Numerische Mathematik - Mitschrieb

bei Prof. Dr. Christof Eck

Jan-Cornelius Molnar, Version: 10. September 2009 16:03

Inhaltsverzeichnis

1 Darstellung von Zahlen, Fehlerfortpflanzung

1-A Darstellung von Zahlen

Um Zahlen in numerischen Verfahren verwenden zu können bedarf es einer Darstellung, die von Computern gespeichert und mit hinreichender Geschwindigkeit verarbeitet werden kann. Die endliche Speicher- und Rechenkapazität von Computern führt unweigerlich dazu, dass Größe und Genauigkeit der verwendbaren Zahlen beschränkt sind.

Auch wenn wir die Details der Informatik überlassen wollen, soll dieser Abschnitt eine Übersicht über die uns zur Verfügung stehenden Darstellungen und deren Grenzen geben. Beim Entwurf von Algorithmen sind diese Grenzen stets im Hinterkopf zu behalten.

■ Ganze Zahlen

Ganze Zahlen können wir durch b-adische Brüche der Form

$$\pm a_n a_{n-1} \dots a_0 \mathrel{\widehat{=}} \sum_{j=0}^{n} a_j b^j,$$

darstellen. Die Basis b wählen wir aus $\{2, 3, \dots\}$, während wir die Ziffern a_j aus $\{0, 1, \dots, b-1\}$ wählen.

Bsp (i) Wohlbekannt ist uns das Dezimalsystem ($b = 10$). Die Zahl 1203 hat hier die Darstellung,

$$1203_{(10)} \mathrel{\widehat{=}} 1 \cdot 10^3 + 2 \cdot 10^2 + 0 \cdot 10^1 + 3 \cdot 10^0.$$

(ii) Computer hingen verwenden zum Speichern von Daten in der Regel das Dualsystem ($b = 2$),

$$10010_{(2)} \mathrel{\widehat{=}} 1 \cdot 2^4 + 1 \cdot 2^1 = 18. \quad ■$$

Bei Computerdatenformaten ist die Anzahl der Ziffern oft vorgegeben und somit sind nur endlich viele Zahlen darstellbar. Beim Über- bzw. Unterschreiten der maximal bzw. minimal darstellbaren Zahl treten sog. Overflowfehler auf. Jede ganze Zahl bis zur minimal bzw. maximal darstellebaren Zahl ist jedoch exakt darstellbar.

Datentyp	Darstellbarer Bereich
int	-32768 ... 32767
long	-2147483648... 2147483647

1 **Mindestanforderungen der Programmiersprache C.**

■ Reelle Zahlen

Reelle Zahlen lassen sich als Fließkommazahlen (engl. floatpoint) darstellen,

$$\underbrace{\pm a_0.a_1\dots a_n}_{\text{Mantisse}}\,\mathbf{E}\,\underbrace{\pm c_m c_{m-1}\dots c_0}_{\text{Exponent}} \mathrel{\hat{=}} \left(\sum_{j=0}^{n} a_j b^{-j}\right) b^{\sum_{n=0}^{m} c_j b^j}.$$

Die Basis b wählen wir aus $\{2,3,\dots\}$, a_j und c_k aus $\{0,1,\dots,b-1\}$. Es gilt die Konvention $a_0 \neq 0$ für eine Zahl ungleich Null.

1.1 **Bsp** Im Dualsystem ($b = 2$)

$$-1.01\mathbf{E}+110 \mathrel{\hat{=}} -(1+1\cdot 2^{-2})\cdot 2^{1\cdot 2^2+1\cdot 2^1} = -\frac{5}{4}\cdot 2^6 = -5\cdot 2^4 = -80. \quad ■$$

Auch hier ist bei Computerdatenformaten die Länge der Mantisse $(n+1)$ und die Länge des Exponenten $(m+1)$ fest vorgegeben. Es sind also nur endlich viele Zahlen darstellbar und es gibt eine größte bzw. kleinste darstellbare Zahl

$$z_{\max} = (b-1).(b-1)(b-1)\dots(b-1)\mathrm{E}(b-1)\dots(b-1).$$

Die kleinste darstellbare Zahl > 0 ist gegeben durch

$$z_{\min} = 1.0\dots 0\mathbf{E}-(b-1)\dots(b-1).$$

Neben den Overflowfehlern bei $|\text{Zahl}| > |z_{\max}|$ können daher auch Underflowfehler auftreten, wenn $|\text{Zahl}| \neq 0$ und $|\text{Zahl}| < z_{\min}$.

Jede Zahl $z \in [z_{\min}, z_{\max}] \cup \{0\} \cup [-z_{\max}, -z_{\min}]$ kann bis auf einen relativen Fehler durch eine Fließkommazahl approximiert werden,

$$\frac{|z - \tilde{z}|}{|z|} \leq \varepsilon = \frac{1}{2} b^{-n},$$

ε heißt Maschinengenaugigkeit.

Typ	$n+1$	$m+1$	$z_{\max}$	$z_{\min}$	ε
"single precision"	23	8	$\approx 10^{38}$	$\approx 10^{-38}$	10^{-7}
"double precision"	52	11	$\approx 10^{308}$	$\approx 10^{-308}$	10^{-16}

2 **Typische Werte bei $b = 2$.**

1-B Fehlerfortpflanzung

Wie wir im letzten Abschnitt gesehen haben, arbeitet man in der numerischen Mathematik lediglich mit Approximationen. Zudem sind häufig bereits die vorliegenden Daten "ungenau", d.h. mit einem gewissen Fehler behaftet. Nun ist es für den Entwurf und die Verwendung von Algorithmen von großem Interesse, wie sich dieser Fehler durch das Anwenden von elementaren Operationen verändert, um Aussagen über die Qualität des Ergebnisses machen zu können.

Dazu definieren wir zunächst für eine exakte Zahl $z \in \mathbb{R}$ und ihre Approximation $\tilde{z} \in \mathbb{R}$ den absoluten Fehler

$$e_A = |z - \tilde{z}|,$$

und den relative Fehler

$$e_R = \frac{|z - \tilde{z}|}{|z|}.$$

Im Folgenden wollen wir untersuchen, wie sich dieser Fehler bei elementaren Operationen verhält. Seien dazu

- $x, y \in \mathbb{R}$ exakte Zahlen,
- $\tilde{x}, \tilde{y}$ Approxmationen,
- $\Delta x = \tilde{x} - x$, $\Delta y = \tilde{y} - y$.

■ Addition

- $z = x + y \in \mathbb{R}$ exakte Summe,
- $\tilde{z} = \tilde{x} + \tilde{y}$ Approxmation.

Der *absolute Fehler* der Addition ist gegeben durch,

$$\begin{aligned} e_A &= |\tilde{z} - z| = |x + y - (x + \Delta x) - (y + \Delta y)| \\ &= |\Delta x + \Delta y| \leq |\Delta x| + |\Delta y| . \end{aligned}$$

e_A ist "klein", wenn $|\Delta x|$ und $|\Delta y|$ "klein" sind. Der *relative Fehler*

$$e_R = \frac{e_A}{|x + y|} = \frac{|\Delta x + \Delta y|}{|x + y|},$$

hingegen kann auch für kleine $|\Delta x|$ und $|\Delta y|$ groß werden, nämlich wenn $|x + y|$ sehr klein wird.

Bsp Sei ein Dezimalsystem mit 6-Stellen Genauigkeit gegeben.

$$\begin{aligned} \tilde{x} &= 1,93251 \cdot 10^{-3}, \\ \tilde{y} &= -1,93248 \cdot 10^{-3}, \\ \tilde{x} + \tilde{y} &= 0,00003 \cdot 10^{-3}. \end{aligned}$$

Von 6-Stellen ist noch genau eine übrig. Diese Phänomen nennt man Auslöschung und tritt immer dann auf, wenn man zwei gleich große Zahlen subtrahiert und daher das Ergebnis nahe bei Null liegt. ■

■ Multiplikation

- $z = x \cdot y$ exaktes Produkt,
- $\tilde{z} = \tilde{x} \cdot \tilde{y}$ Approximation.

Absoluter Fehler.

$$\begin{aligned} e_A &= |\tilde{z} - z| = |x \cdot y + x \cdot \Delta y + \Delta x \cdot y + \Delta x \cdot \Delta y - x \cdot y| \\ &= |x \cdot \Delta y + \Delta x \cdot y + \Delta x \cdot \Delta y| . \end{aligned}$$

Relativer Fehler.

$$e_R = \frac{\tilde{z} - z}{|z|} = \frac{|x \cdot \Delta y + \Delta x \cdot y + \Delta x \cdot \Delta y|}{|xy|}$$
$$\leq \underbrace{\frac{|\Delta y|}{|y|} + \frac{|\Delta x|}{|x|}}_{\text{rel. Fehler von } x \text{ und } y} + \underbrace{\frac{|\Delta x|}{|x|}\frac{|\Delta y|}{|y|}}_{\text{Verstärkung}}.$$

e_R wird durch die Multiplikation nur schwach verstärkt. Die Multiplikation ist daher im Gegensatz zur Addition eine sehr gutartige Operation.

■ Funktionsauswertung

Seien D ein offenes Intervall, $f \in C^1(D \to \mathbb{R})$[1] und

- $x \in D$ exaktes Argument,
- $\tilde{x} \in D$ Approximation,
- $f(x)$ exakter Funktionswert,
- $f(\tilde{x})$ Approximation.

Wir vernachlässigen etwaige Maschinenfehler bei der Funktionsauswertung selbst, da wir lediglich an der Fehlerfortpflanzung durch die Auswertung an $\tilde{x}$ anstatt von x interessiert sind.

f ist differenzierbar in x und besitzt daher die Darstellung

$$f(\tilde{x}) = f(x) + f'(x)\Delta x + o(\Delta x) \approx f(x) + f'(x)\Delta x,$$

mit der wir die Fehler elegant angeben können.

Absoluter Fehler.

$$e_A = |f(\tilde{x}) - f(x)| = |f'(x)|\,|\Delta x|.$$

Der Fehler von x wird also mit $|f'(x)|$ verstärkt. $|f'(x)|$ heißt hier Verstärkungsfaktor.

[1] Die Forderung, dass f für die Funktionsauswertung stetig differenzierbar sein muss, erscheint zunächst scharf; die in der Numerik vorliegenden Funktionen sind jedoch in der Regel hinreichend glatt und die Differenzierbarkeit erlaubt eine elegante Fehlerabschätzung.

Relativer Fehler.

$$e_R = \frac{|f(\tilde{x}) - f(x)|}{|f(x)|} = \frac{|f'(x)|\,|\Delta x|}{|f(x)|} = \frac{|f'(x)|\,|x|}{|f(x)|}\frac{|\Delta x|}{|x|}.$$

Der Verstärkungsfaktor ist hier $\frac{|f'(x)||x|}{|f(x)|}$.

Bsp (i) $f(x) = x^n$, $n \in \mathbb{N}$. Die Verstärkung des absoluten Fehlers beträgt $n\,|x^{n-1}|$, die des relativen Fehlers n.

(ii) $f(x) = \sin x$. Die Verstärkung des absoluten Fehlers ist $|\cos x| \leq 1$, die des relativen Fehlers $\frac{|\cos x|}{|\sin x|}\,|x|$. Letztere wird groß, wenn $x \approx k\pi$ mit $k \in \mathbb{Z} \setminus \{0\}$. ■

■ Numerische Berechnung von Ableitungen

Sei D offenes Intervall und $f \in C^2(D \to \mathbb{R})$. Die erste Ableitung lässt sich approximieren durch, $f'(x) \approx \frac{f(x+h)-f(x)}{h}$.

Fehlerabschätzung. Mithilfe der Taylorentwicklung plus Restglied erhalten wir

$$f(x+h) = f(x) + hf'(x) + \frac{h^2}{2}f''(\xi), \quad \xi \in [x, x+h].$$

Der Fehler der Approximation lässt sich wie folgt abschätzen

$$\left| f'(x) - \frac{f(x+h) - f(x)}{h} \right| = \frac{h}{2}\,|f''(\xi)| \leq \frac{h}{2}C_{f''},$$

wobei $C_{f''}$ eine obere Schranke für $|f''(\xi)|$ ist.

Wie wir sehen, hängt der Fehler von h ab, es ist aber oft nicht sinnvoll h kleinstmöglich zu wählen, da aufgrund der endlichen Maschinengenauigkeit Auslöschung o.ä. Effekte auftreten können. Es stellt sich also die Frage, wie man h bei gegebener Maschinengenauigkeit ε optimal wählen kann.

Realisierung der Approximation.

$$f'(x) \approx \frac{f(x+h) + \varepsilon_1 - (f(x) - \varepsilon_2) + \varepsilon_3}{h} + \varepsilon_4.$$

- ε_1 Fehler der Auswertung von $f(x+h)$,

- ε_2 Fehler der Auswertung von $f(x)$,
- ε_3 Fehler der Subtraktion,
- ε_4 Fehler der Division.

Angenommen $\left|\varepsilon_j\right| \leq \varepsilon$ für $j = 1, \ldots, 4$, dann gilt

$$\begin{aligned}\Delta f' &= \left| f'(x) - \frac{\left| f(x+h) - f(x) + \varepsilon_1 + \varepsilon_2 + \varepsilon_3) \right|}{h} - \varepsilon_4 \right| \\ &\leq \frac{h}{2} C_{f''} + \frac{3\varepsilon}{h} + \varepsilon.\end{aligned}$$

Da $\varepsilon << \frac{\varepsilon}{h}$ für $h < 1$, können wir den letzten Term vernachlässigen und erhalten,

$$\Delta f' \approx \frac{h}{2} C_{f''} + \frac{3\varepsilon}{h}.$$

Der Ausdruck wird minimal für $h = \sqrt{\frac{6\varepsilon}{C_{f''}}} \approx \sqrt{\varepsilon}$. Wollen wir also h optimal wählen, sollte $h \approx \sqrt{\varepsilon}$ sein. Der Fehler der Approximation von f' beträgt in diesem Fall $\sqrt{2\varepsilon C_{f''}}$.

Bsp Bei "**single precision**" ist $\varepsilon \approx 2.4 \cdot 10^{-8}$.
Optimales h $\quad h \approx \sqrt{8} \approx 5 \cdot 10^{-4}$
Genauigkeit $\quad \sqrt{\varepsilon} \approx 5 \cdot 10^{-4}$
Man verliert somit beim Differenzieren ca. die Hälfte der Stellen an Genauigkeit.

2 Lineare Gleichungssysteme

Wir kennen lineare Gleichungssysteme (LGS) bereits aus der Linearen Algebra. Es handelt sich hierbei um ein System von linearen Gleichungen der Form,

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, \quad i = 1, \dots, m,$$

mit $a_{ij}, b_i \in \mathbb{R}$ bekannt und $x_j \in \mathbb{R}$ gesucht. Als Kurzschreibweise verwenden wir die Matrix-Vektor-Notation

$$Ax = b, \qquad A = (a_{ij}) \in \mathbb{R}^{m \times n}, \quad b = (b_j) \in \mathbb{R}^m, \; x = (x_j) \in \mathbb{R}^n.$$

In diesem Kapitel wollen wir numerische Lösungsverfahren für diese Systeme betrachten. Das naheliegendste, die Inverse von A zu bilden und damit die Lösung $x = A^{-1}b$ zu berechnen ist numerisch "teuer", d.h. es sind sehr viele Rechenoperationen notwendig, und zudem "schwierig", da dabei sehr kleine bzw. sehr große Zahlen und damit Effekte wie Auslöschung oder Overflows auftreten können.

Es existieren jedoch zahlreiche numerische Lösungsverfahren mit unterschiedlichen Voraussetzungen an die Matrix A. In der Regel sind allgemeine Verfahren langsamer als diejenigen, die spezielle Anforderungen an A stellen. Kennt man jedoch das Problem und damit die Eigenschaften von A genau, so lässt sich oft ein optimiertes Verfahren wählen.

2-A Gauß-Elimination

Als Lösungsverfahren haben wir ebenfalls in der Linearen Algebra den Gauß Algorithmus kennengelernt. Wir verwenden dazu die erweiterte Matrix Notation

$$Ax = b \to (Ax|b).$$

Ziel ist eine Transformation auf obere Dreiecksgestalt,

$$(A|b) \to (\hat{A}|\hat{b}).$$

Bsp Betrachte eine 4×5-Matrix $(A|b)$ und führe den Gauß-Algorithmus aus,

$$\left(\begin{array}{cccc|c} 1 & 1 & 0 & 1 & 4 \\ 3 & -1 & -1 & 2 & -1 \\ -1 & 3 & 2 & -1 & 2 \\ 5 & 5 & 0 & 2 & 8 \end{array}\right) \sim \left(\begin{array}{cccc|c} 1 & 1 & 0 & 1 & 4 \\ 0 & -4 & -1 & -1 & -13 \\ 0 & 4 & 2 & 0 & 6 \\ 0 & 0 & 0 & -3 & -12 \end{array}\right)$$

$$\sim \left(\begin{array}{cccc|c} 1 & 1 & 0 & 1 & 4 \\ 0 & -4 & -1 & -1 & -13 \\ 0 & 0 & 1 & -1 & -7 \\ 0 & 0 & 0 & -3 & -12 \end{array}\right).$$

Wir lösen das LGS durch Rückwärtsauflösen,

$$\begin{aligned} x_4 &= \frac{-12}{-3} = 4, \\ x_3 &= x_3 - 7 = -3, \\ x_2 &= \frac{1}{-4}(-13 + x_3 + x_4) = \frac{1}{-4}(-13 - 3 + 4) = 3, \\ x_1 &= -x_2 - x_4 + 4 = -3 - 4 + 4 = -3. \end{aligned}$$

Die Lösung hat also die Form,

$$x = \begin{pmatrix} -3 \\ 3 \\ -3 \\ 4 \end{pmatrix}. \quad \blacksquare$$

■ Algorithmus

Wir betrachten die Gaußelimination für $n \times n$ Systeme $Ax = b$ mit $A \in \mathbb{R}^{n \times n}$, $b \in \mathbb{R}^n$ ohne Pivotisierung, d.h. ohne Vertauschung von Zeilen. Dabei gehen wir davon aus, dass auf der Diagonalen bei jedem Schritt der Elimination von Null verschiedene Einträge stehen, das LGS also insbesondere lösbar ist.

Für $i = 1 \ldots n-1$ (Spalten)
  Für $j = i+1, \ldots, n$ (Zeilen)
    $l_{ji} = \dfrac{a_{ji}}{a_{ii}}$

Für $k = i+1, \ldots, n$

$$a_{jk} = a_{jk} - l_{ji} \cdot a_{ik}$$

$$b_j = b_j + l_{ji} b_i$$

Dieser Algorithmus überschreibt die "alte" Matrix durch die "neue" obere rechte Dreiecksmatrix. Die Stellen, in der normalerweise Nullen stehen, bleiben unberührt, tragen also noch die "alten" Werte, sie werden aber auch nicht mehr benötigt.

Rückwärtssubsitution.

$$x_n = \frac{b_n}{a_{nn}}$$

Für $j = n-1, n-2, \ldots, 1$

$$x_j = \tfrac{1}{a_{ji}} \left(b_j - \sum_{k=j+1}^{n} a_{jk} x_k \right)$$

Rechenaufwand. Wir zählen die Multiplikationen und Divisionen, da diese für einen Prozessor aufwändigere Operationen darstellen, als Addieren und Subtrahieren.[2]

Gauß-Elimination:

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \left(1 + \left(\sum_{k=i+1}^{n} 1 \right) + 1 \right) = \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} n - i + 2$$

$$= \sum_{i=1}^{n-1} (n-i)(n-i+2) = \sum_{i=1}^{n-1} i^2 - (2n+2) \sum_{i=1}^{n-1} i + n(n+2) \sum_{i=1}^{n-1} 1$$

$$= \frac{(n-1)n(2n-1)}{6} - (2n+2)\frac{(n-1)n}{2} + n(n+2)(n-1)$$

$$= \frac{(n-1)n}{6} \left[2n - 1 - 6(n+1) + 6(n+2) \right]$$

$$= \frac{(n-1)n}{6} (2n+5) = \frac{(n-1)n(2n+5)}{6}$$

$$\approx \frac{1}{3} n^3 + \mathcal{O}(n^2).$$

[2] Für Menschen gilt dies im Übrigen auch - ein Beweis dafür sind die Logarithmentafeln, die aber heute fast keiner mehr kennt…

Rückwärtsauflösung:

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

Der Aufwand der Rückwärtsauflösung ist mit $\approx \frac{1}{2}n^2 + \mathcal{O}(n)$ gegenüber dem der Gauß-Elimination zu vernachlässigen. Der Aufwand zum Lösen eines LGS mit Gauß-Elimination beträgt also etwa $\frac{1}{3}n^3$ Multiplikationen und Divisionen.

2-B Die LU-Zerlegung

Wir wollen nun ein weiteres Verfahren zur Lösung von linearen Gleichungssystemen betrachten, die LU-Zerlegung. Der Name kommt aus dem Englischen von "lower-upper-seperation". In Deutscher Literatur ist auch oft von links-rechts Zerlegung die Rede.

Bei der Gauß-Elimination ist die Grundoperation die Addition vom $-l$-fachen der Zeile i zur Zeile j. Dies können wir für eine Matrix $A = (a_{jk})$ durch eine Matritzenmultiplikation darstellen,

$$A \mapsto GA, \quad a_{jk} \mapsto a_{jk} - l \cdot a_{ik},$$

mit der Matrix

$$G = \begin{pmatrix} 1 & & & & \\ & \ddots & -l_{ji} & & \\ & & \ddots & & \\ & & & \ddots & \\ & & & & 1 \end{pmatrix} = \mathrm{Id} - l_{ji} e_j e_i^t,$$

wobei $-l_{ji}$ der i-te Eintrag in der j-ten Zeile ist.

Der i-te Schritt der Gauß-Elimination lässt sich also darstellen als $A \to G_i A$,

mit

$$G_i = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & -l_{i+1,i} & \ddots & & \\ & \vdots & & \ddots & \\ & -l_{n,i} & & & 1 \end{pmatrix} = \mathrm{Id} - \sum_{j=l+1}^{n} l_{ji} e_j e_i^t,$$

Das Resultat der Gauß-Elimination lautet,

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

wobei A die ursprüngliche Matrix und U eine obere Dreiecksmatrix darstellen. Dabei sind die G_i untere Dreiecksmatritzen mit Determinante 1, also invertierbar. Setzen wir nun,

$$L = (G_{n-1} G_{n-2} \cdots G_1)^{-1} = G_1^{-1} \cdots G_{n-2}^{-1} G_{n-1}^{-1},$$

so erhalten wir die LU-Zerlegung von A durch, $A = L \cdot U$.

Der nächste Satz liefert uns detaillierte Informationen über die Gestallt der Matrix L.

2.1 **Satz** *Die Matrix L der LU-Zerlegung hat die Form,*

$$L = \begin{pmatrix} 1 & & & & \\ l_{21} & \ddots & & & \\ \vdots & \ddots & \ddots & & \\ \vdots & \ddots & \ddots & \ddots & \\ l_{n1} & \cdots & \cdots & l_{nn-1} & 1 \end{pmatrix},$$

mit den Eliminationsfaktoren $l_{ji} = \frac{a_{ji}^{(i-1)}}{a_{ii}^{(i-1)}}$, wobei $A^{(k)} = (a_{ji}^{(k)}) = G_k \cdots G_1 A$ die Matrix nach dem k-ten Schritt der Gaußelimination darstellt. ⋊

» Wir können die aus der LU-Zerlegung gewonnen G_j auch darstellen als,

$$G_j = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & -l_{j+1,j} & & \\ & & \vdots & \ddots & \\ & & -l_{n,j} & & 1 \end{pmatrix} = \mathrm{Id} - \underbrace{\begin{pmatrix} 0 \\ \vdots \\ 0 \\ -l_{j+1,j} \\ \vdots \\ -l_{n,j} \end{pmatrix}}_{:=b_j} e_j^\top$$

Damit ergibt sich,

$$(\mathrm{Id} - b_j e_j^\top)(\mathrm{Id} + b_j e_j^\top) = \mathrm{Id} + b_j e_j^\top - b_j e_j^\top - b_j e_j^\top b_j e_j^\top = \mathrm{Id} + b_j \underbrace{\left\langle e_j, b \right\rangle}_{=0} e_j^\top = \mathrm{Id},$$

also ist $\mathrm{Id} + be_j^\top$ invers zu $G = \mathrm{Id} - be_j^\top$ und damit hat G^{-1} die Form,

$$G^{-1} = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & l_{j+1,j} & & \\ & & \vdots & \ddots & \\ & & l_{n,j} & & 1 \end{pmatrix}.$$

Wir zeigen nun durch Induktion, dass

$$G_i^{-1} \cdots G_{n-1}^{-1} = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & -l_{i+1,i} & \ddots & \\ & & \vdots & \ddots & \ddots \\ & & -l_{n,i} & \cdots & l_{n,n-1} & 1 \end{pmatrix}.$$

Induktionsanfang. $i = n-1$ ist klar.
Induktionsschritt. $(i+1) \Rightarrow i$.

$$\begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & -l_{i+1,i} & \boxed{\begin{matrix} 1 & & \\ & \ddots & \\ & & 1 \end{matrix}} \\ & & \vdots & & \\ & & -l_{n,i} & & \end{pmatrix} \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & \boxed{\begin{matrix} -l_{i+1,i+1} & 1 & & \\ \vdots & \ddots & \ddots & \\ -l_{n,i+1} & \cdots & l_{n,i+1} & 1 \end{matrix}} \\ & & & & \end{pmatrix}$$

$$= \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & & -l_{i+1,i} & \ddots & \\ & & \vdots & \ddots & \ddots \\ & & -l_{n,i} & \cdots & l_{n,n-1} & 1 \end{pmatrix}.$$

Für $i = 1$ folgt daher,

$$L = G_1^{-1} \cdots G_{n-1}^{-1} = \begin{pmatrix} 1 & & & \\ l_{2,1} & \ddots & & \\ \vdots & \ddots & \ddots & \\ l_{n,1} & \cdots & l_{n,n-1} & 1 \end{pmatrix}. \quad \text{«}$$

■ Praktische Durchführung der LU-Zerlegung

Beim Berechnen der LU-Zerlegung werden die Einträge für L und U in einer einzigen Matrix gespeichert,

$$(L/U) = \begin{pmatrix} u_{1,1} & \cdots & u_{1,n} \\ l_{2,1} & \ddots & \vdots \\ \vdots & \ddots & u_{n,n} \\ l_{n,1} & \cdots & l_{n,n-1} \end{pmatrix}$$

Der Algorithmus entspricht dem der Gauß-Elimination ohne rechte Seite, wobei die Eliminationsfaktoren l_{ji} als neue Werte a_{ji} gespeichert werden.

Für $i = 1, \dots, n-1$
  Für $j = i+1, \dots, n$
    $a_{ji} = \dfrac{a_{ji}}{a_{ii}}$
    Für $k = i+1, \dots, n$
      $a_{jk} = a_{jk} - a_{ji} \cdot a_{ik}$

Rechenaufwand. Die Anzahl der Multiplikationen beträgt,

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} (1 + n - i) = \sum_{i=1}^{n-1} (n-i)(n-i+1) = \sum_{i=1}^{n-1} n^2 - ni + -ni + i^2 - i$$
$$= \sum_{i=1}^{n-1} n^2 - 2ni + n - i + i^2 = \sum_{i=1}^{n-1} (n^2 + n) - (2n+1)i + i^2$$
$$= n(n-1)(n-1) - \frac{(2n+1)(n-1)n}{2} + \frac{1}{6}(n-1)(2n-1)$$
$$= \frac{1}{6} n(n-1)\left[(2n-1) - 3(2n+1) + 6(n-1)\right]$$
$$= \frac{1}{6} n(n-1)\,(2n+2) = \frac{1}{3} n^3 - \frac{1}{3} n. \quad \multimap$$

Bsp LU-Zerlegung einer 4×4-Matrix

$$\begin{pmatrix} 1 & 1 & 0 & 1 \\ 3 & -1 & -1 & 2 \\ -1 & 3 & 2 & -1 \\ 5 & 5 & 0 & 2 \end{pmatrix} \sim \begin{pmatrix} 1 & 1 & 0 & 1 \\ 3 & -4 & -1 & -1 \\ -1 & 4 & 2 & 0 \\ 5 & 0 & 0 & -3 \end{pmatrix} \sim \begin{pmatrix} 1 & 1 & 0 & 1 \\ 3 & -4 & -1 & -1 \\ -1 & -1 & 1 & -1 \\ 5 & 0 & 0 & -3 \end{pmatrix}$$
$$\sim \begin{pmatrix} 1 & 1 & 0 & 1 \\ 3 & -4 & -1 & -1 \\ -1 & -1 & 1 & -1 \\ 5 & 0 & 0 & -3 \end{pmatrix}$$
$$\Rightarrow L = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 3 & 1 & 0 & 0 \\ -1 & -1 & 1 & 0 \\ 5 & 0 & 0 & 1 \end{pmatrix}, \quad U = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 0 & -4 & -1 & -1 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & -3 \end{pmatrix}. \quad \blacksquare$$

Wir wollen nun lineare Gleichungssysteme mit Hilfe der LU-Zerlegung lösen. Seien also $A \in \mathbb{R}^{n\times n}, b \in \mathbb{R}^n$ und $Ax = b$, sowie

$$A = LU \Rightarrow LUx = b.$$

Nun setzen wir $y = Ux$ und lösen $Ly = b$ durch Vorwärtsauflösen und anschließend $Ux = y$ durch Rückwärtsauflösen. Da L und U bereits Dreiecksgestallt haben, ist dieses Vorgehen sehr schnell.

Wir können dazu die folgenden Algorithmen verwenden:
Vorwärtsauflösen.

$$\begin{aligned} &y_1 = b_1 \\ &\text{Für } i = 2, \dots, n \\ &\qquad y_i = b_i - \sum_{j=1}^{i-1} l_{ij} y_j \end{aligned}$$

Rückwärtsauflösen.

$$\begin{aligned} &x_n = \frac{y_n}{u_{nn}} \\ &\text{Für } i = n-1, \dots, 1 \\ &\qquad x_i = \tfrac{1}{u_{ii}} \left(y_i - \sum_{j=i+1}^{n} u_{ij} x_j \right) \end{aligned}$$

Der Rechenaufwand dieser Operationen beträgt,

Rechenaufwand. Vorwärtsauflösen

$$\sum_{i=2}^{n} (i-1) = \sum_{i=1}^{n-1} i = \frac{(n-1)n}{2}.$$

Rückwärtsauflösen

$$\frac{n(n+1)}{2}.$$

Es werden also $\approx n^2$ Operationen (Multiplikationen / Divisionen) benötigt, um das Gleichungssystem mit einer vorhanden LU-Zerlegung zu lösen.

Der Aufwand der Vorwärts- und Rückwärtsauflösung entspricht gerade dem Aufwand einer Matrix-Vektormultiplikation (wie $A^{-1}b$). Die LU-Zerlegung ist also numerisch "genau so viel wert" wie die Inverse der Matrix, aber viel billiger (mit weniger Rechenaufwand) zu bestimmen. Deshalb wird in der Numerik "fast nie" die Inverse einer Matrix bestimmt, um ein Gleichungssystem zu lösen.

Nun stellt sich natürlich die Frage, wann die LU-Zerlegung überhaupt existiert.

2.2 **Satz** *Sei $A \in \mathbb{R}^{n\times n}$. Dann gilt: Die LU-Zerlegung von A existiert und ist eindeutig genau dann wenn die Hauptuntermatrizen $A_m := (a_{ij})_{i,j=1}^{m}$ regulär sind für $m = 1, 2, \ldots, n-1$.* ⋊

» Induktion über n.

Induktionsanfang $n = 1$. $A = (a_{11})$ mit der LU-Zerlegung $= \underbrace{(1)}_{L}\underbrace{(a_{11})}_{U}$.

Induktionsschritt $n-1 \Rightarrow n$. Setze,

$$A = \begin{pmatrix} A_{n-1} & c \\ d^\top & a_{nn} \end{pmatrix}, \quad \text{mit} \quad A_{n-1} \in \mathbb{R}^{n-1\times n-1}, c, d \in \mathbb{R}^{n-1}, a_{nn} \in \mathbb{R}.$$

Ansatz für die LU-Zerlegung:

$$L = \begin{pmatrix} L_{n-1} & 0 \\ l^\top & 1 \end{pmatrix}, \quad \text{mit} \quad L_{n-1} \in \mathbb{R}^{n-1\times n-1}, l \in \mathbb{R}^{n-1},$$
$$U = \begin{pmatrix} U_{n-1} & u \\ 0 & u_{nn} \end{pmatrix}, \quad \text{mit} \quad U_{n-1} \in \mathbb{R}^{n-1\times n-1}, u \in \mathbb{R}^{n-1}$$

Wir zeigen nun, dass $A = LU$

$$\begin{pmatrix} A_{n-1} & c \\ d^\top & a_{nn} \end{pmatrix} = \begin{pmatrix} L_{n-1} & 0 \\ l^\top & 1 \end{pmatrix}\begin{pmatrix} U_{n-1} & u \\ 0 & u_{nn} \end{pmatrix}$$

äquivalent ist zu,

$$\begin{cases} A_{n-1} = L_{n-1}U_{n-1} \\ L_{n-1}u + 0 \cdot u_{nn} = c \\ d^\top = l^\top \cdot U_{n-1} \Leftrightarrow U_{n-1}^\top l = d \\ a_{nn} = l^\top u + u_{nn} \end{cases}$$

"⇐": Wir wenden die Induktionsvoraussetzung auf A_{n-1} an, also existieren L_{n-1} und U_{n-1} und sind eindeutig. A_{n-1} ist regulär also gilt,

$$0 \neq \det(A_{n-1}) = \det(L_{n-1})\det(U_{n-1}),$$

und daher sind L_{n-1} und U_{n-1} regulär. Damit existieren auch,

$$u = L_{n-1}^{-1}c, \qquad l = (U_{n-1}^\top)^{-1}d,$$

und sind eindeutig und daher existiert auch $u_{nn} = a_{nn} - l^\top u$ und ist eindeutig.

"⇒": $A = LU$ existiert und ist eindeutig, also existiert auch $A_{n-1} = L_{n-1}U_{n-1}$ und ist eindeutig. Falls L_{n-1} oder U_{n-1} nicht regulär ist, dann hätten L_{n-1} oder $U_{n-1}^\perp l = d$ keine eindeutige Lösung.

Dann würde die LU-Zerlegung von A nicht existieren bzw. wäre nicht eindeutig. Also wären L_{n-1} oder U_{n-1} nicht regulär und daher gilt,

$$\det A_{n-1} = (\det L_{n-1})(\det U_{n-1}) \neq 0.$$

Anwendung der Induktionsvoraussetzung auf A_{n-1} ergibt, $A_1, \ldots, A_{n-1}$ sind regulär. «

Definition *Eine Matrix A heißt positiv definit, falls $x^\perp Ax \geq 0$, $\forall\ x \in \mathbb{R}^n$ und $x^\perp Ax = 0 \Leftrightarrow x = 0$.* ⋊

Korollar *Für eine positiv definite Matrix $A \in \mathbb{R}^{n\times n}$ existiert eine eindeutige LU-Zerlegung.* ⋊

2-C Pivotisierung

Bisher haben wir stets angenommen, dass bei der Gauß-Elimination bzw. der LU-Zerlegung alle auf der Diagonalen auftretenden Einträge von Null verschieden sind. Unser bisher entwickelter Algorithmus zur Gauß-Eliminitation bzw. LU-Zerlegung funktionieriert nicht, falls eines der Diagonalenelemente $a_{ii}^{(i-1)}$ der Matrix $A^{(i-1)}$ nach dem $i-1$-ten Schritt Null ist. Nun kann dies aber auch bei invertierbaren Matrizen passieren. Ein Beispiel,

$$A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

Hier würde unser Algorithmus bereits im ersten Schritt versagen. Beim Rechnen mit Fließkommazahlen der Fall $a_{ii}^{(i-1)} = 0$ "fast nie" auf. Viel häufiger sind Probleme, die von betragsmäßig sehr kleinen Diagonalenelementen verursacht werden. Da alle Rechenoperationen mit Rundungsfehlern behaftet sind, erhält man anstatt einer exakten Null lediglich Werte nahe bei Null. Eine singuläre Matrix ist

so meißt nach Anwendung einer einzigen Fließkommaoperation nicht mehr singulär. Darüber hinaus sind Operationen mit Zahlen nahe bei Null sehr fehlerbehaftet, weshalb wir daran interessiert sind, diese in unserem LU-Zerlegungsalgorithmus zu vermeiden.

Bsp Lösung des Gleichungssystems mit 3 Stellen Genauigkeit,

$$\begin{pmatrix} 10^{-3} & -1 \\ 1 & 2 \end{pmatrix} x = \begin{pmatrix} -4 \\ 6 \end{pmatrix}$$

Wir erhalten als LU-Zerlegung,

$$A = \begin{pmatrix} 10^{-3} & -1 \\ 1 & 2 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 1000 & 1 \end{pmatrix} \begin{pmatrix} 10^{-3} & -1 \\ 0 & \underbrace{1000}_{1002} \end{pmatrix}.$$

Durch Vorwärtsauflösen ergibt sich,

$$\begin{pmatrix} 1 & 0 \\ 1000 & 1 \end{pmatrix} y = \begin{pmatrix} -4 \\ 6 \end{pmatrix} \Rightarrow \begin{cases} y_1 = -4, \\ y_2 = \underbrace{4010}_{=4006}. \end{cases}$$

Durch Rückwärtssubstitution

$$\begin{pmatrix} 10^{-3} & -1 \\ 0 & 1000 \end{pmatrix} x = \begin{pmatrix} -4 \\ 4010 \end{pmatrix} \Rightarrow \begin{cases} x_2 = \frac{4010}{1000} = 4.01, \\ x_1 = (-4 + 4.01)1000 = 10 \end{cases}$$

erhalten wir die numerische Lösung,

$$x = \begin{pmatrix} 10 \\ 4.01 \end{pmatrix}.$$

Durch Einsetzen in das Gleichungssystem ergibt,

$$Ax = \begin{pmatrix} 10^{-3} \cdot 10 - 1 \cdot 4.01 \\ 1 \cdot 10 + 2 \cdot 4.01 \end{pmatrix} = \begin{pmatrix} -4 \\ 18.02 \end{pmatrix} \neq \begin{pmatrix} -4 \\ 6 \end{pmatrix}.$$

Die exakte Lösung ist,

$$x = \begin{pmatrix} 1.996 \\ 3.998 \end{pmatrix}.$$

Der Fehler ist hier so fatal, da der Abstand von 10^{-3} und 1 etwa der Rechengenauigkeit entspricht. ■

Zur Lösung des Problems vertrauschen wir Zeilen. Im i-ten Schritt wählen wir eine Pivot-Zeile $p \geq i$ mit der Eigenschaft, dass $\left|a_{pi}\right|$ "groß" ist und vertauschen die aktuelle Zeile i mit der Pivot-Zeile p. Um die Pivotzeile auszuwählen, gibt es zahlreiche "Strategien". Wir wollen hier nur sehr einfache vorstellen.

Pivotisierungssatrategien. 1.) Spaltenmaximum.

Wähle p so, dass $\left|a_{pi}\right| \geq \left|a_{ji}\right| \quad \forall\ j \geq i$.

2.) Relatives Spaltenmaximum.

Wähle p so, dass $\frac{|a_{pi}|}{\sum_{j=i}^{n}|a_{pj}|}$ in dieser Spalte maximal ist.

Algorithmus.

Für $i = 1, \ldots, n-1$.
  Bestimme einen Pivot-Index $p_i \in \{i, \ldots, n\}$.
  Falls $p_i \neq i$: Vertausche Zeilen i, p (*)
  Für $j = i+1, \ldots, n$
    $a_{ji} = \dfrac{a_{ji}}{a_{ii}}$
    Für $k = i+1, \ldots, n$
      $a_{jk} = a_{jk} - a_{ji}a_{ik}$

Der Algorithmus entspricht also dem bisherigen bis auf die Vertauschung der Zeilen.

Bemerkung zu (*). Die Eliminiationsfaktoren $l_{jk} \mathrel{\hat{=}} a_{jk}$ für $k < i$ werden ebenfalls vertauscht. ⊸

Bsp LU-Zerlegung mit Pivotisierung nach der relativen Spaltenmaximierungs-

strategie.

$$\begin{pmatrix} 1 & 1 & 0 & 2 \\ \frac{1}{2} & \frac{1}{2} & 2 & -1 \\ -1 & 0 & -\frac{1}{8} & -5 \\ 2 & -6 & 9 & 12 \end{pmatrix} \begin{matrix} \frac{1}{4} \\ \frac{1}{8} \\ \frac{1}{6+\frac{1}{8}} \\ \frac{2}{29} \end{matrix} \to \begin{pmatrix} 1 & 1 & 0 & 2 \\ \frac{1}{2} & 0 & 2-2 & \\ -1 & 1 & -\frac{1}{8}-3 & \\ 2 & -8 & 9 & 8 \end{pmatrix} \begin{matrix} - \\ 0 \\ \frac{1}{4+\frac{1}{8}} \\ \frac{8}{25} \end{matrix}$$

$$\to \begin{pmatrix} 1 & 1 & 0 & 2 \\ 2 & -8 & 9 & 8 \\ -1 & -\frac{1}{8} & 1 & -2 \\ \frac{1}{2} & 0 & 2 & -2 \end{pmatrix} \begin{matrix} - \\ - \\ \frac{1}{3} \\ \frac{2}{4} \end{matrix}$$

$$\to \begin{pmatrix} 1 & 1 & 0 & 2 \\ 2 & -8 & 9 & 8 \\ \frac{1}{2} & 0 & 2 & -2 \\ -1 & -\frac{1}{8} & \frac{1}{2} & -1 \end{pmatrix}$$

Der Vektor der Pivot Indizies ist gegeben durch $\vec{p} = (1, 4, 4)$. ■

2.3 **Definition** *Eine Matrix $A \in \mathbb{R}^{n \times n}$ heißt streng (zeilen)-diagonaldominant, falls*

$$|a_{ii}| > \sum_{j \neq i} \left| a_{ij} \right| . \quad \bowtie$$

2.4 **Satz** *Sei $A \in \mathbb{R}^{n \times n}$ streng diagonaldominant, sei*

$$A^{(k)} = \begin{pmatrix} a_{11} & \cdots & \cdots & \cdots & a_{1n} \\ & \ddots & & & \vdots \\ & & a_{kk}^{(k)} & \cdots & a_{kn}^{(k)} \\ & & \vdots & \ddots & \vdots \\ & & a_{nk}^{(k)} & \cdots & a_{nn}^{(k)} \end{pmatrix}$$

die nach $k-1$ Schritten der Gauß-Elimination erhaltene Matrix. Dann ist $A^{(k)}$ streng diagonaldominant. $\bowtie$

» Wir zeigen die Aussage für $k = 2$: Für $i = 2, \dots, n$ gilt:

$$
\begin{aligned}
& a_{i1}^{(2)} = 0, \\
& a_{ij}^{(2)} = a_{ij} - \frac{a_{i1}}{a_{11}} a_{1j}, \qquad j = 2, \dots, n \\
\Rightarrow & \sum_{j \neq i} \left| a_{ij}^{(2)} \right| = \sum_{\substack{j=2 \\ j \neq i}}^{n} \left| a_{ij}^{(2)} \right| \leq \sum_{\substack{j=2 \\ j \neq i}}^{n} \left| a_{ij} \right| + \left| \frac{a_{i1}}{a_{11}} \right| \sum_{\substack{j=2 \\ j \neq i}}^{n} \left| a_{1j} \right| \\
& = \sum_{\substack{j=1 \\ j \neq i}}^{n} \left| a_{ij} \right| - |a_{i1}| + \frac{|a_{i1}|}{|a_{11}|} \left(\sum_{\substack{j=1 \\ j \neq i}}^{n} \left| a_{1j} \right| - |a_{1i}| \right) \\
& < |a_{ii}| - |a_{i1}| + \left| \frac{a_{i1}}{a_{11}} \right| [|a_{11}| - |a_{1i}|] = |a_{ii}| - \frac{|a_{i1}|\,|a_{1i}|}{|a_{11}|} \leq \left| a_{ii}^{(2)} \right|
\end{aligned}
$$

Es gilt also,

$$
\sum_{\substack{j=1 \\ j \neq i}}^{n} \left| a_{ij}^{(2)} \right| < \left| a_{ii}^{(2)} \right|, \qquad \text{für } i \geq 2.
$$

Der Gauß-Algorithmus verändert die erste Zeile nicht, also gilt die Ungleichung ebenso für $i = 1$. Durch Induktion folgt, dass A auch für $k > 2$ streng diagonaldominant ist. «

2.5 **Korollar** *Sei $A \in \mathbb{R}^{n \times m}$ streng diagonaldominant. Dann gilt:*

(i) Die LU-Zerlegung ist ohne Pivotisierung anwendbar.

(ii) Die relative Spaltenmaximumstrategie liefert sofort das aktuelle Diagonalelement als Pivotelement. ⋊

» "(i)": Mit der Notation von 2.4 gilt:

$$
\left| a_{kk}^{(k)} \right| > \sum_{\substack{j=1 \\ j \neq k}}^{n} \left| a_{kj}^{(k)} \right| \geq 0 \Rightarrow a_{kk}^{(k)} \neq 0.
$$

"(ii)": Wir zeigen: Ist A streng diagonaldominant, dann gilt

$$
\frac{|a_{ii}|}{\sum_{j=1}^{n} \left| a_{ij} \right|} > \frac{1}{2} > \frac{|a_{ki}|}{\sum_{j=1}^{n} \left| a_{kj} \right|}.
$$

Zunächst gilt:

$$\sum_{l=1}^{n} |a_{kl}| = |a_{kk}| + \sum_{\substack{l=1 \\ l\neq k}}^{n} |a_{kl}| < 2\,|a_{kk}|$$

und der erste Teil der Ungleichung ist gezeigt. Für $k \neq i$ gilt,

$$2\,|a_{ki}| \leq |a_{ki}| + |a_{kk}| \leq \sum_{l=1}^{n} |a_{kl}|\,,$$

und damit ist der zweite Teil der Ungleichung gezeigt.

Wir sehen also, dass die Spaltenmaximumstrategie bei Pivotisierung im i-ten Schritt immer die i-te Zeile aussucht, da hier das Zeilenmaximum vorliegt. «

Die Vertauschung zweier Zeilen i, j einer Matrix $A \in \mathbb{R}^{n\times n}$ kann man darstellen durch,

$$A \to P_{(i,j)}A, \quad \text{mit} \quad \begin{pmatrix} \ddots & & & & & & & \\ & 1 & & & & & & \\ & & 0 & & & & 1 & \\ & & & 1 & & & & \\ & & & & \ddots & & & \\ & & & & & 1 & & \\ & & 1 & & & & 0 & \\ & & & & & & & 1 \\ & & & & & & & & \ddots \end{pmatrix}$$

Die LU-Zerlegung mit Pivotisierung liefert

- Einen Vektor $p = (p_i)_{i=1}^{n-1} \in \mathbb{R}^{n-1}$ von Pivot Indizes.
- Die LU-Zerlegung der Matrix PA mit der Eingabematrix A und

$$P = P^{(n-1)}P^{(n-2)}\cdots P^{(1)}, \quad \text{wobei } P^{(i)} = P_{(i,p_i)}.$$

Wir wollen nun mit der entwickelten Theorie lineare Gleichungssysteme lösen. Statt $Ax = b$ löst man

$$PAx = Pb, \quad \text{bzw.} \quad LUx = Pb.$$

1. Schritt. LU-Zerlegung mit Pivotisierung

$$p \in \mathbb{R}^{n-1}, PA = LU,$$

2. Schritt. Berechnung von $Pb : b \mapsto Pb$.

3. Schritt. Vorwärtssubstitution $Ly = Pb \Rightarrow y$.

4. Schritt. Rückwärtsauflösen $Ux = y \Rightarrow x$.

Algorithmus für 2. Schritt

Für $i = 1, \ldots, n-1$
    Falls $p_i \neq i$
        vertausche b_i, b_{p_i}

2-D Dünn besetzte Matrizen

Definition *Eine Matrix heißt dünn besetzt, falls "sehr viele" Einträge der Matrix Null sind. Konkrekt bedeutet dies, es müssen so viele Einträge Null sein, dass es sich lohnt, dies auszunutzen.* ⋊

Diese Definition scheint nicht viel her zu geben, trifft die Sache aber genau. Dünn besetzte Matritzen treten in vielen Anwendungen auf:

- Netwerke (elektrische Netzwerke, elastische Stabwerke, Rohrleitungsnetz).
- Diskretisierungen von Differentialgleichungen.

Gerade die Diskretisierung von Differentialgleichungen erzeugt schon bei Systemen mit wenigen Gleichungen gigantische Matritzen, bei denen aber nur sehr wenige Einträge von Null verschieden sind. Daher sind dünn besetzte Matritzen ein wichtiger Spezialfall, der in vielen Algorithmen separat behandelt wird. Eine Herausforderung ist dabei zu erkennen, wann eine Matrix dünn besetzt ist, denn dies ist nicht so offensichtlich.

Frage: Wenn A dünn besetzt ist, ist dann die LU-Zerlegung ebenfalls dünn besetzt?

Antwort. Leider nein!.

Bsp Eine "dünn besetzte" 4×4-Matrix, die bereits nach dem 1. Schritt der LU-Zerlegung keinen Nicht-Null Eintrag mehr besitzt

$$\begin{pmatrix} 1 & 2 & 3 & 4 \\ -1 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{pmatrix} \sim \begin{pmatrix} 1 & 2 & 3 & 4 \\ -1 & 2 & 3 & 4 \\ -1 & 2 & 4 & 4 \\ -1 & 2 & 3 & 5 \end{pmatrix}. \quad \blacksquare$$

Frage: Welche Nulleinträge einer Matrix bleiben bei der LU-Zerlegung erhalten?

Antwort: Die jeweils ersten Nulleinträge in einer Zeile bzw. Spalte bleiben erhalten. Dies gilt zunächst nur bei Zerlegung ohne Pivotisierung, mit unseren Pivotisierungsstrategien würde man auch diese Nulleinträge zerstört werden. Man kann aber Strategien entwicklen die das Auffüllen von Nichtnull Einträgen reduzieren.

$$\begin{pmatrix} * & * & 0 & * & 0 \\ 0 & * & 0 & * & 0 \\ 0 & * & * & & * \\ * & & & * & * \\ 0 & 0 & * & & * \end{pmatrix}$$

Bei LU-Zerlegung erhaltene Nulleinträge sind blau.

2.6 **Definition** *Die Hülle einer Matrix $A \in \mathbb{R}^{n \times n}$ ist,*

$$H(A) := \{1, \ldots, n\}^2 \setminus \Big\{(i,j) : a_{ik} = 0 \text{ für } k = 1, \ldots, i \text{ oder } a_{kj} = 0 \text{ für } k = 1, \ldots, j\Big\}.$$

Die LU-Zerlegung vergrößert die Hülle einer Matrix nicht. Es genügt also die LU-Zerlegung "auf der Hülle" durchzuführen, wodurch sich unter Umständen sehr viel Rechenaufwand sparen lässt.

■ Bandmatritzen

Wir wollen nun eine einfache Klasse von dünn besetzten Matritzen untersuchen, die Bandmatritzen.

2.7 **Definition** *Eine Matrix $A \in \mathbb{R}^{n\times n}$ heißt Bandmatrix mit Bandweite*

$$m = m_1 + m_2 + 1,$$

falls

$$a_{ij} = 0, \text{ für } j < i - m_1 \text{ oder } j > i + m_2.$$

$$\begin{pmatrix} * & * & \cdots & * & & 0 \\ * & \ddots & \ddots & & \ddots & \\ \vdots & \ddots & \ddots & \ddots & & * \\ * & & \ddots & \ddots & \ddots & \vdots \\ & \ddots & & \ddots & \ddots & * \\ 0 & & * & \cdots & * & * \end{pmatrix}$$

Als Band von A bezeichnet man

$$B(A) = \left\{(i,j) \in \{1,\dots,n\}^2 : i - m_1 \le j \le i + m_2\right\}$$

Die LU-Zerlegung einer Bandmatrix ändert die Einträge außerhalb des Bandes nicht. Es genügt also die LU-Zerlegung auf dem Band durchzuführen. ⋈

Wir benötigen zunächst eine spezielle Datenstruktur zum Abspeichern von $A \in \mathbb{R}^{n\times n}$ in $\tilde{A} \in \mathbb{R}^{n\times m}$.

$$\begin{pmatrix} * & & * & 0 & \\ & \ddots & & \ddots & 0 \\ * & & \ddots & & * \\ 0 & \ddots & & \ddots & \\ & 0 & * & & * \end{pmatrix} \sim \begin{pmatrix} 0 & & * & \cdots & * \\ & \ddots & \vdots & & \vdots \\ * & & \vdots & & \vdots \\ \vdots & & \vdots & & * \\ \vdots & & \vdots & \ddots & \\ * & \cdots & * & & \end{pmatrix}$$

$$a_{ij} = \tilde{a}_{i,j+m_1+1-i}$$
$$\tilde{a}_{ij} = a_{i,j+i-m_1-1}$$

Wir werden nun den bekannten Algorithmus für die LU-Zerlegung,

Für $i = 1, \dots, n-1$

  Für $j = i+1, \dots, n$

    $a_{ji} = \dfrac{a_{ji}}{a_{ii}}$.

    Für $k = i+1, \dots, n$

      $a_{jk} = a_{jk} - a_{ji} a_{ik}$.

modifizieren.

Für $i = 1, \dots, n-1$

  Für $j = i+1, \dots, \min\{i+m_1, n\}$

    $\tilde{a}_{j,i+m_1-j+1} = \dfrac{\tilde{a}_{j,i+m_1-j+1}}{\tilde{a}_{i,m_1+1}}$.

    Für $k = i+1, \dots, \min\{i+m_2, n\}$

      $\tilde{a}_{j,k+m_1+1-j} = \tilde{a}_{j,k+m_1+1-j} - \tilde{a}_{j,i+m_1+1-j} \cdot \tilde{a}_{i,k+m_1+1-i}$.

Abschätzung des Rechenaufwands.

$$\leq \sum_{i=1}^{n-1} \sum_{j=i+1}^{i+m_1} (1+m_2) = (n-1) m_1 (1+m_2) \approx n m_1 m_2. \quad \multimap$$

Bsp $n = 10.000$, $m_1 = m_2 = 100$, d.h. $m = 201$.
Standard LU-Zerlegung $\frac{1}{3}n^3 = \frac{1}{3} \cdot 10^{12}$.
LZ-Zerlegung für Bandmatrix $n m_1^2 = 10^8$.
Die LU-Zerlegung für Bandmatritzen ist um den Faktor $3 \cdot 10^{-4}$ schneller. ■

■ Lösung von Gleichungssystemen mit Bandmatritzen

Wir wollen nun die bestehenden Algorithmen für Vorwärts- und Rückwärtsauflösen an die neue Datenstruktur anpassen.

$$Ly = b, \qquad Ux = y,$$

Vorwärtsauflösen.

$$\begin{pmatrix} 1 & & & \\ * & \ddots & & \\ \vdots & \ddots & \ddots & \\ * & \cdots & * & 1 \end{pmatrix} y = b$$

$y_1 = b_1$

Für $i = 2, \dots, n$

$$y_i = b_i - \sum_{j=\max\{1, i-m_1\}}^{i-1} \tilde{a}_{i,j+m_1+1-i} \cdot y_j$$

Rückwärtsauflösen.

$$x_n = \frac{y_n}{\tilde{a}_{n,m_1+1}}$$

Für $i = n-1, \dots, 1$

$$x_i = \frac{1}{\tilde{a}_{i,m_1+1}} \left(y_i - \sum_{j=i+1}^{\min\{n, i+m_2\}} \tilde{a}_{i,j+m_1+1-i} x_j \right)$$

2-E Matrixnormen

Dieser Abschnitt soll kurz die Elemente aus der Linearen Algebra und der Analysis wiederholen, die zur Definition von Matrixnormen notwendig sind. Mithilfe der Matrixnormen lassen sich beispielsweise Aussagen über Invertiertierbarkeit einer zu einem Gleichungssystem gehörenden Matrix und damit über die Lösbarkeit des Systems selbst machen. Wir werden davon später bei der Betrachtung der Fehlerfortpflanzung der LU-Zerlegung gebrauch machen.

2.8 **Definition** *Eine Abbildung* $\|\cdot\| : \mathbb{R}^n \to \mathbb{R}$, $x \mapsto \|x\|$ *heißt Norm, falls*

(i) $\|x\| \geq 0$ *und* $\|x\| = 0 \Leftrightarrow x = 0, \qquad \forall\, x \in \mathbb{R}^n$

(ii) $\|\alpha x\| = |\alpha|\, \|x\|, \qquad \forall\, x \in \mathbb{R}^n, \alpha \in \mathbb{R}$

(iii) $||x + y|| \leq \|x\| + ||y||, \qquad \forall\, x, y \in \mathbb{R}^n.$ ⋈

Bsp Einige bereits wohl bekannte Normen,

$$\|x\|_2 = \left(\sum_{j=1}^{n} x_j^2\right)^{\frac{1}{2}},$$
$$\|x\|_\infty = \max_{j=1,\dots,n} \left|x_j\right|,$$
$$\|x\|_1 = \sum_{j=1}^{n} \left|x_j\right|. \quad \blacksquare$$

2.9 **Definition/Satz** *Seien $\|\cdot\|_n$ und $\|\cdot\|_m$ Normen auf $\mathbb{R}^n$ und $\mathbb{R}^m$. Dann definiert,*

$$\|A\|_{n,m} := \max_{\substack{x\in\mathbb{R}^m \\ x\neq 0}} \frac{\|Ax\|_n}{\|x\|_m} = \max_{\|x\|_m=1} \|Ax\|_n .$$

eine Norm auf $\mathbb{R}^{n\times m}$, die so genannte Matrixnorm bzw. Operatornorm. ⋈

Bsp 1.) Die Matrixnorm für $\|\cdot\|_1$-Norm in $\mathbb{R}^n$ und $\mathbb{R}^m$ ist die Spaltensummennorm,

$$\|A\|_1 := \max_{j=1,\dots,m} \sum_{i=1}^{n} \left|a_{ij}\right|$$

» Zu zeigen ist

(i) $\|Ax\|_1 \le \|A\|_1 \|x\|_1, \qquad \forall\, x \in \mathbb{R}^m,$

(ii) $\exists\, x \in \mathbb{R}^m \ \|Ax\|_1 = \|x\|_1.$

"(i)":

$$\|Ax\|_1 = \sum_{i=1}^{n} \left|\sum_{j=1}^{m} a_{ij}x_j\right| \le \sum_{i=1}^{n}\sum_{j=1}^{m} \left|a_{ij}\right| \left|x_j\right| = \sum_{j=1}^{m} \left(\sum_{i=1}^{n} \left|a_{ij}\right|\right) \left|x_j\right|$$
$$\le \max_{j=1,\dots,m} \sum_{i=1}^{n} \left|a_{ij}\right| \sum_{j=1}^{m} \left|x_j\right| = \|A\|_1 \|x\|_1 .$$

"(ii)": Sei $j \in \{1,\dots,n\}$, so dass

$$\|A\|_1 = \sum_{i=1}^{n} \left|a_{ij}\right|$$

Für $x = e_j$ gilt,

$$\left\|Ae_j\right\|_1 = \sum_{i=1}^{m} \left|(Ae_j)_i\right| = \sum_{i=1}^{m} \left|a_{ij}\right| = \|A\|_1 = \|A\|_1 \left\|e_j\right\|_1 . \quad «$$

2.) Die Matrixnorm für $\|\cdot\|_\infty$-Norm ist die Zeilensummennorm,

$$\|A\|_\infty := \max_{i=1,\dots,n} \sum_{j=1}^{m} \left|a_{ij}\right| .$$

» Der Nachweis der Normeigenschaften ist eine gute Übung. «

3.) Die vielleicht interessanteste Norm ist die Matrixnorm zur euklidischen Norm in $\mathbb{R}^n$ und $\mathbb{R}^m$,

$$\|A\|_2 := \max_{\substack{x\in\mathbb{R}^m \\ \|x\|=1}} \|Ax\|_2 .$$

Sie heißt Spektralnorm. Im Folgenden wollen wir diese etwas genauer studieren. ■

2.10 **Satz** *(i) Für eine symmetrische, quadratische Matrix $A \in \mathbb{R}^{n\times n}$ gilt,*

$$\|A\|_2 = \max\{|\lambda| : \lambda \text{ ist Eigenwert von } A\} .$$

(ii) Für eine allgemeine Matrix $A \in \mathbb{R}^{m\times n}$ gilt,

$$\|A\|_2 = \max\left\{\sqrt{|\lambda|} : \lambda \text{ ist Eigenwert von } A^\top A\right\} = \sqrt{\|A^\top A\|_2}. \quad ⋈$$

» Zu (i) A ist symmetrisch, es existiert also eine Darstellung

$$A = Q\Lambda Q^\top$$

mit einer orthogonalen Matrix Q, d.h. $Q \cdot Q^\top = \mathrm{Id}$ und

$$\Lambda = \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_n \end{pmatrix} .$$

Es gilt somit

$$\|Ax\|_2 = \left\|Q\Lambda Q^\top x\right\|_2 = \left\|\Lambda \underbrace{Q^\top x}_{:=y}\right\|_2 = \left(\sum_{i=1}^{n} |\lambda_i y_i|^2\right)^{\frac{1}{2}}$$
$$\leq \max_{i=1,\dots,n} |\lambda_i| \, \|y\|_2 = \max_{i=1,\dots,n} |\lambda_i| \, \|Q^\top x\|_2 = \max_{i=1,\dots,n} |\lambda_i| \, \|x\|_2$$

Wobei wir verwendet haben, dass orthogonale Matritzen die euklidische Norm erhalten,

$$\|Qy\|_2^2 = \langle Qy, Qy\rangle = \langle y, Q^\top Qy\rangle = \langle y, y\rangle = \|y\|_2^2 .$$

Für $x = Qe_j$ gilt

$$\|Ax\|_2 = \left\|Q\Lambda Q^\top Qe_j\right\|_2 = \left\|\Lambda e_j\right\|_2 = \left|\lambda_j\right| ,$$
$$\|A\|_2 \geq \left|\lambda_j\right| \quad \forall\ j = 1, \dots, n$$
$$\|A\|_2 \geq \max_{j=1,\dots,n} \left|\lambda_j\right|$$

Zu (ii)

$$\|Ax\|_2^2 = \langle Ax, Ax\rangle = \langle x, A^\top Ax\rangle \overset{(i)}{\leq} \|x\|_2 \left\|A^\top Ax\right\|_2$$
$$\leq \|x\|_2 \left\|A^\top A\right\|_2 \|x\|_2 ,$$
$$\Rightarrow \|Ax\|_2 \leq \sqrt{\|A^\perp A\|_2} \, \|x\|_2 ,$$
$$\Rightarrow \|A\|_2 \leq \sqrt{\|A^\perp A\|_2}.$$

Sei x Eigenvektor von $A^\top A$ zum maximalen Eigenwert $\lambda_{\max}$.

$$\Rightarrow \|Ax\|_2^2 = \langle x, A^\top Ax\rangle = \langle x, \lambda_{\max} x\rangle = \lambda_{\max} \|x\|_2^2$$
$$\Rightarrow \|Ax\|_2 = \sqrt{\lambda_{\max}} \, \|x\|_2 \overset{(i)}{=} \sqrt{\|A^\top A\|_2} \, \|x\|_2$$
$$\Rightarrow \|A\|_2 \geq \sqrt{\|A^\top A\|_2}$$

Wir haben damit das Ergebnis

$$\|A\|_2 = \sqrt{\|A^\top A\|_2}. \quad «$$

2-F Fehlerfortpflanzung

Wir wollen nun die Fehlerfortpflanzung bei der Lösung von linearen Gleichungssystemen mit den bisher entwickelten Methoden betrachten.

Gegeben sei ein exaktes lineares Gleichungssystem

$$Ax = b \tag{2}$$

mit $A \in \mathbb{R}^{n\times n}$ invertierbar und eine numerische Realisierung

$$\tilde{A}\tilde{x} = \tilde{b},$$

mit den Fehlern

$$\Delta A = \tilde{A} - A, \quad \Delta b = \tilde{b} - b, \quad \Delta x = \tilde{x} - x.$$

Ziel ist es nun den relativen Fehler von x abzuschätzen

$$\frac{\|\Delta x\|}{\|x\|} \text{ durch } \frac{\|\Delta b\|}{\|b\|} \text{ und } \frac{\|\Delta A\|}{\|A\|}.$$

Um überhaupt eine Abschätzung durchführen zu können, müssen wir $\tilde{A}$ als invertierbar annehmen, die Störung von A darf also nicht so groß sein, dass die numerische Realisierung nicht mehr lösbar ist.

Wir können die Realisierung wie folgt umformen,

$$\begin{aligned}
&\tilde{A}\tilde{x} = (A+\Delta A)(x+\Delta x) = Ax + \Delta Ax + (A+\Delta A)\Delta x = \tilde{b} = b + \Delta b,\\
\Leftrightarrow &(A+\Delta A)\Delta x = \Delta b - \Delta Ax\\
\Leftrightarrow &\Delta x = (A+\Delta A)^{-1}(\Delta b - \Delta Ax)\\
\Rightarrow &\frac{\|\Delta x\|}{\|x\|} \le \frac{\left\|(A+\Delta A)^{-1}\right\|(\|\Delta b\| + \|\Delta A\|\,\|x\|)}{\|x\|}\\
&= \left\|(A+\Delta A)^{-1}\right\|\left(\frac{\|\Delta b\|}{\|x\|} + \|\Delta A\|\right)\\
&= \left\|(A+\Delta A)^{-1}\right\|\|A\|\left(\frac{\|\Delta b\|}{\|A\|\,\|x\|} + \frac{\|\Delta A\|}{\|A\|}\right)
\end{aligned}$$

Mit $\|b\| = \|Ax\| \le \|A\|\,\|x\|$ folgt

$$\frac{\|\Delta x\|}{\|x\|} \le \left\|(A+\Delta A)^{-1}\right\|\|A\|\left(\frac{\|\Delta b\|}{\|b\|} + \frac{\|\Delta A\|}{\|A\|}\right) \tag{4}$$

2.11 **Satz** *Seien $A, B \in \mathbb{R}^{n\times n}$ und A invertierbar, sowie*

$$\left\|A^{-1}\right\| \|B\| < 1, \quad \text{für eine beliebige aber feste Norm } \|\cdot\|.$$

Dann ist $A + B$ invertierbar und

$$\left\|(A+B)^{-1}\right\| = \frac{\left\|A^{-1}\right\|}{1 - \left\|A^{-1}\right\| \|B\|}. \quad \bowtie$$

» Wir addieren Null geschickt und erhalten so,

$$\begin{aligned} & x = A^{-1}(A+B)x - A^{-1}Bx, \\ \Rightarrow\ & \|x\| \le \left\|A^{-1}\right\| \|(A+B)x\| + \left\|A^{-1}\right\| \|B\| \|x\|, \\ \Rightarrow\ & \underbrace{\left(1 - \left\|A^{-1}\right\| \|B\|\right)}_{>0} \|x\| \le \left\|A^{-1}\right\| \|(A+B)x\|. \end{aligned}$$

Für $x \in \ker(A+B)$ also $(A+B)x = 0$ folgt $\|x\| = 0$, d.h. $\ker(A+B) = (0)$ und daher ist $A + B$ invertierbar.

Für $y \in \mathbb{R}^n$ und $x = (A+B)^{-1}y$ folgt,

$$\begin{aligned} & \left\|(A+B)^{-1}y\right\| \le \frac{\left\|A^{-1}\right\|}{1 - \left\|A^{-1}\right\| \|B\|} \|y\|, \\ \Rightarrow\ & \left\|(A+B)^{-1}\right\| \le \frac{\left\|A^{-1}\right\|}{1 - \left\|A^{-1}\right\| \|B\|} \quad \text{«} \end{aligned}$$

Mit 2.11 folgt aus (4) im Fall $\left\|A^{-1}\right\| \|\Delta A\| < 1$,

$$\begin{aligned} \frac{\|\Delta x\|}{\|x\|} &\le \frac{\left\|A^{-1}\right\| \|A\|}{1 - \left\|A^{-1}\right\| \|\Delta A\|} \left(\frac{\|\Delta b\|}{\|b\|} + \frac{\|\Delta A\|}{\|A\|}\right) \\ &= \frac{\left\|A^{-1}\right\| \|A\|}{1 - \left\|A^{-1}\right\| \|A\| \frac{\|\Delta A\|}{\|A\|}} \left(\frac{\|\Delta b\|}{\|b\|} + \frac{\|\Delta A\|}{\|A\|}\right). \end{aligned}$$

2.12 **Definition** *Sei $A \in \mathbb{R}^{n\times n}$ invertierbar. Die Zahl*

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

heißt Konditionszahl von A bezüglich der Matrixnorm $\|\cdot\|$. $\bowtie$

Der folgende Satz fasst unsere Abschätzung des Fehlers zusammen.

2.13 **Satz** *Sei $A \in \mathbb{R}^{n\times n}$ invertierbar, $b \in \mathbb{R}^n$, $b \neq 0$ und $||A^{-1}||\,\|\Delta A\| < 1$. Dann gilt für die Lösung $\tilde{x} = x + \Delta x$ von (2) die Fehlerabschätzung,*

$$\frac{\|\Delta x\|}{\|x\|} \leq \frac{\kappa(A)}{1 - \kappa(A)\frac{\|\Delta A\|}{\|A\|}} \left(\frac{\|\Delta b\|}{\|b\|} + \frac{\|\Delta A\|}{\|A\|}\right). \quad \bowtie$$

Bemerkungen. 1.) Die Bedingung $||A^{-1}||\,\|\Delta A\| < 1$ ist identisch zu

$$\kappa(A)\frac{\|\Delta A\|}{\|A\|} < 1.$$

Wir sehen also dass der relative Fehler von A klein sein muss, denn sonst ist das Gleichungssystem nicht mehr lösbar.

2.) Für $\kappa(A)\frac{\|\Delta A\|}{\|A\|} << 1$ ist der Verstärkungsfaktor des relativen Fehlers im wesentlichen die Konditionszahl.

3.) Für $\kappa(A)\frac{\|\Delta A\|}{\|A\|} \geq 1$ bzw. etwa 1 ist die Lösung $\tilde{x}$ des Systems in der Regel sehr schlecht. $\multimap$

Gleichungssysteme mit kleiner Konditionszahl sind numerisch gut lösbar, während Systeme mit Konditionszahl auch bei "guten Daten" Probleme hervorrufen können. Die Trennung von lösbar und nicht lösbar ist daher nicht so "scharf" wie in der Linearen Algebra. Es kann sogar passieren, dass ein nicht lösbares Gleichungssystem mit numerischen Verfahren "gelöst" wird - diese "Lösung" ist dann aber zu nichts zu gebrauchen. Die Lösbarkeit ist in jedem Fall von der Qualtiät der Daten abhängig, während ein Gleichungssystem mit $\kappa(A) = \infty$ mit keinem numerischen Verfahren mehr lösbar ist.

2-G Cholesky-Zerlegung

Wir werden nun eine weitere Lösungsmethode für Lineare Gleichungssysteme kennenlernen, die nur für positiv definite Matritzen funktioniert, jedoch deutlich geringeren Rechenaufwand aufweist.

2.14 **Satz** *Sei $A \in \mathbb{R}^{n\times n}$ symmetrisch und positiv definit, dann existiert eine eindeutige Zerlegung,*

$$A = LL^\top,$$

wobei $L \in \mathbb{R}^{n\times n}$ eine linke untere Dreiecksmatrix ist mit $l_{ii} > 0$ für $i = 1, \ldots, n$. Sie heißt Cholesky-Zerlegung. ⋈

Zunächst wollen wir die Notwendigkeit der Bedingungen überprüfen. Die Symmetrie sieht man sofort, denn $(LL^\top)^\top = LL^\top$. Die positive Definitheit ist nicht so leicht einzusehen, wir werden auch später sehen, dass wir diese Voraussetzung noch etwas abschwächen können.

» Wir führen eine Induktion über n.

Induktionsanfang $n = 1$. $A = (a_{11}) = LL^\top$ mit $L = (\sqrt{a_{11}})$. $a_{11} > 0$, da A positiv definit.

Induktionsschritt $n - 1 \to n$. Sei dazu

$$A = \begin{pmatrix} A_{n-1} & a \\ a^\top & a_{nn} \end{pmatrix}, \qquad A_{n-1} \in R^{n-1\times n-1}, a \in \mathbb{R}^{n-1}, a_{nn} \in \mathbb{R}.$$

Ansatz:

$$L = \begin{pmatrix} L_{n-1} & 0 \\ l^\top & \alpha \end{pmatrix}, \qquad L_{n-1} \in R^{n-1\times n-1}, l \in \mathbb{R}^{n-1}, \alpha \in \mathbb{R}. \tag{5}$$

Durch (5) erhalten wir,

$$\begin{aligned} A &= \begin{pmatrix} A_{n-1} & a \\ a^\top & a_{nn} \end{pmatrix} = \begin{pmatrix} L_{n-1} & 0 \\ l^\top & \alpha \end{pmatrix} \begin{pmatrix} L_{n-1}^\top & l \\ 0 & \alpha \end{pmatrix} \\ \Leftrightarrow & A_{n-1} = L_{n-1}L_{n-1}\top \\ & a = L_{n-1}l, \\ & a^\top = l^\top L_{n-1}^\top \\ & a_{nn} = ll^\top + \alpha^2 \end{aligned}$$

Können wir dieses Gleichungssystem eindeutig lösen, so existiert die Zerlegung und ist ebenfalls eindeutig.

Die Induktionsvoraussetzung besagt, dass eine eindeutige linke untere Dreiecksmatrix L_{n-1} existiert, sodass die erste Bedingung

$$A_{n-1} = L_{n-1}L_{n-1}$$

erfüllt ist.

Die Gleichung $a = L_{n-1}l$ ist genau dann eindeutig nach l auflösbar, wenn L_{n-1} invertierbar ist. L_{n-1} ist nach Voraussetzung eine untere Dreiecksmatrix, also invertierbar; man sieht dies auch durch $\det A = (\det L)^2$.

Für α erhalten wir den Ausdruck,

$$\alpha = \sqrt{a_{nn} - \|l\|^2}.$$

Das negative Ergebnis der Wurzel können wir ignorieren, da wir nur an positiven Diagonaleinträgen interessiert sind. Da A postiv definit ist auch $a_{nn} > 0$. Wir müssen jedoch zeigen, dass $a_{nn} > \|l\|^2$.

Aus der positiven Definitheit von A folgt,

$$\begin{aligned} 0 &< \left\langle \begin{pmatrix} A_{n-1}^{-1}a \\ -1 \end{pmatrix}, \begin{pmatrix} A_{n-1} & a \\ a^\top & a_{nn} \end{pmatrix} \begin{pmatrix} A_{n-1}^{-1}a \\ -1 \end{pmatrix} \right\rangle \\ &= \begin{pmatrix} A_{n-1}^{-1}a \\ -1 \end{pmatrix}^\top \begin{pmatrix} a - a \\ a^\top A_{n-1}^{-1}a - a_{nn} \end{pmatrix} = a_{nn} - a^\top A_{n-1}^{-1}a \end{aligned}$$

Mit

$$A_{n-1}^{-1} = (L_{n-1}L_{n-1})^\top = (L_{n-1}^\top)^{-1}L_{n-1}^{-1}$$

folgt,

$$a^\top A_{n-1}^{-1}a = \underbrace{a^\top (L_{n-1}^{-1})^\top}_{l^\top} \underbrace{L_{n-1}^{-1}a}_{l} = \|l\|^2. \quad «$$

■ Berechnung der Cholesky-Zerlegung

Sei $A = LL^\top$ die Cholesky-Zerlegung von A, dann gilt

$$a_{ji} = \sum_{k=1}^{n} l_{ik}l_{kj}^\top = \sum_{k=1}^{n} l_{ik}l_{jk} = \sum_{k=1}^{j} l_{ik}l_{jk}, \qquad i = 1,\dots,n \; j = 1,\dots,i$$

da $l_{jk} = 0$ für $k > j$. Zeilenweise Berechnung der l_{ij}:

$$\begin{aligned} a_{11} &= l_{11}^2 \Rightarrow l_{11} = \sqrt{a_{11}}, \\ a_{21} &= l_{21}l_{11} \Rightarrow l_{21} = \frac{a_{21}}{l_{11}}, \\ a_{22} &= l_{21}l_{21} + l_{22}l_{22} \Rightarrow l_{22} = \sqrt{a_{22} - l_{21}^2} \end{aligned}$$

Für die Zeile i und $j = 1, \dots, i-1$ ergibt sich,

$$a_{ij} = \sum_{k=1}^{j-1} l_{ik} l_{jk} + l_{ij} l_{jj}$$

alle Koeffizienten mit Zeilenindex $< i$ bzw. Zeilenindex i und Spaltenindex $< i$ sind bekannt, also gilt

$$l_{ij} = \frac{1}{l_{jj}} \left(a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{jk} \right).$$

$$a_{ii} = \sum_{k=1}^{l-1} l_{ik}^2 + l_{ii}^2 \Rightarrow l_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} l_{ik}^2}.$$

Algorithmus.

$l_{11} = \sqrt{a_{11}}$
Für $i = 2, \dots, n$
    Für $j = 1, \dots, i-1$
        $l_{ij} = \frac{1}{l_{jj}} \left(a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{jk} \right)$
    $l_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} l_{ik}^2}.$

Bsp Wir betrachten die symmetrische Matrix,

$$A = \begin{pmatrix} 4 & 2 & -2 \\ 2 & 10 & -7 \\ -2 & -7 & 6 \end{pmatrix}.$$

Die positive Definitheit ist nicht ganz offensichtlich, kann aber mithilfe der Determinanten berechnet werden. Die Cholesky-Zerlegung ergibt sich als,

$$L = \begin{pmatrix} 2 & 0 & 0 \\ \frac{1}{2}2 & \sqrt{10 - 1^2} & 0 \\ -\frac{1}{2}2 & \frac{1}{3}(-7+1) & \sqrt{6 - 1 - 4} \end{pmatrix} = \begin{pmatrix} 2 & 0 & 0 \\ 1 & 3 & 0 \\ -1 & -2 & 1 \end{pmatrix}. \quad \blacksquare$$

Rechenaufwand. Die Wurzeln sind hier die aufwändigsten Rechenoperationen. Für jede Diagonale muss eine gezogen werden, die Anzahl ist daher n.

Für die Multiplikationen ergibt sich,

$$\begin{aligned}
&\sum_{i=2}^{n}\sum_{j=1}^{i-1} 1 + (j-1) + i - 1 = \sum_{i=2}^{n}\sum_{j=1}^{i-1} j + i - 1 = \sum_{i=2}^{n} \frac{(i-1)i}{2} + i - 1 \\
&= \sum_{i=2}^{n} \frac{1}{2}i^2 + \frac{i}{2} - 1 = \frac{1}{2}\left(\frac{n(n+1)(2n+1)}{6} + \frac{n(n+1)}{2}\right) - n - 1 - 1 \\
&= \frac{1}{6}n^3 + \mathcal{O}(n^2)
\end{aligned}$$

Im Vergleich zum Gauß Algorithmus $\frac{1}{3}n^3$ haben wir hier $\frac{1}{6}n^3$. Die Wurzeln haben nur einfache Ordnung, also selbst wenn eine Wurzel dem 60-fachen Aufwand einer Multiplikationen entspricht, kann man sie bei hinreichend großen Matritzen vernachlässigen. Die Cholesky-Zerlegung entspricht daher für großes n der Hälfte des Aufwands der LU-Zerlegung.

■ Lösung von Linearen Gleichungssystemen

Gegeben sei ein Gleichungssystem

$$Ax = b,$$

wobei $A \in \mathbb{R}^{n\times n}$ eine positiv definite symmetrische Matrix und $b \in \mathbb{R}^n$ ist.

1. Schritt Cholesky Zerlegung $A = LL^\top$.

2. Schritt Vorwärtseinsetzen $Ly = b$.

3. Schritt Rückwärtsauflösen $L^\top x = y$.

Bemerkungen. 1.) Es gibt eine Variante der Cholesky-Zerlegung für Bandmatritzen.

2.) Pivotisierung ist möglich. Beim Vertauschen von einzelnen Zeilen und Spalten kann die positive Definitheit bzw. die Symmetrie verloren gehen, man muss daher Zeilen und Spalten vertauschen. ⊸

3 Interpolation

Interpolationsaufgabe. Gegeben sind die Datenpaare (x_i, y_i) für $i = 0, \dots, n$ und $x_i, y_i \in \mathbb{R}$.

Bsp x_i - Messpunkte, y_i - Messwerte. ■

Gesucht ist nun eine Funktion $f : \mathbb{R} \to \mathbb{R}$, die an den vorgegebenen Punkten x_i die Werte y_i annimmt,

$$f(x_i) = y_i, \qquad \text{für } i = 0, \dots, n.$$

Die Aufgabe ist aber nicht wirklich sinnvoll formuliert, da unsere Anforderungen an die Funktion f so gering sind, dass wir überabzählbar viele Lösungen angeben können. Im Folgenden wollen wir Anforderungen an die Funktion f erarbeiten, so dass das Problem eindeutig lösbar ist.

3-A Polynominterpolation

Betrachten wir zunächst eine sehr einfache Klasse von Funktionen - die Polynome. Den Raum der Polynome vom Grad $\leq n$ bezeichnen wir mit,

$$\mathcal{P}_n := \left\{ p : \mathbb{R} \to \mathbb{R} \,:\, p(x) = \sum_{j=0}^{n} a_j x^j,\ a_j \in \mathbb{R} \right\}.$$

$\mathcal{P}_n$ ist ein Vektorraum der Dimension $n + 1$.

Wir können unsere Intperpolationsaufgabe also dahingehend einschränken, dass nur Polynome als Lösungen zugelassen sind. Konkret bedeutet dies, dass zu gegebenen Daten (x_i, y_i), $i = 0, \dots, n$ ein Polynom $p \in \mathcal{P}_n$ gesucht ist mit

$$p(x_i) = y_i, \qquad \text{für } i = 0, \dots, n. \tag{1}$$

Als Lösungsansatz wählen wir den einfachst möglichen

$$p(x) = \sum_{j=0}^{n} a_j x^j,$$

mit unbekannten Koeffizienten $a_j \in \mathbb{R}$. Einsetzen in (1) ergibt,

$$\sum_{j=0}^{n} a_j x_i^j = y_i, \qquad i = 0, \ldots, n.$$

Dies ist ein lineares Gleichungssystem für die unbekannten Koeffizienten $a_0, \ldots, a_n$. In Matrix-Vektor-Darstellung erhalten wir,

$$\underbrace{\begin{pmatrix} 1 & x_0 & x_0^1 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^1 & x_1^2 & \cdots & x_1^n \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^1 & x_n^2 & \cdots & x_n^n \end{pmatrix}}_{=M \in \mathbb{R}^{n+1 \times n+1}} \underbrace{\begin{pmatrix} a_0 \\ \vdots \\ \vdots \\ a_n \end{pmatrix}}_{\in \mathbb{R}^{n+1}} = \underbrace{\begin{pmatrix} y_0 \\ \vdots \\ \vdots \\ y_n \end{pmatrix}}_{\in \mathbb{R}^{n+1}}.$$

Man kann zeigen, dass das Gleichungssystem lösbar ist, wenn die x_i paarweise verschieden sind, d.h. falls $x_i \neq x_j$ für $i \neq j$ folgt $\det M \neq 0$.

Mit der bereits entwickelten Theorie können wir eine Lösung berechnen. Die Polynominterpolation durch Lösung dieses Gleichungssystems hat jedoch den Nachteil, dass sie relativ aufwändig ist ($\mathcal{O}(n^3)$). Es gibt bessere Methoden mit denen das Problem schneller lösbar ist.

Als Basis des Matrizenraums haben wir die kanonische gewählt, $\mathcal{B} = \{1, x, x^2, \ldots\}$. Wählt man hingegen eine an das Problem angepasste Basis, kann man den Aufwand zur Lösung erheblich reduzieren.

3-B Lagrange-Interpolation

Wir wählen eine spezielle Basis,

$$\{L_0(x), \ldots, L_n(x)\}, \text{ mit } L_j(x_i) = \delta_{ij}.$$

Die Basiselemente sind Polynome vom Grad $\leq n$. Zu deren Konstruktion wählen wir das Produkt der Nullstellen und normieren es,

$$L_j(x) = \frac{1}{\prod\limits_{\substack{k=0 \\ k \neq j}}^{n} (x_j - x_k)} \prod_{\substack{k=0 \\ k \neq j}}^{n} (x - x_k).$$

Offensichtlich gilt $L_j(x_i) = \delta_{ji}$. Die Menge $\{L_0, \ldots, L_n\}$ bildet eine Basis des $\mathcal{P}_n$, denn

(i) $L_j \in \mathcal{P}_n,\ \forall\ j = 0, \ldots, n.$

(ii) $\dim \mathcal{P}_n = n + 1 = \text{card}\,\{L_0, \ldots, L_n\}.$

(iii) $\{L_0, \ldots, L_n\}$ ist linear unabhängig, denn seien $\lambda_0, \ldots, \lambda_n \in \mathbb{R}$ mit

$$\lambda_0 L_0(x) + \ldots + \lambda_n L_n(x) = 0,\ \forall\ x \in \mathbb{R},$$

so folgt für $x = x_i$, $\lambda_i = 0$.

Als Ansatz für das Interpolationspolynom wählen wir,

$$p(x) = \sum_{j=0}^{n} c_j L_j(x).$$

Einsetzen der Daten in p ergibt,

$$\sum_{j=0}^{n} c_j L_j(x_i) = \sum_{j=0}^{n} c_j \delta_{ij} = y_i$$
$$\Rightarrow c_i = y_i, \qquad i = 0, \ldots, n.$$

D.h. wir müssen das Interpolationspolynom nicht berechnen, sondern es ist sofort und eindeutig durch die Daten bestimmt.

Ergebnis. Lagrange Darstellung des Interpolationspolynoms,

$$p(x) = \sum_{j=0}^{n} y_j L_j(x),\ \text{mit}\ L_j(x) = \frac{1}{\prod\limits_{\substack{k=0 \\ k\neq j}}^{n} (x_j - x_k)} \prod_{\substack{k=0 \\ k\neq j}}^{n} (x - x_k). \tag{2}$$

Diese Darstellung ist für theoretische Zwecke sehr wichtig. ⊸

3.2 **Korollar** *Seien $x_0, \ldots, x_n, y_0, \ldots, y_n \in \mathbb{R}$, $x_i \neq x_j$ für $i \neq j$. Dann hat die Interpolationsaufgabe, finde $p \in \mathcal{P}_n$ mit*

$$p(x_i) = y_i,\ \textit{für}\ i = 0, \ldots, n,$$

genau eine Lösung. ⋊

Bsp Sei $n = 2$ und folgende Daten gegeben,

x_i	1	2	3
y_i	2	3	6

Die Basiselemente haben die Form,

$$L_0(x) = \frac{(x-2)(x-3)}{(1-2)(1-3)} = \frac{1}{2}(x-2)(x-3) = \frac{1}{2}\left(x^2 - 5x + 6\right),$$
$$L_1(x) = \frac{(x-1)(x-3)}{(2-1)(2-3)} = -(x-1)(x-3) = -x^2 + 4x - 3,$$
$$L_2(x) = \frac{(x-1)(x-2)}{(3-1)(3-2)} = \frac{1}{2}(x-1)(x-2) = \frac{1}{2}\left(x^2 - 3x + 2\right).$$

Das Interpolationspolynom ist daher,

$$\begin{aligned} p(x) &= 2\frac{1}{2}\left(x^2 - 5x + 6\right) + 3\left(-x^2 + 4x - 3\right) + 6\frac{1}{2}\left(x^2 - 3x + 2\right) \\ &= x^2 - 2x + 3. \quad \blacksquare \end{aligned}$$

Der Vorteil dieser Methode ist, dass das Interpolationspolynom mit wenig Aufwand aufgestellt werden kann. Der Nachteil ist jedoch, dass die Auswertung sehr rechenintensiv ist.

Die Lagrange-Interpolation ist komplementär zur Polynominterpolation in der kanonischen Basis, dort war die Berechnung des Interpolationspolynom aufwändig, seine Form erlaubt dagegen eine Auswertung mit wenig Aufwand.

In Bezug auf den Rechenaufwand haben wir durch die Lagrange-Interpolation nichts gewonnen, sie ist jedoch aufgrund ihreres hohen theoretischen Werts sehr wichtig.

3-C Newtonsche Interpolationsmethode

Die Idee bei der Newtonschen Interpolationsmethode ist, eine Basis zu wählen, so dass sowohl das Aufstellen des Interpolationspolynoms sowie die Auswer-

tung für $x \in \mathbb{R}$ schnell ist. Wir wählen dazu die spezielle Basis $(q_j)_{j=0}^n$

$$
\begin{aligned}
q_0(x) &= 1 \\
q_1(x) &= x - x_0 \\
q_2(x) &= (x - x_0)(x - x_1) \\
&\vdots \\
q_n(x) &= (x - x_0) \cdots (x - x_{n-1})
\end{aligned}
$$

bzw.

$$
q_j(x) = \prod_{k=0}^{j-1} (x - x_k), \qquad j = 0, \ldots, n. \tag{3}
$$

Die Basiselemente haben eine gewisse Ähnlichkeit zu denen der Lagrange-Interpolation, es werden aber nur die ersten $j-1$ Nullstellen multipliziert. Für die Auswertung gilt,

$$
q_j(x_i) = \begin{cases} 0, & i < j \\ \prod_{k=0}^{j-1} (x_i - x_k), & i \geq j. \end{cases}
$$

Ansatz für das Interpolationspolynom,

$$
p(x) = \sum_{j=0}^{n} c_j q_j(x).
$$

Einsetzen in die Interpolationsbedingung ergibt,

$$
\begin{aligned}
& p(x_i) = y_i, \qquad i = 0, \ldots, n \\
\Rightarrow & c_0 \cdot 1 && = y_0, \\
\Rightarrow & c_0 \cdot 1 + c_1(x_1 - x_0) && = y_1, \\
& \vdots \\
\Rightarrow & c_0 \cdot 1 + c_1(x_n - x_0) + \ldots + c_n(x_n - x_0) \cdots (x_n - x_{n-1}) && = y_n.
\end{aligned}
$$

Wir erhalten also ein lineares Gleichungssystem für $c_0, \ldots, c_n$. In Matrix-Vektor-Schreibweise gilt

$$\underbrace{\begin{pmatrix} 1 & & & & \\ 1 & q_1(x_1) & & & \\ 1 & q_1(x_2) & q_2(x_2) & & \\ \vdots & \vdots & \vdots & \ddots & \\ 1 & q_1(x_n) & q_2(x_n) & \cdots & q_n(x_n) \end{pmatrix}}_{=M} \begin{pmatrix} c_0 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} y_0 \\ \vdots \\ y_n \end{pmatrix} \tag{4}$$

Im Fall $x_i \neq x_j$ für alle $i \neq j$ hat das Gleichungssystem eine eindeutige Lösung. Die Matrix M ist eine linkere untere Dreiecksmatrix, man kann daher das Gleichungssystem ohne Gauß-Algorithmus einfach durch Vorwärtsauflösen lösen. Da die Polynome noch "relativ schön" sind, ist auch die Auswertung "relativ günstig".

Bsp Sei $n = 2$ mit den bekannten Daten.

x_i	1	2	3
y_i	2	3	6

Die Basiselemente sind,

$$q_0(x) = 1,$$
$$q_1(x) = x - 1,$$
$$q_2(x) = (x-1)(x-2).$$

Das Gleichungssystem hat die Form,

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 2 & 2 \end{pmatrix} = \begin{pmatrix} 2 \\ 3 \\ 6 \end{pmatrix}.$$

$$\Rightarrow c_0 = 2, \quad c_1 = 3 - 1 \cdot c_0 = 1, \quad c_2 = \frac{1}{2}(6 - 2 - 2) = 1$$

$$\Rightarrow p(x) = 2 + 1(x-1) + 1(x-1)(x-2) = x^2 - 2x + 3$$

Diese Rechnung ist für Computer jedoch immernoch mit erheblichem Rechenaufwand verbunden, der hauptsächlich durch das Aufstellen des Gleichungssystems und der damit verbunden Berechnung der Polynome $q_i(x)$ zustande kommt. ■

Es gibt einen sehr effizienten Algorithmus zur Berechnung der c_i aus (3), der auf folgender Rekursionsformel basiert.

3.3 **Satz** *Sei $p_{i,j} \in P_j$ das Interpolationspolynom zu den Daten*

$$(x_k, y_k), \qquad \text{für } k = i, \dots, i+j.$$

Dann gilt

$$p_{i,j}(x) = \frac{(x - x_i)p_{i+1,j-1}(x) - (x - x_{i+j})p_{i,j-1}(x)}{x_{i+j} - x_i}, \qquad \text{für } j \geq 1. \quad \bowtie \tag{5}$$

» Wir müssen nachweisen, dass unsere Definition von $p_{i,j}(x)$ die Interpolationseigenschaften erfüllt. Es gilt:

$$\begin{aligned} p_{i+1,j-1}(x_k) &= y_k, && \text{für } k = i+1, \dots, i+j, \\ p_{i,j-1}(x_k) &= y_k, && \text{für } k = i, \dots, i+j-1. \end{aligned}$$

Für $q(x) = p_{i,j}(x)$ folgt,

$$\begin{aligned} q(x_i) &= \frac{-(x_i - x_{i+j})p_{i,j-1}(x_i)}{x_{i+j} - x_i} = y_i, \\ q(x_{i+j}) &= \frac{(x_{i+j} - x_i)p_{i+1,j-1}(x_{i+j})}{x_{i+j} - x_i} = y_j \end{aligned}$$

Für $k = i+1, \dots, i+j-1$ gilt,

$$\begin{aligned} q(x_k) &= \frac{(x_k - x_i)p_{i+1,j-1}(x_k) - (x_k - x_{i+j})p_{i,j-1}(x_k)}{x_{i+j} - x_i} \\ &= \frac{(x_k - x_i)y_k - (x_k - x_{i+j})y_k}{x_{i+j} - x_i} \\ &= \frac{(x_{i+j} - x_i)y_k}{x_{i+j} - x_i} = y_k. \end{aligned}$$

D.h. $q_{i,j}(x)$ ist das Interpolationspolynom. «

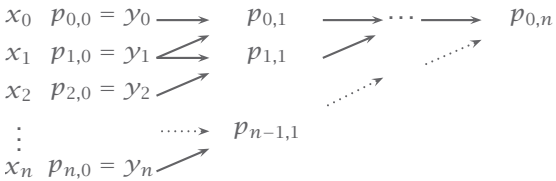
Aus (5) und den Anfangspolynomen

$$p_{i,0}(x) = y_i$$

kann man durch folgendes Schema das Interpolationspolynom

$$p(x) = p_{0,n}(x)$$

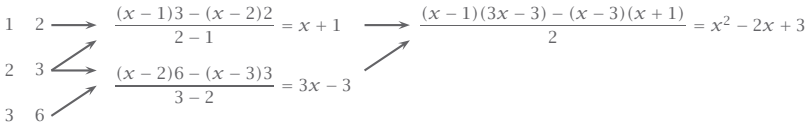
zu den Daten $(x_0, y_0), \ldots, (x_n, y_n)$ bestimmen.



Zur Bestimmung von $p_{i,j}$ werden also die Polynome $p_{i,j-1}$ und $p_{i+1,j-1}$ benötigt.

Bsp Sei $n = 2$ mit den bekannten Daten.

x_i	1	2	3
y_i	2	3	6



Das Interpolationspolynom ist gegeben durch $p(x) = x^2 - 2x + 3$. ■

Wie wir sehen ist unterscheidet sich das Verfahren im Rechenaufwand kaum vom vorherigen, wir können es aber durch eine kleine Modifikation deutlich vereinfachen.

Newtonsche Darstellung des Interpolationspolynoms

$$p(x) = c_0 + c_1(x - x_0) + \ldots + c_n(x - x_0) \cdots (x - x_{n-1}).$$

Darstellung von $p_{0,k}(x)$

$$p_{0,k}(x) = c_0 + c_1(x - x_0) + \ldots + c_k(x - x_0) \cdots (x - x_{k-1})$$

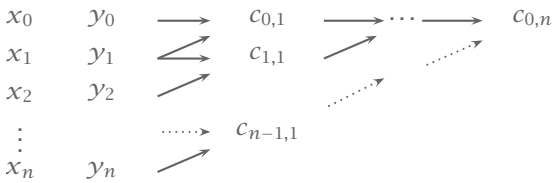
Wichtige Beobarchtung. c_k ist der Koeffizient am Term x^k der höchsten Ordnung von $p_{0,k}(x)$, welcher dem Koeffizent c_k von p entspricht. ⊸

Korollar *Man kann alle c_k, $k = 0, \dots, n$ durch ein Rekursionsverfahren für die Koeffizienten c_{ij} von $p_{i,i+j}$ am Term x^j höchster Ordnung auszurechnen. Die Rekursionsformel ist gegeben durch,*

$$c_{i,j} = \frac{c_{i+1,j-1} - c_{i,j-1}}{x_{i+j} - x_i}.$$

Sie heißt Formel der dividierten Differenzen. Startwerte sind

$$c_{i,0} = y_i, \qquad \text{für } i = 1, \dots, n. \quad \bowtie$$

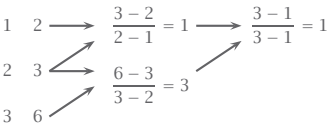


3 **Schema der Rekursionsformel für die Koeffizenten.**

Die Koeffizienten des Interpolationspolynoms sind $c_k = c_{0,k}$.

Bsp

x_i	1	2	3
y_i	2	3	6



Das Interpolationspolynom ist gegeben durch,

$$p(x) = 2 + 1(x-1) + 1(x-1)(x-2) = x^2 - 2x + 3. \quad \blacksquare$$

Bei einer Implementierung wählt man als Parameter für Stufe j,

$$c_i := c_{i-j,j}, \qquad i = j, \dots, n$$

D.h. man lässt bei Stufe j die ersten j Zeilen unverändert. Das macht auch Sinn, denn man benötigt diese Werte am Ende, um das Polynom aufzustellen.

Algorithmus für die Newton-Interpolation.

Für $i = 0, \ldots, n$
  $c_i = y_i$
Für $j = 1, \ldots, n$
  Für $i = n, n-1, \ldots, j$
    $c_i = \dfrac{c_i - c_{i-1}}{x_i - x_{i-j}}$

Das Interpolationspolynom lautet dann,

$$p(x) = c_0 + c_1(x - x_0) + \ldots + c_n(x - x_0)(x - x_1) \cdots (x - x_{n-1}).$$

Rechenaufwand. Wir berücksichtigen nur die Divisionen,

$$\sum_{j=1}^{n} \sum_{i=j}^{n} 1 = \sum_{j=1}^{n} (n - j + 1) = (n+1)n - \frac{n(n-1)}{2} = \frac{n(n-1)}{2}. \quad \multimap$$

Polynom in Newton-Darstellung:

$$p(x) = c_0 + c_1(x - x_0) + \ldots + c_n(x - x_0) \cdots (x - x_{n-1})$$

Zur effizienten Auswertung können wir das Horner-Schema verwenden. Dieses beruht im Prinzip auf einer geschickten Klammerung des Polynoms.

$$p(x) = \Big(\ldots (c_n(x - x_{n-1}) + c_{n-1})(x - x_{n-2}) + \ldots \Big) + c_0.$$

Bsp $n = 3$

$$\begin{aligned} p(x) &= c_0 + c_1(x - x_0) + c_2(x - x_1)(x - x_0) + c_3(x - x_0)(x - x_1)(x - x_2) \\ &= ((c_3(x - x_2) + c_2)(x - x_1) + c_1)(x - x_0) + c_0 \end{aligned}$$ ■

Algorithmus Horner-Schema.

$p = c_n$
Für $k = n-1, n-2, \ldots, 0$
$\quad p = p(x - x_k) + c_k$

Rechenaufwand. Im wesentlichen n Multiplikationen.

Für Berechnungen von Hand wird folgendes Schema verwendet.

c_{n-1}	c_{n-2}	$\ldots c_1$	c_0	
$x - x_{n-1}$	$x - x_{n-2}$	$\ldots x - x_1$	$x - x_0$	
$c_n y_{n-1}$	y_{n-2}	$\ldots y_1$	y_0	$= p(x)$

Mit der Rechenvorschrift $y_k = y_{k+1}(x - x_k) + c_k$.

Bsp

$$p(x) = 2 + 3x - x(x+2) + x(x+2)(x-1).$$

Auswertung in $x = 3$ $(n = 3)$

	-1	3	2	
	$(3-2)$	$(3+2)$	3	
1	1	8	26	$= p(3)$

Wir sehen natürlich auch durch Einsetzen,

$$p(3) = 2 + 9 - 15 + 30 = 26. \quad \blacksquare$$

3-D Approximationseigenschaften

Interpolationsaufgabe. Finde $p \in \mathcal{P}_n$ mit $p(x_i) = y_i$ für $i = 0, \ldots, n$.

Wir wollen dazu annehmen, dass ein $f : \mathbb{R} \to \mathbb{R}$ existiert mit $y_i = f(x_i)$ und f hinreichend "glatt".

Frage: Wie gut approximiert das Polynom p die Funktion f?

3.4 **Satz** *Sei $p \in \mathcal{P}_n$ das Interpolationspolynom mit $p(x_i) = f(x_i)$, $i = 0, \dots, n$ wobei $x_i \in \mathbb{R}\ \forall\ i$, $x_i \neq x_j$ für $i \neq j$ und $f \in C^{n+1}(\mathbb{R})$. Dann gilt*

$$|p(x) - f(x)| = \frac{1}{(n+1)!} \left| f^{n+1}(\xi)(x - x_0)(x - x_1) \cdots (x - x_n) \right| \qquad (6)$$

mit $\xi = \xi(x) \in [\min\{x, x_0, \dots, x_n\}, \max\{x, x_0, \dots, x_n\}]$. ⋈

» Für $x = x_j$ mit $j \in \{0, \dots, n\}$ ist (6) erfüllt. Sei nun $x \neq \{x_0, \dots, x_n\}$. Sei

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

$$g(t) = f(t) - p(t) - \frac{\omega(t)}{\omega(x)}(f(x) - p(x)).$$

Die Funktion g hat die Nullstellen,

$$g(x) = 0, \quad g(x_j) = 0, \quad i = 0, \dots, n,$$

d.h. g hat $n + 2$ verschiedene Nullstellen. Wenden wir den Satz von Rolle auf g an, erhalten wir, dass g' mindestens $n+1$ verschiedene Nullstellen hat. Iteration ergibt, dass $g^{(k)}$ mindestens $n - k + 2$ verschiedene Nullstellen hat, $g^{(n+1)}$ hat somit mindestens eine Nullstelle $\xi \in [a, b]$.

$$0 = g^{(n+1)}(\xi) = f^{(n+1)}(\xi) - p^{(n+1)}(\xi) - \frac{\omega^{(n+1)}(\xi)}{\omega(x)}(f(x) - p(x))$$

$$\Rightarrow 0 = f^{(n+1)}(\xi) - \frac{(n+1)!}{\omega(x)}(f(x) - p(x))$$

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

Problem. Leider impliziert (6) nicht die Konvergenz des Interpolationspolynoms für $n \to \infty$,

$$|p(x) - f(x)| \to 0, \qquad n \to \infty,$$

denn $\left| f^{(n+1)}(\xi) \right|$ kann schneller wachsen als

$$\left(\frac{1}{(n+1)!} |(x - x_0) \cdots (x - x_n)| \right)^{-1}. \quad \multimap$$

In der Praxis kommt es häufig zu Oszillationen von p, vor allem nahe des Randes von $[\min\{x_0, \dots, x_n\}, \max\{x_0, \dots, x_n\}]$.

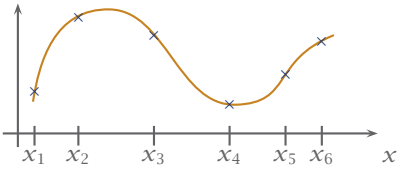
3-E Interpolation durch Splines

Interpolationsaufgabe. Finde eine Funktion

$$f : \mathbb{R} \to \mathbb{R}, \quad \text{mit } f(x_i) = y_i, \qquad i = 0, \ldots, n,$$

wobei $a = x_0 < x_1 < \ldots < x_n = b$.

Konzept der Spline-Interpolation *Man bestimmt n Polynome vom Grad m mit $m << n$ auf den Teilintervallen $[x_{i-1}, x_i]$ für $i = 1, \ldots, n$.* ⋈



4 **Spline-Interpolation.**

3.5 **Definition** *Eine Funktion $f : [a, b] \to \mathbb{R}$ mit $f \in C^{m-1}([a, b])$ und*

$$f\Big|_{[x_{i-1}, x_i]} \in \mathcal{P}_m, \qquad \text{für } i = 1, \ldots, n,$$

heißt Spline der Ordnung m zum Datensatz $X = (x_0, x_1, \ldots, x_n)$. Den Raum der Splines bezeichnen wir mit

$$S_m(X) = \left\{ f \in C^{m-1}([a, b]) : f\Big|_{[x_{i-1}, x_i]} \in \mathcal{P}_m, \text{ für } i = 1, \ldots, n, \right\}. \quad ⋈$$

Gegeben seien die Daten (x_i, y_i), $i = 0, \ldots, n$. Finde $s \in S^m(\{x_0, \ldots, x_m\})$ mit

$$s(x_j) = y_j, \qquad \text{für } j = 0, \ldots, n.$$

Darstellung eines Splines

$$s(x) = p_j(x) = \sum_{k=0}^{m} a_k^{(j)} x^k \quad \text{für } x \in [x_{j-1}, x_j],\ j = 1, \ldots, m.$$

Anzahl der Koeffizienten $(m+1)n$. Die Bedingungen für die Spline-Interpolation sind

$$p_j(x_j) = y_j, \quad p_j(x_{j-1}) = y_{j-1}$$
$$p_j^{(k)}(x_j) = p_{j-1}^{(k)}(x_j), \quad \text{für } j = 1,\dots,n-1, k = 1,\dots,m-1,$$

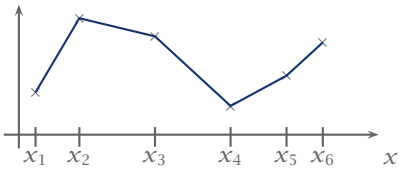
wir haben somit $2n + (m-1)(n-1) = 2n + mn - m - n + 1 = (m+1)n - m + 1$ Bedingungen, das sind $m-1$ Bedingungen weniger als Koeffizienten. Für eine eindeutige Lösung benötigen wir daher zusätzliche Bedingungen. Eine übliche Wahl sind Bedingungen für die Ableitungen an den Endpunkten x_0, x_n.

Bsp Lineare Splines ($m = 1$). Das Interpolationspolynom hat somit die Darstellung,

$$s(x) = p_j(x) = a_j x + b_j, \qquad \text{für } x \in [x_{j-1}, x_j].$$

$p_j(x_j) = y_j, \quad p_j(x_{j-1}) = y_{j-1}$

$$p_j(x) = y_j \frac{x - x_{j-1}}{x_j - x_{j-1}} + y_{j-1} \frac{x - x_j}{x_{j-1} - x_j}, \qquad j = 1,\dots,n.$$



5 **Interpolation durch lineare Splines.**

Bsp Kubische Splines ($m = 3$).

Die kubischen Splines sind ein sehr wichtiges Beispiel für Spline Interpolation. Oft wird in der Literatur auch von Splines geredet, während kubische Splines gemeint sind.

Hier ist $m = 3$, es fehlen also zwei Bedingungen. Es gibt nun zahlreiche Möglichkeiten, diese zu wählen.

1.) Natürliche Splines.

$$p_1''(x_0) = 0, \quad p_n''(x_n) = 0, \tag{R1}$$

d.h. keine Krümmung an den Endpunkten.

2.) Hermitsche Randbedingungen (Eingespannter Spline). Wähle $\alpha, \beta \in \mathbb{R}$,

$$p_1'(x_0) = \alpha, p_n'(x_n) = \beta. \tag{R2}$$

3.) Periodische Splines.

$$\begin{aligned} p_1'(x_0) &= p_n'(x_n) \\ p_1''(x_0) &= p_n''(x_n) \end{aligned} \tag{R3}$$

Man erhält eine C^2-periodische Lösung. Dies macht natürlich nur Sinn, falls die Funktion an den Endpunkten den selben Wert annimmt.

■ Berechnung von kubischen Splines

Wir wollen ein Verfahren zur Berechnung der kubischen Splines anhand von natürlichen Splines erarbeiten.

$$p_1''(x_0) = 0, \quad p_n''(x_n) = 0.$$

Wir wollen hier eine spezielle Darstellung wählen, die für die Berechnung jedoch äußerst geschickt ist,

$$s(x) = p_j(x) = a_j(x - x_{j-1})^3 + b_j(x - x_{j-1})^2 + c_j(x - x_{j-1}) + d_j,$$

für $x \in [x_{j-1}, x_j]$, $j = 1, \ldots, n$.

$$\begin{aligned} &p_j(x_{j-1}) = y_{j-1}, && p_j(x_j) = y_j, && j = 1, \ldots, n \\ &p_j'(x_j) = p_{j+1}'(x_j), && p_j''(x_j) = p_{j+1}''(x_j), && j = 1, \ldots, n-1 \\ &p_1''(x_0) = 0,\ p_n''(x_n) = 0 \end{aligned}$$

Es ist nicht effizient Gleichungssysteme für a_j, b_j, c_j und d_j zu bestimmen. Viel geschickter ist die Idee $M_j = s''(x_j)$, $j = 0, \dots, n$ als Variablen zu verwenden. Zusätzlich verwenden wir die Notation

$$h_j := x_j - x_{j-1}.$$

Wir können a_j, b_j, c_j und d_j nun durch Differenzieren von p_j bestimmen,

$$\begin{aligned} p_j''(x_{j-1}) &= 2b_j := M_{j-1} \\ p_j''(x_j) &= 6a_j h_j + 2b_j = M_j \\ \Rightarrow a_j &= \frac{M_j - M_{j-1}}{6h_j}. \end{aligned}$$

Des Weiteren gilt für $j = 1, \dots, n-1$

$$\begin{aligned} p_j(x_{j-1}) &= d_j = y_{j-1} \\ p_j(x_j) &= a h_j^3 + b_j h_j^2 + c_j h_j + d_j = y_j \\ \Rightarrow c_j &= \frac{y_j - d_j}{h_j} - a_j h_j^2 - b_j h_j = \frac{y_j - y_{j-1}}{h_j} - \left(\frac{M_j - M_{j-1}}{6h_j}\right) h_j^2 - \frac{M_{j-1}}{2} h_j \\ &= \frac{y_j - y_{j-1}}{h_j} - h_j \left(\frac{M_j}{6} - \frac{M_{j-1}}{6} + \frac{3M_{j-1}}{6}\right) \\ &= \frac{y_j - y_{j-1}}{h_j} - h_j \left(\frac{M_j}{6} + \frac{M_{j-1}}{3}\right) \end{aligned}$$

$$\begin{aligned} p_j'(x_j) &= 3a_j h_j^2 + 2b_j h_j + c_j \\ &= 3\left(\frac{M_j - M_{j-1}}{6h_j}\right) h_j^2 + M_{j-1} h_j + \frac{y_j - y_{j-1}}{h_j} - h_j \left(\frac{M_j}{6} + \frac{M_{j-1}}{3}\right) \\ &= \frac{y_j - y_{j-1}}{h_j} + h_j \left(\frac{M_j}{3} + \frac{M_{j-1}}{5}\right) \end{aligned}$$

Verwenden wir nun die Stetigkeitsbedingung der Ableitung, ergibt sich,

$$p_{j+1}'(x) = c_{j+1} = \frac{y_{j+1} - y_j}{h_{j+1}} - h_{j+1} \left(\frac{M_{j+1}}{6} + \frac{M_j}{3}\right) \overset{!}{=} p_j'(x_j)$$

$$\frac{h_j + h_{j+1}}{3} M_j + \frac{h_j}{6} M_{j-1} + \frac{h_{j+1}}{6} M_{j+1} = \frac{y_{j+1} - y_j}{h_{j+1}} - \frac{y_j - y_{j-1}}{h_j}$$

Wir haben also ein Gleichungssystem für M_j, für $j = 1, \dots, n-1$. Zusammengefasst ergibt sich,

$$h_j M_{j-1} + 2(h_j + h_{j+1}) M_j + h_{j+1} M_{j+1} = 6 \left(\frac{y_{j+1} - y_j}{h_{j+1}} - \frac{y_j - y_{j-1}}{h_j} \right),$$

wobei für natürliche Splines $M_0 = M_n = 0$.

$$\begin{pmatrix} 2(h_1+h_2) & h_2 & & & \\ h_2 & 2(h_2+h_3) & h_3 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & h_{n-1} \\ & & & h_{n-1} & 2(h_{n-1}+h_n) \end{pmatrix} \begin{pmatrix} M_1 \\ \vdots \\ \vdots \\ \vdots \\ M_{n-1} \end{pmatrix}$$

$$= \begin{pmatrix} \frac{6}{h_2}(y_2 - y_1) - \frac{6}{h_1}(y_1 - y_0) \\ \\ \vdots \\ \vdots \\ \vdots \\ \frac{6}{h_n}(y_n - y_{n-1}) - \frac{6}{h_{n-1}}(y_{n-1} - y_{n-2}) \end{pmatrix} \tag{7}$$

Für äquidistante Punkte $x_j = x_0 + jh$, $j = 0, \dots, n$, vereinfacht sich die Matrix deutlich,

$$\begin{pmatrix} 4 & 1 & & \\ 1 & \ddots & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & 4 \end{pmatrix} \begin{pmatrix} M_1 \\ \vdots \\ \vdots \\ M_{n-1} \end{pmatrix} = \frac{6}{h^2} \begin{pmatrix} y_2 - 2y_1 + y_0 \\ y_3 - 2y_2 + y_1 \\ \vdots \\ y_n - 2y_{n-1} + y_{n-2} \end{pmatrix}.$$

Das Gleichungssystem (7) hat eine strikt diagonaldeterminante Matrix, ist also eindeutig lösbar.

Bsp

x_i	1	2	3
y_i	2	3	6

Es sind $M_0 = M_2 = 0$. Für M_1 ergibt sich,

$$4M_1 = 6(6 - 2 \cdot 3 + 2) = 12 \Rightarrow M_1 = 3.$$

Der Spline hat die Gestalt,

$$s(x) = \begin{cases} p_1(x), & x \in [1,2] \\ p_2(x), & x \in [2,3], \end{cases}$$

wobei

$$\begin{aligned} & p_1(x) = a_1(x-1)^3 + b_1(x-1)^2 + c_1(x-1) + d_1 \\ \Rightarrow & a_1 = \frac{M_1 - M_0}{6h} = \frac{3-0}{6 \cdot 1} = \frac{1}{2} \\ & b_1 = \frac{M_0}{2} = 0 \\ & c_1 = \frac{3-2}{1} - 1\left(\frac{3}{6} + \frac{0}{3}\right) = \frac{1}{2} \\ & d_1 = y_0 = 2. \end{aligned}$$

Wir erhalten somit

$$p_1(x) = \frac{1}{2}(x-1)^2 + \frac{1}{2}(x-1) + 2.$$

Für p_2 erhalten wir,

$$\begin{aligned} & p_2(x) = a_2(x-2)^3 + b_2(x-2)^2 + c_2(x-2) + d_2 \\ \Rightarrow & a_2 = \frac{-3}{6} = -\frac{1}{2} \\ & b_2 = \frac{M_1}{2} = \frac{3}{2} \\ & c_2 = \frac{6-3}{1} - \left(0 + \frac{3}{3}\right) = 2 \\ & d_2 = y_1 = 3 \\ \Rightarrow & p_2(x) = -\frac{1}{2}(x-2)^3 + \frac{3}{2}(x-2)^2 + 2(x-2) + 3. \end{aligned}$$ ■

■ Approximationseigenschaften von Splines

Interpolationsaufgabe. Gegeben sind die Daten,

$$a = x_0 < x_1 < x_2 < \ldots < x_n = b,$$
$$y_0, \ldots, y_n \in \mathbb{R}.$$

Finde $s \in S_3(X)$, $X = (x_0, x_1, \ldots, x_n)$ mit

$$s(x_j) = y_j \quad \text{für } j = 0, \ldots, n,$$

die eine der folgenden Randbedingungen erfüllt,

$$s''(a) = s''(b) = 0, \tag{R1}$$
$$s'(a) = \alpha, (b) = \beta, \qquad \alpha, \beta \in \mathbb{R} \text{ gegeben}, \tag{R2}$$
$$s'(a) = s'(b),\ s''(a) = s''(b), \tag{R3}$$

wobei (R3) nur sinnvoll ist, wenn $s(a) = s(b)$.

Wir wollen nun Funktionenräume definieren, die unseren Randbedigungen entsprechen.

$$V_1 = \left\{ f \in C^2([a,b]) : f(x_j) = y_j, \quad \text{für } j = 0, \ldots, n \right\}$$
$$V_2 = \{ f \in V_1 : f'(a) = \alpha,\ f'(b) = \beta \}$$
$$V_3 = \{ f \in V_1 : f'(a) = f'(b),\ f''(a) = f''(b) \}$$

Wir können damit die Interpolationsaufgabe auch als, finde $s \in S_3 \cap V_j$, definieren.

3.6 **Optimalitätskriterium** *Sei $j \in \{1,2,3\}$, $s \in V_j \cap S_3$ die Splineinterpolation für $f \in V_j$ und $s''(a) = s''(b) = 0$ falls $j = 1$. Dann gilt*

$$\int_a^b |s''(x)|^2 \, \mathrm{d}x \le \int_a^b |f''(x)|^2 \, \mathrm{d}x.$$

D.h. s löst das Optimierungsproblem

$$\min_{g \in V_j} \int_a^b |g''(x)|^2 \, \mathrm{d}x. \quad \rtimes$$

» Sei $f \in V_j$, $s \in S_3 \cap V_j$. Für

$$\int_a^b |s''(x)|^2 \, \mathrm{d}x = 0$$

folgt die Behauptung sofort. s ist abschnittsweise ein Polynom vom Grad 3, die 4. Ableitung verschwindet also. Die Idee ist nun, das Integral durch partielle Integration so umzuformen, dass Ableitungen von s 4. Grades auftreten,

$$\begin{aligned}\int_a^b (f-s)''(x)s''(x)\,\mathrm{d}x &= \sum_{j=1}^{n} \int_{x_{j-1}}^{x_j} (f-s)''(x)s''(x)\,\mathrm{d}x \\ &\overset{\text{part.int.}}{=} \sum_{j=1}^{n} \left(\left[(f-s)'(x)s''(x)\right]_{x_{j-1}}^{x_j} - \int_{x_{j-1}}^{x_j} (f-s)'(x)s'''(x)\,\mathrm{d}x \right).\end{aligned}$$

Mit

$$\sum_{j=1}^{n} \left[(f-s)'(x)s''(x)\right]_{x_{j-1}}^{x_j} = (f-s)'(b)s''(b) - (f-s)'(a)s''(a) = 0$$

$$\text{wegen} \begin{cases} s''(a) = s''(b) = 0, & j=1, \\ (f-s)'(a) = (f-s)'(b) = 0, & j=2, \\ (f-s)'(a) = (f-s)'(b),\ s''(a) = s''(b), & j=3, \end{cases}$$

folgt

$$\begin{aligned}&\int_a^b (f-s)''(x)s''(x)\,\mathrm{d}x \\ &= \sum_{j=1}^{n} - \underbrace{\left[(f-s)(x)p_j'''(x)\right]_{x_{j-1}}^{x_j}}_{0 \text{ wegen } f(x_j)=s(x_j)} + \int_{x_{j-1}}^{x_j} (f-s)(x) \underbrace{p_j^{(4)}(x)}_{=0,\ \text{da } \deg p_j = 3}\,\mathrm{d}x,\end{aligned}$$

wobei $s'''(x)\big|_{[x_{j-1},x_j]} = p_j'''(x)$. Insgesamt gilt also

$$\begin{aligned}&\int_a^b (s''(x))^2\,\mathrm{d}x = \int_a^b f''(x)s''(x)\,\mathrm{d}x \\ &\le \left(\int_a^b (f''(x))^2\,\mathrm{d}x\right)^{1/2} \left(\int_a^b (s''(x))^2\,\mathrm{d}x\right)^{1/2} \\ \Rightarrow &\int_a^b (s''(x))^2\,\mathrm{d}x \le \int_a^b (f''(x))^2\,\mathrm{d}x \quad \text{«}\end{aligned}$$

Physikalische Interpretation. Wir betrachten den Graphen des Spline als Kurve

$$\Gamma = \left\{ \begin{pmatrix} x \\ s(x) \end{pmatrix} : x \in [a,b] \right\}.$$

Die Krümmung des Splines ist gegeben durch

$$\kappa(x) = \frac{s''(x)}{\sqrt{1 + (s'(x))^2}^3}.$$

Wenn man eine dünne Latte in die Form Γ bringt, hat man die Biegeenergie

$$\begin{aligned} E &= \gamma \int_\Gamma |\kappa(x)|^2 \, \mathrm{d}x = \gamma \int_a^b \frac{|s''(x)|^2}{(1 + (s'(x)^2))^3} \sqrt{1 + (s'(x))^2} \, \mathrm{d}x \\ &= \gamma \int_a^b \frac{|s'(x)|^2}{(1 + (s'(x))^2)^{5/2}} \, \mathrm{d}x \end{aligned}$$

Für *kleines* s' gilt näherungsweise

$$E \approx \gamma \int_a^b |s''(x)|^2 \, \mathrm{d}x.$$

Korollar *Kubische Splines approximieren das Minimum der Biegeenergie eines dünnen Splines.* ⋊ ⊸

Notation.

$$\begin{aligned} \|u\|_{L^2(a,b)} &= \left(\int_a^b |u(x)|^2 \, \mathrm{d}x \right)^{1/2} \\ \|u\|_{L^\infty(a,b)} &= \sup_{x \in (a,b)} |u(x)|. \end{aligned}$$

Man kann zeigen, dass so eine Norm auf einem geeignet definierten Funktionenraum, dem $\mathcal{L}^p((a,b))$ definiert wird. ⊸

3.7 **Satz** *Sei* $a \le x_0 < x_1 < \ldots < x_n = b$.

$$h = \max_{j=1,\ldots,n} \left| x_j - x_{j-1} \right|, \quad u \in C^k([a,b]) \text{ mit } k < n,$$
$$u(x_j) = 0, \quad j = 0, \ldots, n.$$

Dann gilt

$$\left\| u^{(l)} \right\|_{L^2(a,b)} \le \frac{k!}{l!} h^{k-l} \left\| u^{(k)} \right\|_{L^2(a,b)}, \qquad l = 0, 1, \ldots, k \tag{8}$$

$$\left\| u^{(l)} \right\|_{L^\infty(a,b)} \le \frac{k!}{l!\sqrt{k}} h^{k-l-1/2} \left\| u^{(k)} \right\|_{L^2((a,b))}, \qquad l = 0, 1, \ldots, k. \quad \rtimes \tag{9}$$

» Die Beweisidee besteht zunächst in einer mehrfachen Anwendung des Satz von Rolle. u hat $n+1$ Nullstellen mit maximalem Abstand h. u' hat n Nullstellen mit maximalem Abstand $2h$. $u^{(l)}$ hat $n+1-l$ Nullstellen mit maximalem Abstand $(l+1)h$.

Wir machen folgende Fallunterscheidung. Für $l = 0$ sei,

$$y_j = x_j, \quad j = 0, \ldots, n = N_0.$$

Für $l > 0$ seien die Nullstellen von $u^{(l)}$,

$$y_1 < y_2 < \ldots < y_{n+1-l},$$

sowie

$$y_0 = a,\ y_{n+2-l} = b,\ N_l = n + 2 - l.$$

Zu (8): Für $x \in [y_{j-1}, y_j]$ gilt

$$\left| u^{(l)}(x) \right| \overset{j \ge 2}{=} \left| \int_{y_{j-1}}^{x} u^{(l+1)}(t)\,\mathrm{d}t \right| \le \left(\int_{y_{j-1}}^{x} 1^2\,\mathrm{d}t \right)^{1/2} \left(\int_{y_{j-1}}^{x} \left| u^{(l+1)}(t) \right|^2 \mathrm{d}t \right)^{1/2}$$
$$\le ((l+1)h)^{1/2} \left(\int_{y_{j-1}}^{x} \left| u^{(l+1)}(t) \right|^2 \mathrm{d}t \right)^{1/2}, \tag{10}$$

bzw. für $j = 1$

$$\left| u^{(l)}(x) \right| = \left| \int_{x}^{y_1} u^{(l+1)}(t)\,\mathrm{d}t \right| \le ((l+1)h)^{1/2} \left(\int_{x}^{y_1} \left| u^{(l+1)}(t) \right|^2 \mathrm{d}t \right)^{1/2}$$

Insgesamt:

$$\begin{aligned}
&\int_a^b \left|u^{(l)}(x)\right|^2 \mathrm{d}x = \sum_{j=1}^{N_l} \int_{y_{j-1}}^{y_j} \left|u^{(l)}(x)\right|^2 \mathrm{d}x \\
&\leq \sum_{j=1}^{N_l} (l+1)h \int_{y_{j-1}}^{y_j} \int_{y_{j-1}}^{y_j} \left|u^{(l+1)}(t)\right|^2 \mathrm{d}t \, \mathrm{d}x \\
&\leq \sum_{j=1}^{N_e} (l+1)h \int_{y_{j-1}}^{y_j} \left|u^{(l+1)}(t)\right|^2 \mathrm{d}t \underbrace{\int_{y_{j-1}}^{y_j} 1 \, \mathrm{d}x}_{\leq (l+1)h} \\
&\leq (l+1)^2 h^2 \int_a^b \left|u^{(l+1)}(x)\right|^2 \mathrm{d}x, \\
\Rightarrow &\left\|u^{(l)}\right\|_{L^2(a,b)} \leq (l+1)h \left\|u^{(l+1)}\right\|_{L^2(a,b)}.
\end{aligned}$$

Wir können dies nun zukzessiv anwenden und erhalten,

$$\left\|u^{(l)}\right\|_{L^2(a,b)} \leq (l+1)h(l+2)h \cdots kh \left\|u^{(k)}\right\|_{L^2(a,b)} = \frac{k!}{l!} h^{k-l} \left\|u^{(k)}\right\|_{L^2(a,b)}.$$

Zu (9): Für $x \in [y_{j-1}, y_j]$ ersetzen wir (10) durch,

$$\begin{aligned}
\left|u^{(l)}(x)\right| &\leq \int_{y_{j-1}}^{y_j} \left|u^{(l+1)}(t)\right| \mathrm{d}t \leq (y_j - y_{j-1}) \sup_{x \in [a,b]} \left|u^{(l+1)}(x)\right| \\
&\leq (l+1)h \sup_{x \in [a,b]} \left|u^{(l+1)}(x)\right|,
\end{aligned}$$

d.h.

$$\left\|u^{(l)}\right\|_{L^\infty(a,b)} \leq (l+1) \left\|u^{(l+1)}\right\|_{L^\infty(a,b)}.$$

für $l \leq k-1$ und

$$\left|u^{(l)}(x)\right|^2 \overset{(10)}{\leq} (l+1)h \int_{y_{j-1}}^{y_j} \left|u^{(l+1)}(t)\right|^2 \mathrm{d}t \leq (l+1)h \int_a^b \left|u^{(l+1)}(t)\right|^2 \mathrm{d}t$$

d.h.

$$\left\|u^{(l)}\right\|^2_{L^\infty(a,b)} \le (l+1)h \left\|u^{(l+1)}\right\|_{L^2(a,b)}.$$

Insgesamt

$$\begin{aligned}\left\|u^{(l)}\right\|_{L^\infty(a,b)} &\le (l+1)h(l+2)h\cdots(k-1)h\sqrt{kh}\left\|u^{(k)}\right\|_{L^2(a,b)}\\ &= \frac{k!}{l!\sqrt{k}}h^{k-l-1/2}. \quad «\end{aligned}$$

3.8 **Satz** *Sei $s \in S_3(X)$ ein interpolierender Spline zu den Daten*

$$(x_j, f(x_j)), \qquad j = 0,\dots,n,$$

mit $f \in C^2([a,b])$ und einer der Randbedingungen $(R1)$-$(R3)$.
Im Fall $(R2)$ gelte,

$$f'(a) = \alpha,\ f'(b) = \beta,$$

und im Fall $(R3)$,

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

Dann gilt mit $h = \max\limits_{j=1,\dots,n} \left|x_j - x_{j-1}\right|$,

$$\left\|(s-f)^{(l)}\right\|_{L^2(a,b)} \le 2h^{2-l}\,\|f''\|_{L^2(a,b)}.$$

und

$$\left\|(s-f)^{(l)}\right\|_{L^\infty(a,b)} \le \sqrt{2}h^{2-l-1/2}\,\|f''\|_{L^2(a,b)},$$

jeweils für $l = 0, 1$. ⋊

» Anwendung von Satz 3.7 auf $u = s - f$ für $k = 2$ ⇒

$$\begin{aligned}\left\|s - f^{(l)}\right\|_{L^2((a,b))} &\le 2h^{2-l}\,\|(s-f)''\|_{L^2((a,b))},\\ \left\|s - f^{(l)}\right\|_{L^\infty((a,b))} &\le \sqrt{2}h^{2-l-1/2}\,\|(s-f)''\|_{L^2((a,b))}.\end{aligned}$$

Aus dem Beweis von 3.6 folgt,

$$\begin{aligned}
&\int_a^b s''(f''-s'')\,\mathrm{d}x = 0\\
\Leftrightarrow &\int_a^b (s'')^2\,\mathrm{d}x = \int_a^b s''f''\,\mathrm{d}x\\
&||(s-f)''||^2_{L^2(a,b)} = \int_a^b |\,s''-f''\,|^2\,\mathrm{d}x = \int_a^b (s'')^2 - 2s''f''(f'')^2\,\mathrm{d}x\\
&\qquad = \int_a^b (f'')^2 - (s'')^2\,\mathrm{d}x \le \int_a^b |\,f''\,|^2\,\mathrm{d}x = ||f''||_{L^2(a,b)}\,. \quad «
\end{aligned}$$

Bemerkung. Für eingespannte Splines d.h. Randbedingung $(R2)$, gilt sogar

$$||s-f||_{L^\infty((a,b))} \le h^4 \left|\left|f^{(4)}\right|\right|_{L^\infty(a,b)}.$$

» Siehe Schaback, Wendland, Numerische Mathematik Satz 11.13. « ⊸

3.9 **Satz** *Die Spline-Interpolationsaufgabe, finde $s \in S_3(X)$,*

$$\begin{aligned}
&\text{mit } s(x_j) = y_j, \qquad \text{für } j = 0,1,\ldots,n,\\
&\text{mit } a = x_0 < x_1 < \ldots < x_n = b,
\end{aligned}$$

hat für jede der Randbedingungen $(R1)$-$(R3)$ eine eindeutige Lösung. ⋈

» Die Interpolationsaufgabe entspricht einem linearen Gleichungssystem.

$$Mc = b, \qquad c \in \mathbb{R}^{4n},\ M \in \mathbb{R}^{4n\times 4n}$$

c ist der Vektor der Koeffizienten von $p_j = s\big|_{[x_{j-1},x_j]} \in \mathcal{P}^3$, M die Matrix mit den

$$4n = (2n) + (n-1) + (n-2) + 2$$

Bedingungen. Die Lösung von $Mc = 0$ liefert den Spline zu den Daten $s(x_j) = 0$, $j = 0, \ldots, n$ und im Fall $(R2)$ noch die Bedingung $s'(a) = s'(b) = 0$.

Wenden wir nun den Satz 3.8 mit $f \equiv 0$ an, so folgt

$$\|s - 0\|_{\mathcal{L}^\infty(a,b)} = \sqrt{2} h^{2-1/2} \, \|0''\|_{\mathcal{L}^2(a,b)} = 0.$$

D.h. $f = 0 \Rightarrow c = 0$, also ist $\ker M = (0)$, M invertierbar und daher ist das Gleichungssystem eindeutig lösbar. «

4 Numerische Integration

Aufgabenstellung. Gegeben sind die Datensätze $(x_i, f(x_i))$, $i = 0, 1, \ldots, n$ von Werten einer Funktion $f : [a, b] \to \mathbb{R}$.

Gesucht ist eine Approximation des Integrals,

$$\int_a^b f(x)\,\mathrm{d}x.$$

Anwendungen.

- Approximation von Integralen, die man nicht analytisch lösen kann.

  Bsp $\int_0^1 e^{x^2}\,\mathrm{d}x.$ ■

- Integration von Funktionen, die nicht vollständig bekannt sind, sondern nur durch gemessene Daten oder eine im Computer implementierte Funktion.

Die allgemeine Sturktur einer numerischen Integration ist,

$$\int_a^b f(x)\,\mathrm{d}x \approx \sum_{i=0}^{n} \omega_i f(x_i),$$

mit den Stützstellen x_i und (Integrations)-gewichten ω_i. Eine solche Formel nennt man auch Quadraturformel.

4-A Interpolatorische Quadraturformeln

Naheliegend ist es, Interpolationstechniken zu nutzen, um die Quadraturformel herzuleiten. Dazu integrieren wir das Interpolationspolynom zu den Daten

$$(x_i, f(x_i)), \qquad \text{für } i = 0, \ldots, n.$$

■ Lagrange Darstellung

Wir verwenden die Interpolationspolynome von Lagrange

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

mit

$$L_i(x) = \prod_{\substack{k=0 \\ k \neq i}}^{n} \frac{x - x_k}{x_i - x_k}, \quad L_i(x_k) = \delta_{ik}.$$

Aufgrund der Linearität der Integration erhalten wir für das Integral,

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

Bsp $[a,b] = [0,1]$, $n = 2$ und $x_0 = 0$, $x_1 = \frac{1}{2}$, $x_2 = 1$.

$$\begin{aligned}
\omega_0 &= \int_0^1 L_0(x)\,\mathrm{d}x = \int_0^1 \frac{(x - \frac{1}{2})(x-1)}{\frac{1}{2} \cdot 1}\,\mathrm{d}x \\
&= \int_0^1 2\left(x^2 - x - \frac{1}{2}x + \frac{1}{2}\right)\mathrm{d}x = 2\left(\frac{1}{3} - \frac{3}{4} + \frac{1}{2}\right) = \frac{1}{6}, \\
\omega_1 &= \int_0^1 L_0(x)\,\mathrm{d}x = \int_0^1 \frac{x(x-1)}{\frac{1}{2} \cdot \frac{-1}{2}}\,\mathrm{d}x \\
&= \int_0^1 -4\left(x^2 - x\right)\mathrm{d}x = -\frac{4}{3} + 2 = \frac{2}{3}, \\
\omega_2 &= \int_0^1 L_0(x)\,\mathrm{d}x = \int_0^1 \frac{x(x - \frac{1}{2})}{1 \cdot \frac{1}{2}}\,\mathrm{d}x \\
&= \int_0^1 2\left(x^2 - \frac{1}{2}x\right)\mathrm{d}x = \frac{2}{3} - \frac{1}{2} = \frac{1}{6}.
\end{aligned}$$

Für die Quadraturformel erhalten wir

$$Q(f) = \frac{1}{6}\left(f(0) + 4f\left(\frac{1}{2}\right) + f(1)\right). \quad \blacksquare$$

Ergebnis. Die interpolatorische Quadraturformel zu dem Integrationsintervall $[a, b]$ und den Stützstellen $x_0, \ldots, x_n$ ist,

$$Q(f) = \sum_{i=0}^{n} \omega_i f(x_i), \tag{1}$$

mit

$$\omega_i = \int_a^b \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j}\, \mathrm{d}x. \tag{2}$$

■ Newton-Cotes-Formeln

Newton-Cotes-Formeln sind interpolatorische Quadraturformeln mit *äquidistanten Stützstellen*, d.h. $x_i - x_{i-1} = h$.

Definition *Man unterscheidet zwischen abgeschlossenen Newton-Cotes-Formeln*

$$x_0 = a,\ x_n = b,\ x_i = a + i \cdot h,\ h = \frac{b-a}{n},$$

und offenen Newton-Cotes-Formeln

$$x_1 = a + \frac{h}{2},\ x_n = b - \frac{h}{2},\ x_i = a + \frac{2i-1}{2}h,\ h = \frac{b-a}{n}.$$

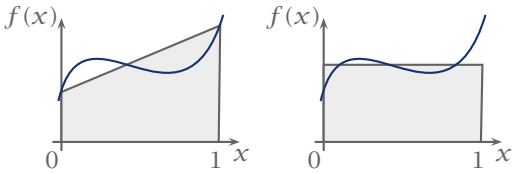
Die Quadraturformel hat die Form von (1),

$$Q(f) = \sum_{i=1}^{n} \omega_i f(x_i),$$

die ω_i sind durch (2) gegeben. ⋊

n	ω_i abgeschlossen		ω_i offen	
1	$\frac{1}{2}, \frac{1}{2}$	Trapezregel	1	Mittelpunktsregel
2	$\frac{1}{6}, \frac{2}{3}, \frac{1}{6}$	Simpson-Regel	$\frac{1}{2}, \frac{1}{2}$	
3	$\frac{1}{8}, \frac{3}{8}, \frac{3}{8}, \frac{1}{8}$	$\frac{3}{8}$-Regel	$\frac{3}{8}, \frac{1}{4}, \frac{3}{8}$	

6 **Integrationsgewichte der abgeschlossenen und offenen Formeln für $[a,b] = [0,1]$**



7 **Trapetz- und Mittelpunktsregel.**

Bemerkung. Es genügt die Quadraturformeln des Integrationsgebiets $[0,1]$ zu kennen. Die Formeln für alle weiteren Gebiete folgen aus der Transformationsformel für Integrale,

$$\int_a^b f(x)\,\mathrm{d}x \overset{x=a+(b-a)y}{=} \int_0^1 f\left(a+(b-a)y\right)(b-a)\,\mathrm{d}y$$
$$\approx \sum_{i=0}^{n} \underbrace{(b-a)\omega_i}_{\tilde{\omega}_i} f\left(\underbrace{a+(b-a)y}_{\tilde{x}_i}\right). \quad \multimap$$

Probleme. Bei der Verwendung von Newton-Cotes-Formeln können für großes n ($n \geq 8$) *negative* Integrationsgewichte auftreten. Dies kann zu Auslöschung bzw. zu numerischer Instabilität führen.

Außerdem konvergiert das Interpolationspolynom nicht notwendigerweise gegen die approximierte Funktion.

■ Zusammengesetzte Newton-Cotes-Formeln

Die Idee ist, das Integrationsgebiet $[a,b]$ in Teilintervalle $[y_{i-1}, y_i]$, $i = 1,\ldots,m$ mit $a = y_0 < y_1 < \ldots < y_n = b$ zu unterteilen, auf denen das jeweilige Interpo-

lationspolynom die Funktion gut approximiert. Verwende nun für jedes Teilintervall eine Newton-Cotes-Formel.

Bsp *Zusammengesetzte Trapetzregel.* Setze

$$y_i = a + ih, \quad i = 0, \dots, m, \quad h = \frac{b-a}{m}.$$

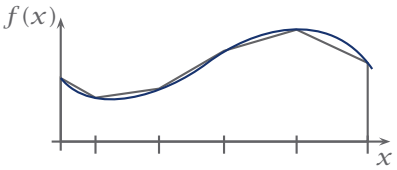
Für das Integral folgt,

$$\begin{aligned}\int_a^b f(x)\,\mathrm{d}x &= \sum_{i=1}^{m} \int_{y_{i-1}}^{y_i} f(x)\,\mathrm{d}x \approx \sum_{i=1}^{m} \frac{y_i - y_{i-1}}{2}\left(f(y_{i-1}) + f(y_i)\right) \\ &= \frac{h}{2} f(a) + h \sum_{i=1}^{m-1} f(a+ih) + \frac{h}{2} f(b).\end{aligned}$$

Die zusammengesetzte Integrationsformel hat also die Form,

$$T_m(f) = h\left[\frac{1}{2}f(a) + \sum_{i=1}^{m-1} f(a+ih) + \frac{1}{2}f(b)\right], \qquad h = \frac{b-a}{m}.$$

Wir sehen sowohl optisch als auch in den Fehlern ein deutlich besseres Konver-



8 **Zusammengesetzte Trapetzregel.**

genzverhalten. ■

Bsp *Zusammengesetzte Simpsonregel.* Setze

$$\begin{aligned}y_i &= a + i2h, \quad i = 0, \dots, \frac{m}{2}, \quad \frac{m}{2} \in \mathbb{N}, \quad h = \frac{b-a}{m}, \\ x_i &= a + ih, \quad i = 0, \dots, m.\end{aligned}$$

$$
\begin{aligned}
s_m(f) &= h\Big[\frac{1}{6}f(x_0) + \frac{2}{3}f(x_1) + \frac{1}{6}f(x_2) + \frac{1}{6}f(x_2) + \frac{2}{3}f(x_3) + \frac{1}{6}f(x_4) \\
&\qquad + \ldots + \frac{1}{6}f(x_{m-2}) + \frac{2}{3}f(x_{m-1}) + \frac{1}{6}f(x_n)\Big] \\
&= \frac{h}{6}\left(f(a) + 4\sum_{i=1}^{m/2} f(x_{2i}-1) + 2\sum_{i=1}^{m/2-1} f(x_{2i}) + f(b)\right).
\end{aligned}
$$

Man kann von der zusammengesetzten Simpsonsformel ein deutlich schnelleres Konvergenzverhalten erwarten als von der zusammengesetzten Trapetzregel. Wir werden dies später bei der Fehleranalyse sehen. ■

■ Approximationseigenschaften

Durch Anwendung von Satz 3.4, der Fehlerabschätzung für Interpolationspolynome, erhalten wir auch eine Fehlerabschätzung für die Quadraturformel:

4.1 **Satz** *Sei* $Q(f) = \sum_{i=0}^{m} \omega_i f(x_i)$ *eine Quadraturformel mit*

$$
Q(p) = \int_a^b p(x)\,\mathrm{d}x, \qquad \text{für alle } p \in \mathcal{P}_m.
$$

Dann gilt für $f \in C^{m+1}([a,b])$

$$
\left|\int_a^b f(x)\,\mathrm{d}x - Q(f)\right| \le \frac{1}{(m+1)!}\left\|f^{(m+1)}\right\|_\infty \int_a^b \prod_{j=0}^{m}(x - x_j)\,\mathrm{d}x. \quad \bowtie
$$

» Sei p das Interpolationspolynom zu

$$
(x_i, f(x_i)), \qquad i = 0, \ldots, m,
$$

d.h. $p \in \mathcal{P}_m$. Für die Quadraturformel gilt nach Vorassetzung,

$$
Q(f) = Q(p) = \int_a^b p(x)\,\mathrm{d}x.
$$

Mit Satz 3.4 folgt somit,

$$\left|\int_a^b f(x)\,\mathrm{d}x - Q(f)\right| = \left|\int_a^b f(x) - p(x)\,\mathrm{d}x\right|$$
$$\leq \int_a^b \frac{1}{(m+1)!}\left|f^{(m+1)}(\xi(x))\right|\left|\prod_{j=0}^{m}(x-x_j)\right|\,\mathrm{d}x$$
$$\leq \frac{1}{(m+1)!}\left\|f^{(m+1)}\right\|_\infty \int_a^b \prod_{j=1}^{m}\left|x-x_j\right|\,\mathrm{d}x. \quad «$$

Bsp *Trapezregel*

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

Hier ist $m = 1$, $x_0 = a$, $x_1 = b$,

$$\int_a^b |(x-x_0)(x-x_1)|\,\mathrm{d}x = \int_a^b |(x-a)(x-b)|\,\mathrm{d}x = \int_a^b (x-a)(b-x)\,\mathrm{d}x$$
$$= \int_0^{b-a}(b-a-x)\,\mathrm{d}x = \left[-\frac{1}{3}x^3 + \frac{b-a}{2}x^2\right]_0^{b-a} = \frac{1}{6}(b-a)^3.$$
$$\Rightarrow \left|\int_a^b f(x)\,\mathrm{d}x - Q(f)\right| \leq \frac{(b-a)^3}{12}\,||f''||_\infty.$$

Zusammengesetzte Trapezregel mit Schrittweite $h = \frac{b-a}{n}$, $x_j = a + jh$.

$$\left|\int_a^b f(x)\,\mathrm{d}x - \Gamma_n(f)\right| = \left|\sum_{i=1}^{n}\int_{x_{i-1}}^{x_i} f(x)\,\mathrm{d}x - Q_i(f)\right|,$$

wobei Q_i die Trapezregel für das entsprechende Intervall $[x_{i-1}, x_i]$ darstellt.

$$\leq \sum_{i=1}^{n}\frac{(x_i - x_{i-1})^3}{12}\,||f''||_\infty \leq n\frac{h^3}{12}\,||f''||_\infty = \frac{b-a}{h}\frac{h^3}{12}\,||f''||_\infty$$
$$= \frac{b-a}{12}h^2\,||f''||_\infty. \quad ■$$

Bsp *Simpsonregel*

$$Q(f) = \frac{b-a}{2}\left(f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right).$$

Hier ist $m = 2$, $x_0 = a$, $x_1 = \frac{a+b}{2}$, $x_2 = b$,

$$\begin{aligned}
\int_a^b \left|\prod_{j=0}^{2}(x-x_j)\right| \mathrm{d}x &= \int_a^b \left|(x-a)\left(x-\frac{a+b}{2}\right)(x-b)\right| \mathrm{d}x \\
&= 2\int_a^{\frac{a+}{2}} \left|(x-a)\left(x-\frac{a+b}{2}\right)(x-b)\right| \mathrm{d}x \\
&= 2\int_0^{\frac{b-a}{2}} \left|x\left(x-\frac{b-a}{2}\right)(x-b-a)\right| \mathrm{d}x \\
&= 2\left[\frac{x^4}{2} - \frac{3}{2}(b-a)\frac{x^3}{3} + \frac{(b-a)^2}{2}\frac{x^2}{2}\right]_0^{\frac{b-a}{2}} \\
&= 2\frac{1}{2^4}\left[\left(\frac{1}{4}-1+1\right)(b+a)^4\right] = \frac{1}{32}(b-a)^4 \\
\Rightarrow \left|\int_a^b f(x)\,\mathrm{d}x - Q(f)\right| &\le \frac{(b-a)^4}{192}\,||f'''||_\infty .
\end{aligned}$$

Zusammengesetze Simpsonregel mit Schrittweite $h = \frac{b-a}{n}$.

$$\left|\int_a^b f(x)\,\mathrm{d}x - S_n(f)\right| \le n\frac{h^4}{192}\,||f'''||_\infty = \frac{b-a}{192}h^3\,||f'''||_\infty .$$

Die Abschätzung ist *nicht optimal*. Numerische Ergebnisse lassen einen Faktor $\sim h^4$ erwarten. Wir können unsere Fehlerabschätzung verbessern, wenn wir berücksichtigen, dass die Simpsonregel auch Polynome vom Grad 3 exakt integriert. ■

4.2 **Definition** *Eine Quadraturformel der Form*

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

heißt symmetrisch genau dann, wenn

$$x_{m-j} - x_j = a + b, \quad \text{und } \omega_{m-j} = \omega_j, \quad j = 0, \dots, m.$$

Spezialfall. $[a, b] = [-1, 1]$. *Dann heißt*

$$Q(f) = \sum_{i=0}^{m} \omega_i f(x_i)$$

symmetrisch genau dann, wenn

$$x_{m-j} = -x_j, \quad \omega_{m-j} = \omega_j. \quad \bowtie$$

Es gilt dann für k ungerade, dass

$$\int_{-1}^{1} x^k \, \mathrm{d}x = 0.$$

$$Q(x \mapsto x^k) = \sum_{i=0}^{m} \omega_i x_i^k = \sum_{i=0}^{\left[\frac{m-1}{2}\right]} \omega_i \underbrace{\left(x_i^k + (-x_i)^k\right)}_{=0} + \begin{cases} 0, & m \text{ ungerade}, \\ \omega_{\frac{m}{2}} \cdot \underbrace{x_{\frac{m}{2}}^k}_{0^k}, & m \text{ gerade}. \end{cases}$$

Korollar *Q integriert Polynome der Art $p(x) = x^k$, k ungerade, exakt (unabhängig vom Grad).* $\bowtie$

4.3 **Definition** *Eine Quadraturformel Q für das Intervall $[a, b]$ heißt exakt auf $\mathcal{P}_k$ genau dann, wenn*

$$Q(p) = \int_a^b p(x) \, \mathrm{d}x, \qquad \forall\ p \in \mathcal{P}_k. \quad \bowtie$$

4.4 **Satz** *Eine symmetrische Quadraturformel, die auf $\mathcal{P}_{2l}$, $l \in \mathbb{N}_0 := \mathbb{N} \cup \{0\}$ exakt ist, ist auch auf $\mathcal{P}_{2l+1}$ exakt.* $\bowtie$

» Sei $Q(f) = \sum_{i=0}^{m} \omega_i f(x_i)$ eine symmetrische Quadraturfromel auf $[a,b]$. Wir betrachten zunächst den Spezialfall

$$q(x) = \left(x - \frac{a+b}{2}\right)^{2l+1},$$

$$\int_a^b q(x)\,\mathrm{d}x = \int_a^b \left(x - \frac{a+b}{2}\right)^{2l+1} \mathrm{d}x = \int_{-\frac{b-a}{2}}^{\frac{b-a}{2}} y^{2l+1}\,\mathrm{d}y = 0.$$

Daher ist

$$Q(q) = \sum_{i=0}^{m} \underbrace{\omega_i}_{\omega_{m-i}} \underbrace{\left(x_i - \frac{a+b}{2}\right)^{2l+1}}_{-\left(x_{m-i} - \frac{a+b}{2}\right)} = 0,$$

aufgrund der Symmetrie.

Für $p \in \mathcal{P}_{2l+1}$ existiert ein $\alpha \in \mathbb{R}$ und $r \in \mathcal{P}_{2l}$ mit $p = \alpha q + r$.

$$\begin{aligned} \Rightarrow Q(p) &= \sum_{i=0}^{m} \omega_i \left(\alpha q + r\right)(x_i) = \alpha \sum_{i=0}^{m} \omega_i q(x_i) + \sum_{i=0}^{m} \omega_i r(x_i) \\ &= \alpha \int_a^b q(x)\,\mathrm{d}x + \int_a^b r(x)\,\mathrm{d}x = \int_a^b p(x)\,\mathrm{d}x. \quad « \end{aligned}$$

Korollar *Eine Newton-Cotes-Formel*

$$Q(f) = \sum_{i=0}^{m} \omega_i f(x_i),$$

der Stufe m ist

(i) symmetrisch,

(ii) exakt auf $\begin{cases} \mathcal{P}_m, & \text{für } m \text{ ungerade,} \\ \mathcal{P}_{m+1}, & \text{für } m \text{ gerade.} \end{cases}$ ⋊

» Der Beweis der Symmetrie sei als Übung überlassen. «

Insbesondere ist die Simpsonformel exakt auf $\mathcal{P}_3$.

4.5 **Fehlerdarstellung von Peano** *Sei* $Q(f) = \sum_{i=0}^{m} \omega_i f(x_i)$ *eine auf* $\mathcal{P}_k$ *exakte Integrationsformel für* $[a,b]$. *Dann gilt für* $f \in C^{k+1}([a,b])$

$$Q(f) - \int_a^b f(x)\,\mathrm{d}x = \int_a^b K(t) f^{(k+1)}(t)\,\mathrm{d}t\,,$$

$$K(t) = \frac{1}{k!}\left[\sum_{i=0}^{m} \omega_i (x_i - t)_+^k - \int_a^b (x-t)_+^k\,\mathrm{d}x\right]$$

$$= \frac{1}{k!}\left[\sum_{i=0}^{m} Q\left(x \mapsto (x-t)_+^k\right) - \int_a^b (x-t)_+^k\,\mathrm{d}x\right] \quad \bowtie$$

» Taylorentwicklung mit Integralrestglied ergibt,

$$f(x) = \sum_{l=0}^{k} \frac{f^{(l)}(a)}{l!}(x-a)^l + \int_a^x \frac{(x-t)^k}{k!} f^{(k+1)}(t)\,\mathrm{d}t$$

$$= \sum_{l=0}^{k} \frac{f^{(l)}(a)}{l!}(x-a)^l + \int_a^b \frac{(x-t)_+^k}{k!} f^{(k+1)}(t)\,\mathrm{d}t\,.$$

Wenden wir darauf die Integrationsformel an,

$$Q(f) = \sum_{l=0}^{k} \frac{f^{(l)}(a)}{l!} \underbrace{\sum_{i=0}^{m} \omega_i (x_i - a)^l}_{Q((x-a)^l)} + \int_a^b \sum_{i=0}^{m} \omega_i (x_i - t)_+^k \frac{f^{(k+1)}(t)}{k!}\,\mathrm{d}t$$

$$= \sum_{l=0}^{k} \frac{f^{(l)}(a)}{l!} \int_a^b (x-a)^l\,\mathrm{d}x + \int_a^b \sum_{i=0}^{m} \omega_i (x_i - t)_+^k \frac{f^{(k+1)}(t)}{k!}\,\mathrm{d}t\,.$$

Das Integral über f können wir auch darstellen als,

$$\int_a^b f(x)\,\mathrm{d}x = \sum_{l=0}^{k} \frac{f^{(l)}(a)}{l!} \int_a^b (x-a)^l\,\mathrm{d}x + \int_a^b \int_a^b \frac{(x-t)_+^k}{k!} f^{(k+1)}(t)\,\mathrm{d}x\,\mathrm{d}t\,.$$

Somit ergibt sich die Behauptung,

$$Q(f) - \int_a^b f(x)\,\mathrm{d}x = \int_a^b \frac{1}{k!}\left[\sum_{i=0}^{m} \omega_i (x_i - t)_+^k - \int_a^b (x-t)_+^k\,\mathrm{d}x\right] f^{(k+1)}(t)\,\mathrm{d}t\,. \quad «$$

Bsp *Trapezregel.*

$$Q(f) = \frac{b-a}{2}\left(f(a)+f(b)\right), \quad k=1$$

$$\Rightarrow K(t) = \frac{b-a}{2}\left[(a-t)_+ + (b-t)_+\right] - \int_a^b (x-t)_+ \,\mathrm{d}x$$

$$\overset{\text{für } t\in(a,b)}{=} \frac{b-a}{2}(b-t) - \int_t^b (x-t)\,\mathrm{d}x = \frac{b-t}{2}\left[b-a-(b-t)\right]$$

$$= \frac{1}{2}(b-t)(t-a).$$

Wegen $K(t) \geq 0$ für $t \in [a,b]$ folgt,

$$\int_a^b K(t)f''(t)\,\mathrm{d}t = f''(\xi)\int_a^b K(t)\,\mathrm{d}t = f''(\xi)\int_0^{b-a} \frac{y(b-a-y)}{2}\,\mathrm{d}y$$

$$= f''(\xi)\,\frac{\frac{1}{2}by^2 - \frac{1}{2}ay^2 - \frac{1}{3}y^3}{2}\Bigg|_0^{b-a}$$

$$= f''(\xi)\,\frac{3b(b-a)^2 - 3a(b-a)^2 - 2(b-a)^3}{12}$$

$$= \frac{f''(\xi)}{12}(b-a)^3$$

Simpsonregel.

$$Q(f) = \frac{b-a}{6}\left(f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right)$$

$$\Rightarrow K(t) = \begin{cases} \frac{(t-a)^3}{72}(a+2b-3t), & a \leq t \leq \frac{a+b}{2}, \\ \frac{(b-t)^3}{72}(3t-2a-b), & \frac{a+b}{2} \leq t \leq b, \end{cases}$$

$$Q(f) - \int_a^b f(x)\,\mathrm{d}x = \frac{(b-a)^5}{2880} f^{(4)}(\xi) \quad \blacksquare$$

Bemerkungen. 1.) Falls Q exakt auf $\mathcal{P}_0$, also

$$\sum_{i=0}^{m} \omega_i = b-a = \int_a^b 1\,\mathrm{d}x,$$

und $\omega_i \geq 0$ für $i = 0, \ldots, m$, dann

$$\begin{aligned} |K(t)| &\leq \frac{1}{k!}\left[\sum_{i=0}^{m} \omega_i (x_i - t)_+^k + \int_a^b (x-t)_+^k \,\mathrm{d}x\right] \\ &\leq \frac{1}{k!}\left[\sum_{i=0}^{m} \omega_i (b-a)^k + \int_a^b (b-a)^k \,\mathrm{d}x\right] \\ &\leq \frac{2}{k!}(b-a)^{k+1} \end{aligned}$$

Einsetzen in die Fehlerabschätzung ergibt,

$$\left| Q(f) - \int_a^b f(x)\,\mathrm{d}x \right| \leq \int_a^b |K(t)| \left| f^{(k+1)}(t) \right| \,\mathrm{d}t \leq \left\| f^{(k+1)} \right\|_\infty \frac{2}{k!}(b-a)^{k+2}.$$

Die Abschätzung ist jedoch nicht so genau, hier erhalten wir für die Konstante $\frac{2}{3}$ und dies ist weit weg von $\frac{1}{2280}$

2.) Falls $K(t)$ *keinen* Vorzeichenwechsel auf $[a,b]$ hat, dann gilt

$$\int_a^b K(t)^{(k+1)}(t)\,\mathrm{d}t = f^{(k+1)}(\xi) \int_a^b K(t)\,\mathrm{d}t$$

mit $\xi \in [a,b]$. ⊸

■ Anwendung auf zusammengesetzte Quadraturformeln

Betrachte das Integrationsintervall $[a,b]$ mit der Unterteilung

$$[y_{j-1}, y_j], \quad j = 1, \ldots, n,$$

mit $y_j = a + jh$ und Schrittweite $h = \frac{b-a}{h}$.

Definition *Eine zusammengesetzte Quadraturformel ist gegeben durch,*

$$Q(f) = \sum_{i}^{n} Q_i(f),$$

wobei die Q_i Quadraturformeln für $[y_{i-1}, y_i]$ sind. ⋊

Voraussetzung. Q_j ist exakt auf $\mathcal{P}_k$.

$$\Rightarrow \left| Q(f) - \int_a^b f(x)\,\mathrm{d}x \right| \leq \sum_{i=1}^{n} \left| Q_i(f) - \int_{y_{i-1}}^{y_i} f(x)\,\mathrm{d}x \right| \leq nCh^{k+2} \left\| f^{(k+1)} \right\|_\infty$$
$$= (b-a) \left\| f^{(k+1)} \right\|_\infty h^{k+1}$$

Dabei hängt C von Q ab, es gilt

$$C \leq \frac{2}{k!}.$$

Die obere Schranke muss nicht gut sein - wie wir bei der Simpsonregel gesehen haben - sie funktioniert dafür aber allgemein.

4-B Gauß-Quadratur

Bisher waren die Stützstellen (x_i) einer Quadraturformel

$$Q(f) = \sum_{i=0}^{m} \omega_i f(x_i)$$

fest vorgegeben.

Frage: Kann man die Genauigkeit einer Quadraturformel verbessern, indem man die Stützstellen x_i geschickt wählt?

Hilfsmittel *Sei $[a,b]$, $a < b$ ein Intervall. Dann wird auf $\mathcal{P}_m$ durch*

$$\langle p, q \rangle = \int_a^b p(x)q(x)\,\mathrm{d}x.$$

ein Skalarprodukt definiert, d.h.

(i) $\langle p+q, r \rangle = \langle p, q \rangle + \langle q, r \rangle, \quad \forall\ p,q,r \in \mathcal{P}_m$
$\langle \alpha p, r \rangle = \alpha \langle p, r \rangle, \quad \forall\ p,r \in \mathcal{P}_m, \alpha \in \mathbb{R},$

(ii) $\langle p, q \rangle = \langle q, p \rangle, \quad \forall\ p,q \in \mathcal{P}_m,$

(iii) $\langle p, p \rangle \geq 0, \quad \forall\ p \in \mathcal{P}_m.$

Die zugehörige Norm ist,

$$||p|| = \sqrt{\langle q, q\rangle} = \left(\int_a^b p(x)q(x)\,\mathrm{d}x\right)^{1/2}.$$

Zwei Polynome p, q heißen orthogonal, falls

$$\langle q, p\rangle = 0. \quad \bowtie$$

Das Skalarprodukt und die Orthogonalitätsbeziehung hängen vom Intervall $[a, b]$ ab, d.h. $p \perp q$ auf $[a, b]$ erzwingt nicht $p \perp q$ auf $[c, d]$.

Im Folgenden sei $[a, b] = [0, 1]$.

4.6 **Definition** *Die durch*

$$p_n(x) = x^n + c_{n-1}x^{n-1} + \ldots + c_0, \quad n = 0, 1, 2, \ldots$$

skalierten orthogonalen Polynome auf $[-1, 1]$ heißen Legendre-Polynome. $\bowtie$

4.7 **Satz** *(i) Die Legendre-Polynome $p_0, p_1, \ldots$ sind eindeutig definiert.*

(ii) $\{p_0, p_1, \ldots, p_n\}$ ist eine Basis von $\mathcal{P}_n$.

(iii) Für $q \in \mathcal{P}_{n-1}$ gilt $p_n \perp q$, d.h.

$$\langle p_n, q\rangle = \sqrt{\int_{-1}^{1} p_n(x)q(x)\,\mathrm{d}x} = 0.$$

(iv) p_n hat n verschiedene Nullstellen $x_1, \ldots, x_n \in (-1, 1)$. $\bowtie$

» "(i),(ii)": *Induktion über n. $n = 0$:* Dann ist $p_0(x) = 1$.

$n - 1 \to n$: $\{p_0, \ldots, p_{n-1}\}$ ist eine Basis von $\mathcal{P}_{n-1}$, d.h. wir können folgenden Ansatz wählen,

$$p_n(x) = x^n + \sum_{j=0}^{n-1} \lambda_j p_j(x).$$

Wir müssen also nur noch zeigen, dass

$$\begin{aligned}
&\langle p_n, p_i\rangle = 0, \qquad \text{für } i = 0,\dots,n-1.\\
\Rightarrow &\langle p_n, p_i\rangle = \langle [\cdot]^n, p_i\rangle + \sum_{j=0}^{n-1} \lambda_j \left\langle p_j, p_i\right\rangle = \langle [\cdot]^n, p_i\rangle + \lambda_i \left|\left|p_i\right|\right|^2\\
\Rightarrow &\lambda_i = -\frac{\langle [\cdot]^n, p_i\rangle}{\left|\left|p_i\right|\right|^2} = -\frac{\int_{-1}^{1} x^n p_i(x)\,\mathrm{d}x}{\int_{-1}^{1} \left|p_i(x)\right|^2\,\mathrm{d}x}.
\end{aligned}$$

Wir erhalten so eine eindeutige Darstellung der λ_i. Der Beweis ist konstruktiv, d.h. wir können mit dieser Vorschrift beliebige p_k berechnen.

"(iii)": Sei $q \in \mathcal{P}_{n-1}$, dann können wir q darstellen als,

$$q(x) = \sum_{j=0}^{n-1} \alpha_j p_j(x).$$

Bilden wir jetzt das Skalarprodukt,

$$\langle p_n, q\rangle = \sum_{j=0}^{n-1} \alpha_j \underbrace{\left\langle p_n, p_j\right\rangle}_{=0} = 0.$$

"(iv)": Angenommen $z \in \mathbb{R} \setminus (-1,1)$ sei Nullstelle von p_n, d.h. wir können p_n faktorisieren,

$$\begin{aligned}
&q(x) = \frac{p_n(x)}{x-z}, \quad q \in \mathcal{P}_{n-1},\\
\Rightarrow &0 = \langle q, p_n\rangle = \int_{-1}^{1} \frac{p_n(x)^2}{x-z}\,\mathrm{d}x,
\end{aligned}$$

nun ist der Integrand ≥ 0 für $z \leq -1$ und $\forall\, x \in (-1,1)$ und ≤ 0 für $z \geq 1$ und $\forall\, x \in (-1,1)$, also muss bereits $p_n(x) = 0$ für alle $x \in (-1,1)$ sein. $\lightning$

Also kann z keine einfache Nullstelle von p_n sein.

Angenommen $z \in (-1,1)$ sei 2- bzw. mehr-fache Nullstelle von p_n, so können wir erneut faktorisieren,

$$q(x) = \frac{p_(x)}{(x-z)^2}, \quad q \in \mathcal{P}_{n-2}.$$

Wir können dieselbe Argumentation wie eben verwenden und erhalten $p_n(x) = 0$ für $\forall\, x \in (-1,1)$.

Angenommen $z \in \mathbb{C}\setminus\mathbb{R}$ sei Nullstelle von p_n, d.h. $\overline{z} \neq z$ ist ebenfalls Nullstelle. Faktorisiere

$$q(x) = \frac{p_n(x)}{(x-z)(x-\overline{z})} = \frac{p_n(x)}{|x-z|^2}, \quad q \in \mathcal{P}_{n-2}.$$

Analog erhalten wir

$$0 = \langle q, p_n\rangle = \int_{-1}^{1} \frac{p_n(x)^2}{|x-z|^2}\,\mathrm{d}x.$$

Der Integrand ist ≥ 0, d.h. $p_n(x)$ muss Null sein $\forall\, x \in (-1,1)$. ↯

Da $\mathbb{C}$ algebraisch abgeschlossen über $\mathbb{R}$, hat p_n nach dem Hauptsatz der Algebra genau n verschiedene Nullstellen in $\mathbb{C}$. Wir haben gezeigt, dass diese nur im Intervall $(-1,1) \subseteq \mathbb{R}$ liegen können. «

Bemerkungen. 1) Der Satz gilt auch für entsprechende orthogonale Polynome auf einem allgemeinen Intervall.

2) Man kann zeigen (für $[a,b] = [-1,1]$)

$$p_n(x) = \frac{n!}{(2n)!}\frac{\mathrm{d}^n}{\mathrm{d}x^n}\left[(x^2-1)^n\right].$$

■ Gauß'sche Quadraturformeln

Wir wählen

- als Stützstellen die Nullstellen $x_1, \ldots, x_m$ des Legendre-Polynoms p_m.
- als Integrationsgewichte $\omega_1, \ldots, \omega_m$ mit

$$\int_{-1}^{1} q(x)\,\mathrm{d}x = \sum_{i=1}^{m} \omega_i q(x_i), \qquad \forall\, q \in \mathcal{P}_{m-1}.$$

d.h. wir können zur Berechnung der Integrationsgewichte die Lagrangedarstellung wählen,

$$\omega_i = \int_{-1}^{1} \prod_{\substack{j=1 \\ j\neq i}}^{m} \frac{x - x_j}{x_i - x_j} \, \mathrm{d}x.$$

Die Gauß'sche Quadraturformel der Stufe m ist gegeben durch

$$Q(f) = \sum_{i=1}^{m} \omega_i f(x_i),$$

für $\int_{-1}^{1} f(x) \, \mathrm{d}x$.

4.8 **Satz** *Die Gauß'sche Quadraturformel der Stufe m ist exakt auf $\mathcal{P}_{2m-1}$.* ⋈

» Sei $p \in \mathcal{P}_{2m-1}$. Dann gilt (Polynomdivision)

$$p(x) = p_m(x)q(x) + r(x), \quad \text{mit } q, r \in \mathcal{P}_{m-1}.$$

Es folgt,

$$\int_{-1}^{1} p(x) \, \mathrm{d}x = \int_{-1}^{1} p_m(x)q(x) \, \mathrm{d}x + \int_{-1}^{1} r(x) \, \mathrm{d}x = \underbrace{\langle p_m, q \rangle}_{=0} + \int_{-1}^{1} r(x) \, \mathrm{d}x.$$

Wir müssen noch nachprüfen, ob die Quadraturformel die selbe Eigenschaft hat.

$$Q(f) = \sum_{j=1}^{m} \omega_j p(x_j) = \sum_{j=1}^{m} \omega_j \underbrace{p_m(x_j)q(x_j)}_{=0} + \sum_{j=1}^{m} \omega_j r(x_j) = \sum_{j=1}^{m} \omega_j r(x_j)$$

$$= \int_{-1}^{1} r(x) \, \mathrm{d}x. \quad «$$

Bsp $n = 2$. Die Legendrepolynome sind gegeben durch,

$$p_0(x) = 1$$

$$p_1(x) = x - \frac{\int_{-1}^{1} 1s \,\mathrm{d}s}{\int_{-1}^{1} 1^2 \,\mathrm{d}s} = x - 0 = x - \frac{0}{2} = x$$

$$p_2(x) = x^2 - \frac{\int_{-1}^{1} s^2 \,\mathrm{d}s}{\int_{-1}^{1} 1^2 \,\mathrm{d}s} - \frac{\int_{-1}^{1} ss^2 \,\mathrm{d}s}{\int_{-1}^{1} s^2 \,\mathrm{d}s} = x^2 - \frac{\frac{2}{3}}{2} - 0 = x^2 - \frac{1}{3}.$$

Integrationsgewichte

$$\omega_1 = \int_{-1}^{1} \frac{x - \frac{1}{\sqrt{3}}}{-\frac{1}{\sqrt{3}} - \frac{1}{\sqrt{3}}} \,\mathrm{d}x = -\frac{1}{2} \int_{-1}^{1} \sqrt{3}x - 1 \,\mathrm{d}x = 1.$$

$$\omega_2 = \ldots = 1.$$

D.h. die Quadraturformel ist gegeben durch,

$$Q(f) = f\left(-\frac{1}{\sqrt{3}}\right) + f\left(\frac{1}{\sqrt{3}}\right). \quad \blacksquare$$

m	x_i	ω_i
2	$-\frac{1}{\sqrt{3}}, \frac{1}{\sqrt{3}}$	$1, 1$
3	$-\sqrt{\frac{3}{5}}, 0, \sqrt{\frac{3}{5}}$	$\frac{5}{9}, \frac{8}{9}, \frac{5}{9}$
4	$x_4 = -x_1 = 0.861136311,$	$\omega_1 = \omega_4 = 0.3478548451$
	$x_3 = -x_2 = 0.339981$	$\omega_2 = \omega_3 = 0.6521451549$

■ Allgemeine Integrationsgebiete

Wir führen für $x \in [-1, 1]$ die Transformation

$$x \mapsto \frac{a+b}{2} + x\frac{b-a}{2} \in [a, b]$$

durch und erhalten somit für die Quadraturformel.

$$\Rightarrow \int_a^b f(x)\,\mathrm{d}x = \frac{b-a}{2} \int_{-1}^{1} f\left(\frac{a+b}{2} + x\frac{b-a}{2}\right)\mathrm{d}x$$

$$\approx \sum_{i=1}^{m} \underbrace{\frac{b-a}{2}\omega_i}_{\tilde{\omega}_i} f\left(\underbrace{\frac{a+b}{2} + \frac{b-a}{2}x}_{\hat{=}\tilde{x}_i}\right) = \sum_{i=1}^{m} \tilde{\omega}_i f(\tilde{x}_i).$$

Zur Fehlerabschätzung verwenden wir die Formel von Peano (Satz 4.5) und die anschließende Bemerkung. Für die Gaußsche Integrationsformel mit m Stützstellen gilt,

$$\left| Q(f) - \int_a^b f(x)\,\mathrm{d}x \right| \leq c_m\,|b-a|^{2m+1} \left\| f^{(2m)} \right\|_\infty,$$

für $f \in C^{2m}([a,b])$. Dabei ist $c_m \leq \frac{2}{(2m)!}$. Der Exaktheitsgrad ist $k = 2m-1$ ($2m+1 \mathrel{\hat{=}} k+2$, $2m \mathrel{\hat{=}} k+1$).

■ Zusammengesetze Gauß-Quadratur

Sei $a < b$, $y_j = a + jh$, $h = \frac{b-a}{n}$ für $j = 0,\ldots,n$. Sei Q_j die Quadraturformel mit m Sützstellen für das Intervall $[y_{j-1}, y_j]$, $j = 1,\ldots,n$.

$$Q(f) = \sum_{j=1}^{n} Q_j(f).$$

Dann gilt für $f \in C^{2m}([a,b])$

$$\left| Q(f) - \int_a^b f(x)\,\mathrm{d}x \right| \leq c_m (b-a) h^{2m} \left\| f^{(2m)} \right\|_\infty.$$

Die Gauß-Quadraturformel ist also substantiell “besser” als die Newton-Cotes-Formeln.

Ein weiterer Vorteil der Gauß-Quadratur ist, dass die Integrationsgewichte alle positiv sind. D.h. Fehler durch Auschlöschung werden vermieden.

Fazit. Wenn die Stützstellen frei wählbar sind, ist die Gauß-Quadratur anwendbar und daher den anderen Quadraturformeln vorzuziehen. ⊸

5 Nichtlineare Gleichungssysteme

Ziel dieses Kapitels ist die Lösung von nichtlinearen Gleichungen und nichtlinearen Gleichungssystem. Lineare Gleichungssysteme erhält man in der Regel nur, wenn die Abhängigkeiten linear sind. Bei nichtlinearen Abhängigkeiten z.b. Verformung erhält man auch nichtlineare Gleichungen. Wie wir bereits gesehen haben, können dies abhängig vom Problem sehr viele Gleichungen werden. Dabei wollen wir im Folgenden zwischen nichtlinearen Gleichungen

$$f(x) = 0, \qquad f : D \to \mathbb{R}, \quad D \subseteq \mathbb{R}$$

und Gleichungssystemen

$$f(x) = 0, \qquad f : D \to \mathbb{R}^n, \quad D \subseteq \mathbb{R}^n$$

unterscheiden.

Bei linearen Gleichungen konnten wir - sofern sie lösbar waren - stets eine geschlossene Form für die Lösung angeben. Im Falle der nichtlinearen Gleichungen können wir jedoch nicht mehr davon ausgehen. Darüber hinaus lässt sich in vielen Fällen die Lösung nicht exakt berechnen, sondern lediglich durch numerische Verfahren approximieren. In diesem Fall sind Fehlerabschätzungen sehr wichtig, die angeben wie groß der "Abstand" von der exakten Lösung nach n Iterationen ist.

Im Folgenden wollen wir einige gänge Verfahren zum Lösen von nichlinearen Gleichungen und Gleichungssystem untersuchen. Es gibt grob gesprochen zwei unterschiedliche Typen. Der erste Typ von Verfahren garantiert Konvergenz, die Konvergenzgeschwindigkeit ist jedoch klein. Der zweite Typ hat eine viel höhere Konvergenzgeschwindigkeit, die Konvergenz hängt dabei aber entscheidend vom Startwert - also der Wert von dem das Verfahren als vermutete Lösung ausgeht - ab. Diese Verfahren sind nur "gut", wenn der Startwert bereits nahe bei der Lösung liegt.

Bsp (i) Berechnung von $\sqrt{a}$.

$$x^2 = a, \quad \text{d.h. } f(x) = x^2 - a; \quad D = [0, +\infty), \quad a > 0.$$

(ii) System aus zwei Gleichungen.

$$x^2 - e^{-xy} = 1,$$
$$xy + \sin(xy) = 0,$$

d.h.

$$f\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x^2 - e^{-xy} - 1 \\ xy + \sin(xy) \end{pmatrix}, \quad f : \mathbb{R}^2 \to \mathbb{R}^2. \quad \blacksquare$$

5-A Verfahren für skalare Gleichungen

Wir betrachten zunächst skalare nichtlineare Gleichungen,

$$f(x) = 0, \qquad f : D \to \mathbb{R}, \quad D \subseteq \mathbb{R}. \tag{1}$$

■ Das Bisektionsverfahren

Damit das Verfahren überhaupt sinnvoll angewendet werden kann, wollen wir davon ausgehen, dass $f : [a,b] \to \mathbb{R}$ stetig und $f(a) < 0$, $f(b) > 0$ oder $f(a) > 0$, $f(b) < 0$ (d.h. $f(a)f(b) < 0$). Der Zwischenwertsatz garantiert nun

$$\exists\, x \in (a,b) \text{ mit } f(x) = 0.$$

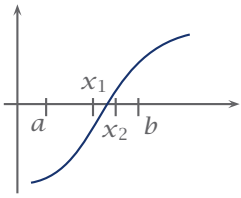
Um dieses x zu approximieren, konstruieren wir eine Folge von Intervallen $[a_n, b_n]$ mit

$$f(a_n)f(b_n) < 0, \quad d.h.\ \exists\, x \in [a_n, b_n] : f(x) = 0.$$

Start: $[a_0, b_0] = [a, b]$.
Schritt $n \to n+1$: Setze $x_n = \frac{1}{2}(a_n + b_n)$

$$\begin{aligned} f(a_n)f(x_n) < 0 \quad &\Rightarrow [a_{n+1}, b_{n+1}] = [a_n, x_n], \\ f(a_n)f(x_n) = 0 \quad &\Rightarrow x_n \text{ ist Lösung}, \\ \text{sonst} \quad &\Rightarrow [a_{n+1}, b_{n+1}] = [x_n, b_n] \end{aligned}$$

Der Fehler fällt nicht monoton, da man für x_{n+1} den Intervallmittelpunkt wählt.



9 **Bisektionsverfahren.**

Approximationseigenschaften. Es gilt

$$|b_{n+1} - a_{n+1}| = \frac{1}{2}|b_n - a_n|$$
$$\Rightarrow |b_n - a_n| = \frac{1}{2^n}|b_0 - a_0|.$$

Für $x_n = \frac{1}{2}(b_n - a_n)$ und jede Lösung $x^* \in [a_n, b_n]$ von $f(x^*) = 0$ gilt

$$|x_n - x^*| \leq \frac{1}{2^{n+1}}|b_0 - a_0|.$$

Das Bisektionsverfahren konvergiert sicher gegen eine Lösung. Es kann jedoch schwierig werden, bei mehreren Lösungen alle zu bekommen.

■ Das Sekantenverfahren

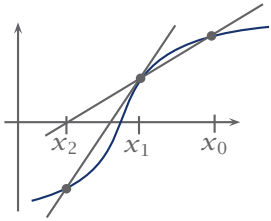
Konstruktion einer Folge $(x_n)_{n\geq 0}$ von Näherungslösungen aus zwei verschiedenen Startwerten $x_0 \neq x_1$.

Schritt $n \to n+1$. Bilde die Sekante $s = s(x)$ durch $(x_{n-1}, f(x_{n-1}))$ und $(x_n, f(x_n))$ und löse $s(x) = 0 \Rightarrow x_{n+1}$.

Darstellung von $s(x)$:

$$s(x) = f(x_n - 1)\frac{x - x_n}{x_{n-1} - x_n} + f(x_n)\frac{x - x_{n-1}}{x_n - x_{n-1}} \stackrel{!}{=} 0,$$
$$\Rightarrow x_{n+1}\frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}} + \frac{f(x_{n-1})x_n - f(x_n)x_{n-1}}{x_n - x_{n-1}} \stackrel{!}{=} 0$$
$$\Rightarrow x_{n+1} = \frac{f(x_n)x_{n-1} - f(x_{n-1})x_n}{f(x_n) - f(x_{n-1})} \tag{2}$$

5.1 **Konvergenz des Sekantenverfahrens** *Sei $f \in C^2([a,b])$, $x^* \in (a,b)$ Lösung von $f(x) = 0$ und $f'(x^*), f''(x^*) \neq 0$.*



10 **Sekantenverfahren.**

Dann existiert ein $\delta > 0$, so dass das Sekantenverfahren für alle Startwerte $x_0,\ x_1 \in U_\delta(x^*)$ konvergiert und es gilt

$$|x_{n+1} - x^*| \le C\,|x_n - x^*|^p\,, \qquad \text{für } n \ge 1$$

mit $p = \frac{1}{2}(1+\sqrt{5})$ und $C > 0$. ⋈

» 1) Wir zeigen zunächst, aus $x_n \neq x_{n-1}$ und $f(x_n) \neq 0$ folgt $x_{n+1} \neq x_n$.

Angenommen $x_{n+1} = x_n$. Dann gilt:

$$\begin{aligned} & x_{n+1}\,(f(x_n) - f(x_{n-1})) = f(x_n)x_{n-1} - f(x_{n-1})x_n \\ \Rightarrow & x_n f(x_n) = f(x_n)x_{n-1}.\ \lightning. \end{aligned}$$

Folgerung: Falls $x_0 \neq x_1$, dann gilt $x_n \neq x_{n-1}$ so lange x_n wohldefiniert ist.

2) *Konvergenz von (x_n).* Aus $f \in C^1([a,b])$ und $f'(x^*) \neq 0$ folgt,

$$\exists\ \varepsilon > 0\ \forall\ x \in U_\varepsilon(x^*) : |f'(x)| \ge \frac{1}{2}\,|f'(x^*)| > 0.$$

D.h. f ist in $U_\varepsilon(x^*)$ streng monoton und daher ist für $x_n, x_{n-1} \in U_\varepsilon(x^*)$, x_{n+1} wohldefiniert, denn $f(x_n) \neq f(x_{n-1})$.

Sei $e_n := x_n - x^*$. Aus (2) folgt,

$$\begin{aligned} e_{n+1} &= \frac{f(x_n)e_{n-1} - f(x_{n-1})e_n}{f(x_n) - f(x_{n-1})} + x^*\frac{f(x_n) - f(x_{n-1})}{f(x_n) - f(x_{n-1})} - x^* \\ &= \frac{f(x_n)e_{n-1} - f(x_{n-1})e_n}{f(x_n) - f(x_{n-1})} \end{aligned}$$

$$\Rightarrow \frac{e_{n+1}}{e_n e_{n-1}} = \frac{1}{f(x_n) - f(x_{n-1})} \left[\frac{f(x_n)}{x_n - x^*} - \frac{f(x_{n-1})}{x_{n-1} - x^*} \right]$$
$$= \frac{g(x_n) - g(x_{n-1})}{f(x_n) - f(x_{n-1})}$$

mit $g(x) := \frac{f(x)}{x-x^*} = \frac{f(x)-f(x^*)}{x-x^*}$.

Anwendung des Mittelwertsatzes auf die Funktion,

$$h(x) := \big(g(x_n) - g(x_{n-1}) f(x) - (f(x_n) - f(x_{n-1}))g(x)\big)$$

ergibt,

$$\begin{aligned} &h(x_n) - h(x_{n-1}) = h'(\xi_n)(x_n - x_{n-1}) \\ &= (g(x_n) - g(x_{n-1}))f'(\xi_n) - (f(x_n) - f(x_{n-1}))g'(\xi_n) \\ &= -g(x_{n-1})f(x_n) + f(x_{n-1})g(x_n) - \big(g(x_n)f(x_{n-1}) - f(x_n)g(x_{n-1})\big) \\ &= 0. \end{aligned}$$

Wir erhalten somit die Gleichung

$$\frac{g(x_n) - g(x_{n-1})}{f(x_n) - f(x_{n-1})} = \frac{g'(\xi_n)}{f'(\xi_n)}.$$

Mit $g'(x) = \frac{f'(x)(x-x^*)-f(x)}{(x-x^*)^2}$ und Taylorentwicklung von f in x,

$$0 = f(x^*) = f(x) + f'(x)(x^* - x) + \frac{1}{2} f''(\eta)(x^* - x)^2,$$

η zwischen x und x^*, folgt

$$\begin{aligned} g'(x) &= \frac{f'(x - x^*) - \left(-f'(x)(x^* - x) - \frac{1}{2} f''(\eta)(x^* - x)^2\right)}{(x - x^*)^2} \\ &= \frac{1}{2} f''(\eta) \\ \Rightarrow g'(\xi_n) &= \frac{1}{2} f''(\eta_n). \\ \Rightarrow \frac{e_{n+1}}{e_n e_{n-1}} &= \frac{1}{2} \frac{f''(\eta_n)}{f'(\xi_n)} \end{aligned} \tag{3}$$

mit ξ_n, η_n zwischen x_{n-1} und x_n. Sei

$$C := \frac{\sup_{\eta \in U_\varepsilon(x^*)} |f''(\eta)|}{f'(x^*)}$$
$$\Rightarrow \frac{e_{n+1}}{e_n e_{n-1}} \leq C.$$

Für $\delta = \min\left\{\varepsilon, \frac{1}{2C}\right\}$ gilt

$$|e_{n-1}| \leq \delta \Rightarrow |e_{n+1}| \leq C\,|e_n|\,|e_{n-1}| \leq \frac{1}{2}\,|e_n|\,.$$

Aus $|e_0|, |e_1| < \delta$, d.h. $x_0, x_1 \in U_\delta(x^*)$, folgt (mit Induktion)

$$|e_{n-1}| \leq \frac{1}{2^{n-1}}\,|e_1|\,.$$
$$\Rightarrow e_n \overset{n\to\infty}{\to} 0 \Rightarrow x_n \overset{n\to\infty}{\to} x^*.$$

3) *Konvergenzgeschwindigkeit.* Setze $y_n := \frac{|e_n|}{|e_{n-1}|^p}$, dann folgt mit (3),

$$y_{n+1} = \alpha_n \frac{|e_n|\,|e_{n-1}|}{|e_n|^p},$$

mit $\alpha_n = \frac{1}{2}\left|\frac{f''(\eta_n)}{f'(\xi_n)}\right|$.

$$\Rightarrow y_{n+1} = \alpha_n \left(\frac{|e_n|}{|e_{n-1}|^p}\right)^{-(p-1)},$$

falls $p(p-1) = 1$. D.h. $p^2 - p - 1 = 0 \Leftrightarrow p_{1,2} = \frac{1}{2} \pm \sqrt{\frac{1}{4} + 1} = \frac{1}{2} \pm \frac{\sqrt{5}}{2}$.
Setze $p = \frac{1+\sqrt{5}}{2}$, dann

$$y_{n+1} = \alpha_n y_n^{-(p-1)}.$$

Für δ klein genug gilt wegen $f''(x^*) \neq 0$:

$$\frac{1}{c_0} \leq \alpha_n \leq c_0, \qquad \text{mit } c_0 > 0,$$

falls $\eta_n, \xi_n \in U_\delta(x^*)$. Für $y_n = \log y_n$ folgt,

$$y_{n+1} = \log(\alpha_n) + \underbrace{(p-1)}_{=\frac{1}{p}}\, y_n$$

Mit Induktion folgt,

$$\begin{aligned}
y_2 &= \log \alpha_1 - \frac{1}{p} y_1, \\
y_3 &= \log \alpha_2 - \frac{1}{p} \left(\log \alpha_1 - \frac{1}{p} y_1 \right) \\
&\vdots \\
y_{n+1} &= \sum_{j=1}^{n} \left(-\frac{1}{p} \right)^{n-j} \log \alpha_j + \left(-\frac{1}{p} \right)^{n} y_1
\end{aligned}$$

Das gilt für $|e_0|, |e_1| \leq \delta$. Wegen $\left| \log \alpha_j \right| \leq \left| \log c_0 \right| = a$ folgt,

$$\begin{aligned}
&\left| y_{n+1} \right| \leq \sum_{j=1}^{n} \left(\frac{1}{p} \right)^{n-j} a + y_1 \leq \frac{a}{1 - \frac{1}{p}} + y_1 =: c_1 \\
&\Rightarrow -c_1 \leq \log y_{n+1} \leq c_1 \\
\Rightarrow &\frac{1}{e^{c_1}} \leq y_{n+1} = \frac{|e_{n+1}|}{|e_n|} \leq e^{c_1} \\
\Rightarrow &\left| x_{n+1} - x^* \right| \leq e^{c_1} \left| x_n - x^* \right|^p . \quad «
\end{aligned}$$

Problem. Das Verfahren konvergiert nur, wenn der Startwert sich bereits nahe der Nullstelle befindet. Man muss also erst einmal einen geeigneten Startwert "raten". ⊸

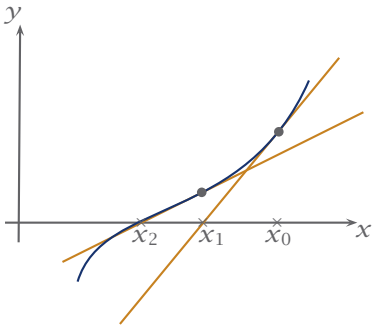
■ Das Newton-Verfahren

Das Newton-Verfahren liefert eine Folge von Näherungslösungen $(x_n)_{n \geq 0}$.

Startwert sei x_0. Im Schritt $n \mapsto n+1$ konstruiere die Tangente des Graphen $\{(x, f(x)) \in D\}$ in $(x_n, f(x_n))$,

Tangente $\{(x, t(x)) : x \in D\}$ und löse anschließend $t(x_{n+1}) = 0$. Es gilt

$$\begin{aligned}
t(x) &= f(x_n) + f'(x_n)(x - x_n) \stackrel{!}{=} 0 \\
&\Rightarrow x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \qquad (5)
\end{aligned}$$



11 **Newtonverfahren.**

Bsp $x^2 = a$, $f(x) = x^2 - a \overset{!}{=} 0$. Es folgt,

$$x_{n+1} = x_n - \frac{x_n^2 - a}{2x_n} = \frac{1}{2}\left(x_n + \frac{a}{x_n}\right).$$

Dieses Iterationsverfahren zur Berechnung von $\sqrt{a}$ heißt Heronsches Verfahren. Es ist seit bereits 2000 Jahren bekannt. ■

5.2 **Konvergenz des Newton-Verfahrens** *Sei $f \in C^2((a,b) \to \mathbb{R})$, $x^* \in (a,b)$, $f(x^*) = 0$, $f'(x^*) \neq 0$. Dann existiert ein $\delta > 0$ so, dass das Newton-Verfahren für alle Startwerte $x_0 \in U_\delta(x^*)$ konvergiert und es gilt,*

$$|x_{n+1} - x^*| \leq C\,|x_n - x^*|^2, \quad \text{mit } C > 0. \quad \bowtie$$

» Sei $\varepsilon > 0$ so, dass $|f'(x)| \geq \frac{1}{2}|f'(x^*)|$ für alle $x \in U_\varepsilon(x^*)$. Sei $|x_n - x^*| \leq \varepsilon$. Dann gilt

$$x_{n+1} - x^* = x_n - x^* - \frac{f(x_n) - f(x^*)}{f'(x_n)} = \left(1 - \frac{f'(\xi_n)}{f'(x_n)}\right)(x_n - x^*)$$

mit ξ_n zwischen x_n und x^*. Daraus folgt,

$$x_{n+1} - x^* = \frac{f'(x_n) - f'(\xi_n)}{f'(x_n)}(x_n - x^*) = \frac{f''(\eta_n)}{f'(x_n)}(x_n - \xi_n)(x_n - x^*).$$

mit η_n zwischen x_n und ξ_n. Sei

$$C := 2 \sup_{x \in U_\varepsilon(x^*)} \frac{|f''(x)|}{f'(x^*)},$$

dann gilt

$$|x_{n+1} - x^*| \leq C\,|x_n - x^*|^2.$$

Sei

$$\delta := \min\left\{\varepsilon, \frac{1}{2C}\right\}$$

dann gilt

$$|x_n - x^*| \leq \delta \Rightarrow |x_{n+1} - x^*| \leq \frac{1}{2}\,|x_n - x^*|.$$

Für $|x_0 - x^*| \leq \delta$ folgt (Induktion),

$$|x_n - x^*| \leq \left(\frac{1}{2}\right)^n |x_0 - x^*| \overset{n\to\infty}{\to} 0. \quad «$$

5-B Nichtlineare Gleichungssysteme

■ Fixpunktiteration

Betrachte das nichtlineare Gleichungssystem

$$f(x) = 0, \quad f : D \to \mathbb{R}^n,\ D \subseteq \mathbb{R}^m.$$

Eine Umformulierung in eine äquivalente Fixpunktgleichung ergibt,

$$f(x) + x = x$$

oder

$$x - f(x) = x$$

oder

$$x - \omega f(x) = x, \quad \text{mit } \omega \in \mathbb{R}, \omega \neq 0.$$

5.3 **Banachscher Fixpunktsatz** *Sei $D \subseteq \mathbb{R}^n$ abgeschlossen, $\phi : D \to D$ eine Kontraktion, d.h.*

$$||\phi(x) - \phi(y)|| \leq L\,||x - y||, \qquad \forall\ x, y \in \mathrm{D},$$

mit $L < 1$. Dann gilt

(i) Die Fixpunktgleichung $x = \phi(x)$ hat genau eine Lösung $x^* \in D$.

(ii) Die Fixpunktiteration ist gegeben durch

$$x^{n+1} = \phi(x^n)$$

und konvergiert für jeden Startwert $x^0 \in D$ gegen x^*. Es gelten die Fehlerabschätzungen,

$$||x^n - x^*|| \leq \frac{L^n}{1-L} \left|\left|x^0 - x^*\right|\right|, \qquad \textit{a priori}, \tag{6}$$

$$||x^n - x^*|| \leq \frac{L}{1-L} \left|\left|x^n - x^{n-1}\right|\right|, \qquad \textit{a posteriori}. \quad \bowtie \tag{7}$$

» *Existenz.* Sei $x^0 \in D$, $x^{n+1} = \phi(x^n)$ für $n = 0, 1, 2, \ldots$. Es folgt,

$$\left|\left|x^{n+1} - x^n\right|\right| = \left|\left|\phi(x^n) - \phi(x^{n-1})\right|\right| \leq L \left|\left|x^n - x^{n-1}\right|\right| \leq L^n \left|\left|x^1 - x^0\right|\right|.$$

Wir zeigen, dass die Folge (x_n) Cauchy ist,

$$\begin{aligned}\left|\left|x^{n+l} - x^n\right|\right| &\leq \sum_{j=0}^{l-1} \left|\left|x^{n+j+1} - x^{n+j}\right|\right| \leq \sum_{j=0}^{l-1} L^{n+j} \left|\left|x^1 - x^0\right|\right| \\ &\leq L^n \sum_{j=0}^{l-1} L^j \left|\left|x^1 - x^0\right|\right| \leq \frac{L^n}{1-L} \left|\left|x^1 - x^0\right|\right|,\end{aligned}$$

d.h. (x_n) konvergiert gegen $x^* = \lim_{n\to\infty} x^n$. Da D abgeschlossen liegt $x^* \in D$.
Betrachte nun

$$x^{n+1} = \phi(x^n) \Rightarrow x^* \phi(x^*).$$

Eindeutigkeit. Seien x^* und y^* zwei Fixpunkte. Dann gilt

$$\begin{aligned}&||x^* - y^*|| = ||\phi(x^*) - \phi(y^*)|| \leq L\,||x^* - y^*|| \\ &\Rightarrow (1-L)\,||x^* - y^*|| \leq 0.\end{aligned}$$

Nun ist $1 - L > 0$ also $||x^* - y^*|| = 0$ und daher $x^* = y^*$.
Fehlerabschätzungen. Betrachte

$$\lim_{l\to\infty} \left|\left|x^{n+l} - x^n\right|\right| \leq \frac{L^n}{1-L} \left|\left|x^0 - x^1\right|\right| \Rightarrow (6)$$

Weiter folgt,

$$\left\|x^{n+l}-x^*\right\| \le \sum_{j=0}^{l-1}\left\|x^{n+j+1}-x^{n+j}\right\| \le \sum_{j=0}^{l-1} L^{j+1}\left\|x^n-x^{n-1}\right\|$$
$$\le L\sum_{j=0}^{l-1} L^j\left\|x^n-x^{n-1}\right\| \le \frac{L}{1-L}\left\|x^n-x^{n-1}\right\|,$$

mit dem Grenzwert $l\to\infty$ folgt (7). «

Anwendung auf ein nichtlineares Gleichungssystem,

$$f(x)=0, \quad f: D\to\mathbb{R}^n, \quad D\subseteq\mathbb{R}^n. \tag{8}$$

Annahmen an f. (i) f ist lipschitz stetig, d.h.

$$||f(x)-f(y)|| \le L\,||x-y||, \quad \forall\, x,y\in D,\ L\in\mathbb{R}.$$

$L<1$ ist hier *nicht* verlangt.

(ii) f ist strikt monoton, d.h.

$$(f(x)-f(y))(x-y) \ge \gamma\,||x-y||^2, \quad \forall\, x,y\in D,$$

mit $\gamma>0$.

Lemma *Sei $f\in C^1(D)$ und D offen. Dann ist f genau dann strikt monoton, wenn $\mathrm{D}f=\left(\frac{\partial f_i}{\partial x_j}\right)_{i,j=1}^n$ gleichmäßig positiv definit ist, d.h.*

$$y^\top \mathrm{D}f(x)y \ge \gamma\,||y||^2, \qquad \forall\, y\in\mathbb{R}^n\ \forall\, x\in D$$

mit γ unabhängig von x. ⋊

» "⇒": Sei also f strikt monoton. Dann gilt

$$y^\top \mathrm{D}f(x)y = \lim_{h\to 0} y^\top \frac{1}{h}\left(f(x+hy)-f(y)\right)$$
$$= \lim_{h\to 0}\frac{1}{h^2}\left(f(x+hy)-f(x)\right)(x+hy-x)$$
$$\ge \lim_{h\to 0}\frac{\gamma}{h^2}\,||x+hy-x||^2 = \gamma\,||y||^2$$

"⇐":

$$(f(x)-f(y))(x-y) = \int_0^1 \mathrm{D}f(y+t(x-y))(x-y)(x-y)\,\mathrm{d}t$$
$$\geq \int_0^1 \gamma \|x-y\|^2 \,\mathrm{d}t = \gamma \|x-y\|^2 . \quad «$$

Sei f Lipschitz-stetig und strikt monoton. Transformiere das Gleichungssystem

$$f(x) = 0$$

in die äquivalente Fixpunktgleichung

$$x = x - \omega f(x) = \phi(x),$$

mit einem Relaxationsparameter $\omega > 0$.

Nachprüfen der Kontraktionseigenschaft:

$$\begin{aligned}\|\phi(x)-\phi(y)\|^2 &= \|x-y-\omega(f(x)-f(y))\|^2 \\ &= \|x-y\|^2 - 2(x-y)\omega(f(x)-f(y)) + \omega^2\|f(x)-f(y)\|^2 \\ &\leq \|x-y\|^2 - 2\omega\gamma\|x-y\|^2 + \omega^2 L^2\|x-y\|^2 \\ &= \underbrace{\left(1-2\omega\gamma+\omega^2L^2\right)}_{=:L_\omega^2}\|x-y\|^2 .\end{aligned}$$

Es muss gelten $L_\omega < 1$

$$\Leftrightarrow 1-2\omega\gamma+L^2\omega^2 < 1 \Leftrightarrow L^2\omega^2 < 2\omega\gamma \Leftrightarrow \omega < \frac{2\gamma}{L^2}.$$

Zur optimalen Wahl von ω bestimme das Minimum,

$$-2\omega + 2L^2\omega = 0 \Leftrightarrow \omega_{\text{opt}} = \frac{\gamma}{L^2}.$$

Optimale Kontraktionsrate

$$L_{\text{opt}} = L_{\omega_{\text{opt}}} = \sqrt{1-2\frac{\gamma}{L^2}\gamma+\frac{\gamma^2}{L^4}L^2} = \sqrt{1-\frac{\gamma^2}{L^2}} < 1.$$

5.4 **Satz** *Sei $f : \mathbb{R}^n \to \mathbb{R}^n$ lipschitzstetig mit Konstante L und strikt monoton mit Konstante $\gamma > 0$. Dann hat*

$$f(x) = 0$$

genau eine Lösung x^*. Das Iterationsverfahren

$$x^{n+1} = x^n - \omega f(x^n), \quad \text{mit } 0 < \omega < \frac{2\gamma}{L^2},$$

konvergiert für jedes $x^0 \in \mathbb{R}^n$ gegen x^* und es gilt

$$||x^n - x^*|| \leq L_\omega^n \left|\left|x^0 - x^1\right|\right|$$

mit

$$L_\omega = \sqrt{1 - 2\omega\gamma + \omega^2 L^2}. \quad \bowtie$$

Bsp Betrachte $f(x) = x - e^{-x^2}$. Für die erste Ableitung gilt,

$$f'(x) = 1 + 2xe^{-x^2}.$$

Nullstellen der zweiten Ableitung,

$$f''(x) = (2 - 4x^2)e^{-x^2} = 0 \Leftrightarrow x = \pm\frac{x}{\sqrt{2}}$$

Als untere Schranke der Ableitung ergibt sich,

$$\Rightarrow \underbrace{1 - \sqrt{2}e^{-1/2}}_{\gamma} \leq f'(x) \leq 1 + \sqrt{2}e^{-1/2} = L.$$

Damit f positiv definit ist, muss f durch eine positive Zahl nach unten beschränkt sein,

$$e^{1/2} > 1 + \frac{1}{2}, \quad \sqrt{2} < \frac{3}{2} \Rightarrow \frac{\sqrt{2}}{e^{1/2}} < \frac{3/2}{3/2} = 1 \Rightarrow \gamma > 0$$

Die optimalen Konstanten sind,

$$\omega_{\text{opt}} = \frac{\gamma}{L^2} \approx 0.041\ldots$$

$$L_{\text{opt}} = \sqrt{1 - \frac{\gamma^2}{L^2}} \approx \text{Übung}. \quad \blacksquare$$

Bemerkung. Das Verfahren garantiert Konvergenz für jedes $x^0 \in D$, jedoch kann die Konvergenz unter umständen sehr langsam sein. Man sieht dieses Schema immer wieder, robuste Verfahren garantieren Konvergenz, diese ist aber u.U. sehr langsam, weniger robuste Verfahren konvergieren dagegen meist sehr schnell.

Es ist so wie im echten Leben, der Traktor ist sehr robust und daher auch fürs Feld geeignet, während man für die Autobahn doch lieber den weniger robusten dafür aber schnellen Porsche wählt. ⊸

5.5 **Definition** *Ein Interationsverfahren,*

$$x^{n+1} = \phi(x^n)$$

mit $\phi : D \to D$, $D \subseteq \mathbb{R}^n$ heißt lokal konvergent gegen $x^* \in \mathbb{R}^n$ genau dann, wenn eine Umgebung U von x^* existiert, so dass die Folge $(x^n)_{n\geq 0}$ für jedes $x^0 \in U$ gegen x^* konvergiert.

Das Verfahren heißt global konvergent genau dann, wenn die Folge für jedes $x^0 \in D$ gegen x^* konvergiert.

Das Verfahren hat die Konvergenzordnung $p \geq 1$ genau dann, wenn

$$\exists\, c > 0 : \left|\left|x^{n+1} - x^*\right|\right| \leq C \left|\left|x^n - x^*\right|\right|^p, \quad \forall\ n \geq 0.$$

Dabei muss im Fall $p = 1$ auch $C < 1$ gelten. ⋈

Bsp Von den bereits untersuchten Verfahren sind folgende global konvergent,

- Bisektionsverfahren $p = 1$,
- Fixpunktiteration $p = 1$,

und folgende lokal konvergent,

- Sekantenverfahren $p = \frac{1}{2}(1 + \sqrt{5})$,
- Newtonverfahren $p = 2$. ■

■ Das Newton-Verfahren für nichtlineare Systeme

Betrachten das nichtlinerare System gegeben durch,

$$f(x) = 0, \quad f : D \to \mathbb{R}^n, \quad D \subseteq D \subseteq \mathbb{R}^n.$$

Linearisierung von f um die aktuelle Approximation x^n,

$$f(x) \approx f(x^n) + \mathrm{D}f(x^n)(x - x^n) =: T_f(x).$$

Berechnung von x^{n+1} aus

$$\begin{aligned} & T_f(x^{n+1}) = 0 \\ \Rightarrow & \mathrm{D}f(x^n)(x^{n+1} - x^n) = -f(x^n). \end{aligned}$$

Algorithmus für das Newton-Verfahren.

Start mit $x^0 \in D$
Schritt $n \to n + 1$
Löse
$$\mathrm{D}f(x^n)d^n = -f(x^n)$$
Setze
$$x^{n+1} = x^n + d^n$$

Man kann zeigen, dass dieses Verfahren - wie das Newton Verfahren - quadratisch konvergiert.

Bemerkungen. (i) Die Matrix $\mathrm{D}f(x^n)$ wird *nicht* invertiert. Es wird stattdessen ein lineares Gleichungssystem gelöst.

(ii) Wenn $\mathrm{D}f(x)$ nicht bekannt bzw. schwer zu berechnen ist, kann man $\mathrm{D}f(x)$ durch Differenzenquotienten approximieren.

$$(\mathrm{D}f(x))_{i,j} = \frac{\partial f_i(x)}{\partial x_j} \approx \frac{f_i(x + he_j) - f_i(x)}{h}.$$

e_j bezeichnet hier den j-ten Einheitsvektor. ⊸

5.6 **Konvergenz des Newton Verfahrens im $\mathbb{R}^n$** *Sei $f \in C^2(D \to \mathbb{R}^n)$, $D \subseteq \mathbb{R}^n$ offen, $f(x^*) = 0$ und $\det \mathrm{D}f(x^*) \neq 0$.*

Dann konvergiert das Newtonverfahren lokal gegen x^* mit Konvergenzordnung $p = 2$. ⋊

» Sei $\delta_0 > 0$ so, dass $\|\mathrm{D}f^{-1}(x)\| \leq C_0 < \infty$ für alle $x \in B_{\delta_0(x^*)}$ und $\mathrm{D}^2 f$ beschränkt in $B_{\delta_0}(x^*)$,

$$\left| \frac{\partial^2 f_i}{\partial x_j \partial x_k}(x) \right| \leq C_1, \quad \forall\ i,j,k = 1,\dots,n,\ x \in B_{\delta_0}(x^*).$$

Für $x^n \in B_{\delta_0}(x^*)$ gilt,

$$x^{n+1} - x^* = x^n - x^* - (\mathrm{D}f(x^n))^{-1}(f(x^n) - f(x^*)).$$

Weiter folgt,

$$\begin{aligned} f(x^n) - f(x^*) &= \int_0^1 \frac{\mathrm{d}}{\mathrm{d}t} f(tx^n + (1-t)x^*)\,\mathrm{d}t \\ &= \int_0^1 \mathrm{D}f(tx^n + (1-t)x^*)(x^n - x^*)\,\mathrm{d}t \\ &= \mathrm{D}f(x^n)(x^n - x^*) \\ &\quad + \int_0^1 \left(\mathrm{D}f(tx^n + (1-t)x^*) - \mathrm{D}f(x^n)\right)(x^n - x^*)\,\mathrm{d}t \end{aligned}$$

Wir können somit den Abstand von x^{n+1} und x^* abschätzen,

$$\begin{aligned} \left\|x^{n+1} - x^*\right\| &\leq \underbrace{\left\|x^n - x^* - (\mathrm{D}f(x^n))^{-1}\mathrm{D}f(x^n)(x^n - x^*)\right\|}_{=0} \\ &\quad + \left\| (\mathrm{D}f(x^n))^{-1} \int_0^1 \underbrace{\mathrm{D}f(tx^n + (1-t)x^*) - \mathrm{D}f(x^n)}_{\|\cdot\| \leq C_3(1-t)\|x^n - x^*\|} (x^n - x^*)\,\mathrm{d}t \right\| \\ &\leq C_0 C_2 \left\|x^n - x^*\right\|^2, \end{aligned}$$

denn f ist C^2 und daher ist $\mathrm{D}f$ lipschitz. Damit ist die quadratische Konvergenz gezeigt.

Für $S := \min\left\{\frac{1}{C_0 C_2}, \delta_0\right\}$ und $x^n \in B_\delta(x^*)$ folgt,

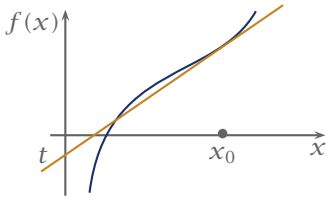
$$\left\|x^{n+1} - x^*\right\| \leq \frac{1}{2}\|x^n - x^*\| \Rightarrow \|x^n - x^*\| \leq \frac{1}{2^n}\left\|x^0 - x^*\right\| \overset{n\to\infty}{\to} 0.$$

für $x_0 \in B_\delta(x^*)$. Damit ist die lokale Konvergenz gezeigt. «

■ Dämpfung des Newton-Verfahrens

Um den "Einzugsbereich" der Konvergenz des Newton-Verfahrens zu vergrößern, kann man folgendene Strategie benutzen:

Schritt $n \to n+1$
Löse
$$Df(x^n)d^n = -f(x^n)$$
Wähle $\lambda \in (0,1]$ so, dass
$$\|f(x^n + \lambda_n d^n)\| \leq \|f(x^n)\|.$$
Setze
$$x^{n+1} = x^n + \lambda_n d^n.$$



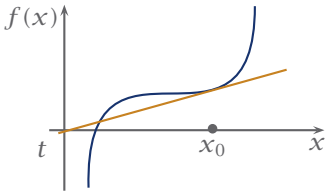
12 **Zur Veranschaulichung des Verfahrens.**

Es gilt,

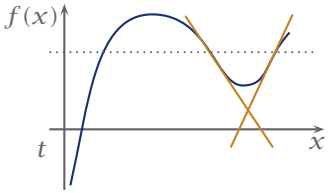
$$f(x^n + \lambda_n d^n) = f(x^n) + \underbrace{Df(x^n)}_{-\lambda_n f(x^n)}\lambda_n d^n + \mathcal{O}(\lambda_n^2) = (1-\lambda_n)f(x^n) + \mathcal{O}(\lambda_n^2)$$

Für λ_n klein genug und $f(x^n) \neq 0$ kann

$$\|f(x^n + \lambda d^n)\| \leq \|f(x^n)\|$$



13 **Hier würde die Dämpfung eine Approximation des Sattelpunkts anstatt der Nullsstelle bewirken.**



14 **Hier würde die Dämpfung verhindern, dass x^n aus dem lokalen Minimum hinauszukommt.**

garantiert werden. Für ein $\beta \in (0,1)$ und λ klein genug gilt sogar,

$$||f(x^n + \lambda_n d^n)|| \leq (1 - \beta\lambda_n)\, ||f(x^n)||$$

Armijo-Regel zur Wahl von λ_n *Seien $\alpha, \beta \in (0,1)$. Setze*

$$\lambda_n = \alpha^k, \quad \textit{mit } k = \min\left\{ j \in \mathbb{N}_0 \,:\, \left|\left| f(x^n + x^j d^n) \right|\right| \leq (1 - \beta\alpha^j)\, ||f(x^n)|| \right\} \quad \bowtie$$

Damit lässt sich jedoch immernoch keine Konvergenz garantieren.

5-C Abbruchkriterien

Jedes Iterationsverfahren der Form,

$$x^{n+1} = \phi(x^n)$$

muss beendet werden, wenn x^n "nahe genug" an der Lösung x^* liegt.

Das optimal Abbruchkriterium wäre,

$$||x^n - x^*|| < \varepsilon = \text{ vorgegebene Toleranz,}$$

doch dieses ist leider unbrauchbar, da x^* unbekannt.

Mögliche Abbruchkriterien

(i) $||f(x^n)|| < \varepsilon$,

(ii) $||x^{n+1} - x^n|| \leq \varepsilon$,

(iii) $||(\mathrm{D}f(x^n))^{-1} f(x^n)|| < \varepsilon$.

Beim klassischen Newtonverfahren sind (ii) und (iii) identisch, denn

$$x^{n+1} - x^n = d^n = -(\mathrm{D}f(x^n))^{-1} f(x^n).$$

Rechtfertigung für (iii):

$$\begin{aligned} 0 = f(x^*) &= f(x^n) + \mathrm{D}f(x^n)(x^* - x^n) + \dots \\ \Rightarrow x^* - x^n &= -(\mathrm{D}f(x^n))^{-1} f(x^n) + \dots \end{aligned}$$