}$$

Weiter erhalten wir

$$\begin{aligned}
\|x^{k+1} - x^0\| &\leq \|x^{k+1} - x^k\| + \ldots + \|x^1 - x^0\| \\
&\leq \alpha(1 + q + q^3 + q^7 + \ldots + q^{2^k-1}) \\
&\leq \frac{\alpha}{1-q} \\
&\leq 2\alpha \\
&= r.
\end{aligned}$$

Damit ist der Induktionsschritt von $k \to k+1$ gezeigt, d.h., die beiden Ungleichungen (5.9) sind gültig für $k+1$.

(iii) Existenz eines Grenzwertes $x^k \to z$ durch Nachweis der Cauchy-Eigenschaft:
Es sei $m > 0$. Unter Ausnutzung von $q < \frac{1}{2}$ (Voraussetzung!) gilt

$$\begin{aligned}
\|x^k - x^{k+m}\| &\leq \|x^k - x^{k+1}\| + \ldots + \|x^{k+m-1} - x^{k+m}\| \\
&\leq \alpha(q^{2^k-1} + q^{2^{k+1}-1} + \ldots + q^{2^{m+k-1}-1}) \\
&= \alpha q^{2^k-1}(1 + q^{2^k} + \ldots + (q^{2^k})^{2^{m-1}-1}) \\
&\leq 2\alpha q^{2^k-1}
\end{aligned} \tag{5.10}$$

Damit ist gezeigt, dass $(x^k) \subset D_0$ eine Cauchy-Folge ist, da $q < \frac{1}{2}$. Da D endlichdimensional (hier ist jeder Raum ein Banachraum - also vollständig!), existiert der Limes

$$z = \lim_{k\to\infty} x^k$$

Im Grenzübergang $k \to \infty$ erhalten wir dann in der ersten Ungleichung in (5.9):

$$\|z - x^0\| \leq r,$$

so dass $z \in B_r(x_0)$. Im Grenzübergang $m \to \infty$ in (5.10), verifizieren wir die Fehlerabschätzung des Satzes:

$$\|x^k - z\| \leq 2\alpha q^{2^k-1}, \quad k = 0, 1, \ldots.$$

(iv) Zeige, dass $z \in B_r(x_0)$ eine Nullstelle von f ist:
Die Newton-Iterationsvorschrift sowie Bedingung (5.4) liefern

$$\begin{aligned}\|f(x^k)\| &= \|f'(x^k)(x^k - x^{k-1})\| \\ &\leq \|f'(x^k) - f'(x^0) + f'(x^0)\| \, \|x^{k+1} - x^k\| \\ &\leq (\gamma\|x^k - x^0\| + \|f'(x^0)\|) \, \|x^{k+1} - x^k\| \\ &\to 0, \quad k \to \infty.\end{aligned}$$

Daher gilt

$$f(x^k) \to 0, \quad k \to \infty.$$

Die Stetigkeit von f impliziert dann $f(z) = 0$.

(v) Eindeutigkeit der Nullstelle $z \in B_r(x_0)$:
Die Eindeutigkeit wird mit Hilfe der Kontraktionseigenschaft und der Formulierung der Newton-Iteration als Fixpunktiteration gezeigt. Die gefundene Nullstelle $x^k \to z$ von $f(x)$ ist auch Fixpunkt der vereinfachten Iteration:

$$\bar{g}(x) := x - f'(x^0)^{-1} f(x).$$

Diese Iteration ist eine Kontraktion, denn aus

$$\bar{g}(x) - \bar{g}(y) = x - y - f'(x^0)^{-1} f(x) + f'(x^0)^{-1} f(y) = f'(x^0)^{-1}(f(y) - f(x) - f'(x^0)(y - x)),$$

folgt mit den Bedingungen (5.5) und (5.6) sowie (5.8) die Abschätzung:

$$\|\bar{g}(x) - \bar{g}(y)\| \leq \beta\beta\gamma r\|y - x\| \leq 2q\|y - x\|, \quad \forall x, y \in B_r(x_0).$$

Da $q < \frac{1}{2}$ ist $\bar{g}$ eine Kontraktion. Eine Kontraktionsabbildung kann höchstens einen Fixpunkt haben. Da dieser Fixpunkt die Nullstelle von f ist, haben wir die Nullstelle $z \in B_r(x_0)$ eindeutig bestimmt. Damit ist alles gezeigt. □

Bemerkung 5.13. *Der Satz von Newton-Kantorovich unterscheidet sich in einigen Punkten wesentlich von Satz 2.10 über das eindimensionale Newton-Verfahren. Der wesentliche Unterschied ist die Regularität von f, welches beim Newton-Kantorovich nur über eine Lipschitz-stetige Jacobi-Matrix anstelle von zweimal stetiger Differenzierbarkeit verfügen muss. Ferner muss die Existenz einer Nullstelle nicht vorausgesetzt werden, sie folgt beim Newton-Kantorovich als Ergebnis des Satzes.*

Daher nun eine der Hauptanwendungen des vorherigen Satzes 5.12: das folgende lokale Konvergenzresultat:

Korollar 5.14. *Es sei $D \subset \mathbb{R}^n$ offen und $f : D \subset \mathbb{R}^n \to \mathbb{R}^n$ zweimal stetig-differenzierbar. Wir nehmen an, dass $z \in D$ eine Nullstelle mit regulärer Jacobi-Matrix $f'(z)$ ist. Dann ist das Newton-Verfahren lokal konvergent, d.h. es existiert eine Umgebung B um z, so dass das Newton-Verfahren für alle Startwerte $x^0 \in B$ konvergiert.*

BEWEIS: Den Beweis stellen wir als Übungsaufgabe. □

Korollar 5.15. *Korollar 5.14 stellt den Zusammenhang zu dem ein-dimensionalen Resultat her: Dazu sei $z \in G$ eine Nullstelle von f und $\|\cdot\|_\infty$ die Maximumsnorm. Damit können die Konstanten*

$$m = \frac{1}{\beta}, \quad M = \gamma$$

bestimmt werden. Dann kann gilt zusätzlich zur a priori Fehlerabschätzung, die a posteriori Schranke:

$$\|x^k - z\|_\infty \leq \frac{1}{m}\|f(x^k)\|_\infty \leq \frac{M}{2m}\|x^k - x^{k-1}\|_\infty^2, \quad k = 1, 2, \ldots.$$

Beispiel 5.16. *Newton im $\mathbb{R}^n$ Wir suchen die Nullstelle der Funktion*

$$f(x_1, x_2) = \begin{pmatrix} 1 - x^2 - y^2 \\ (x - 2y)/(1/2 + y) \end{pmatrix},$$

mit der Jacobi-Matrix

$$f'(x) = \begin{pmatrix} -2x & -2y \\ \frac{2}{1+2y} & -\frac{4+4x}{(1+2y)^2} \end{pmatrix}.$$

Die Nullstellen von f ist gegeben durch:

$$x \approx \pm(0.894427, 0.447214).$$

Wir starten die Iteration mit $x^0 = (1, 1)^T$ und erhalten die Iterierten:

$$x^1 \approx \begin{pmatrix} 1.14286 \\ 0.357143 \end{pmatrix}, \quad x^2 \approx \begin{pmatrix} 0.92659 \\ 0.442063 \end{pmatrix}, \quad x^3 \approx \begin{pmatrix} 0.894935 \\ 0.447349 \end{pmatrix}, \quad x^4 \approx \begin{pmatrix} 0.894427 \\ 0.447214 \end{pmatrix}.$$

Nach nur vier Iterationen sind die ersten sechs Stellen exakt.

5.2.3 Vereinfachtes und gedämpftes Newton-Verfahren

Analog zum ein-dimensionalen Fall können wir auch im $\mathbb{R}^n$ das vereinfachte und das gedämpfte Newton-Verfahren formulieren. Diese Verallgemeinerungen dienen im Wesentlichen wieder zwei Zwecken:

- Durch festhalten der Inversen $f'(c)^{-1}$ in einem Punkt $c \in \mathbb{R}^n$ kann in jedem folgenden Newton-Schritt ein lineares Gleichungssystem mit derselben Matrix gelöst werden.
- Durch Einfügen eines Dämpfungsparameters kann der Einzugsbereich des Newton-Verfahrens vergrößert werden (Globalisierung).

Vereinfachtes Newton-Verfahren Im höher-dimensionalen Newton-Verfahren liegt der Hauptaufwand in der Lösung des linearen Gleichungssystems in jedem Newton-Schritt. Daher macht sich hier die Verwendung eines vereinfachten Newton-Verfahrens deutlicher bemerkbar als im eindimensionalen Fall. Wir wählen ein $c \in \mathbb{R}^n$ möglichst nahe an der gesuchten Nullstelle $c \approx z$ und iterieren:

$$f'(c)(x^{k+1} - x^k) = -f(x^k), \quad k = 1, 2, \ldots$$

Dabei kann z.B. $c = x^0$ gewählt werden. Das "einfrieren" der Inversen hat zwei Vorteile: zunächst muss diese seltener berechnet werden. Dies kann bei komplizierten Ableitungen, die möglicherweise nur numerisch bestimmt werden können ein wesentlicher Vorteil sein. Noch wichtiger ist jedoch, dass die Matrix $f'(c)$ nur einmal zerlegt werden muss, etwa $f'(c) = LR$. Diese Zerlegung kann dann in jedem Schritt zum Lösen des linearen Gleichungssystems genutzt werden. Da ein Erstellen der Zerlegung $O(n^3)$ und das anschließende Lösen nur $O(n^2)$ Operationen benötigt, kann selbst dann ein enormer Effizienzgewinn erreicht werden, falls das vereinfachte Newton-Verfahren weit mehr Schritte benötigt. Das vereinfachte Newton-Verfahren kann nur noch linear konvergieren und die Konvergenz kann einfach durch Anwenden des Banachschen Fixpunktsatzes auf die Iterationsvorschrift

$$x^{k+1} = x^k - f'(c)^{-1} f(x^k)$$

gesichert werden.

Bei einem konkreten Verfahren mit x^0 und fester Matrix A kann während der Berechnung mit Hilfe des Banachschen Fixpunktsatzes geklärt werden, ob Konvergenz vorliegt. Hierzu kann der Kontraktionsfaktor q a posteriori berechnet werden:

$$q_k = \frac{\|g(x^k) - g(x^{k-1})\|}{\|x^k - x^{k-1}\|} = \frac{\|x^{k+1} - x^k\|}{\|x^k - x^{k-1}\|}, \quad k = 1, 2, \ldots.$$

Die zugrundeliegende Norm kann die Maximums-Norm oder l_1-Norm sein (siehe zu den Vektornormen auch Kapitel 4). Falls der Schätzfaktor $q_k \ll 1$, dann ist das vereinfachte Newton-Verfahren sehr wahrscheinlich konvergent. Falls $q_k \approx 1$ bzw. $q_k \geq 1$, dann liegt wahrscheinlich keine Konvergenz vor.

Hier hat man nun folgende Möglichkeiten:

- Wahl eines besseren Startwertes x^0,
- Bessere Wahl von $f'(c)$ als Approximation für $f'(x)$,
- Oder ein vereinfachtes Newton-Verfahren in dem $f'(c)$ z.B. alle 4 Schritte neu aufgebaut wird, um so ab und zu die Matrix anzupassen.

Die letzte Methodik wird sehr häufig bei nicht-linearen Problemen im Rahmen von partiellen Differentialgleichungen verwendet. Denn hier ist die kluge Wahl eines Startwertes oftmals mit sehr viel Aufwand verbunden, weshalb die Möglichkeit 1 ausgeschlossen wird. Daher ist die Wahl von x^0 oftmals sehr schlecht, weshalb aber auch $A := f'(x^0)$ eine sehr schlechte Approximation von $f'(z)$ ist. Deshalb sollte man nach z.B. 3 Schritten $A := f'(x^3)$ wählen, um eine bessere Approximation der Jacobi-Matrix zu erhalten.

Gedämpftes Newton-Verfahren Die Vergrößerung des Konvergenzbereiches (Globalisierung) kann mit Hilfe eines Dämpfungsparameters erreicht werden. Hierzu wird die Newton-Iteration abgewandelt:

$$x^{k+1} = x^k - \omega_k f'(x^k)^{-1} f(x^k).$$

Der Dämpfungsparameter ω_k ist dabei im Intervall $(0, 1]$ zu wählen. Dieser wird am Anfang klein gewählt, um das Konvergenzgebiet zu vergrößern. Allerdings konvergiert das Verfahren dann nur mit linearer Ordnung. Quadratische Konvergenz wird nur dann erreicht, falls für $k \to \infty$ auch $\omega_k \to 1$ gilt. Der folgenden Satz gibt ein konstruktives Kriterium, den Dämpfungsparameter ω_k a posteriori bestimmen zu können.

Satz 5.17 (Gedämpftes Newton-Verfahren)**.** *Unter den Voraussetzungen von Satz 5.12 erzeugt die gedämpfte Newton-Iteration (siehe 2.4.4) mit*

$$\omega_k := \min\{1, \frac{1}{\alpha_k \beta \gamma}\}, \quad \alpha_k := \|f'(x^k)^{-1} f(x^k)\|$$

eine Folge $(x^k)_{k\in\mathbb{N}}$, für die nach t_* Schritten

$$q_* := \alpha_{k_*} \beta \gamma < \frac{1}{2}$$

erfüllt ist. Ab dann konvergiert x^k quadratisch und es gilt die a priori Fehlerabschätzung

$$\|x^k - z\| \leq \frac{\alpha}{1 - q_*} q_*^{2^k - 1}, \quad k \geq k_*.$$

Beweis: In [9]. □

In der praktischen Anwendung wird der Dämpfungsparameter oft über die sogenannte *Line-Search*-Strategie bestimmt:

Algorithmus 5.18 (Newton-Verfahren mit Line-Search)**.** *Gegeben sei ein Startwert $x^0 \in \mathbb{R}^n$ sowie $\sigma \in (0, 1)$. Iteriere für $k = 0, 1, \ldots$*

(i) Löse $f'(x^k) w^k = -f(x^k)$

(ii) Starte mit $\omega_0 = 1$ und iteriere für $l = 0, 1, \ldots$

$$x^{k+1} = x^k + \omega_l w^k, \quad \omega_{l+1} = \sigma \omega_l,$$

solange bis $|f(x^{k+1})| < |f(x^k)|$.

Der Line-Search Algorithmus versucht zunächst ohne Dämpfung zu iterieren. Besitzt die neue Approximation allerdings ein größeres Residuum $|f(x^{k+1})| > |f(x^k)|$, so wird der Dämpfungsparameter schrittweise reduziert. Auf diese Weise wird monotone Konvergenz im Residuum erzwungen.

5.3 Iterative Lösungsverfahren für lineare Gleichungssysteme

Als zweite Hauptanwendung des Banachschen Fixpunktsatzes besprechen wir in diesem Kapitel die iterative Lösung linearer Gleichungssysteme. Die in Kapitel 4 kennengelernten Zerlegungsverfahren (also LR, Cholesky sowie QR-Zerlegung) haben alle den Nachteil kubischer Laufzeit $O(n^3)$. Gerade bei sehr großen Problemen wächst der Aufwand so schnell an, dass die Lösung in sinnvoller Zeit nicht zu erreichen ist, siehe Tabelle 4.1. Neben der Laufzeit spielt auch der Speicheraufwand eine Rolle. Zur Speicherung einer voll besetzten Matrix $A \in \mathbb{R}^{n\times n}$ mit $n = 1\,000\,000$ sind bei doppelter Genauigkeit etwa 7 Terabyte (!!!) Speicher notwendig. Dies übersteigt jeden noch so modernen Computer. Die linearen Gleichungssysteme die aus den meisten Anwendungen resultieren (etwa bei der Diskretisierung von Differentialgleichungen) sind sehr dünn besetzt, d.h., in jeder Zeile stehen nur einige wenige Einträge. Eine dünn besetzte Matrix mit $n = 1\,000\,000$ aber nur 100 Einträgen pro Zeile benötigt zur Speicherung nur etwa 750 MB und passt in jeden Laptop. In Abschnitt 4.2.5 haben wir Sortierverfahren kennengelernt, um dieses dünne Besetzungsmuster auch für eine LR-Zerlegung nutzbar zu machen. Im Allgemeinen können die Matrizen L und R aber voll besetzt sein und somit den zur Verfügung stehenden Speicher wieder bei weitem übersteigen.

Ein weiterer Nachteil der Zerlegungsverfahren sind numerische Stabilitätsprobleme. Durch Rundungsfehler beim Zerlegungsprozess gilt für die LR-Zerlegung üblicherweise nur $A \neq \tilde{L}\tilde{R}$. D.h., obwohl die LR-Zerlegung eine direkte Methode darstellt, kann das Gleichungssystem nicht exakt gelöst werden. Mit der Nachiteration haben wir in Abschnitt 4.3 eine Methode kennengelernt, um diesen Fehlereinfluss durch sukzessive Iteration zu verringern. Mit der gestörten LR-Zerlegung $A \approx \tilde{L}\tilde{R}$ haben wir die Iteration

$$x^{k+1} = x^k + \tilde{R}^{-1}\tilde{L}^{-1}(b - Ax^k), \quad k = 0, 1, 2, \ldots$$

definiert. Obwohl $\tilde{L}$ und $\tilde{R}$ nicht exakt sind, konvergiert diese Iteration (falls das Residuum $d^k := b - Ax^k$ exakt berechnet werden kann), siehe Satz 4.45.

In diesem Abschnitt werden wir auf der Basis der Nachiteration eine eigene Verfahrensklasse zur iterativen Lösung großer Gleichungssysteme entwickeln. Dazu sei $C \approx A^{-1}$ eine Approximation an die Inverse (etwa $C := \tilde{R}^{-1}\tilde{L}^{-1}$). Dann definieren wir:

Definition 5.19 (Fixpunktverfahren zum Lösen linearer Gleichungssysteme)**.** *Es sei $A \in \mathbb{R}^{n\times n}$ sowie $b \in \mathbb{R}^n$ und $C \in \mathbb{R}^{n\times n}$. Für einen beliebigen Startwert $x^0 \in \mathbb{R}^n$ iteriere für $k = 1, 2, \ldots$*

$$x^k = x^{k-1} + C(b - Ax^{k-1}). \tag{5.11}$$

Alternativ führen wir die Bezeichnungen $B := I - CA$ und $c := Cb$ ein. Dann gilt:

$$x^k = Bx^{k-1} + c.$$

Aufgrund der Konstruktion kann man sich einfach klarmachen, dass durch die Vorschrift $g(x) = Bx + c = x + C(b - Ax)$ wirklich eine Fixpunktiteration mit der Lösung von $Ax = b$

als Fixpunkt gegeben ist. Die Konvergenz von allgemeinen (auch linearen) Fixpunktiterationen kann leicht mit dem Banachschen Fixpunktsatz untersucht werden. Hierzu ist die Kontraktionseigenschaft nachzuweisen:

$$\|g(x) - g(y)\| \leq \|B\| \, \|x - y\|.$$

Es ergibt sich jedoch das Dilemma, das je nach verwendeter Matrixnorm $\|\cdot\|$ unterschiedliche Konvergenzresultate vorhergesagt werden. Für eine Matrix $B \in \mathbb{R}^{n\times n}$ kann etwa $\|B\|_2 < 1$ aber $\|B\|_\infty > 1$ gelten. Diese Beobachtung steht im Widerspruch zur Normäquivalenz im $\mathbb{R}^n$ welche insbesondere eine Äquivalenz von Konvergenzausdrücken besagt. Um uns in der Analyse von konkreten Matrixnormen zu befreien beweisen wir zunächst einen Hilfsatz:

Hilfsatz 5.20 (Matrixnorm und Spektralradius)**.** *Zu jeder beliebigen Matrix $B \in \mathbb{R}^{n\times n}$ und zu jedem $\epsilon > 0$ existiert eine natürliche Matrixnorm $\|\cdot\|_\epsilon$, so dass gilt:*

$$\operatorname{spr}(B) \leq \|B\|_\epsilon \leq \operatorname{spr}(B) + \epsilon.$$

Beweis: Für den allgemeinen Fall verweisen wir auf [9]. Hier machen wir uns nur klar, dass die Aussage für symmetrische Matrizen gilt. Denn in diesem Fall, siehe Satz 4.14 gilt sogar $\operatorname{spr}(B) = \|B\|_2$. □

Mit diesem Hilfsatz zeigen wir das fundamentale Resultat über allgemeine lineare Fixpunktiterationen:

Satz 5.21 (Fixpunktverfahren zum Lösen linearer Gleichungssysteme)**.** *Die Iteration (5.11) konvergiert für jeden Startwert $x^0 \in \mathbb{R}^n$ genau dann gegen die Lösung $x \in \mathbb{R}^n$ von $Ax = b$ falls $\rho := \operatorname{spr}(B) < 1$. Dann gilt das asymptotische Konvergenzverhalten*

$$\limsup_{k\to\infty} \left(\frac{\|x^k - x\|}{\|x^0 - x\|} \right)^{1/k} = \operatorname{spr}(B).$$

Beweis: Wir führen den Beweis in drei Schritten. Zunächst *(i)* gehen wir davon aus, dass $\operatorname{spr}(B) < 1$ und zeigen, dass $x^k \to x$. In Schritt *(ii)* zeigen wir die Rückrichtung und schließlich in Schritt *(iii)* die Konvergenzaussage.

(0) Zunächst weisen wir nach, dass die Iteration überhaupt eine Fixpunktiteration ist. Für die Lösung $x \in \mathbb{R}^n$ von $Ax = b$ gilt:

$$Bx + c = (I - CA)x + Cb = x - C \underbrace{(Ax - b)}_{=0} = x.$$

Eine Konvergenzaussage erhalten wir jetzt sofort über den Fixpunktsatz von Banach.

Weiter definieren den Fehler $e^k := x^k - x$ und erhalten bei Ausnutzen der Fixpunkteigenschaft $x = Bx + c$ die Iterationsvorschrift:

$$e^k = x^k - x = Bx^{k-1} + c - (Bx + c) = Be^{k-1}.$$

Entsprechend gilt:

$$e^k = B^k e^0 = B^k(x^0 - x). \tag{5.12}$$

(i) Hilfsatz 5.20 besagt, dass zu jedem $\epsilon > 0$ eine natürlichen Matrixnorm $\|\cdot\|_\epsilon$ existiert mit

$$\mathrm{spr}(B) \leq \|B\|_\epsilon \leq \mathrm{spr}(B) + \epsilon.$$

Es sei nach Voraussetzung $spr(B) < 1$, dann existiert ein $\epsilon > 0$ mit

$$\|B\|_\epsilon \leq \mathrm{spr}(B) + \epsilon < 1,$$

und aus (5.12) erhalten wir sofort bei $k \to \infty$ in der entsprechenden induzierten Vektornorm $\|\cdot\|_\epsilon$:

$$\|e^k\|_\epsilon = \|B^k e^0\|_\epsilon \leq \|B^k\|_\epsilon \|e^0\|_\epsilon \leq \|B\|_\epsilon^k \|e^0\|_\epsilon \quad \to \quad 0.$$

Da im $\mathbb{R}^n$ alle Normen äquivalent sind, konvergiert also $x^k \to x$ für $k \to \infty$.

(ii) Es sei nach Voraussetzung die Iteration konvergent. Hieraus folgt (für beliebige Startwerte) mit der Wahl $x^0 = w + x$ mit einem Eigenvektor $w \in \mathbb{R}^n \setminus \{0\}$ zum betragsmäßig größten Eigenwert λ von B mit (5.12):

$$\lambda^k w = B^k w = B^k e^0 = e^k \quad \to 0 \quad (k \to \infty).$$

Dies bedeutet notwendig $|\lambda| < 1$ für $\lambda \in \sigma(B)$, d.h. $spr(B) < 1$.

(iii) Fehlerabschätzung: Aufgrund der Äquivalenz aller Normen existieren Konstanten m, M mit $m \leq M$ so dass:

$$m\|x\| \leq \|x\|_\epsilon \leq M\|x\|, \quad x \in \mathbb{R}^n.$$

Damit gilt

$$\|e^k\| \leq \frac{1}{m}\|e^k\|_\epsilon = \frac{1}{m}\|B^k e^0\|_\epsilon \leq \frac{1}{m}\|B\|_\epsilon^k \|e^0\|_\epsilon \leq \frac{M}{m}(\mathrm{spr}(B) + \epsilon)^k \|e^0\|.$$

Wegen

$$\left(\frac{M}{m}\right)^{1/k} \quad \to 1, \quad (k \to \infty)$$

folgt damit

$$\limsup_{k\to\infty} \left(\frac{\|e^k\|}{\|e^0\|}\right)^{1/k} \leq \mathrm{spr}(B) + \epsilon.$$

Da $\epsilon > 0$ beliebig klein gewählt wird, folgt hiermit die Behauptung. □

5.3.1 Konstruktion von Fixpunktverfahren

Satz 5.21 legt das theoretische Fundament für allgemeine Fixpunkt-Iterationen. Für den Spektralradius $\rho := \operatorname{spr}(B) = \operatorname{spr}(I - CA)$ muss gelten $\rho < 1$. Ziel dieses Abschnittes ist die Konstruktion von Iterationsmatrizen C, welche

- möglichst Nahe an der Inversen $C \approx A^{-1}$ liegen, damit $\operatorname{spr}(I - CA) \ll 1$,
- eine möglichst einfache Berechnung der Iteration $x^k = Bx^{k-1} + c$ ermöglichen.

Die erste Forderung ist einleuchtend und $C = A^{-1}$ stellt in diesem Sinne die optimale Matrix dar. Die zweite Bedingung beschreibt den Aufwand zum Durchführen der Fixpunktiteration. Wählen wir etwa $C = \tilde{R}^{-1}\tilde{L}^{-1}$ so bedeutet jeder Schritt ein Vorwärts- und ein Rückwärtseinsetzen, d.h. einen Aufwand der Größenordnung $O(n^2)$. Hinzu kommen die $O(n^3)$ Operationen zum einmaligen Erstellen der Zerlegung. Die Wahl $C = I$ erlaubt eine sehr effiziente Iteration mit $O(n)$ Operationen in jedem Schritt und Bedarf keines zusätzlichen Aufwands zum Erstellen der Iterationsmatrix C. Die Einheitsmatrix I ist jedoch eine sehr schlechte Approximation von A^{-1}. Die beiden Forderungen sind Maximalforderungen und nicht gleichzeitig zu erreichen.

Zur Konstruktion einfacher iterativer Verfahren spalten wir die Matrix A additiv auf zu $A = L + D + R$, mit

$$A = \underbrace{\begin{pmatrix} 0 & & \dots & 0 \\ a_{21} & \ddots & & \\ \vdots & \ddots & \ddots & \\ a_{n1} & \dots & a_{n,n-1} & 0 \end{pmatrix}}_{=:L} + \underbrace{\begin{pmatrix} a_{11} & & \dots & 0 \\ & \ddots & & \\ & & \ddots & \\ 0 & \dots & & a_{nn} \end{pmatrix}}_{=:D} + \underbrace{\begin{pmatrix} 0 & a_{21} & \dots & a_{n1} \\ & \ddots & \ddots & \vdots \\ & & \ddots & a_{n-1,n} \\ 0 & \dots & & 0 \end{pmatrix}}_{=:R}.$$

Diese additive Zerlegung ist nicht mit der multiplikativen LR-Zerlegung zu verwechseln! Ist die Matrix A bekannt, so sind auch die additiven Bestandteile L, D, R unmittelbar verfügbar.

Wir definieren die zwei wichtigsten Iterationsverfahren:

Definition 5.22 (Jacobi-Verfahren)**.** *Zur Lösung von $Ax = b$ mit $A = L + D + R$ sei $x^0 \in \mathbb{R}^n$ ein beliebiger Startwert. Iteriere für $k = 1, 2, \dots$:*

$$x^k = x^{k-1} + D^{-1}(b - Ax^{k-1}),$$

bzw. mit der Jacobi-Iterationsmatrix $J := -D^{-1}(L + R)$

$$x^k = Jx^{k-1} + D^{-1}b.$$

Definition 5.23 (Gauß-Seidel-Verfahren)**.** *Zur Lösung von $Ax = b$ mit $A = L + D + R$ sei $x^0 \in \mathbb{R}^n$ ein beliebiger Startwert. Iteriere für $k = 1, 2, \ldots$:*

$$x^k = x^{k-1} + (D + L)^{-1}(b - Ax^{k-1}),$$

bzw. mit der Gauß-Seidel-Iterationsmatrix $H := -(D + L)^{-1}R$

$$x^k = Hx^{k-1} + (D + L)^{-1}b.$$

Diese beiden Fixpunktverfahren sind einfach, aber dennoch sehr gebräuchlich. Zum Aufstellen der Iterationsmatrix C sind keine Rechenoperationen notwendig. Es gilt:

Satz 5.24 (Durchführung des Jacobi- und Gauß-Seidel-Verfahrens)**.** *Ein Schritt des Jacobi- bzw. Gauß-Seidel-Verfahrens ist jeweils in $n^2 + O(n)$ Operationen durchführbar. Für jeden Schritt des Jacobi-Verfahren gilt die Index-Schreibweise*

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

für das Gauß-Seidel-Verfahren gilt die Vorschrift:

$$x_i^k = \frac{1}{a_{ii}} \left(b_i - \sum_{j<i} a_{ij} x_j^k - \sum_{j>i} a_{ij} x_j^{k-1} \right), \quad i = 1, \ldots, n.$$

Beweis: Übung! □

Jeder Schritt dieser Verfahren benötigt mit $O(n^2)$ größenordnungsmäßig genauso viele Operationen wie die Nachiteration mit der LR-Zerlegung. Bei Jacobi- und Gauß-Seidel sind jedoch weit schlechtere Konvergenzraten zu erwarten (wenn die Verfahren überhaupt konvergieren, dieser Nachweis steht hier noch aus!). Der Vorteil der einfachen Iterationsverfahren zeigt sich erst bei dünn besetzten Matrizen. Hat eine Matrix $A \in \mathbb{R}^{n\times n}$ nur $O(n)$ Einträge, so benötigt jeder Iterationsschritt nur $O(n)$ Operationen.

5.3.2 Konvergenzkriterium für Jacobi- und Gauß-Seidel-Iteration

Zur Untersuchung der Konvergenz muss gemäß Satz 5.21 der Spektralradius der Iterationsmatrizen $J = -D^{-1}(L+R)$ sowie $H := -(D+L)^{-1}R$ untersucht werden. Im Einzelfall ist diese Untersuchung nicht ohne weiteres möglich und einer Matrix A kann der entsprechende Spektralradius nur schwer angesehen werden. Daher leiten wir in diesem Abschnitt einfach überprüfbare Kriterien her, um eine Aussage über die Konvergenz der beiden Verfahren treffen zu können. Zunächst gilt:

Satz 5.25 (Starkes Zeilensummenkriterium)**.** *Falls die Zeilensummen der Matrix* $A \in \mathbb{R}^{n\times n}$ *der* strikte Diagonaldominanz *genügt*

$$\sum_{k=1,k\neq j}^{n} |a_{jk}| < |a_{jj}|, \quad j = 1,\dots,n,$$

so gilt $spr(J) < 1$ *bzw.* $spr(H) < 1$*, sprich Jacobi- und Gauß-Seidel-Verfahren konvergieren.*

Beweis: Wir beweisen den Satz für beide Verfahren gleichzeitig. Es seien $\lambda \in \sigma(J)$ bzw. $\mu \in \sigma(H)$ und v bzw. w die zugehörigen Eigenvektoren. Das bedeutet für das Jacobi-Verfahren

$$\lambda v = Jv = D^{-1}(L+R)v,$$

und für das Gauß-Seidel-Verfahren

$$\mu w = Hw = -(D+L)^{-1}Rw \quad \Leftrightarrow \quad \mu w = -D^{-1}(\mu L + R)w.$$

Falls $\|v\|_\infty = \|w\|_\infty = 1$, dann folgt hieraus für das Jacobi-Verfahren

$$|\lambda| \leq \|D^{-1}(L+R)\|_\infty \|v\|_\infty = \|D^{-1}(L+R)\|_\infty \leq \max_{j=1,\dots,n} \left\{ \frac{1}{|a_{jj}|} \sum_{k=1,k\neq j}^{n} |a_{jk}| \right\} < 1$$

und für das Gauß-Seidel-Verfahren

$$|\mu| \leq \|D^{-1}(\mu L+R)\|_\infty \|w\|_\infty = \|D^{-1}(\mu L+R)\|_\infty \leq \max_{1\leq j\leq n} \left\{ \frac{1}{|a_{jj}|} [\sum_{k<j} |\mu||a_{jk}| + \sum_{k>j} |a_{jk}|] \right\}.$$

Hier muss jetzt noch $|\mu| < 1$ gezeigt werden. Für $|\mu| \geq 1$ ergäbe sich der Widerspruch

$$|\mu| \leq |\mu| \|D^{-1}(L+R)\|_\infty < |\mu|$$

woraus notwendig $|\mu| < 1$ folgt. □

Dieses Kriterium ist einfach zu überprüfen und erlaubt sofort eine Einschätzung, ob Jacobi- und Gauß-Seidel-Verfahren bei einer gegebenen Matrix konvergieren. Es zeigt sich jedoch, dass die strikte Diagonaldominanz eine zu starke Forderung darstellt: die schon bekannte Modellmatrix der Form

$$A = \begin{pmatrix} B & -I & & \\ -I & B & -I & \\ & -I & B & -I \\ & & -I & B \end{pmatrix}, \quad B = \begin{pmatrix} 4 & -1 & & \\ -1 & 4 & -1 & \\ & -1 & 4 & -1 \\ & & -1 & 4 \end{pmatrix}, \quad I = \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{pmatrix},$$

ist nur Diagonaldominant (siehe Definition 4.31), jedoch nicht strikt Diagonaldominant. (Es zeigt sich aber, dass sowohl Jacobi- als auch Gauß-Seidel-Verfahren dennoch konvergieren). Eine Abschwächung der Konvergenzaussage erhalten wir mit Hilfe der folgenden Definition:

Definition 5.26 (Irreduzibel)**.** *Eine Matrix $A \in \mathbb{R}^{n\times n}$ heißt irreduzibel, wenn es keine Permutationsmatrix P gibt, so dass die Matrix A durch Spalten- und Zeilen in eine Block-Dreiecksmatrix zerfällt:*

$$PAP^T = \begin{pmatrix} \tilde{A}_{11} & 0 \\ \tilde{A}_{21} & \tilde{A}_{22} \end{pmatrix}$$

mit den Matrizen $\tilde{A}_{11} \in \mathbb{R}^{p\times p}$, $\tilde{A}_{22} \in \mathbb{R}^{q\times q}$, $\tilde{A}_{21} \in \mathbb{R}^{q\times p}$, mit $p, q > 0, p + q = n$.

Ob eine gegebene Matrix $A \in \mathbb{R}^{n\times n}$ irreduzibel ist lässt sich nicht unmittelbar bestimmen. Oft hilft das folgende äquivalente Kriterium, welches einfacher zu überprüfen ist:

Satz 5.27 (Irreduzibel)**.** *Eine Matrix $A \in \mathbb{R}^{n\times n}$ ist genau dann irreduzibel, falls der zugehörige* gerichtete Graph

$$\mathcal{G}(A) = \{\textit{Knoten: } \{1, 2, \ldots, n\}, \textit{Kannten: } \{i, j\} \textit{ falls } a_{ij} \neq 0\}$$

zusammenhängend ist. Dies heißt: zu zwei beliebigen Knoten *i und j existiert ein Pfad $\{i, i_1\} =: \{i_0, i_1\}, \{i_1, i_2\}, \ldots, \{i_{m-1}, i_m\} := \{i_{m-1}, j\}$ mit $\{i_{k-1}, i_k\} \in \mathcal{G}(A)$.*

Für den Beweis verweisen wir auf [9]. Nach dieser alternativen Charakterisierung bedeutet die Irreduzibilität, dass zu je zwei Indizes i, j ein Pfad $i = i_0, i_1, \ldots, i_m =: j$ besteht, so dass $a_{i_{k-1},i_k} \neq 0$ ist. Anschaulich entspricht dies einem abwechselnden Springen in Zeilen und Spalten der Matrix A, wobei nur Einträge ungleich Null getroffen werden dürfen.

Nun gilt:

Satz 5.28 (Schwaches Zeilensummenkriterium)**.** *Die Matrix $A \in \mathbb{R}^{n\times n}$ sei irreduzibel und es gelte das* schwache Zeilensummenkriterium*, d.h., die Matrix A sei diagonaldominant:*

$$\sum_{k=1,k\neq j}^{n} |a_{jk}| \leq |a_{jj}|, \quad j = 1, \ldots, n,$$

und in mindestens einer Zeile $r \in \{1, \ldots, n\}$ gelte strikte Diagonaldominanz:

$$\sum_{k=1,k\neq r}^{n} |a_{rk}| < |a_{rr}|.$$

Dann ist A regulär und es gilt $spr(J) < 1$ bzw. $spr(H) < 1$. D.h., Jacobi- und Gauß-Seidel-Verfahren konvergieren.

Beweis: *(i)* Durchführbarkeit der Verfahren:
Aufgrund der Irreduzibilität von A gilt notwendig

$$\sum_{k=1}^{n} |a_{jk}| > 0, \quad j = 1, \ldots, n.$$

Wegen der vorausgesetzten Diagonaldominanz folgt dann hieraus $a_{jj} \neq 0$ für $j = 1, 2, \ldots, n$.
(ii) Zeige $spr(J) \leq 1$ und $spr(H) \leq 1$:
Diese Aussage erhalten wir entsprechend zum Vorgehen im Beweis zu Satz 5.25. Es bleibt zu zeigen, dass kein Eigenwert den Betrag Eins hat.

(iii) Nachweis, dass $|\lambda| < 1$:
Angenommen, es gebe einen Eigenwert $\lambda \in \sigma(J)$ mit $|\lambda| = 1$. Es sei $v \in \mathbb{C}^n$ der zugehörige normierte Eigenvektor mit $\|v\|_\infty = 1$. Insbesondere gelte $|v_s| = \|v\|_\infty = 1$ für ein $s \in \{1, \ldots, n\}$. Dann erhalten wir aufgrund der Struktur der Iterationsmatrix (hier nun explizit für Jacobi)

$$J = -D^{-1}(L+R) = \begin{pmatrix} 0 & \frac{-a_{12}}{a_{11}} & \frac{-a_{13}}{a_{11}} & \cdots & \frac{-a_{1n}}{a_{11}} \\ \frac{-a_{21}}{a_{22}} & 0 & \frac{-a_{23}}{a_{22}} & \cdots & \frac{-a_{2n}}{a_{22}} \\ \vdots & & \ddots & & \vdots \\ \frac{-a_{n1}}{a_{nn}} & \cdots & \cdots & & 0 \end{pmatrix}$$

die folgende Abschätzung:

$$|v_i| = \underbrace{|\lambda|}_{=1} |v_i| \leq \sum_{k \neq i} \frac{|a_{ik}|}{|a_{ii}|} |v_k| \leq \sum_{k \neq i} \frac{|a_{ik}|}{|a_{ii}|} \leq 1, \quad i = 1, 2, \ldots, n, \tag{5.13}$$

wobei wir die Struktur von J in der ersten Ungleichung, $|v_i| = \|v\|_\infty$ (in der zweiten) und die schwache Diagonaldominanz in der letzten Abschätzung verwendet haben.

Aufgrund der Irreduzibilität gibt es zu je zwei Indizes s, r stets eine Kette von Indizes $i_1, \ldots, i_m \in \{1, \ldots, n\}$ (siehe Beweis zur Irreduzibilität), so dass

$$a_{s,i_1} \neq 0, a_{i_1,i_2} \neq 0, \ldots, a_{i_m,r} \neq 0.$$

Durch mehrfache Anwendung von (5.13) folgt der Widerspruch (nämlich, dass $|\lambda| = 1$):

$$\begin{aligned}
|v_r| = |\lambda v_r| &\leq \frac{1}{|a_{rr}|} \sum_{k \neq r} |a_{rk}| \, \|v\|_\infty < \|v\|_\infty \quad \text{(strikte DD in einer Zeile)}, \\
|v_{i_m}| = |\lambda v_{i_m}| &\leq \frac{1}{|a_{i_m,i_m}|} \left[\sum_{k \neq i_m, r} |a_{i_m,k}| \, \|v\|_\infty + |a_{i_m,r}| \, |v_r| \right] < \|v\|_\infty, \\
&\vdots \\
|v_{i_1}| = |\lambda v_{i_1}| &\leq \frac{1}{|a_{i_1,i_1}|} \left[\sum_{k \neq i_1, i_1} |a_{i_1,k}| \, \|v\|_\infty + |a_{i_1,i_2}| \, |v_{i_2}| \right] < \|v\|_\infty, \\
\|v\|_\infty = |\lambda v_s| &\leq \frac{1}{|a_{ss}|} \left[\sum_{k \neq s, i_1} |a_{s,k}| \, \|v\|_\infty + |a_{s,i_1}| \, |v_{i_1}| \right] < \|v\|_\infty.
\end{aligned}$$

Daher muss $spr(J) < 1$. Mit analoger Schlussfolgerung wird $spr(H) < 1$ bewiesen. Hierzu nutzt man ebenfalls die spezielle Struktur der Iterationsmatrix H sowie die umgekehrte Dreiecksungleichung, um (5.13) zu erhalten. Die restliche Argumentation erfolgt dann auf analogem Wege. □

Bemerkung 5.29. *Es ist (hoffentlich!) klar, dass starke und schwache Spaltensummenkriterien mit analoger Argumentation hergeleitet werden können.*

Der vorherige Satz sichert die Konvergenz von Jacobi- sowie Gauß-Seidel-Verfahren für die Modellmatrizen. Denn diese sind z.B. in erster und letzter Zeile strikt Diagonaldominant. Eine Abschätzung des Spektralradius $\mathrm{spr}(J)$ sowie $\mathrm{spr}(H)$ ist nicht ohne weiteres möglich. Im allgemeinen Fall konvergieren beide Verfahren sehr langsam.

Beispiel 5.30 (Jacobi- und Gauß-Seidel-Verfahren bei der Modellmatrix)**.** *Wir betrachten das lineare Gleichungssystem $Ax = b$ mit der Modellmatrix $A \in \mathbb{R}^{n\times n}$*

$$A = \begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{pmatrix}, \tag{5.14}$$

sowie der rechten Seite $b \in \mathbb{R}^n$ mit $b = (1, \ldots, 1)^T$. Zu i, j mit $i < j$ gilt

$$a_{i,i} \neq 0 \to a_{i,i+1} \neq 0 \to a_{i+1,i+2} \neq 0 \to a_{i+2,i+3} \neq 0 \to \cdots \to a_{j-1,j} \neq 0.$$

Die Matrix ist also irreduzibel, weiter diagonaldominant und in erster und letzter Zeile auch stark diagonaldominant. Jacobi und Gauß-Seidel-Verfahren konvergieren. Experimentell bestimmen wir für die Problemgröße $n = 10 \cdot 2^k$ für $k = 2, 3, \ldots$ die Anzahl der notwendigen Iterationsschritte, sowie die Rechenzeit (Core i7-Prozessor '2011) zur Approximation des Gleichungssystems mit einer Fehlertoleranz $\|x^k - x\| < 10^{-4}$:

Matrixgröße	*Jacobi*		*Gauß-Seidel*	
	Schritte	*Zeit (sec)*	*Schritte*	*Zeit (sec)*
80	*9 453*	0.02	*4 727*	0.01
160	*37 232*	0.13	*18 617*	0.06
320	*147 775*	0.92	*73 888*	0.43
640	*588 794*	7.35	*294 398*	3.55
1 280	*2 149 551*	58	*1 074 776*	29
2 560	*8 590 461*	466	*4 295 231*	233

Es zeigt sich, dass für das Gauß-Seidel-Verfahren stets halb so viele Iterationsschritte notwendig sind, wie für das Jacobi-Verfahren. Weiter steigt die Anzahl der notwendigen Iterationsschritte mit der Matrixgröße n. Bei doppelter Matrixgröße steigt die Anzahl der Iterationen etwa um den Faktor 4. Der Zeitaufwand steigt noch stärker mit einem Faktor von etwa 8, da jeder einzelne Schritt einen größeren Aufwand bedeutet. Diesen Zusammenhang zwischen Matrixeigenschaft und Konvergenzgeschwindigkeit werden wir später genauer untersuchen.

Wir merken hier noch an, dass Jacobi- und Gauß-Seidel-Verfahren effizient unter Ausnutzung der dünnen Besetzungsstruktur programmiert wurden. Dennoch steigt der Gesamtaufwand zur Approximation mit vorgegebener Genauigkeit mit dritter Ordnung in der Problemgröße n.

5.3.3 Relaxationsverfahren: das SOR-Verfahren

Das vorangehende Beispiel zeigt, dass Jacobi- sowie Gauß-Seidel-Verfahren sehr langsam konvergieren. Für die Modellmatrix $A \in \mathbb{R}^{n\times n}$ steigt die Anzahl der notwendigen Iterationsschritte (zum Erreichen einer vorgegebenen Genauigkeit) quadratisch $O(n^2)$. Obwohl jeder einzelne Schritt sehr einfach ist und äußerst effizient in $O(n)$ Operationen durchgeführt werden kann, sind diese Verfahren den direkten nicht überlegen. Oft, etwa bei Tridiagonalsystemen sind direkte Löser mit einem Aufwand von $O(n)$ sogar unschlagbar schneller.

Das SOR-Verfahren ist eine Weiterentwicklung der Gauß-Seidel-Iteration durch die Einführung eines *Relaxationsparameters* $\omega > 0$. Das i-te Element berechnet sich laut Satz 5.24 als:

$$x_i^{k,\mathrm{GS}} = \frac{1}{a_{ii}}\left(b_i - \sum_{j<i} a_{ij}x_j^{k,\mathrm{GS}} - \sum_{j>i} a_{ij}x_j^{k-1}\right), \quad i = 1,\ldots,n.$$

Zur Bestimmung der SOR-Lösung verwenden wir nicht unmittelbar diese Approximation $x_i^{k,\mathrm{GS}}$, sondern führen einen Relaxationsparameter $\omega > 0$ ein und definieren

$$x_i^{k,\mathrm{SOR}} = \omega x_i^{k,GS} + (1-\omega)x_i^{k-1}, \quad i = 1,\ldots,n$$

als einen gewichteten Mittelwert zwischen Gauß-Seidel Iteration und alter Approximation. Dieser Relaxationsparameter ω kann nun verwendet werden, um die Konvergenzeigenschaften der Iteration wesentlich zu beeinflussen. Im Fall $\omega = 1$ ergibt sich gerade das Gauß-Seidel-Verfahren. Im Fall $\omega < 1$ spricht man von *Unterrelaxation*, im Fall $\omega > 1$ von *Überrelaxation*. Das SOR-Verfahren steht für *Successive Over Relaxation*, verwendet also Relaxationsparameter $\omega > 1$. Successive (also schrittweise) bedeutet, dass die Relaxation für jeden einzelnen Index angewendet wird. Die Vorstellung zunächst die komplette Gauß-Seidel Approximation $x^{k,\mathrm{GS}}$ zu berechnen und $x^{k,\mathrm{SOR}} = \omega x^{k,\mathrm{GS}} + (1-\omega)x^{k-1}$ zu bestimmen ist falsch! Stattdessen definieren wir in Indexschreibweise:

$$x_i^{k,\mathrm{SOR}} = \omega\frac{1}{a_{ii}}\left(b_i - \sum_{j<i} a_{ij}x_j^{k,\mathrm{SOR}} - \sum_{j>i} a_{ij}x_j^{k-1}\right) + (1-\omega)x_i^{k-1}, \quad i = 1,\ldots,n.$$

In Vektorschreibweise gilt:

$$x^{k,\mathrm{SOR}} = \omega D^{-1}(b - Lx^{k,\mathrm{SOR}} - Rx^{k-1}) + (1-\omega)x^{k-1}.$$

Trennen der Terme nach $x^{k,\mathrm{SOR}}$ sowie x^{k-1} ergibt

$$(D+\omega L)x^{k,\mathrm{SOR}} = \omega b + [(1-\omega)D - \omega R]x^{k-1},$$

also die Iteration

$$x^{k,\mathrm{SOR}} = H_\omega x^{k-1} + \omega[D+\omega L]^{-1}b, \quad H_\omega := [D+\omega L]^{-1}[(1-\omega)D - \omega R].$$

Dieses Verfahren, mit der Iterationsmatrix H_ω passt wieder in das Schema der allgemeinen Fixpunktiterationen und gemäß Satz 5.21 hängt die Konvergenz des Verfahrens an $\rho_\omega := \mathrm{spr}(H_\omega) < 1$.

Die Schwierigkeit bei der Realisierung des SOR-Verfahrens ist die Bestimmung von guten Relaxationsparametern, so dass die Matrix H_ω einen möglichst kleinen Spektralradius besitzt. Es gilt die erste Abschätzung:

Satz 5.31 (Relaxationsparameter des SOR-Verfahrens)**.** *Es sei $A \in \mathbb{R}^{n\times n}$ mit regulärem Diagonalteil $D \in \mathbb{R}^{n\times n}$ Dann gilt:*

$$\operatorname{spr}(H_\omega) \geq |\omega - 1|, \quad \omega \in \mathbb{R}.$$

Für $\operatorname{spr}(H_\omega) < 1$ muss gelten $\omega \in (0, 2)$.

Beweis: Wir nutzen die Matrix-Darstellung der Iteration:

$$H_\omega = [D + \omega L]^{-1}[(1-\omega)D - \omega R] = (I + w \underbrace{D^{-1}L}_{=:\tilde{L}})^{-1} \underbrace{D^{-1}D}_{=I}[(1-\omega)I - \omega \underbrace{D^{-1}R}_{=:\tilde{R}}].$$

Die Matrizen $\tilde{L}$ sowie $\tilde{R}$ sind echte Dreiecksmatrizen mit Nullen auf der Diagonale. D.h., es gilt $\det(I+\omega\tilde{L}) = 1$ sowie $\det((1-\omega)I-\omega\tilde{R}) = (1-\omega)^n$, also Nun gilt für die Determinante von H_ω

$$\det(H_\omega) = 1 \cdot (1-\omega)^n.$$

Für die Eigenwerte λ_i von H_ω gilt folglich

$$\prod_{i=1}^{n} \lambda_i = \det(H_\omega) = (1-\omega)^n \quad \Rightarrow \quad \operatorname{spr}(H_\omega) = \max_{1\leq i\leq n} |\lambda_i| \geq \left(\prod_{i=1}^{n} |\lambda_i|\right)^{\frac{1}{n}} = |1-\omega|.$$

Die letzte Abschätzung nutzt, dass das geometrische Mittel von n Zahlen kleiner ist, als das Maximum. □

Dieser Satz liefert eine erste Abschätzung für die Wahl des Relaxationsparameters, hilft jedoch noch nicht beim Bestimmen eines Optimums. Für die wichtige Klasse von positiv definiten Matrizen erhalten wir ein sehr starkes Konvergenzresultat:

Satz 5.32 (SOR-Verfahren für positiv definite Matrizen)**.** *Es sei $A \in \mathbb{R}^{n\times n}$ eine symmetrisch positiv definite Matrix. Dann gilt:*

$$\operatorname{spr}(H_\omega) < 1 \quad \textit{für } 0 < \omega < 2.$$

SOR-Verfahren und auch Gauß-Seidel-Verfahren sind konvergent.

Beweis: Siehe [9]. □

Für die oben angegebene Modellmatrix ist die Konvergenz von Jacobi- sowie Gauß-Seidel-Iteration auch theoretisch abgesichert. Für diese Matrizen (und allgemein für die Klasse der *konsistent geordneten Matrizen*, siehe [9]) kann für die Jacobi- J und Gauß-Seidel-Iteration H_1 der folgende Zusammenhang gezeigt werden:

$$\operatorname{spr}(J)^2 = \operatorname{spr}(H_1). \tag{5.15}$$

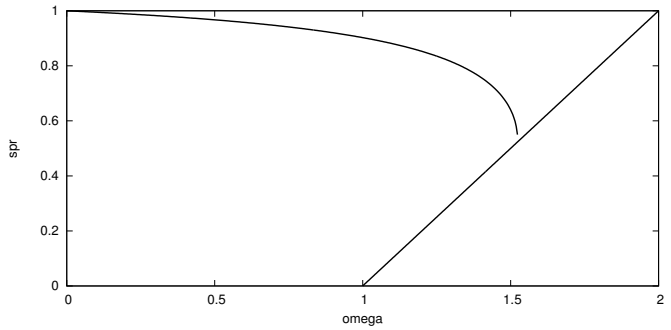

Abbildung 5.1: Bestimmung des optimalen Relaxationsparameters ω_{opt}.

Ein Schritt der Gauß-Seidel-Iteration führt zu der gleichen Fehlerreduktion wie zwei Schritte der Jacobi-Iteration. Dieses Resultat findet sich in Beispiel 5.30 exakt wieder. Weiter kann für diese Matrizen ein Zusammenhang zwischen Eigenwerten der Matrix H_ω sowie den Eigenwerten der Jacobi-Matrix J hergeleitet werden. Angenommen, es gilt $\rho_J := \text{spr}(J) < 1$. Dann gilt für den Spektralradius der SOR-Matrix:

$$\text{spr}(H_\omega) = \begin{cases} \omega - 1 & \omega \leq \omega_{\text{opt}}, \\ \frac{1}{4}(\rho_J \omega + \sqrt{\rho_J^2 \omega^2 - 4(\omega - 1)})^2 & \omega \geq \omega_{\text{opt}} \end{cases}$$

Ist der Spektralradius der Matrix J bekannt, so kann der optimale Parameter ω_{opt} kann als Schnittpunkt dieser beiden Funktionen gefunden werden, siehe Abbildung 5.1. Es gilt:

$$\omega_{\text{opt}} = \frac{2(1 - \sqrt{1 - \rho_J^2})}{\rho_J^2}. \tag{5.16}$$

Beispiel 5.33 (Modellmatrix mit SOR-Verfahren)**.** *Wir betrachten wieder die vereinfachte Modellmatrix aus Beispiel 5.30. Für die Jacobi-Matrix $J = -D^{-1}(L + R)$ gilt:*

$$J = \begin{pmatrix} 0 & \frac{1}{2} & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ & \ddots & \ddots & \ddots & \\ & & \frac{1}{2} & 0 & \frac{1}{2} \\ & & & \frac{1}{2} & 0 \end{pmatrix}.$$

Zunächst bestimmen wir die Eigenwerte λ_i und Eigenvektoren w^i für $i = 1, \ldots, n$ dieser Matrix. Hierzu machen machen wir den Ansatz:

$$w^i = (w_k^i)_{k=1,\ldots,n}, \quad w_k^i = \sin\left(\frac{\pi i k}{n+1}\right).$$

Dann gilt mit dem Additionstheoremen $\sin(x \pm y) = \sin(x)\cos(y) \pm \cos(x)\sin(y)$:

$$\begin{aligned}(Jw^i)_k &= \frac{1}{2} w_{k-1}^i + \frac{1}{2} w_{k+1}^i = \frac{1}{2}\left(\sin\left(\frac{\pi i(k-1)}{n+1}\right) + \sin\left(\frac{\pi i(k+1)}{n+1}\right)\right) \\ &= \frac{1}{2}\sin\left(\frac{\pi i k}{n+1}\right)\left(\cos\left(\frac{\pi i}{n+1}\right) + \cos\left(\frac{\pi i}{n+1}\right)\right) = w_k^i \cos\left(\frac{\pi i}{n+1}\right).\end{aligned}$$

Man beachte, dass diese Gleichung wegen $w_0^i = w_{n+1}^i = 0$ auch für die erste und letzte Zeile, d.h. für $i = 1$ sowie $i = n$ gültig ist. Es gilt $\lambda_i = \cos(\pi i/(n+1))$ und der betragsmäßig größte Eigenwert von J wird für $i = 1$ sowie $i = n$ angenommen. Hier gilt mit der Reihenentwicklung des Kosinus:

$$\lambda_{max} = \lambda_1 = \cos\left(\frac{\pi}{n+1}\right) = 1 - \frac{\pi^2}{2(n+1)^2} + O\left(\frac{1}{(n+1)^4}\right).$$

Der größte Eigenwert geht mit $n \to \infty$ quadratisch gegen 1. Hieraus bestimmen wir mit (5.16) für einige Schrittweiten aus Beispiel 5.30 die optimalen Relaxationsparameter:

n	$\lambda_{max}(J)$	ω_{opt}
320	*0.9999521084*	*1.980616162*
640	*0.9999879897*	*1.990245664*
1280	*0.9999969927*	*1.995107064*

Schließlich führen wir für diese Parameter das SOR-Verfahren mit optimalem Relaxationsparameter durch und fassen die Ergebnisse in folgender Tabelle zusammen:

Matrixgröße	*Jacobi*		*Gauß-Seidel*		*SOR*	
	Schritte	*Zeit (sec)*	*Schritte*	*Zeit (sec)*	*Schritte*	*Zeit (sec)*
320	*147 775*	0.92	*73 888*	0.43	*486*	$\ll 1$
640	*588 794*	7.35	*294 398*	3.55	*1 034*	0.02
1 280	*2 149 551*	58	*1 074 776*	29	*1937*	0.05
2 560					*4 127*	0.22
5 120			*zu aufwendig*		*8 251*	0.90
10 240					*16 500*	3.56

Die Anzahl der notwendigen Schritte steigt beim SOR-Verfahren nur linear in der Problemgröße. Dies ist im Gegensatz zum quadratischen Anstieg beim Jacobi- sowie beim Gauß-Seidel-Verfahren ein wesentlicher Fortschritt. Da der Aufwand eines Schrittes des SOR-Verfahrens mit dem von Jacobi- und Gauß-Seidel vergleichbar ist für das SOR-Verfahren zu einem Gesamtaufwand von nur $O(n^2)$ Operationen. Dieses positive Resultat gilt jedoch nur dann, wenn der optimale SOR-Parameter bekannt ist.

5.3.4 Praktische Aspekte

Wir fassen zunächst die bisher vorgestellten Verfahren zusammen:

Beispiel 5.34 (Einfache Iterationsverfahren)**.** *Es gilt in allgemeiner Darstellung*

$$x^{k+1} = x^k + C^{-1}(b - Ax^k) = \underbrace{(I - C^{-1}A)}_{=B} x^k + C^{-1}b.$$

Ausgehend von der natürlichen Aufspaltung $A = L + D + R$ sowie mit einem Relaxationsparameter ω sind Richardson-, Jacobi-, Gauß-Seidel- sowie SOR-Verfahren gegeben als als:

- *Gedämpftes Richardson Verfahren:*

$$C^{-1} = \omega I, \quad B = I - \omega A,$$

mit der Index-Schreibweise:

$$x_i^k = \omega b_i + x_i^{k-1} - \omega \sum_{j=1}^{n} a_{ij} x_j^{k-1}, \quad i = 1, \ldots, n.$$

- *Jacobi-Verfahren:*

$$C^{-1} = D^{-1}, \quad B = -D^{-1}(L + R),$$

mit der Index-Schreibweise:

$$x_i^k = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1, j \neq i}^{n} a_{ij} x_j^{k-1} \right), \quad i = 1, \ldots, n.$$

- *Gauß-Seidel-Verfahren*

$$C^{-1} = [D + L]^{-1}, \quad B = -(D + L)^{-1} R$$

mit der Index-Schreibweise:

$$x_i^k = \frac{1}{a_{ii}} \left(b_i - \sum_{j<i} a_{ij} x_j^k - \sum_{j>i} a_{ij} x_j^{k-1} \right), \quad i = 1, \ldots, n.$$

- *SOR-Verfahren (englisch. Successive Over-Relaxation):*

$$C = [D + \omega L]^{-1}, \quad B = [D + \omega L]^{-1}[(1 - \omega)D - \omega R], \quad \omega = \omega_{opt} \in (0, 2),$$

mit der Index-Schreibweise:

$$x_i^k = \omega \frac{1}{a_{ii}} \left(b_i - \sum_{j<i} a_{ij} x_j^k - \sum_{j>i} a_{ij} x_j^{k-1} \right) + (1 - \omega) x_i^{k-1}, \quad i = 1, \ldots, n.$$

Zur einfachen Durchführung der Verfahren eignet sich stets die Index-Schreibweise. Die Matrix-Form dient insbesondere der einfachen Charakterisierung sowie zum Herleiten von Konvergenzaussagen.

Als iterative Verfahren werden die Gleichungssysteme nur im (praktisch irrelevanten) Fall $n \to \infty$) wirklich gelöst. Üblicherweise muss die Iteration nach einer bestimmten Anzahl von Schritten abgebrochen werden. Als Kriterium für ein Abbrechnen kann zunächst die

asymptotische Konvergenzaussage aus Satz 5.21 herangezogen werden. Mit $\rho := \mathrm{spr}(B)$ gilt im Grenzfall:

$$\|x^k - x\| \approx \rho^k \|x^0 - x\|.$$

Und bei vorgegebener Toleranz TOL kann die notwendige Zahl an Iterationsschritten abgeschätzt werden:

$$\|x^k - x\| < TOL \quad \Rightarrow \quad k = \frac{\log\left(\frac{TOL}{\|x^0 - x\|}\right)}{\log(\rho)}.$$

Die Toleranz TOL gibt hier an, um welchen Faktor der Anfängliche Fehler $\|x^0 - x\|$ reduziert wird. Dieses Vorgehen ist in der praktischen Anwendung wenig hilfreich, da der Spektralradius ρ der Iterationsmatrix B im Allgemeinen nicht bekannt ist.

Ein alternatives allgemeines Kriterium liefert die Abschätzung aus Satz 4.44 für den Defekt $d^k := b - Ax^k$:

$$\frac{\|x^k - x\|}{\|x\|} \leq \mathrm{cond}(A) \frac{\|b - Ax^k\|}{\|b\|}.$$

Hier entsteht jedoch ein ähnliches Problem: die Konditionszahl der Matrix A ist im Allgemeinen nicht bekannt, so kann auch keine quantitativ korrekte Abschätzung hergeleitet werden. Dieser einfache Zusammenhang zwischen Defekt und Fehler kann jedoch genutzt werden um eine *relative Toleranz* zu erreichen:

Bemerkung 5.35 (Relative Toleranz)**.** *Bei der Durchführung von iterativen Lösungsverfahren werden als Abbruchskriterium oft* relative Toleranzen *eingesetzt. Die Iteration wird gestoppt, falls gilt:*

$$\|x^k - x\| \leq TOL \, \|x^0 - x\|.$$

Als praktisch durchführbares Kriterium werden die unbekannten Fehler durch die Defekte ersetzt:

$$\|b - Ax^k\| \leq TOL \, \|b - Ax^0\|.$$

5.3.5 Abstiegs & Gradientenverfahren

Die bisher kennengelernten Iterationsverfahren zur Approximation von linearen Gleichungssystemen haben alle den Nachteil, dass die Konstruktion nicht durch einen fundierten Zugang erfolgt, sondern auf Kontraktionsprinzipien beruht, die von Fall zu Fall untersucht werden müssen. In diesem abschließenden Abschnitt werden wir zur Vorbereitung von leistungsfähigeren Verfahren einige Grundlagen entwickeln.

Alle bisherigen Fixpunktiterationen lassen sich allgemeiner in folgender Form schreiben

$$x^{k+1} = x^k + d^k, \quad k = 1, 2, \ldots,$$

wobei d^k in jedem Schritt die Richtung angibt, in der die Lösung verbessert wird. Beim Jacobi-Verfahren bestimmt sich diese Richtung z.B. als $d^k = D^{-1}(b - Ax^k)$, beim Gauß-Seidel Verfahren als $d^k = (D+L)^{-1}(b - Ax^k)$. Um diese allgemeine Iteration zu verbessern

setzen wir an zwei Punkten an: zunächst fügen wir in jedem Schritt der Iteration einen Relaxationsparameter ω^k ein

$$x^{k+1} = x^k + \omega^k d^k, \quad k = 1, 2, \ldots,$$

welchen wir Schritt für Schritt optimal bestimmen werden. Anschließen versuchen wir neue Suchrichtungen d^k auf eine systematische Art und Weise zu entwickeln. D.h., wir suchen eine Richtung d^k, in der die größte Fehlerreduktion zu erwarten ist. In diesem Abschnitt beschränken wir uns auf symmetrisch positiv definite Matrizen $A \in \mathbb{R}^{n\times n}$. Zentral für das gesamte Kapitel ist die folgende Charakterisierung zur Lösung eines linearen Gleichungssystems mit symmetrisch, positiv definiter Matrix:

Satz 5.36 (Lineares Gleichungssystem und Minimierung)**.** *Es sei $A \in \mathbb{R}^{n\times n}$ eine symmetrische positiv definite Matrix. Das lineare Gleichungssystem $Ax = b$ ist äquivalent zur Minimierungsaufgabe:*

$$Q(x) \le Q(y) \quad \forall y \in \mathbb{R}^n, \quad Q(y) = \frac{1}{2}(Ay, y)_2 - (b, y)_2.$$

Beweis: *(i)* Zunächst sei x Lösung des linearen Gleichungssystems $Ax = b$. Dann gilt mit beliebigem $y \in \mathbb{R}^n$:

$$\begin{aligned} 2Q(y) - 2Q(x) &= (Ay, y)_2 - 2(b, y)_2 - (Ax, x)_2 + 2(b, x) \\ &= (Ay, y)_2 - 2(Ax, y)_2 + (Ax, x)_2 \\ &= (A(y - x), y - x)_2 \ge 0, \end{aligned}$$

d.h. $Q(y) \ge Q(x)$.

(ii) Umgekehrt sei $Q(x)$ nun Minimum. D.h. $x \in \mathbb{R}^n$ ist stationärer Punkt der quadratischen Form $Q(x)$, also:

$$0 \stackrel{!}{=} \frac{\partial}{\partial x_i} Q(x) = \frac{\partial}{\partial x_i}\Big\{\frac{1}{2}(Ax, x)_2 - (b, x)_2\Big\} = 2(Ax)_i - 2b_i, \quad i = 1, \ldots, n.$$

D.h., x ist Lösung des linearen Gleichungssystems. □

Anstelle ein lineares Gleichungssystem $Ax = b$ zu lösen betrachten wir die Minimierung des "Energiefunktionals" $Q(x)$. Dieser Zugang ist Grundlage der im Folgenden diskutieren Verfahren und auch Basis der allgemeinen Klasse von Krylow-Raum-Verfahren.

Wir betrachten zunächst nur symmetrisch positiv definite Matrizen, daher ist durch $\|x\|_A := \sqrt{(Ax, x)_2}$ eine Norm, die sogenannte *Energienorm* gegeben. Die Minimierung des Energiefunktionals $Q(\cdot)$ ist auch äquivalent zur Minimierung des Fehlers $x^k - x$ in der zugehörigen. Denn, angenommen $x \in \mathbb{R}^n$ sei die Lösung und $x^k \in \mathbb{R}^n$ eine Approximation, dann gilt:

$$\|x^k - x\|_A^2 = (A(x^k - x), x^k - x)_2 = (Ax^k, x^k) - \underbrace{2(Ax^k, x)}_{=2(Ax, x^k) = 2(b, x^k)} + (Ax, x) = 2Q(x^k) + (Ax, x).$$

Abstiegsverfahren Wir gehen zunächst davon aus, dass die Suchrichtungen d^k durch ein gegebenes Verfahren (etwa Jacobi oder Gauß-Seidel) bestimmt sind und stellen uns der Frage, den nächsten Schritt optimal zu gestalten, also in der Iteration $k \to k+1$ mit

$$x^{k+1} = x^k + \omega^k d^k,$$

den skalaren Faktor ω^k möglichst optimal zu bestimmen, so dass die neue Approximation x^{k+1} eine möglichst geringe "Energie" $Q(x^{k+1})$ aufweist:

$$\omega^k = \arg\min_{\omega \in \mathbb{R}} Q(x^k + \omega d^k).$$

Das gesuchte Minimum ω können wir wieder als Extrempunkt des quadratischen Funktionals bestimmen:

$$0 \overset{!}{=} \frac{\partial}{\partial \omega} Q(x+\omega d) = \frac{\partial}{\partial \omega}\Big\{\frac{1}{2}(A(x+\omega d), x+\omega d)_2 - (b, x+\omega d)_2\Big\} = \omega(Ad,d)_2 + (Ax-b,d)_2$$

Hieraus bestimmt sich ω zu:

$$\omega = \frac{(b-Ax,d)_2}{(Ad,d)_2}.$$

Wir fassen zusammen:

Algorithmus 5.37 (Abstiegsverfahren)**.** *Es sei $A \in \mathbb{R}^{n\times n}$ symmetrisch positiv definit, $x^0, b \in \mathbb{R}^n$ sowie für $k = 1, 2, \ldots$ durch $d^k \in \mathbb{R}^n$ Abstiegsrichtungen gegeben. Iteriere:*

1. *Bestimme ω^k als*
$$\omega^k = \frac{(b-Ax^k, d^k)_2}{(Ad^k, d^k)_2},$$
2. *Berechne*
$$x^{k+1} = x^k + \omega^k d^k.$$

Ein konkretes Verfahren entsteht durch Wahl der Abstiegsrichtungen d^k. Es zeigt sich, dass die Kombination des Abstiegsverfahrens mit den bisher eingeführten Methoden wie Jacobi und Gauß-Seidel nicht zu wesentlichen Verbesserungen in der Konvergenzgeschwindigkeit führt. Wir betrachten hierzu ein Beispiel:

Beispiel 5.38 (Abstiegsverfahren, Jacobi & Gauß-Seidel)**.** *Es sei $Ax = b$ mit*

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

Mit dem Startvektor $x^0 = 0$ führen wir jeweils 10 Schritte mit Jacobi-, Gauß-Seidel-Verfahren sowie jeweils mit den entsprechenden Kombinationen unter Verwendung des optimalen Abstiegs-Schritts ω^k. In Abbildung 5.2 links fassen wir für alle Verfahren die Fehler zusammen. Auf der rechten Seite der Abbildung stellen wir den Approximationsverlauf $x^k \in \mathbb{R}^3$ für Jacobi- sowie Jacobi-Abstiegsverfahren graphisch dar. Obwohl der Verlauf des Jacobi-Abstiegsverfahrens wesentlich "gradliniger" scheint, konvergiert dieses Verfahren ebenso langsam wie das Jacobi-Verfahren selbst. Nur im Falle des Gauß-Seidel-Verfahrens wird die Konvergenz durch Wahl optimaler Schrittweite ω^k wesentlich beschleunigt.

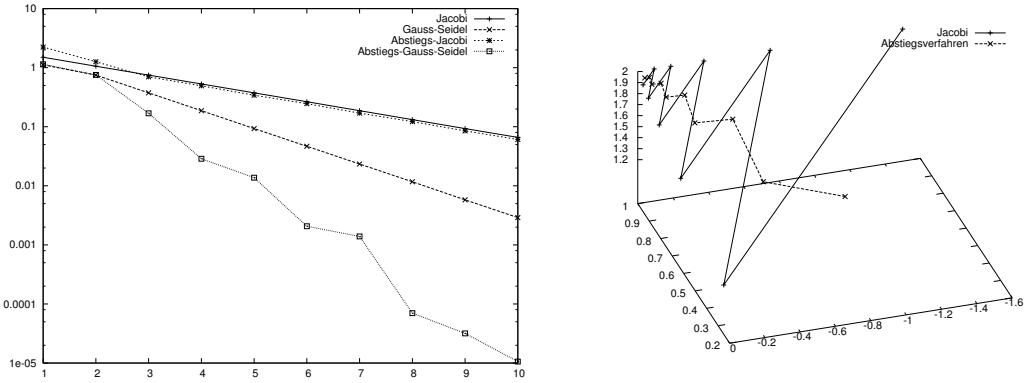

Abbildung 5.2: Links: Konvergenz von Jacobi-, Gauß-Seidel- sowie den entsprechenden Abstiegsverfahren. Rechts: Vergleich der Annäherung bei Jacobi- und Jacobi-Abstiegs-Verfahren an die Lösung $x = (1, 0, 2)^T$.

Gradientenverfahren Abschließend werden wir ein erstes Verfahren entwickeln, welches die neue Suchrichtung $d^k \in \mathbb{R}^n$ systematisch so bestimmt, dass das quadratische Funktional $Q(x)$ möglichst stark minimiert werden kann. Wir suchen also die Richtung des *stärksten Abfalls*. Zu einem Punkt $x \in \mathbb{R}^n$ ist dies gerade die Richtung $d \in \mathbb{R}^n$, die normal auf den Niveaumenge $N(x)$ steht:

$$N(x) := \{y \in \mathbb{R}^n : \ Q(y) = Q(x)\}$$

In einem Punkt x ist die Niveaumenge aufgespannt durch alle Richtungen $\delta x \in \mathbb{R}^n$ mit:

$$0 \stackrel{!}{=} Q'(x) \cdot \delta x = (\nabla Q(x), \delta x) = (b - Ax, \delta x).$$

Die Vektoren δx, welche die Niveaumenge aufspannen stehen orthogonal auf dem Defekt $b - Ax$, dieser zeigt daher in Richtung der stärksten Änderung von $Q(\cdot)$. Wir wählen $d^k := b - Ax^k$. Die so gefundene Suchrichtung wird dann mit dem Abstiegsverfahren kombiniert, d.h. wir iterieren:

$$d^k := b - Ax^k, \quad \omega^k := \frac{\|d^k\|_2^2}{(Ad^k, d^k)_2}, \quad x^{k+1} := x^k + \omega^k d^k.$$

Wir definieren das *Gradientenverfahren*:

Algorithmus 5.39 (Gradientenverfahren)**.** *Es sei $A \in \mathbb{R}^{n\times n}$ symmetrisch positiv definit, $b \in \mathbb{R}^n$. Es sei $x^0 \in \mathbb{R}^n$ beliebig und $d^0 := b - Ax^0$. Iteriere für $k = 0, 1, 2, \ldots$*

1. $r^k := Ad^k$
2. $\omega^k = \frac{\|d_k\|_2^2}{(r^k, d^k)_2}$
3. $x^{k+1} = x^k + \omega^k d^k$.
4. $d^{k+1} = d^k - \omega^k r^k$.

Durch Einführen eines zweiten Hilfsvektors $r^k \in \mathbb{R}^n$ kann in jeder Iteration ein Matrix-Vektor-Produkt gespart werden. Für Matrizen mit Diagonalanteil $D = \alpha I$ ist das Gradientenverfahren gerade das Jacobi-Verfahren in Verbindung mit dem Abstiegsverfahren. Daher kann für dieses Verfahren im Allgemeinen auch keine verbesserte Konvergenzaussage erreicht werden. Es stellt jedoch den Einstieg in eine ganze Klasse von fortgeschrittenen Verfahren, die Krylow-Unterraum-Verfahren dar. Wir zeigen:

Satz 5.40 (Gradientenverfahren)**.** *Es sei $A \in \mathbb{R}^{n\times n}$ symmetrisch positiv definit. Dann konvergiert das Gradientenverfahren für jeden Startvektor $x^0 \in \mathbb{R}^n$ gegen die Lösung des Gleichungssystems $Ax = b$.*

Beweis: Es sei $x^k \in \mathbb{R}^n$ eine gegebene Approximation. Weiter sei $d := b - Ax^k$. Dann berechnet sich ein Schritt des Gradientenverfahrens als:

$$x^{k+1} = x^k + \frac{(d,d)}{(Ad,d)}d.$$

Für das Energiefunktional gilt:

$$\begin{aligned}
Q(x^{k+1}) &= \frac{1}{2}(Ax^{k+1}, x^{k+1}) - (b, x^{k+1}) \\
&= \frac{1}{2}(Ax^k, x^k) + \frac{1}{2}\frac{(d,d)^2}{(Ad,d)^2}(Ad,d) + \frac{(d,d)}{(Ad,d)}(Ax^k,d) - (b,x^k) - \frac{(d,d)}{(Ad,d)}(b,d) \\
&= Q(x^k) + \frac{(d,d)}{(Ad,d)}\left\{\frac{1}{2}(d,d) + (Ax^k,d) - (b,d)\right\} \\
&= Q(x^k) + \frac{(d,d)}{(Ad,d)}\left\{\frac{1}{2}(d,d) + (\underbrace{Ax^k - b}_{=-d}, d)\right\}
\end{aligned}$$

Also folgt:

$$Q(x^{k+1}) = Q(x^k) - \frac{(d,d)^2}{2(Ad,d)}.$$

Wegen der positiven Definitheit von A gilt $\lambda_{\min}(A)(d,d) \leq (Ad,d) \leq \lambda_{\max}(A)(d,d)$ und schließlich ist mit

$$Q(x^{k+1}) \leq Q(x^k) - \underbrace{\frac{(d,d)}{2\lambda_{\max}}}_{\geq 0},$$

die Folge $Q(x^k)$ monoton fallend. Weiter ist $Q(x^k)$ nach unten durch $Q(x)$ beschränkt. Also konvergiert die Folge $Q(x^k) \to c \in \mathbb{R}^n$. Im Grenzwert muss gelten $0 = (d,d) = \|b - Ax\|^2$, also $Ax = b$. □

Schließlich zitieren wir noch zur Abschätzung der Konvergenzgeschwindigkeit die folgende Fehlerabschätzung:

Satz 5.41 (Konvergenz des Gradientenverfahrens)**.** *Es sei $A \in \mathbb{R}^{n\times n}$ eine symmetrisch positiv definite Matrix. Dann gilt für das Gradientenverfahren zur Lösung von $Ax = b$ die Fehlerabschätzung*

$$\|x^k - x\|_A \leq \left(\frac{1 - 1/\kappa}{1 + 1/\kappa}\right)^k, \quad \kappa := \mathrm{cond}_2(A) = \frac{\lambda_{max}(A)}{\lambda_{min}(A)}.$$

Beweis: Siehe [9] □

Die asymptotische Konvergenzrate des Gradientenverfahrens wir durch die Kondition der Matrix bestimmt. Für die Modellmatrix gilt $\kappa = O(n^2)$, siehe Beispiel 5.33. Also gilt:

$$\rho = \frac{1 - \frac{1}{n^2}}{1 + \frac{1}{n^2}} = 1 - \frac{2}{n^2} + O\left(\frac{1}{n^4}\right).$$

Die Konvergenz ist demnach ebenso langsam wie die des Jacobi-Verfahrens (wir haben bereits diskutiert, dass es für die Modellmatrix mit dem Jacobi-Abstiegsverfahren übereinstimmt). Für das Gradientenverfahren gilt jedoch der folgende Zusammenhang, der Basis des CG-Verfahrens ist:

Satz 5.42 (Abstiegsrichtungen im Gradientenverfahren)**.** *Es sei $A \in \mathbb{R}^{n\times n}$ symmetrisch positiv definit. Dann stehen je zwei aufeinanderfolgende Abstiegsrichtungen d^k und d^{k+1} des Gradientenverfahren orthogonal aufeinander.*

Beweis: Zum Beweis, siehe Algorithmus 5.39. Es gilt:

$$d^{k+1} = d^k - \omega^k r^k = d^k - \frac{(d^k, d^k)}{(Ad^k, d^k)} Ad^k.$$

Also gilt:

$$(d^{k+1}, d^k) = (d^k, d^k) - \frac{(d^k, d^k)}{(Ad^k, d^k)}(Ad^k, d^k) = (d^k, d^k) - (d^k, d^k) = 0.$$

□

Das CG-Verfahren Der Zusammenhang aus Satz 5.42 gilt nur für jeweils aufeinander folgende Abstiegsrichtungen, im Allgemeinen gilt jedoch $d^k \not\perp d^{k+2}$. In Abbildung 5.2 rechts ist der Approximationsverlauf des Abstiegs-Jacobi-Verfahren, welches mit dem Gradientenverfahren übereinstimmt dargestellt. Zwei aufeinander folgende Richtungen sind zwar je orthogonal, die dritte Richtung steht jedoch wieder nahezu parallel auf der ersten. Dies führt dazu, dass das Gradientenverfahren im Allgemeinen sehr langsam konvergiert. Das CG-Verfahren, oder "Verfahren der konjugierten Gradienten", entwickelt diesen Ansatz weiter und wählt Suchrichtungen $\{d^1, \ldots, d^k\}$ die paarweise orthogonal sind. Orthogonalität wird dabei im A-Skalarprodukt erreicht:

$$(Ad^r, d^s) = 0 \quad \forall r \neq s.$$

Im k-ten Schritt wird die Approximation $x^k = x^0 + \sum_{i=1}^k \alpha_i d^i$ als das Minimum über alle $\alpha_1, \ldots, \alpha_k$ bezüglich $Q(x^k)$ gesucht:

$$\min Q\left(x^0 + \sum_{i=1}^k \alpha_i d^i\right) = \min\left\{\frac{1}{2}\left(Ax^0 + \sum_{i=1}^k \alpha_i A d^i, x^0 + \sum_{i=1}^k \alpha_i d^i\right) - \left(b, x^0 + \sum_{i=1}^k \alpha_i d^i\right)\right\}.$$

Der stationäre Punkt ist bestimmt durch:

$$0 \stackrel{!}{=} \frac{\partial}{\partial \alpha_j} Q(x^k) = \left(Ax^0 + \sum_{i=1}^k \alpha_i A d^i, d^j\right) - (b, d^j) = -\left(b - Ax^k, d^j\right), \quad j = 1, \ldots, k.$$

D.h., das neue Residuum $b - Ax^k$ steht orthogonal auf allen Suchrichtungen d^j für $j = 1, \ldots, k$. Diese Gleichung

$$(b - Ax^k, d^j) = 0 \quad \forall j = 1, \ldots, k, \tag{5.17}$$

wird *Galerkin-Gleichung* genannt. Beim Entwurf des CG-Verfahrens ist es nun wichtig, dass die neu gewählte Suchrichtung d^{k+1} nicht im Erzeugnis der bisherigen Suchrichtungen $\text{span}\{d^1, \ldots, d^k\}$ enthalten ist. Denn in diesem Fall wird der Suchraum nicht größer und die Approximation kann nicht verbessert werden. Daher wählt man für das CG-Verfahren ausgehend von einer Startapproximation $x^0 \in \mathbb{R}^n$ mit $d^0 := b - Ax^0$ den *Krylow-Raum* $K_k(d^0, A)$:

$$K_k(d^0, A) := \text{span}\{d^0, Ad^0, \ldots, A^{k-1}d^0\}.$$

Es gilt:

Hilfsatz 5.43. *Angenommen es gilt $A^k d^0 \in K_k$. Dann liegt auch für die Lösung $x \in \mathbb{R}^n$ von $Ax = b$ im k-ten Krylow-Raum $K_k(d_0, A)$.*

Beweis: Es sei K_k gegeben und $x^k \in x_0 + K_k$ die beste Approximation, welche die Galerkin-Gleichung (5.17) erfüllt. Es sei $r^k := b - Ax^t$. Wegen

$$r^k = b - Ax^k = \underbrace{b - Ax^0}_{=d^0} + A\underbrace{(x^0 - x^k)}_{\in K_k} \in d^0 + AK_k,$$

gilt $r^k \in K_{k+1}$. Angenommen nun, $K_{k+1} \subset K_k$. Dann gilt $r^k \in K_k$. Die Galerkin-Gleichung besagt $r^k \perp K_k$, d.h. es gilt zwingend $r^k = 0$ und $Ax^k = b$. □

Falls das CG-Verfahren abbricht weil keine neuen Suchrichtungen hinzukommen, so ist die Lösung gefunden. Angenommen, die A-orthogonalen Suchrichtungen $\{d^0, d^1, \ldots, d^{k-1}\}$ liegen vor, so kann die CG-Approximation durch Ausnutzen der Basisdarstellung $x^k = x^0 + \sum \alpha_i d^i$ aus der Galerkin-Gleichung berechnet werden:

$$\left(b - Ax^0 - \sum_{i=1}^k \alpha_i A d^i, d_j\right) = 0 \quad \Rightarrow \quad (b - Ax^0, d^j) = \alpha_j (Ad^j, d^j) \quad \Rightarrow \quad \alpha_j = \frac{(d^0, d^j)}{(Ad^j, d^j)}.$$

Die A-orthogonale Basis $\{d^0, \ldots, d^{k-1}\}$ des Krylow-Raums $K_k(d^0, A)$ kann z.B. mit dem Gram-Schmidt-Verfahren berechnet werden. Der Nachteil dieser Methode ist zum einen der quadratische (in k) Aufwand des Gram-Schmidt-Verfahrens sowie dessen numerische Instabilität. Seine Leistungsfähigkeit erlangt das CG-Verfahren durch ausnutzen einer zweistufigen Rekursionsformel, welche die A-orthogonale Basis effizient und stabil berechnet:

Hilfsatz 5.44 (2-stufige Rekursionsformel zur Orthogonalisierung)**.** *Es sei $A \in \mathbb{R}^{n\times n}$ symmetrisch positiv definit, sowie $x^0 \in \mathbb{R}^n$ und $d^0 := b - Ax^0$. Dann wird durch die Iteration*

$$r^k := b - Ax^k, \quad \beta_{k-1} := -\frac{(r^k, Ad^{k-1})}{(d^{k-1}, Ad^{k-1})}, \quad d^k := r^k - \beta_{k-1}d^{k-1}, \quad k = 1, 2, \ldots$$

eine A-orthogonale Basis mit $(Ad^r, d^s) = 0$ für $r \neq s$ erzeugt. Dabei ist x^k in Schritt k definiert als die Galerkin-Lösung $(b - Ax^k, d^j) = 0$ für $j = 0, \ldots, k-1$.

Beweis: Es sei durch $\{d^0, \ldots, d^{k-1}\}$ eine A-orthogonale Basis des $K_k(d^0, A)$ gegeben. Weiter sei $x^k \in x^0 + K_k(d^0, A)$ die Galerkin-Lösung zu (5.17). Es sei $r^k := b - Ax^k \in K_{k+1}$ und wir fordern, dass $g^k \notin K_k(d^0, A)$. Ansonsten bricht die Iteration nach Hilfsatz 5.43 ab. Wir bestimmen d^k mit dem Ansatz:

$$d^k = r^k - \sum_{j=0}^{k-1} \beta_j^{k-1} d^j. \tag{5.18}$$

Die Orthogonalitätsbedingung besagt:

$$0 \overset{!}{=} (d^k, Ad^i) = (r^k, Ad^i) + \sum_{j=1}^{k-1} \beta_j^{k-1}(d^j, Ad^i) = (r^k, Ad^i) + \beta_i^{k-1}(d^i, Ad^i), \quad i = 0, \ldots, k-1.$$

Es gilt $(r^k, Ad^i) = (b - Ax^k, Ad^i) = 0$ für $i = 0, \ldots, k-2$, da $Ar^k \perp K_{k-1}$. Hieraus folgt $\beta_i^{k-1} = 0$ für $i = 0, 1, \ldots, k-2$. Für $i = k-1$ gilt:

$$\beta_{k-1} := \beta_{k-1}^{k-1} = -\frac{(r^k, Ad^{k-1})}{(d^{k-1}, Ad^{k-1})}.$$

Schließlich gilt mit (5.18) $d^k = r^k - \beta_{k-1}d^{k-1}$. □

Mit diesen Vorarbeiten können wir alle Bestandteile des CG-Verfahrens zusammensetzen. Es sei also mit x^0 eine Startlösung und mit $d^0 := b - Ax^0$ der Startdefekt gegeben. Angenommen, $K_k := \operatorname{span}\{d^0, \ldots, d^{k-1}\}$ sowie $x^k \in x^0 + K_k$ und der Defekt $r^k = b - Ax^k$ liegen vor. Dann berechnet sich d^k gemäß Hilfsatz 5.44 als

$$\beta_{k-1} = -\frac{(r^k, Ad^{k-1})}{(d^{k-1}, Ad^{k-1})}, \quad d^k = r^k - \beta_{k-1}d^{k-1}. \tag{5.19}$$

Für den neuen Entwicklungskoeffizienten α_k in $x^{k+1} = x^0 + \sum_{i=0}^{k} \alpha_i d^i$ gilt durch Testen der Galerkin-Gleichung (5.17) mit d^k:

$$\left(\underbrace{b - Ax^0}_{=d^0} - \sum_{i=0}^{k} \alpha_i A d^i, d^k \right) = (b - Ax^0, d^k) - \alpha_k (Ad^k, d^k) = (b - Ax^0 + A \underbrace{(x^0 - x^k)}_{\in K_k}, d^k) - \alpha_k (Ad^k, d^k).$$

Also:

$$\alpha_k = \frac{(r^k, d^k)}{(Ad^k, d^k)}, \quad x^{k+1} = x^k + \alpha_k d^k. \tag{5.20}$$

Hieraus lässt sich auch unmittelbar der neue Defekt r^{k+1} bestimmen:

$$r^{k+1} = b - Ax^{k+1} = b - Ax^k - \alpha_k Ad^k = r^k - \alpha_k Ad^k. \tag{5.21}$$

Wir fassen (5.19-5.21) zusammen und formulieren das klassische CG-Verfahren:

Algorithmus 5.45 (CG-Verfahren)**.** *Es sei $A \in \mathbb{R}^{n \times n}$ symmetrisch positiv definit, $x^0 \in \mathbb{R}^0$ und $r^0 = d^0 = b - Ax^0$ gegeben. Iteriere für $k = 0, 1, \ldots$*

1. $\alpha_k = \frac{(r^k, d^k)}{(Ad^k, d^k)}$
2. $x^{k+1} = x^k + \alpha_k d^k$
3. $r^{k+1} = r^k - \alpha_k r^k$
4. $\beta_k = \frac{(r^{k+1}, Ad^k)}{(d^k, Ad^k)}$
5. $d^{k+1} = r^{k+1} - \beta_k d^k$

Das CG-Verfahren liefert eine Lösung nach (höchstens) n Schritten für ein n-dimensionales Problem und kann daher prinzipiell als direktes Verfahren betrachtet werden. Allerdings sind schon für relativ kleine Probleme (z.B. $n = 1000$) im ungünstigsten Fall 1000 Iterationen notwendig. Deshalb wird das CG-Verfahren üblicherweise approximativ eingesetzt. In jedem Iterationsschritt sind Rundungsfehler zu erwarten, daher werden die Suchrichtungen $\{d^0, \ldots, d^{k-1}\}$ nie wirklich orthogonal sein. Die Konvergenzanalyse des CG-Verfahrens gestaltet sich als sehr aufwendig. Daher zitieren wir hier nur das Ergebnis:

Satz 5.46 (Konvergenz des CG-Verfahrens)**.** *Es sei $A \in \mathbb{R}^{n \times n}$ symmetrisch positiv definit. Dann gilt für beliebigen Startvektor $x^0 \in \mathbb{R}^n$ die Fehlerabschätzung*

$$\|x^k - x\|_A \leq 2 \left(\frac{1 - 1/\sqrt{\kappa}}{1 + 1/\sqrt{\kappa}} \right)^k \|x^0 - x\|_A, \quad k \geq 0,$$

mit der Spektralkondition $\kappa = \mathrm{cond}_2(A)$ der Matrix A.

Die Konvergenz des CG-Verfahrens ist gerade doppelt so schnell wie die des Gradientenverfahrens. Es gilt:

$$\rho := \frac{1 - 1/\sqrt{\kappa}}{1 + 1/\sqrt{\kappa}} = 1 - 2\sqrt{\kappa} + O(\kappa).$$

Für die Modellmatrix folgt $\rho = 1 - \frac{1}{n}$. Das CG-Verfahren ist für dünn besetzte symmetrische Gleichungssysteme eines der effizientesten Iterationsverfahren. Die Konvergenz hängt wesentlich von der Kondition $\text{cond}_2(A)$ der Matrix A ab. Für $\text{cond}_2(A) \approx 1$ ist das Verfahren optimal: zur Reduktion des Fehlers um einen gegebenen Faktor ϵ ist eine feste Anzahl von Schritten notwendig. Verallgemeinerungen des CG-Verfahrens für nicht symmetrische Matrizen müssen die Orthogonalisierung der Suchrichtungen entweder aufwendig durch volle Orthogonalisierungsverfahren wie Gram-Schmidt sicherstellen (dies führt auf das GMRES-Verfahren) oder die Orthogonalitätsbeziehung wird verletzt (dies führt zum Beispiel auf das BiCGStab-Verfahren).

Abschließend diskutieren wir anhand der Modellmatrix die verschiedenen Verfahren im Vergleich:

Beispiel 5.47 (LGS mit der Modellmatrix)**.** *Es sei $A \in \mathbb{R}^{n\times n}$ mit $n = m^2$ die Modellmatrix:*

$$A = \begin{pmatrix} B & -I & & \\ -I & B & -I & \\ & -I & B & -I \\ & & -I & B \end{pmatrix}, \quad B = \begin{pmatrix} 4 & -1 & & \\ -1 & 4 & -1 & \\ & -1 & 4 & -1 \\ & & -1 & 4 \end{pmatrix}, \quad I = \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{pmatrix},$$

mit $B, I \in \mathbb{R}^{m\times m}$. Die Matrix A ist eine Bandmatrix mit Bandbreite $2m$. Weiter ist die Matrix A symmetrisch positiv definit. Sie ist irreduzibel und diagonaldominant und erfüllt in den Rändern der Blöcke das starke Zeilensummenkriterium. Alle bisher diskutieren Verfahren können auf die Matrix A angewendet werden. Größter sowie kleinster Eigenwert und Spektralkondition von A berechnen sich zu:

$$\lambda_{min} = \frac{2\pi^2}{n} + O(n^{-2}), \quad \lambda_{max} = 8 - \frac{2\pi^2}{n} + O(n^{-2}), \quad \kappa \approx 2\pi^2 n$$

Direkte Verfahren *Die LR-Zerlegung ist (da A positiv definit) ohne Permutierung durchzuführen. Gemäß Satz 4.39 beträgt der Aufwand hierzu $N_{LR} = nm^2 = 4n^2$ Operationen. Die Lösung ist dann bis auf Rundungsfehlereinflüsse exakt gegeben. Alternativ kann die Cholesky-Zerlegung von A erstellt werden. Hierzu sind $N_{LL} = 2n^2$ Operationen notwendig. LR- bzw. Cholesky-Zerlegung sind dich besetzte Bandmatrizen. Das anschließende Lösen mit Vorwärts und Rückwärtselimination benötigt weitere $2nm$ Operationen. Wir fassen die notwendigen Operationen in folgender Tabelle zusammen:*

$n = m^2$	N_{LR}	N_{LL}
100	$5 \cdot 10^4$	$2 \cdot 10^4$
10000	$5 \cdot 10^9$	$2 \cdot 10^9$
1000000	$5 \cdot 10^{12}$	$2 \cdot 10^{12}$

Bei effizienter Implementierung auf moderner Hardware ist das Gleichungssystem mit $10\,000$ *Unbekannten in wenigen Sekunden, das größte Gleichungssystem in wenigen Stunden lösbar.*

Einfache Fixpunktiterationen *Wir schätzen zunächst für Jacobi- sowie Gauß-Seidel-Verfahren die Spektralradien ab. Mit einem Eigenwert* λ *und Eigenvektor* w *von* A *gilt:*

$$Aw = \lambda w \quad \Rightarrow \quad Dw + (L+R)w = \lambda w \quad \Rightarrow \quad -D^{-1}(L+R)w = -D^{-1}(\lambda I - D)w.$$

D.h. wegen $D_{ii} = 4$ *gilt*

$$Jw = \frac{\lambda - 4}{4} w,$$

und die Eigenwerte von J *liegen zwischen*

$$\lambda_{min}(J) = \frac{1}{4}(\lambda_{min}(A) - 4) \approx -1 + \frac{\pi^2}{2n}, \quad \lambda_{max}(J) = \frac{1}{4}(\lambda_{max}(A) - 4) \approx 1 - \frac{\pi^2}{2n}.$$

Für die Konvergenzrate gilt:

$$\rho_J := 1 - \frac{\pi^2}{2n}.$$

Aus (5.15) folgt:

$$\rho_H := \rho_J^2 \approx 1 - \frac{\pi^2}{n}.$$

Zur Reduktion des Fehlers um einen gegebenen Faktor ϵ *sind* t *Schritte erforderlich:*

$$\rho_J^{t_J} = \epsilon \quad \Rightarrow \quad t_J = \frac{\log(\epsilon)}{\log(\rho)} \approx \frac{2}{\pi^2} \log(\epsilon) n, \quad t_H \approx \frac{1}{\pi^2} \log(\epsilon) n,$$

Jeder Schritt hat den Aufwand eines Matrix-Vektor Produktes, d.h. im gegebenen Fall $5n$*. Schließlich bestimmen wir gemäß (5.16) den optimalen SOR-Parameter zu*

$$\omega_{opt} \approx 2 - 2\frac{\pi}{\sqrt{n}}.$$

Dann gilt:

$$\rho_\omega = \omega_{opt} - 1 \approx 1 - 2\frac{\pi}{\sqrt{n}}.$$

Hieraus erhalten wir:

$$t_\omega \approx \frac{\log(\epsilon)}{2\pi} \sqrt{n}.$$

Der Aufwand des SOR-Verfahrens entspricht dem des Gauß-Seidel-Verfahrens mit einer zusätzlichen Relaxation, d.h. $6n$ *Operationen pro Schritt. Wir fassen zusammen für die drei Verfahren Konvergenzrate, Anzahl der notwendigen Schritte und Gesamtaufwand zusammen. Dabei ist stets* $\epsilon = 10^{-4}$ *gewählt:*

	Jacobi		*Gauß-Seidel*		*SOR*		
$n = m^2$	t_J	N_j	t_H	N_H	ω_{opt}	t_J	N_j
100	*180*	10^5	*90*	$5 \cdot 10^4$	*1.53*	*9*	10^5
10000	*18500*	10^9	*9300*	$5 \cdot 10^8$	*1.94*	*142*	10^7
1000000	*1866600*	10^{13}	*933000*	$5 \cdot 10^{12}$	*1.99*	*1460*	10^9

Während das größte Gleichungssystem mit dem optimalen SOR-Verfahren in wenigen Sekunden gelöst werden kann, benötigen Jacobi und Gauß-Seidel-Verfahren etliche Stunden.

Abstiegsverfahren *Schließlich betrachten wir Gradienten und CG-Verfahren. Es gilt:*

$$\rho_{GR} \approx 1 - 2\kappa \approx 1 - 4\pi^2 n, \quad \rho_{CG} \approx 1 - 2\sqrt{\kappa} \approx 1 - 2\pi\sqrt{n}.$$

Bei effizienter Implementierung benötigt das Gradientenverfahren pro Iterationsschritt eine Matrix-Vektor Multiplikation ($5n$ *Operationen), zwei Skalarprodukte (*$2n$ *Operationen) sowie 2 Vektor-Additionen (*$2n$ *Operationen), insgesamt somit* $9n$ *Operationen. Der Aufwand des CG-Verfahrens ist mit* $15n$ *Operationen etwas größer. Die Anzahl der Schritte bestimmt sich zu*

$$\rho_{GR}^t = 10^{-4} \quad \Rightarrow \quad t_{GR} \approx \frac{n}{\pi^2}, \quad t_{CG} \approx \frac{2}{\pi\sqrt{n}}.$$

Wir fassen zusammen:

	Gradienten		*CG*	
$n = m^2$	t_{GR}	N_{GR}	t_{CG}	N_{CG}
100	*18*	10^4	*9*	10^4
10000	*2 330*	10^8	*140*	10^7
1000000	*233 300*	10^{12}	*1400*	10^{10}

Wie bereits bekannt ist das Gradienten-Verfahren ebenso ineffektiv wie das Jacobi-Verfahren. Das CG-Verfahren erreicht etwa die gleiche Effizienz wie das SOR-Verfahren bei optimaler Wahl des Relaxationsparameters. Dieses Ergebnis darf nicht falsch interpretiert werden: im Allgemeinen ist dieser Relaxationsparameter nicht verfügbar und muss grob approximiert werden. D.h., in der praktischen Anwendung wird das SOR-Verfahren weit schlechter konvergieren und das CG-Verfahren ist im Allgemeinen stark überlegen!

6 Anwendungsbeispiel: dünner Balken

In diesem Kapitel fassen wir alle vorherigen Kapitel mit Hilfe eines physikalisch motivierten Anwendungsbeispiels zusammen. Das Beispiel kann als typischer prototypischer Vertreter für zahlreiche Anwendungsfälle numerischer Verfahren angesehen werden. Grundsätzlich sind für den Numeriker dabei die folgenden Schritte durchzuführen, wobei der Schwerpunkt von Numerik 0 immer auf Diskretisierung und numerischer Lösung liegt.

1. **Problem/Aufgabenstellung:** Biegung eines dünnen Balkens (z.B. Brücke oder Membran),

2. **Modellierung:** Herleitung eines physikalischen Modells (Abschnitt 6.1),

3. **Diskretisierung** (Abschnitt 6.2) mit Hilfe der Methoden aus Kapitel 3 (Interpolation, stückweise Interpolation, Differenzenverfahren zur Approximation von Ableitungen),

4. **Numerische Lösung** (Abschnitt 6.3) des diskretisierten Problems. Bei linearen Problemen können sofort die Methoden aus Kapitel 4 genutzt werden. Bei nichtlinearen Problemen wird mittels eines Fixpunktverfahrens als Nullstellenaufgabe geschrieben (siehe Kapitel 5 bzw. 2) und hierin Lösung der linearen Gleichungen mit den Methoden aus Kapitel 4,

5. **Analyse der Ergebnisse** mit Vergleich zu anderen Methoden für dasselbe Problem oder Abgleich mit Experimenten.

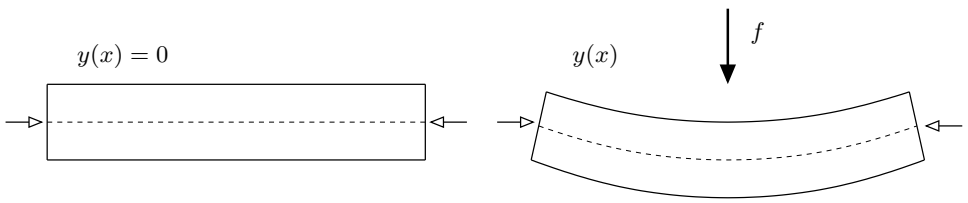

Abbildung 6.1: Balken im Ruhezustand und rechts durch Kraft f ausgedehnter Balken. Der Balken ist an den Endpunkten $x = 0$ sowie $x = 1$ fixiert und wird über die Position $y(x)$ der Mittellinie parametrisiert.

6.1 Modellierung eines elastischen Balkens

In Abbildung 6.1 zeigen wir zunächst die Konfiguration. Ein Balken mit Länge $L = 1$ (im Ruhezustand) ist auf beiden Seiten eingespannt. Unter Einwirkung eines Biegemoments f wird der Balken ausgelenkt. Dabei bezeichnen wir mit $y(x)$ die Deformation der Mittellinie in jedem Punkt $x \in [0, 1]$ des undeformierten Balkens. Dabei gilt stets $y(0) = y(1) = 0$ aufgrund der Einspannung. Die Biegung eines Balkens kann in jedem Punkt durch den *Krümmungsradius* $\rho := \rho(x)$ gekennzeichnet werden. Der Krümmungsradius ist der Radius desjenigen Kreises, der in $(x, y(x))$ die Mittellinie des Balkens bei gleicher Krümmung berührt, siehe Abbildung 6.2.

Die Krümmung erzeugt eine Dehnung $\epsilon := \delta e/e$, welche die relative Längenänderung einer Balkenlinie angibt, die radial von der Mittellinie verschoben ist, siehe Abbildung 6.2 Mitte. Für eine Linie, die um y verschoben ist, gilt mit dem Strahlensatz:

$$\frac{\rho + y}{\rho} = \frac{e + \delta e}{e} \quad \Leftrightarrow \quad \frac{y}{\rho} = \frac{\delta e}{e}. \tag{6.1}$$

Das *Hooke'sche Gesetz* besagt, dass die Dehnung ϵ eines Balkens proportional zur Deformation y und der einwirkenden Kraft f ist:

$$\epsilon = \mu f y,$$

wobei $\mu \in \mathbb{R}$ ein Parameter ist, der die Materialeigenschaften des Balkens beschreibt (die Biegesteifigkeit). Aus dem Zusammenhang zwischen Dehnung und Krümmungsradius (6.1) folgt:

$$\mu f = \frac{1}{\rho}. \tag{6.2}$$

Schließlich werden wir den Krümmungsradius $\rho(x)$ lokal durch die Deformation $y(x)$ ausdrücken. Siehe hierzu die rechte Skizze in Abbildung 6.2. Zunächst gilt aus geometrischen Überlegungen einerseits den Anstiegswinkel

$$\tan(\alpha) = \frac{\delta y}{\delta x},$$

sowie für die Bogenlänge:

$$\delta s = \delta\alpha = \sqrt{\delta x^2 + \delta y^2} = \delta x \sqrt{1 + \frac{\delta y^2}{\delta x^2}} \quad \Rightarrow \quad \frac{\delta\alpha}{\delta x} = \sqrt{1 + \left(\frac{\delta y}{\delta x}\right)^2}.$$

Wir betrachten alle Änderungen δx sowie δy und $\delta\alpha$ als infinitesimal. D.h., es gilt

$$\begin{aligned}
\frac{\delta y^2}{\delta x^2} &= \frac{\partial^2}{\partial^2 x} y(x) = \frac{\partial}{\partial x}\frac{\delta y}{\delta x} \\
&= \frac{\partial}{\partial x}\tan(\alpha) = (1 + \tan^2(\alpha))\frac{\partial\alpha}{\partial x} \\
&= (1 + \tan^2(\alpha))\frac{\delta\alpha}{\delta x} \\
&= \left(1 + \left(\frac{\delta y}{\delta x}\right)^2\right)\frac{\delta\alpha}{\delta x}
\end{aligned}$$

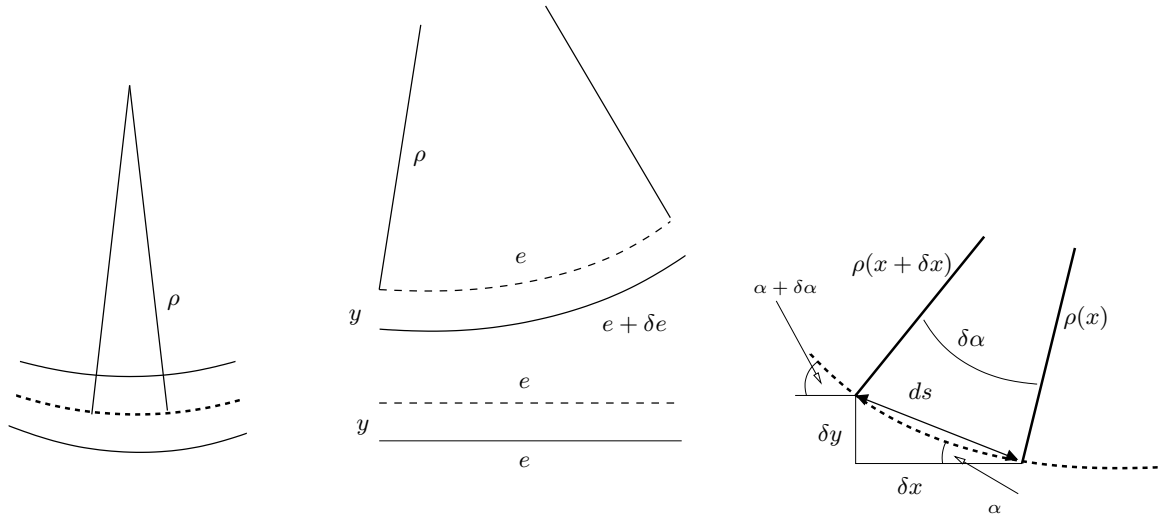

Abbildung 6.2: Links: Krümmungsradius ρ als Radius desjenigen Kreises mit der gleichen Krümmung. Mitte: Herleitung der Dehnung $\epsilon := \delta e/e$ als relative Längenänderung einer um radial verschobenen Linie. Rechts: Herleitung der Beziehung zwischen Deformation $y(x)$ und Krümmungsradius $\rho(x)$.

Hiermit können wir $\delta\alpha/\delta x$ mit Hilfe der 2. Ableitung ersetzen und mit $y' := \partial_x y$ sowie mit $y'' := \partial_{xx} y$ gilt für die Krümmung

$$\kappa := \lim_{\Delta s \to 0} \frac{\Delta\alpha}{\Delta s} = \frac{\delta\alpha}{\delta s} = \frac{\delta\alpha/\delta x}{\delta s/\delta x} = \frac{y''(x)}{\sqrt{1 + y'(x)^2}^3}.$$

und mit dem Zusammenhang zum Krümmungsradius $\kappa := \frac{1}{\rho}$ und dem Materialgesetz 6.2 erhalten wir die (nicht-lineare) *Differentialgleichung* für die Deformation eines die Balkens:

$$\frac{y''(x)}{(1 + y'(x)^2)^{\frac{3}{2}}} = \mu f. \tag{6.3}$$

Für sehr kleine Auslenkungen (dann gilt $y'(x) \ll 1$) erhalten wir hieraus die linearisierte Variante:

$$y''(x) = \mu f \tag{6.4}$$

die in vielen Büchern als Laplace-Gleichung (Modellgleichung) bekannt ist.

6.2 Diskretisierung

Wir suchen also eine Funktion $y \in C^2([0,1])$, welche die Differentialgleichung (6.3) in jedem Punkt $x \in [0,1]$ erfüllt. Diese Differentialgleichung ist für gegebene Kraftverteilung f im Allgemeinen nicht analytisch lösbar und muss mit numerischen Verfahren approximiert

werden. Die Lösung $y \in C^2([0,1])$ ist ein unendlich dimensionales Objekt und kann mit diskreten Methoden nie komplett beschrieben werden. Daher müssen wir in einem ersten Schritt das Problem *diskretisieren*, also in ein endlich dimensionales Problem überführen.

Der übliche Zugang hierzu ist, die Funktion $y \in C^2([0,1])$ durch eine Interpolierende $y_h(x)$ zu ersetzen. Hierzu zerlegen wir das Intervall $I = [0,1]$ zunächst in $n+1$ diskrete Punkte

$$0 = x_0 < x_1 < \cdots < x_n = 1, \quad h = x_i - x_{i-1} = \frac{1}{n}, \quad x_i = ih.$$

Wir wählen die Zerlegung äquidistant, d.h., je zwei benachbarte Punkte haben den Abstand h. Zur Interpolation der Funktion $y(x)$ in den Stützstellen $y_i := y(x_i)$ wählen wir zwei verschiedene Zugänge aus Kapitel 3. Zunächst wählen wir die globale Lagrange-Interpolation:

$$y^n \in P_n(I) = \operatorname{span}\{1, x, \ldots, x^n\}.$$

Zu gegebenen Stützstellenpaaren $(x_i, y(x_i))$ ist diese Interpolation stets eindeutig bestimmt, siehe Satz 3.6. Wir wählen die Lagrange-Darstellung:

$$y^n(x) = \sum_{i=0}^{n} y_i L_i^{(n)}(x), \quad L_i^{(n)}(x) = \prod_{j=1, j\neq i}^{n} \frac{x - x_j}{x_i - x_j}. \tag{6.5}$$

Alternativ setzen wir $y^h(x)$ als stückweise lineare Interpolierende zusammen:

$$y^h(x)\Big|_{[x_{i-1},x_i]} = \frac{x - x_i}{x_{i-1} - x_i} y_{i-1} + \frac{x - x_{i-1}}{x_i - x_{i-1}} y_i. \tag{6.6}$$

Im Falle einer hinreichend regulären Lösung $y(x)$ gelten bei exakter Interpolation (d.h. falls $y_i = y(x_i)$ exakt bekannt sind) die Abschätzungen

$$\|y^n - y\|_\infty \leq \frac{\|y^{(n+1)}\|_\infty}{(n+1)!}, \quad \|y^h - y\|_\infty \leq \frac{h^2}{2} \|y''\|_\infty, \tag{6.7}$$

siehe Satz 3.4 sowie (3.6). Bei der ersten Abschätzung haben wir wegen $|x - x_j| \leq 1$ ganz grob abgeschätzt mit:

$$\prod_{i=0}^{n} |x - x_i| \leq 1.$$

Diese Abschätzung ist sehr pessimistisch, da die meisten Faktoren $|x - x_j| \ll 1$ sehr viel kleiner als eins sind.

Die Diskretisierung der Aufgabe besteht nun darin, die Klasse der möglichen Lösungen zu verringern. Anstelle eines $y \in C^2([0,1])$ lassen wir nur noch Polynome gemäß (6.5) bzw. (6.6) zu. Das unendlich-dimensionale Problem wird durch ein endlich-dimensionales Problem ersetzt. Die beiden diskreten Funktionen haben jeweils $n+1$ Freiheiten, von denen die Randpunkte $y_0 = y_n = 1$ bereits bestimmt sind. Diese $n+1$ Freiheitsgrade werden durch $n+1$ Gleichungen in den Stützstellen beschrieben:

$$y^n(0) = 0, \quad \frac{y^{n\prime\prime}(x_i)}{(1 + y^{n\prime}(x_i)^2)^{\frac{3}{2}}} = \mu f(x_i), \quad i = 1, \ldots, n-1, \quad y^n(1) = 0.$$

Das Ergebnis ist ein nichtlineares Gleichungssystem mit $n+1$ Gleichungen und den $n+1$ unbekannten Koeffizienten $y_0, \ldots, y_n$.

6.2.1 Diskretes Modell mit globaler Interpolation

Zum Aufstellen des diskreten Gleichungssystems müssen die Ableitungen von $y^{n\prime}(x)$ und $y^{n\prime\prime}(x)$ in den Gitterpunkten x_i berechnet werden:

$$y^{n\prime}(x_i) = \sum_{k=0}^{n} y_k \underbrace{\partial_x L_k^{(n)}(x_i)}_{=:a^{(1)}(i,k,n)}, \quad y^{n\prime\prime}(x_i) = \sum_{k=0}^{n} y_k \underbrace{\partial_{xx} L_k^{(n)}(x_i)}_{=:a^{(2)}(i,k,n)}.$$

Die Koeffizienten $a^{(k)}(i,k,n)$, also die k-te Ableitung der k-ten Lagrange-Basisfunktion im Punkt x_i, müssen für jede Kombination von i,k und n berechnet werden. Für kleine Indizes kann diese Berechnung einfach analytisch geschehen. Für große Indizes muss diese Berechnung numerisch, z.B. mit der Newton'schen Darstellung aus Abschnitt 3.1.2 erfolgen. Diese Koeffizienten hängen nicht von der diskreten Funktion y^n ab, sondern lediglich vom Ansatzgrad n und der Position der Stützstellen x_i. Mit dieser Schreibweise gilt für das nichtlineare Gleichungssystem:

$$\begin{aligned} y_0 &= 0, \\ \sum_{k=0}^{n} a^{(2)}(i,k,n) y_k - \mu f(x_i) \left(1 + \left(\sum_{k=0}^{n} a^{(1)}(i,k,n) y_k\right)^2\right)^{\frac{3}{2}} &= 0, \quad i = 1,\dots,n-1, \\ y_n &= 0. \end{aligned} \tag{6.8}$$

Mit dem Koeffizientenvektor $\mathbf{y} \in \mathbb{R}^{n+1}$ und entsprechender vektorwertiger Funktion $F^n : \mathbb{R}^{n+1} \to \mathbb{R}^{n+1}$ schreiben wir kurz

$$F^n(\mathbf{y}) = 0.$$

6.2.2 Diskretes Modell mit stückweiser Interpolation

Wir betrachten nun den stückweisen linearen Ansatz $y^h(x)$. Im Gegensatz zur globalen Interpolation $y^n(x)$ dürfen wir diese Ansatzfunktion nicht in die Differentialgleichung (6.3) einsetzen, da $y^h(x)$ in den Stützstellen gar nicht differenzierbar ist!

Stattdessen werden wir zunächst die Ableitungen $y'(x)$ und $y''(x)$ in den Stützstellen durch geeignete Differenzenquotienten approximieren. In Abschnitt 3.3 haben wir für die erste Ableitung zunächst die einseitigen Differenzenquotienten kennengelernt:

$$y'(x) = \frac{y(x+h) - y(x)}{h} + \frac{h}{2} y''(\xi) + O(h^2).$$

Dieser Differenzenquotient ist von erster Ordnung. D.h., bei Halbierung der Gitterweite ist eine Halbierung des Fehlers zu erwarten. Für den gewählten Diskretisierungsansatz mit stückweise linearen Funktionen ist diese erste Ordnung nicht optimal. Denn Abschätzung (6.7) besagt, dass sich der Interpolationsfehler zwischen y^h und y quadratisch in h verhält. Zur besseren Balancierung wählen wir den zentralen Differenzenquotienten:

$$y'(x) = \frac{y(x+h) - y(x-h)}{2h} + \frac{h^2}{6} y'''(\xi) + O(h^4).$$

Zur Approximation der Ableitung in einem Punkt x_i ersetzen wir die Funktionswerte $y(x)$ durch die Koeffizienten der Lagrange-Darstellung:

$$(y^h)'(x_i) \approx \frac{y_{i+1} - y_{i-1}}{2h}.$$

Bei der zweiten Ableitung gehen wir entsprechend vor und approximieren mit dem zentralen Differenzenquotienten zweiter Ordnung:

$$(y^h)''(x_i) \approx \frac{-2y_i + y_{i+1} + y_{i-1}}{h^2}.$$

Wir setzen beide Approximationen in die Differentialgleichung (6.3) ein und erhalten das nichtlineare Gleichungssystem:

$$\begin{aligned} y_0 &= 0, \\ \frac{-2y_i + y_{i-1} + y_{i+1}}{h^2} - \mu f(x_i) \left(1 + \left(\frac{y_{i+1} - y_{i-1}}{2h}\right)^2\right)^{\frac{3}{2}} &= 0, \quad i = 1, \ldots, n-1, \\ y_n &= 0. \end{aligned} \tag{6.9}$$

Dieses Gleichungssystem können wir wieder kurz mit einer nichtlinearen Funktion $F^h : \mathbb{R}^{n+1} \to \mathbb{R}^{n+1}$ schreiben.

6.2.3 Vergleich und Diskussion der beiden Modelle

Beide Diskretisierungsansätze führen jeweils auf ein nichtlineares Gleichungssystem mit $n+1$ Unbekannten und $n+1$ Gleichungen. Der Lösungsvektor $\mathbf{y} \in \mathbb{R}^{n+1}$ steht jeweils für Approximationen an die Lösung in den Stützstellen $y_i \approx y(x_i)$. Das zweite Modell $F^h(\cdot)$ ist von zweiter Ordnung in der Gitterweite h. Dabei geht die Ordnung gleich zweimal ein: zunächst kann eine stückweise lineare Funktion höchstens quadratisch in $h \to 0$ gegen $y(x)$ konvergieren. Auf der anderen Seite beruht die Diskretisierung auf Differenzenquotienten zweiter Ordnung in h. Das Modell F^n ist zunächst eine Approximation von Grad n. Falls die Lösung $y(x)$ hinreichend regulär ist, so ist durch (6.7) eine weit bessere, nämlich exponentielle Konvergenz in n zu erwarten.

Das erste Modell hat jedoch zwei wesentliche Nachteile: die Koeffizienten des nichtlinearen Gleichungssystems (6.8) können nicht direkt angegeben werden, da die Faktoren $a^{(k)}(i, j, n)$ jeweils (numerisch) berechnet werden müssen. Zweitens kommen in jeder Gleichung sämtliche Koeffizienten $y_0, \ldots, y_n$ vor. D.h., das Gleichungssystem $F^n(\mathbf{y}) = 0$ ist global gekoppelt: jeder Koeffizient y_i steht in direkter Kopplung mit jedem anderen Koeffizienten y_j. Bei Modell $F^h(\mathbf{y}) = 0$ koppeln dagegen nur direkt benachbarte Koeffizienten. Wir kommen auf diesen wesentlichen Punkt bei der Diskussion der Lösungsverfahren zurück.

6.3 Lösungsverfahren

Wir betrachten nun allgemein das Problem $F(\mathbf{y}) = 0$ mit $\mathbf{y} \in \mathbb{R}^{n+1}$ und $F : \mathbb{R}^{n+1} \to \mathbb{R}^{n+1}$. Dieses nichtlineare Gleichungssystem ist als Nullstellenproblem formuliert. Die Lösung kann nicht direkt bestimmt werden. Daher approximieren wir $\mathbf{y}$ mit Hilfe eines Iterationsverfahrens. Hier bietet sich das Newton-Verfahren im $\mathbb{R}^{n+1}$ an. Wir wählen eine Startlösung, z.B. $\mathbf{y}^0 = 0$ und iterieren für $t = 0, 1, 2, \ldots$

$$\mathbf{y}^{t+1} = \mathbf{y}^t - F'(\mathbf{y}^t)^{-1} F(\mathbf{y}^t).$$

Die Ableitung $\mathbf{A}^t := F'(\mathbf{y}^t)$ ist eine Matrix. Daher formulieren wir das Verfahren in Form einer Defekt-Korrektur:

$$\mathbf{A}^t(\mathbf{y}^{t+1} - \mathbf{y}^t) = -F(\mathbf{y}^t), \quad \mathbf{A}^t := F'(\mathbf{y}^t).$$

Zunächst müssen wir die Voraussetzungen des Newton-Verfahrens überprüfen. Die Funktion $F(\cdot)$ muss (siehe Satz 5.12):

1. Stetig differenzierbar sein. Wir berechnen die Ableitungen von F^n und F^h:

$$\mathbf{A}^t_{ij} := \frac{\partial F^n_i}{\partial y_j} = a^{(2)}(i,j,n) - \frac{3}{2}\mu f(x_i)\sqrt{1 + \left(\sum_{k=0}^{n} a^{(1)}(i,k,n) y_k\right)^2}\, 2 \left(\sum_{k=0}^{n} a^{(1)}(i,k,n) y_k\right) a^{(1)}(i,j,n),$$

$$\frac{\partial F^h_i}{\partial y_j} = \begin{cases} \frac{1}{h^2} + \frac{3}{2h}\mu f(x_i)\sqrt{1 + \left(\frac{y_{i+1} - y_{i-1}}{2h}\right)^2}\left(\frac{y_{i+1} - y_{i-1}}{2h}\right) & j = i-1, \\ -\frac{2}{h^2} & j = i, \\ \frac{1}{h^2} - \frac{3}{2h}\mu f(x_i)\sqrt{1 + \left(\frac{y_{i+1} - y_{i-1}}{2h}\right)^2}\left(\frac{y_{i+1} - y_{i-1}}{2h}\right) & j = i+1, \\ 0 & \text{sonst.} \end{cases}$$

   Der Term unter der Wurzel ist stets größer gleich eins. Daher existiert die Ableitung für alle $\mathbf{y}$.

2. Eine gleichmäßig beschränkte und invertierbare Jacobimatrix besitzen. Die Invertierbarkeit dieser Matrix ist schwer zu prüfen. Im Fall F^h ist die Analyse verhältnismäßig einfach: für sehr kleines h ist der erste Anteil der Matrix $2h^{-2}$ und $-h^{-2}$ dominant. Die Matrix ist dann also eine kleine Störung der bereits bekannten Modellmatrix, deren Invertierbarkeit wir bereits kennen. Eine kleine Störung einer Matrix ist nach Hilfsatz 4.17 wieder invertierbar.

Da die Funktion $F(\cdot)$ nichtlinear ist, hängt die Jacobi-Matrix von der Approximation $\mathbf{y}$ ab.

6.4 Ein Beispiel

Wir werden mit einem Beispiel den Diskretisierungsansatz prüfen. Als Beispiel wollen wir eine Funktion $f(x)$ sowie einen Materialparameter μ so wählen, dass wir die exakte Lösung $y(x)$ analytisch bestimmen können. So können wir die numerischen Approximationen y^h sowie y^n auf ihre Genauigkeit überprüfen. Da nichtlineare Differentialgleichungen im Allgemeinen nicht einfach analytisch gelöst werden können, gehen wir hier umgekehrt vor: wir wählen eine gewünschte Lösung $y(x)$ und bestimmen durch Einsetzen in die Differentialgleichung (6.3) die zugehörige rechte Seite $f(x)$. Wir wählen $\mu = 1$ und bestimmen die Lösung $y(x)$ als $y(x) = \sin(2\pi x)$. Diese Lösung erfüllt die Randdaten $y(0) = y(1) = 0$. Es gilt:

$$f(x) = \frac{y''(x)}{(1 + y'(x)^2)^{\frac{3}{2}}} = -\frac{4\sin(2\pi x)\pi^2}{(1 + 4\cos(2\pi x)^2\pi^2)^{\frac{3}{2}}}.$$

Im folgenden bestimmen wir die zugehörigen Approximationen y^n sowie y^h jeweils bei Unterteilung des Intervalls $I = [0, 1]$ in $n = 4$ bzw. in $n = 8$ Teilintervalle.

6.4.1 Globaler Polynomansatz

Wir starten das Newton-Verfahren mit der Anfangsnäherung $\mathbf{y}^0 = 0$, da keine bessere Näherung bekannt ist. Im Folgenden führen wir solange Newton-Schritte $\mathbf{y}^1, \mathbf{y}^2, \ldots$ durch, bis ein Newton-Residuum $|F(\mathbf{y}^t)| < 10^{-6}$ erreicht wird.

- Schritt 1. Es gilt für $n = 4$ sowie $n = 8$ bei $\mathbf{y}^0 = 0$:

  $$\|F^4(\mathbf{y}^0)\| \approx 10, \quad \|F^8(\mathbf{y}^0)\| \approx 30.$$

  Die beiden Einträge $F_0 = F_8 = 0$ sind Null, da hier die Gleichung einfach durch $y_0 = y_n = 0$ gegeben ist. Diese ist bereits erfüllt. Im nächsten Schritt berechnen wir die Jacobimatrix $\mathbf{A}^0 := f'(\mathbf{y}^0)$:

  $$\mathbf{A}^{n=4} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 14.67 & -26.67 & 8 & 5.33 & -1.33 \\ -1.33 & 21.33 & -40 & 21.33 & -1.33 \\ -1.33 & 5.33 & 8 & -26.67 & 14.67 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix},$$

und im Fall $n = 8$:

$$\mathbf{A}^0 \approx \begin{bmatrix} 1.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 41.49 & 1.625 & -265.6 & 489.6 & -470.2 & 300.8 & -124.8 & 30.43 & -3.314 \\ -3.314 & 71.31 & -117.7 & 12.80 & 72.0 & -52.62 & 22.40 & -5.486 & 0.5968 \\ 0.5968 & -8.686 & 92.80 & -167.8 & 88.0 & -3.200 & -2.489 & 0.9143 & -0.1143 \\ -0.1143 & 1.625 & -12.80 & 102.4 & -182.2 & 102.4 & -12.80 & 1.625 & -0.1143 \\ -0.1143 & 0.9143 & -2.489 & -3.200 & 88.0 & -167.8 & 92.80 & -8.686 & 0.5968 \\ 0.5968 & -5.486 & 22.40 & -52.62 & 72.0 & 12.80 & -117.7 & 71.31 & -3.314 \\ -3.314 & 30.43 & -124.8 & 300.8 & -470.2 & 489.6 & -265.6 & 1.625 & 41.49 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix}$$

Diese 9×9-Matrix ist mit $\text{cond}_2(A)^{n=4} \approx 90$ sowie $\text{cond}_2(A)^{n=8} \approx 2\,500$ sehr schlecht konditioniert. Es zeigt sich, dass die Konditionszahl von $\mathbf{A}$ exponentiell in n steigt für $n = 16$ gilt bereits $\text{cond}_2(A) \approx 10^6$, für $n = 32$ bereits 10^{13}. Da die Matrizen $\mathbf{A}^t$ weder dünn besetzt sind noch eine spezielle Struktur aufweisen bietet sich zum Lösen nur ein direktes Verfahren an. Beide Matrizen sind nicht diagonaldominant. Einfache iterative Verfahren werden somit vermutlich nicht konvergieren. Wir verwenden die LR-Zerlegung und berechnen:

$$\mathbf{y}^1 - \mathbf{y}^0 = -[\mathbf{A}^0]^{-1} F(\mathbf{y}^0).$$

Wir geben hier die Zwischenschritte nicht an.

- Schritt 2:
$$\|F^4(\mathbf{y}^1)\|_2 \approx 4, \quad \|F^8(\mathbf{y}^1)\|_2 \approx 10.$$
- Schritt 3:
$$\|F^4(\mathbf{y}^2)\|_2 \approx 0.7, \quad \|F^8(\mathbf{y}^2)\|_2 \approx 5.$$
- Schritt 4:
$$\|F^4(\mathbf{y}^3)\|_2 \approx 0.05, \quad \|F^8(\mathbf{y}^3)\|_2 \approx 2.5$$
- Schritt 5:
$$\|F^4(\mathbf{y}^4)\|_2 \approx 3 \cdot 10^{-4}, \quad \|F^8(\mathbf{y}^4)\|_2 \approx 0.9$$
- Schritt 6:
$$\|F^4(\mathbf{y}^5)\|_2 \approx 1 \cdot 10^{-8}, \quad \|F^8(\mathbf{y}^5)\|_2 \approx 0.03.$$

Für $n = 4$ ist das gewünschte Residuum erreicht und die approximative Lösung lautet:

$$\mathbf{y}^{n=4} \approx \begin{pmatrix} 0 \\ 0.70853 \\ 1.0025 \\ 0.70853 \\ 0 \end{pmatrix}$$

Diese Lösung ist mit dem folgenden Fehler versehen:

$$\max_{i=0,\dots,n} \frac{|y_i^{n=4} - y(x_i)|}{|y(x_i)|} = 0.0025,$$

- Schritt 7: wir fahren mit $n = 8$ fort:

$$\|F^8(\mathbf{y}^6)\|_2 \approx 5 \cdot 10^{-6},$$

- Schritt 8:

$$\|F^8(\mathbf{y}^7)\|_2 \approx 10^{-10},$$

mit der Lösung:

$$\mathbf{y}^{n=8} \approx \begin{pmatrix} 0 \\ 0.382681 \\ 0.707105 \\ 0.923878 \\ 0.999998 \\ 0.923878 \\ 0.707105 \\ 0.382681 \\ 0 \end{pmatrix}.$$

Diese Lösung ist mit dem folgenden relativen Fehler versehen:

$$\max_{i=0,\dots,n} \frac{|y_i^{n=8} - y(x_i)|}{|y(x_i)|} = 0.0000054.$$

Mit den berechneten Koeffizienten $\mathbf{y}$ kann das Lösungspolynom in der Lagrangedarstellung angegeben werden. Durch Verdoppelung der Anzahl der Unbekannten von $n = 4$ auf $n = 8$ wird die Approximation um den Faktor 500 verbessert! Dieser große Zugewinn liegt an der exponentiellen Fehlerabschätzung der globalen Interpolation 6.7.

6.4.2 Stückweiser Polynomansatz

Wir wählen wieder die Diskretisierungen $n = 4$ sowie $n = 8$ und starten die Iteration mit $\mathbf{y}^0 = 0$. Wir iterieren mit dem Newton-Verfahren bis das Residuum die Schwelle 10^{-6} erreicht:

- Schritt 1: Es gilt mit (6.9) für das erste Residuum

$$\|F^4(\mathbf{y}^0)\|_\infty \approx 10, \quad \|F^8(\mathbf{y}^0)\| \approx 10.$$

Die Jacobi-Matrizen ergeben sich zu

$$\mathbf{A}^4 := \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 16 & -32 & 16 & 0 & 0 \\ 0 & 16 & -32 & 16 & 0 \\ 0 & 0 & 16 & -32 & 16 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix},$$

im Fall von $n = 4$ und

$$\mathbf{A}^8 := \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 64 & -128 & 64 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 64 & -128 & 64 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 64 & -128 & 64 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 64 & -128 & 64 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 64 & -128 & 64 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 64 & -128 & 64 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 64 & -128 & 64 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix},$$

im Fall $n = 8$. Für die Konditionszahlen gilt $\text{cond}_2(\mathbf{A}^4) \approx 90$ und $\text{cond}_2(\mathbf{A}^8) \approx 500$. Die Matrizen sind Tridiagonal-Matrizen. Das LGS kann z.B. sehr effizient mit der LR-Zerlegung für Bandmatrizen, in diesem Fall mit dem Thomas-Algorithmus (Satz 4.37) gelöst werden. Für $\mathbf{y}^0 = 0$ erfüllen darüber hinaus beide Matrizen das schwache Zeilensummenkriterium (siehe Satz 5.28), d.h., Jacobi- sowie Gauß-Seidel-Verfahren könnten ebenso eingesetzt werden. Wir wählen hier den direkten Löser.

Wir berechnen das Update:

$$\mathbf{A}(\mathbf{y}^1 - \mathbf{y}^0) = -F(\mathbf{y}^0),$$

ohne die Zwischenschritte hier anzugeben.

- Schritt 2: es gilt:
$$\|F^4(\mathbf{y}^1)\|_2 \approx 1.63, \quad \|F^8(\mathbf{y}^1)\|_2 \approx 2.38$$

- Schritt 3: es gilt:
$$\|F^4(\mathbf{y}^2)\|_2 \approx 0.22, \quad \|F^8(\mathbf{y}^2)\|_2 \approx 0.73$$

- Schritt 4: es gilt:
$$\|F^4(\mathbf{y}^3)\|_2 \approx 0.01, \quad \|F^8(\mathbf{y}^3)\|_2 \approx 0.21$$

- Schritt 5: es gilt:
$$\|F^4(\mathbf{y}^4)\|_2 \approx 10^{-10}, \quad \|F^8(\mathbf{y}^4)\|_2 \approx 0.015$$

Die Lösung $\mathbf{y}^{n=4}$ ist gegeben als:

$$\mathbf{y}^{n=4} \approx \begin{pmatrix} 0 \\ 0.5373 \\ 0.8457 \\ 0.5473 \\ 0 \end{pmatrix},$$

und ist mit folgendem relativen Fehler versehen:

$$\frac{\|\mathbf{y} - \mathbf{y}^{n=4}\|_\infty}{\|\mathbf{y}\|_\infty} \approx 0.17.$$

- Schritt 6: wir fahren mit $n = 8$ fort:

$$\|F^8(\mathbf{y}^5)\|_2 \approx 5 \cdot 10^{-4},$$

- Schritt 7: für $n = 8$:

$$\|F^8(\mathbf{y}^6)\|_2 \approx 10^{-10},$$

mit der Approximation:

$$\mathbf{y}^{n=8} = \begin{pmatrix} 0 \\ 0.320 \\ 0.604 \\ 0.808 \\ 0.885 \\ 0.808 \\ 0.604 \\ 0.320 \\ 0 \end{pmatrix},$$

mit einem Fehler:

$$\frac{\|\mathbf{y} - \mathbf{y}^{n=8}\|_\infty}{\|\mathbf{y}\|_\infty} \approx 0.11.$$

Da der Fehler bei der zweiten Approximation $n = 8$ immer noch sehr groß ist, bestimmen wir - ohne die Zwischenschritte anzugeben - weitere Approximationen. Hier zeigt sich quadratische Konvergenz in h:

n	$\|y^n - y\|_\infty / \|y\|_\infty$	$\|y^h - y\|_\infty / \|y\|_\infty$
4	$2.5 \cdot 10^{-3}$	$2 \cdot 10^{-1}$
8	$5 \cdot 10^{-6}$	$1 \cdot 10^{-1}$
16		$2 \cdot 10^{-2}$
32		$5 \cdot 10^{-3}$

6.4.3 Analyse und Vergleich

Beide Diskretisierungsansätze eignen sich prinzipiell zur Approximation des Anwendungsproblems. Es zeigen sich jedoch wesentliche Unterschiede. Das globale Polynom vierten Grades erzeugt etwa die gleiche Approximationsgüte wie das stückweise Polynom mit $n = 32$ Teilintervallen! Dieser große Unterschied beruht auf der exponentiellen Konvergenzrate der Interpolation im Fall $n \to \infty$.

Die gute Approximation muss durch eine aufwändige Lösung erkauft werden: in jedem Schritt der Newton-Iteration muss ein lineares Gleichungssystem mit voll besetzter Matrix gelöst werden. Spezielle Bandstrukturen können nicht ausgenutzt werden. Der Stückweise Ansatz hingegen erzeugt sehr dünne Bandmatrizen, die auch für sehr große Systeme effizient invertiert werden können. Hinzu kommt, dass der globale Polynomansatz eine sehr schlecht konditionierte Matrix erzeugt. Dies spielt bei den hier betrachteten sehr kleinen Problemen noch keine wesentliche Rolle. Die schlechte Kondition kommt aber z.B. bei der Diskretisierung von zwei- oder dreidimensionalen Problemen schnell zu tragen. Der globale Polynomansatz wird bei $n > 20$ aufgrund von Rundungsfehleranfälligkeit versagen. Der stückweise Ansatz kann (und wird) auch für sehr große $n > 1\,000\,000$ Systeme durchgeführt.

Schließlich wird die hier dargestellte Situation durch das gewählte Beispiel verzerrt. Die Lösung $y(x)$ ist analytisch vorgegeben und es gilt $y \in C^\infty(I)$. Nur bei dieser hohen Differenzierbarkeit kann der globale Polynomansatz seine volle Ordnung erreichen. In der praktischen Anwendung ist die Lösung meist nicht bekannt. Stattdessen werden Kräfte vorgeben. Eine übliche Kraft kann z.B. die stückweise Belastung des Balkens sein:

$$f(x) = \begin{cases} 0 & x < \frac{1}{4} \\ -1 & \frac{1}{4} \le x \le \frac{1}{2} \\ 0 & x > \frac{1}{2} \end{cases}.$$

Diese Funktion f ist nicht mehr differenzierbar, sie ist sogar nicht mehr stetig. Es zeigt sich, dass in diesem Fall auch die Lösung $y(x)$ nicht mehr über beliebige Regularität verfügt. Die Vereinfachung $y''(x) \approx f(x)$ legt nahe, dass $y(x)$ die zwei-fache Stammfunktion zu $f(x)$ ist und dass daher $y \in C^1(I)$ gilt. In diesem Fall kann der globale Polynomansatz nicht mehr die volle Ordnung erreichen und eine stückweise Diskretisierung ist überlegen. In Abbildung 6.3 zeigen wir für dieses Beispiel die Lösungen zu $n = 4$ und $n = 8$, jeweils mit globalem Polynomansatz und mit stückweise definiertem Ansatz. Der globale Ansatz zeigt ein unphysikalisches Verhalten: obwohl die Kraft nur nach unten geht, wird der Balken am Rand nach oben ausgelenkt. Dies ist gerade die numerische Instabilität der globalen Lagrange-Interpolation an den Intervallenden!

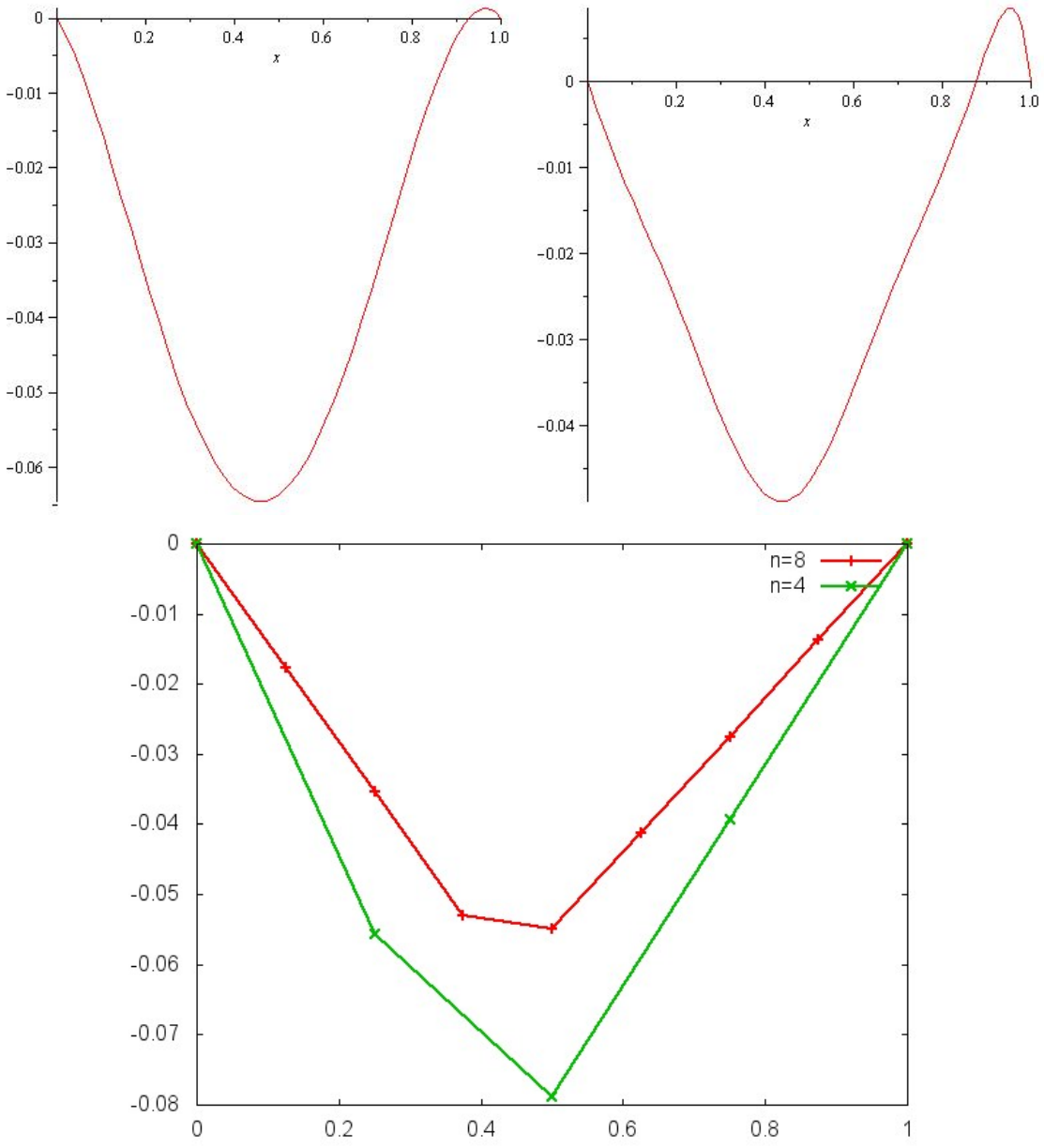

Abbildung 6.3: Oben: Diskretisierung mit globalem Polynomansatz. Links $n = 4$ und rechts $n = 8$. Unten: Diskretisierung mit stückweise linearen Polynomen.

Index