Grundkurs Numerische Mathematik (MODUL M4)

für Fachrichtungen Bio-Informatik und Lehramt Wintersemester 2007/08

Martin Arnold
Martin–Luther–Universität Halle–Wittenberg
Naturwissenschaftliche Fakultät III
Institut für Mathematik

16. Januar 2008

Modul M4

Grundkurs Numerische Mathematik

Fachrichtungen Bio-Informatik und Lehramt

Institut für Mathematik, Arbeitsgruppe Numerische Mathematik

- Prof. Dr. M. Arnold (martin.arnold@mathematik.uni-halle.de)
- Dipl.-Tech. math. B. Burgermeister (bernhard.burgermeister@ ...)

Georg-Cantor-Haus (Heide Süd), Theodor-Lieser-Str. 5, Räume 221, 224

Internet: Stichwort „Lehre“ unter http://www.mathematik.uni-halle.de/~arnold/

Übungsbesprechung	Mi 10.10.2007, 08.15 Uhr, HS 1.04
Abschlussklausur	Donnerstag, den 14.02.08, 10-13 Uhr

Martin-Luther-Universität Halle-Wittenberg, NWF III, Institut für Mathematik
Martin Arnold: Grundkurs Numerische Mathematik (WiS 2007/08)

Hinweis Dieses Vorlesungsskript ist ausschließlich für den persönlichen Gebrauch der Hörerinnen und Hörer der Lehrveranstaltung „Grundkurs Numerische Mathematik (Modul M4) für Fachrichtungen Bio-Informatik und Lehramt“ (Wintersemester 2007/08) an der Martin–Luther–Universität Halle–Wittenberg bestimmt.
Teile des vorliegenden Materials wurden sinngemäß, z. T. auch wörtlich, der angegebenen Lehrbuchliteratur entnommen ohne dies in jedem Einzelfall durch Quellenangaben zu belegen.

Inhaltsverzeichnis

1 Einführung

1.1 Grundlagen

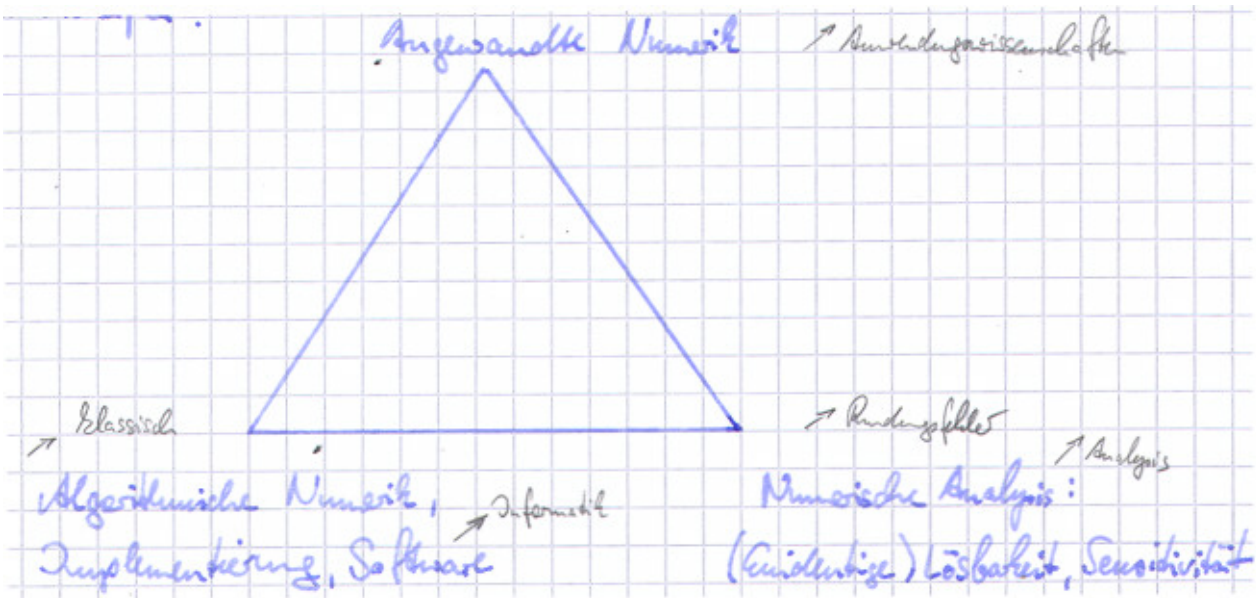
Bemerkung 1.1 (Numerische Mathematik)

a) **Im engeren Sinn:** zahlenmäßige Auswertung mathematischer Zusammenhänge

z. B.
- Lösung von linearen und nichtlinearen Gleichungssystemen
- Numerische Integration und Differentiation
- Näherungsweise Auswertung reeller Funktionen
- Numerische Lösung von Differentialgleichungen
- Numerische Lösung von Optimierungsproblemen

b) **Typisches Ziel:** Näherungen für die exakte Lösung eines mathematischen Problems, deren Fehler beliebig klein gemacht werden kann und für die verlässliche Fehlerschranken vorliegen.

c) **praktisch:** wesentliche Komponente des Wissenschaftlichen Rechnens (engl.: scientific computing): Computersimulation auf der Grundlage mathematischer Modelle in den Anwendungswissenschaften: Naturwissenschaften, Ingenieurwissenschaften, Medizin, Wirtschaftswissenschaften.

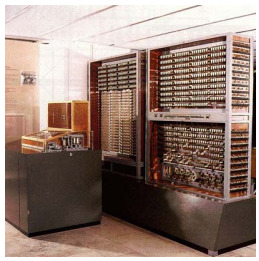


Bemerkung 1.2 (Entwicklung der Rechentechnik)

- Entwicklung der Numerik untrennbar verknüpft mit Entwicklung der Rechentechnik
- Grundlagen: verstärkt ab 18. Jahrhundert

Bemerkung 1.2: Entwicklung der Rechentechnik

Z3 (1941)



ENIAC (1946)

Abbildung 1.1: Klassische Rechentechnik.

Bemerkung 1.2: Entwicklung der Rechentechnik (II)

Hitachi SR8000-F1

Abbildung 1.2: Moderner Hochleistungsrechner am LRZ München.

- 1941 Z3 (K. Zuse)
  1946 ENIAC (J. v. Neumann)
  1958 erster Mikrochip
  1967 erster Taschenrechner
  1976 Home–Computer „Apple“
  1981 erster Personal Computer (PC)
  heute: leistungsfähige Arbeitsplatzrechner (PC, Workstation)
  Vektor- und Parallelrechner für High performance computing
- seit 1971: Anzahl der elementaren Transistorfunktionen je Sekunde verdoppelt sich etwa nach jeweils 18 Monaten

Bemerkung 1.3 (Literatur)

[1] Th. Huckle and S. Schneider. *Numerische Methoden. Eine Einführung für Informatiker, Naturwissenschaftler, Ingenieure und Mathematiker.* Springer–Verlag, Berlin, 2nd edition, 2006.

[2] R.W. Freund and R.H.W. Hoppe. *Stoer / Bulirsch: Numerische Mathematik 1.* Springer–Verlag, Berlin Heidelberg, 10th edition, 2007.

[3] J. Stoer and R. Bulirsch. *Numerische Mathematik 2.* Springer–Verlag, Berlin Heidelberg New York, 5th edition, 2005.

[4] P. Deuflhard and A. Hohmann. *Numerische Mathematik I. Eine algorithmisch orientierte Einführung.* Walter de Gruyter, Berlin New York, 3rd edition, 2002.

[5] A. Quarteroni, R. Sacco, and F. Saleri. *Numerische Mathematik 1.* Springer–Verlag, Berlin, 2002.

[6] A. Quarteroni, R. Sacco, and F. Saleri. *Numerische Mathematik 2.* Springer–Verlag, Berlin, 2002.

[7] G.H. Golub and Ch.F. van Loan. *Matrix Computations.* The Johns Hopkins University Press, Baltimore London, 3rd edition, 1996.

1.2 Klassische Polynominterpolation

Bemerkung 1.4 (Problemstellung)

geg.: $n+1$ *Stützpunkte* (x_j, y_j), $(\, j = 0, 1, \ldots, n\,)$ mit *Stützstellen* x_j und *Stützwerten* y_j, zwischen denen ein (oft auch nur vermuteter) funktionaler Zusammenhang besteht: $y_j = f(x_j)\,, \quad (\, j = 0, 1, \ldots, n\,)$. Praktisch oft: Messdaten y_j

ges.: Polynom $\Phi^{(n)}(x)$ höchstens n-ten Grades, das die $n+1$ *Interpolationsbedingungen*

$$y_j = \Phi^{(n)}(x_j)\,, \quad (\, j = 0, 1, \ldots, n\,)$$

erfüllt.

Bemerkung 1.4: Polynominterpolation

Beispiel $n = 2\,,\ x_0 = 0\,,\ x_1 = 1/2\,,\ x_2 = 1$

$y_j = \sin \pi x_j\,,\ (j = 0, 1, 2)$

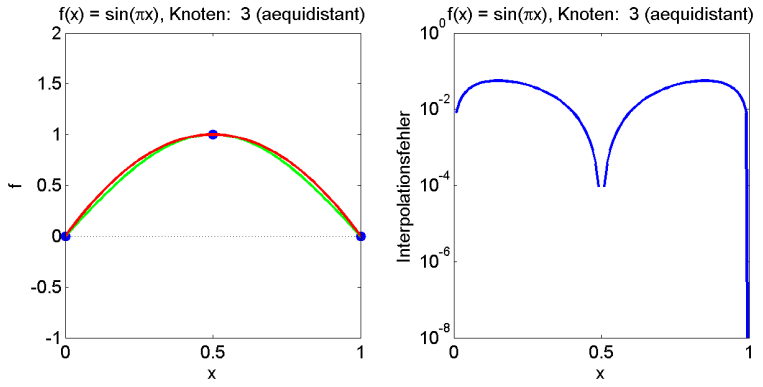


Abbildung 1.3: Interpolationspolynom $\Phi^{(2)}(x)$ zu $f(x) = \sin \pi x\,,\ (x \in [0, 1]\,)$.

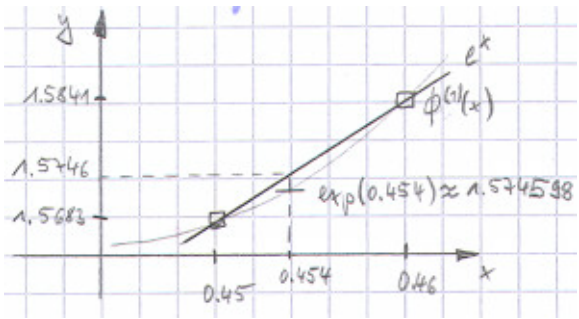
Beispiel Tabellierte Daten, z. B.
$n = 1,\ f(x) = \mathrm{e}^x,\ x_0 = 0.45,\ x_1 = 0.46,\ y_0 = 1.5683 \approx \exp(0.45),\ y_1 = 1.5841 \approx \exp(0.46)$

Lineare Interpolation

$$\exp(x) \approx \Phi^{(1)}(x) = 1.5683 + \frac{x - 0.45}{0.46 - 0.45}(1.5841 - 1.5683)$$

Beispiel

$$\Phi^{(1)}(0.454) = 1.5746$$



Bemerkung 1.4: Polynominterpolation (II)

Beispiel $n = 3\,,\ x_0 = 0\,,\ x_1 = 1/3\,,\ x_2 = 2/3\,,\ x_3 = 1$

$y_j = \sin \pi x_j\,,\ (j = 0, 1, 2, 3)$

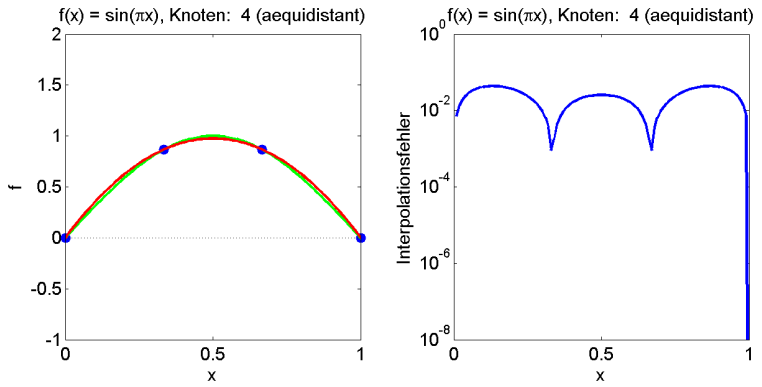


Abbildung 1.4: Interpolationspolynom $\Phi^{(3)}(x)$ zu $f(x) = \sin \pi x\,,\ (x \in [0, 1]\,)$.

Bemerkung 1.4: Polynominterpolation (III)

Beispiel $n = 4\,,\ x_0 = 0\,,\ x_1 = 1/4\,,\ x_2 = 1/2\,,\ x_3 = 3/4\,,\ x_4 = 1$

$y_j = \sin \pi x_j\,,\ (j = 0, 1, 2, 3, 4)$

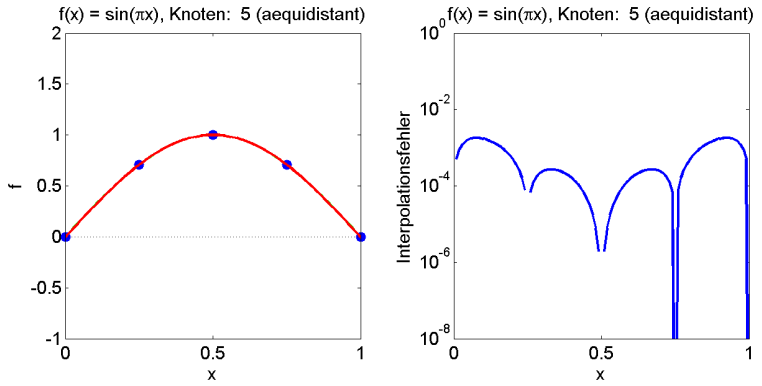


Abbildung 1.5: Interpolationspolynom $\Phi^{(4)}(x)$ zu $f(x) = \sin \pi x\,,\ (x \in [0, 1]\,)$.

Bemerkung 1.4: Polynominterpolation (IV)

Beispiel $n = 5\,,\ x_j = j/5\,,\ (j = 0,1,2,3,4,5)$
$y_j = \sin \pi x_j\,,\ (j = 0,1,2,3,4,5)$

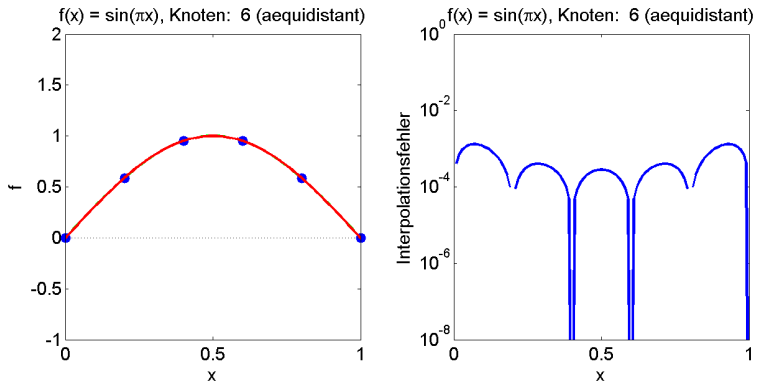


Abbildung 1.6: Interpolationspolynom $\Phi^{(5)}(x)$ zu $f(x) = \sin \pi x\,,\ (x \in [0,1]\,)$.

Bemerkung 1.4: Polynominterpolation (V)

Beispiel $n = 6\,,\ x_j = j/6\,,\ (j = 0,1,2,3,4,5,6)$
$y_j = \sin \pi x_j\,,\ (j = 0,1,2,3,4,5,6)$

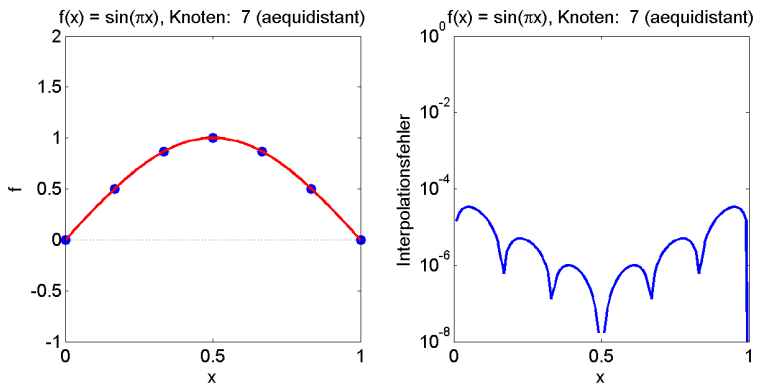


Abbildung 1.7: Interpolationspolynom $\Phi^{(6)}(x)$ zu $f(x) = \sin \pi x\,,\ (x \in [0,1]\,)$.

Bemerkung 1.5 (Eindeutigkeit des Interpolationspolynoms)
Sind die Stützstellen $x_0, x_1, \ldots, x_n$ paarweise voneinander verschieden, so ist das Interpolationspolynom $\Phi^{(n)}(x)$ aus Bemerkung 1.4. eindeutig bestimmt, denn gilt für zwei Polynome $\Phi_1^{(n)}, \Phi_2^{(n)} \in \Pi_n$

$$\Phi_1^{(n)}(x_j) = \Phi_2^{(n)}(x_j) = y_j\,, \quad (\,j = 0, 1, \ldots, n\,)\,,$$

so ist $\Phi_1^{(n)} - \Phi_2^{(n)} \in \Pi_n$ ein Polynom mit den $n+1$ Nullstellen $x_0, x_1, \ldots, x_n$
$\Rightarrow\ \Phi_1^{(n)} - \Phi_2^{(n)} = 0$ (Fundamentalsatz der Algebra), $\Phi_1^{(n)}(x) \equiv \Phi_2^{(n)}(x)$.

Bemerkung 1.6 (Horner–Schema)
Auswertung des Interpolationspolynoms mit je n Multiplikationen und Additionen:

$$\begin{aligned}\Phi^{(n)}(x) &= \sum_{j=0}^{n} a_j x^j = a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n \\ &= a_0 + x\,(a_1 + x\,(a_2 + \ldots + x\,(a_{n-1} + x a_n)\cdots))\end{aligned}$$

Horner–Schema

	a_n	a_{n-1}	a_{n-2}	$\cdots$	a_0
x		$x \cdot a_n$	$x \cdot (a_{n-1} + x a_n)$	$\cdots$	$x \cdot (\ldots)$
	a_n	$a_{n-1} + x a_n$	$a_{n-2} + x(a_{n-1} + x a_n)$	$\cdots$	$\ldots = \Phi^{(n)}(x)$

Beispiel Auswertung von $\Phi(x) = (x-1)^3 = x^3 - 3x^2 + 3x - 1$ an der Stelle $x = 5$:

	1	-3	3	-1
5		$5 \cdot 1 = 5$	$5 \cdot 2 = 10$	$5 \cdot 13 = 65$
	1	$-3 + 5 = 2$	$3 + 10 = 13$	$-1 + 65 = 64 = \Phi(5)$

Algorithmus

```
p := a_n
for i = n : -1 : 1
  | p := a_{i-1} + x * p
```

Matlab–Code Speicherschema für Vektoren in Matlab $(a_0, a_1, \ldots, a_n)$ = `a(1:(n+1))`

```
p = a(n+1);
for i=n:-1:1,
   p = a(i) + x * p;
end;
```

Bemerkung 1.7 (Elementarer Zugang)

Sei $\Phi^{(n)}(x) = \sum_{i=0}^{n} a_i x^i$ mit den zunächst unbekannten Koeffizienten $a_0, a_1, \ldots, a_n$. Die Interpolationsbedingungen $y_j = \Phi^{(n)}(x_j)$, $(\, j = 0, 1, \ldots, n\,)$ sind äquivalent zu dem linearen Gleichungssystem

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{pmatrix}$$

Lösung mit dem Gaußschen Algorithmus, vgl. Abschnitt 2.1.

Ziel: Transformation in ein äquivalentes lineares Gleichungssystem mit Dreiecksgestalt

Schritt 1 Addiere Vielfache der 1. Zeile zu Zeilen $2, \ldots, n+1$ so, dass in der 1. Spalte alle Elemente unterhalb der Hauptdiagonale verschwinden:

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 0 & x_1 - x_0 & x_1^2 - x_0^2 & \cdots & x_1^n - x_0^n \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & x_n - x_0 & x_n^2 - x_0^2 & \cdots & x_n^n - x_0^n \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 - y_0 \\ \vdots \\ y_n - y_0 \end{pmatrix}$$

Rechenaufwand: $\approx n^2$ Rechenoperationen zur Transformation von n Zeilen mit je n Spalten

Schritt *k* Addiere Vielfache der k-ten Zeile zu Zeilen $k+1, \ldots, n+1$ so, dass in der k-ten Spalte alle Elemente unterhalb der Hauptdiagonale verschwinden.

Rechenaufwand: $\approx (n+1-k)^2$ Rechenoperationen zur Transformation von $n+1-k$ Zeilen mit je $n+1-k$ Spalten

gesamt n Gaußschritte mit insgesamt

$$\sum_{k=1}^{n} (n+1-k)^2 = \sum_{l=1}^{n} l^2 = \frac{n(n+1)(2n+1)}{6}$$

Rechenoperationen.

Ergebnis: Rechenaufwand $\dfrac{n^3}{3} + \mathcal{O}(n^2)$ Rechenoperationen, wächst kubisch mit n.

Bemerkung 1.8 (Landau–Symbole)

Sei $x_0 \in \mathbb{R}$ und $g : I \to \mathbb{R}$ in einer Umgebung von x_0 definiert. Gibt es für ein $p \in \mathbb{R}$, $p \geq 0$ eine positive Konstante $\bar{c} \in \mathbb{R}$, so dass für alle x in einer hinreichend kleinen Umgebung von x_0 die Abschätzung

$$|g(x)| \leq \bar{c} \cdot |x - x_0|^p$$

erfüllt ist, so schreibt man

$$g(x) = \mathcal{O}(\,(x - x_0)^p\,)\,, \quad (\,x \to x_0\,)$$

sprich: „$g(x)$ ist groß O von $(x - x_0)^p$ “.

Beispiel $\sin x = \mathcal{O}(x)\,, \quad (\,x \to 0\,)$

Existiert $\lim\limits_{x\to x_0} \dfrac{g(x)}{(x - x_0)^p}$ und ist $\lim\limits_{x\to x_0} \dfrac{g(x)}{(x - x_0)^p} = 0$, so schreibt man

$$g(x) = o(\,(x - x_0)^p\,)\,, \quad (\,x \to x_0\,)$$

sprich: „klein o“.

Beispiel $\sqrt{x}^3 = o(x)\,, \quad (\,x \to 0\,)$

Entsprechend bedeutet

$$v(n) = \mathcal{O}(n^p)\,, \quad (\,n \to \infty\,)$$

für eine Funktion $v : \mathbb{N} \to \mathbb{R}$, dass für alle $n \geq n_0$ die Abschätzung $|v(n)| \leq \bar{c} n^p$ mit einer gewissen positiven Konstanten $\bar{c} \in \mathbb{R}$ erfüllt ist, und

$$v(n) = o(n^p)\,, \quad (\,n \to \infty\,)$$

steht für

$$\lim_{n\to\infty} \frac{v(n)}{n^p} = 0\,.$$

Beispiel $v(n) = \dfrac{n(n+1)(2n+1)}{6}$

Es gilt $v(n) < n \cdot 2n \cdot 3n \,/\, 6 = n^3$, also $v(n) = \mathcal{O}(n^3)$. Genauer gilt $v(n) = n^3/3 + w(n)$ mit $w(n) = \frac{1}{2}n^2 + \frac{1}{6}n = \mathcal{O}(n^2)$, man schreibt kurz: $v(n) \doteq n^3/3$.

Bemerkung 1.9 (Klassische Polynominterpolation: Lagrange–Darstellung)
Elementarer Zugang aus Bemerkung 1.7 $\Rightarrow$ Interpolationspolynom $\Phi^{(n)}$ in monomialer Basis $\{\,1, x, x^2, \ldots, x^n\,\}$:

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

Lagrange–Darstellung

$$\Phi^{(n)}(x) = \sum_{j=0}^{n} y_j L_j^{(n)}(x)$$

mit den *Lagrangeschen Basispolynomen*

$$L_j^{(n)}(x) := \prod_{\substack{i=0\\ i\neq j}}^{n} \frac{x - x_i}{x_j - x_i} = \frac{(x - x_0)\,(x - x_1)\,\cdots\,(x - x_{i-1})\,(x - x_{i+1})\,\cdots\,(x - x_n)}{(x_j - x_0)(x_j - x_1)\cdots(x_j - x_{i-1})(x_j - x_{i+1})\cdots(x_j - x_n)}\,,$$

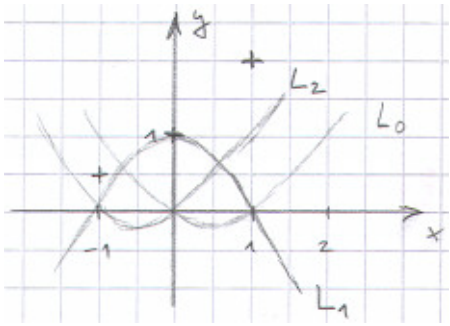
die die Interpolationsbedingungen

$$L_j(x_k) = \delta_{kj} = \begin{cases} 1 & \text{falls } k = j\,, \\ 0 & \text{sonst,} \end{cases} \qquad (\,k = 0, 1, \ldots, n\,)$$

erfüllen.

Beispiel $n = 2\,,\ \ x_0 = -1\,,\ \ x_1 = 0\,,\ \ x_2 = 1$

$$\begin{aligned} L_0^{(2)}(x) &= \frac{(x-0)(x-1)}{(-1-0)(-1-1)} \\ L_1^{(2)}(x) &= \frac{(x+1)(x-1)}{(0+1)(0-1)} \\ L_2^{(2)}(x) &= \frac{(x+1)(x-0)}{(1+1)(1-0)} \end{aligned}$$



Bemerkung 1.10 (Rekursive Auswertung des Interpolationspolynoms)
Sei $\Phi_{i,\ldots,i+l} \in \Pi_l$ das Interpolationspolynom zu Stützpunkten

$$(x_k, y_k)\,, \quad (\,k = i, i+1, \ldots, i+l\,)\,.$$

Dann gilt

$$\begin{aligned} \Phi_{i+1,\ldots,i+l}(x_k) &= y_k\,, \quad (\,k = i+1, \ldots, i+l\,)\,, \\ \Phi_{i,\ldots,i+l-1}(x_k) &= y_k\,, \quad (\,k = i, \ldots, i+l-1\,)\,, \end{aligned}$$

also

$$\Phi_{i,\ldots,i+l}(x) = \frac{(x - x_i)\Phi_{i+1,\ldots,i+l}(x) - (x - x_{i+l})\Phi_{i,\ldots,i+l-1}(x)}{x_{i+l} - x_i}\,, \qquad (*)$$

denn der rechts stehende Ausdruck erfüllt $\Phi_{i,\ldots,i+l}(x_k) = y_k$ für $k = i$, für $k = i + l$ und für $k = i+1, \ldots, i+l-1$.

Neville–Schema

$$\begin{array}{lll} x_0 & y_0 = \Phi_0(x) & \\ & & \Phi_{01}(x) = \frac{(x-x_0)y_1 - (x-x_1)y_0}{x_1 - x_0} \\ x_1 & y_1 = \Phi_1(x) & \\ & & \Phi_{12}(x) = \frac{(x-x_1)y_2 - (x-x_2)y_1}{x_2 - x_1} \\ x_2 & y_2 = \Phi_2(x) & \\ \vdots & \vdots & \vdots \\ & & \Phi_{n-1,n}(x) = \ldots \\ x_n & y_n = \Phi_n(x) & \end{array} \qquad \begin{array}{l} \\ \\ \Phi_{012}(x) \\ \\ \vdots \\ \\ \ldots \end{array} \qquad \begin{array}{l} \\ \\ \\ \\ \ldots \\ \\ \end{array} \qquad \begin{array}{l} \\ \\ \\ \\ \\ \\ \Phi(x) \end{array}$$

mit $\Phi(x) = \Phi_{01\cdots n}(x)\,.$

Beispiel 1.11 (Polynominterpolation: Neville–Schema)

Idee Approximiere $\sqrt{2} = 2^{1/2}$ durch $\Phi^{(2)}(1/2)$ mit dem Interpolationspolynom $\Phi^{(2)} \in \Pi_2$ zu $(-1, 1/2)$, $(0, 1)$ und $(1, 2)$.

Lagrange–Darstellung

$$\Phi^{(2)}(\frac{1}{2}) = \sum_{j=0}^{2} y_j L_j^{(2)}(\frac{1}{2}) = \frac{1}{2} \cdot L_0^{(2)}(\frac{1}{2}) + 1 \cdot L_1^{(2)}(\frac{1}{2}) + 2 \cdot L_2^{(2)}(\frac{1}{2})$$

mit den Lagrangeschen Basispolynomen $L_j^{(2)}(x)$ aus Bemerkung 1.9.

Neville

$$\begin{array}{cccccc} -1 & & & & & \\ & \frac{1}{2} & \searrow & \frac{(\frac{1}{2}-(-1))\cdot 1 - (\frac{1}{2}-0)\cdot\frac{1}{2}}{0-(-1)} = \frac{5}{4} & \searrow & \frac{(\frac{1}{2}-(-1))\cdot\frac{3}{2} - (\frac{1}{2}-1)\cdot\frac{5}{4}}{1-(-1)} = \frac{23}{16} \\ 0 & & \nearrow & & & \\ & 1 & \searrow & \frac{(\frac{1}{2}-0)\cdot 2 - (\frac{1}{2}-1)\cdot 1}{1-0} = \frac{3}{2} & \nearrow & \\ 1 & & \nearrow & & & \\ & 2 & & & & \end{array} \qquad = 1.4375$$

Bemerkung 1.12 (Klassische Polynominterpolation: Newton–Darstellung)

Problem Neville–Schema erfordert $\mathcal{O}(n^2)$ Rechenoperationen, ebenso die Auswertung von $\sum_j y_j L_j^{(n)}(x)$, aber

$$\sum_{j=0}^{n} a_j x^j = a_0 + x\,(a_1 + x\,(a_2 + \ldots + x\,(a_{n-1} + x a_n)\cdots))$$

kann mit $\mathcal{O}(n)$ Rechenoperationen ausgewertet werden.

gesucht Darstellung des Interpolationspolynoms, die man effizient bestimmen kann und die mit Horner–artigem Schema ausgewertet werden kann.

Ansatz Es gilt

$$\Phi_{i,i+1,\ldots,i+l}(x) = \Phi_{i,i+1,\ldots,i+l-1}(x) + (x - x_i)(x - x_{i+1})\cdots(x - x_{i+l-1}) f_{i,i+1,\ldots,i+l}$$

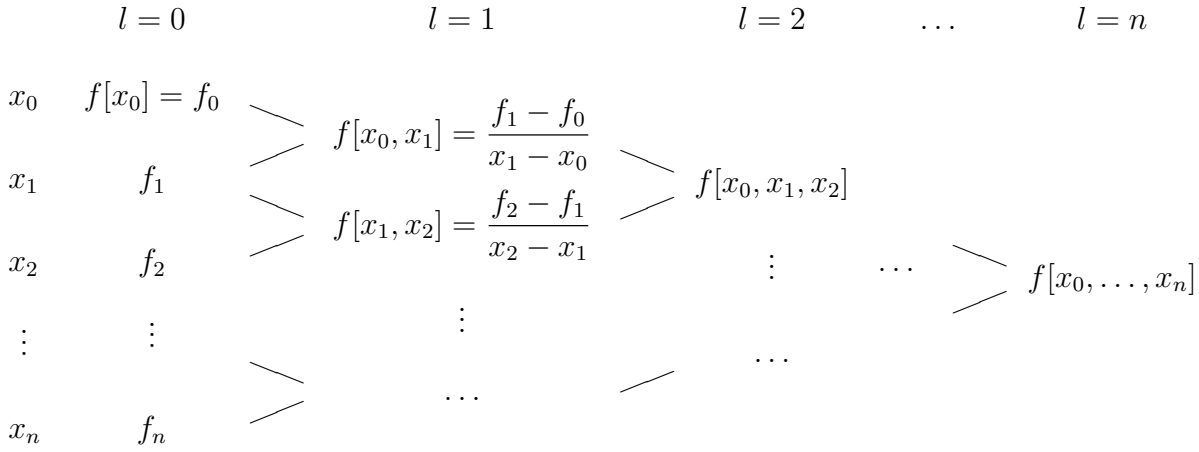
mit einem $f_{i,i+1,\ldots,i+l} \in \mathbb{R}$, denn

$$\Phi_{i,i+1,\ldots,i+l}(x_k) = \Phi_{i,i+1,\ldots,i+l-1}(x_k)\,, \quad (\,k = i, i+1, \ldots, i+l-1\,)\,.$$

Durch Vergleich der Koeffizienten von x^l in $(*)$ aus Bemerkung 1.10 folgt

$$f_{i,i+1,\ldots,i+l} = \frac{f_{i+1,\ldots,i+l} - f_{i,\ldots,i+l-1}}{x_{i+l} - x_i}\,.$$

Für $f_k = f(x_k)$ bezeichnet man diese *dividierten Differenzen* mit $f[x_i, \ldots, x_{i+l}]$.
Rekursive Berechnung mittels *Steigungsschema*



mit

$$f[x_0, x_1, x_2] = \frac{\dfrac{f_2 - f_1}{x_2 - x_1} - \dfrac{f_1 - f_0}{x_1 - x_0}}{x_2 - x_0}, \quad \ldots$$

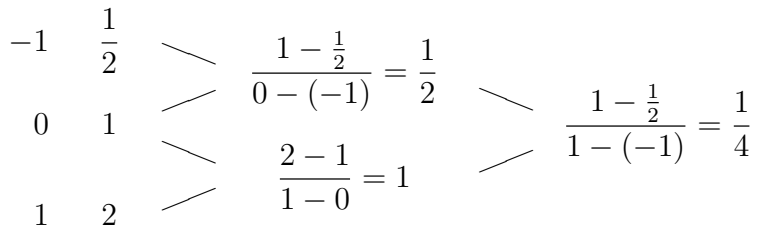
Newtonsche Darstellung

$$\Phi(x) = f[x_0] + (x - x_0)\, f[x_0, x_1] + (x - x_0)(x - x_1)\, f[x_0, x_1, x_2] + \ \ldots \ + \\ + (x - x_0)(x - x_1) \cdots (x - x_{n-1})\, f[x_0, x_1, \ldots, x_n]$$

Newton–Horner–Schema

$$\Phi(x) = f[x_0] + (x - x_0)\,(f[x_0, x_1] + (x - x_1)\,(f[x_0, x_1, x_2] + \ \ldots \\ + (x - x_{n-2})(f[x_0, x_1, \ldots, x_{n-1}] + (x - x_{n-1})\, f[x_0, x_1, \ldots, x_n]) \cdots))$$

Beispiel vgl. Beispiel 1.11



$$\Phi(x) = \frac{1}{2} + (x - (-1)) \cdot \frac{1}{2} + (x - (-1))\,(x - 0) \cdot \frac{1}{4} = \frac{1}{2} + (x + 1) \cdot (\,\frac{1}{2} + x \cdot \frac{1}{4}\,)$$

$$\Phi(\,\frac{1}{2}\,) = \frac{1}{2} + (\,\frac{1}{2} + 1) \cdot (\,\frac{1}{2} + \frac{1}{2} \cdot \frac{1}{4}\,) = \frac{1}{2} + \frac{3}{2} \cdot \frac{5}{8} = \frac{23}{16}$$

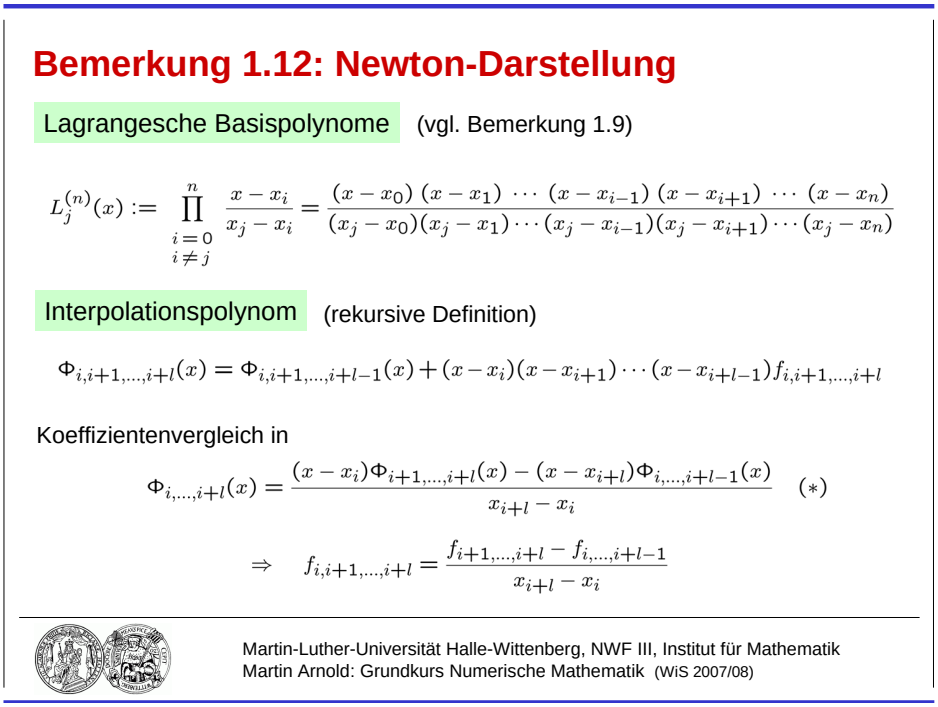


Abbildung 1.8: Rekursive Bestimmung der $f_{i,i+1,\ldots,i+l}$: Beweisidee.

Algorithmen

Steigungsschema

```
for i = n : −1 : 0
    a_i := f_i
    for j = i + 1 : n
        a_j := (a_j − a_{j−1}) / (x_j − x_i)
```

Newton–Horner–Schema

```
p := a_n
for i = n − 1 : −1 : 0
    p := a_i + (x − x_i) p
```

Satz 1.13 (Restglied der Polynominterpolation)
Sei $f \in C^{n+1}[a,b]$ und Φ das Interpolationspolynom zu den Stützstellen

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

d. h. $\Phi(x_k) = f(x_k)$, $(\,k = 0,1,\ldots,n\,)$. Dann gibt es zu jedem $\bar{x} \in [a,b]$ ein $\xi \in [a,b]$ mit

$$f(\bar{x}) - \Phi(\bar{x}) = \frac{f^{(n+1)}(\xi)}{(n+1)!}(\bar{x}-x_0)(\bar{x}-x_1)\cdots(\bar{x}-x_n)\,.$$

Beweis Die Behauptung ist trivial für $x = x_k$, $(\, k = 0, 1, \ldots, n\,)$.
Andernfalls betrachtet man

$$g(x) := f(x) - \Phi(x) - K(x - x_0)(x - x_1)\cdots(x - x_n)$$

mit

$$K := \frac{f(\bar{x}) - \Phi(\bar{x})}{(\bar{x} - x_0)(\bar{x} - x_1)\cdots(\bar{x} - x_n)}\,.$$

Die Funktion g hat in $[a, b]$ (mindestens) $n+2$ Nullstellen: $x_0, x_1, \ldots, x_n, \bar{x}$. Nach dem Satz von Rolle hat g' mindestens $n + 1$ Nullstellen usw. und schließlich $g^{(n+1)}(x)$ mindestens eine Nullstelle $\xi \in [a, b]$. Wegen

$$\begin{aligned} 0 &= g^{(n+1)}(\xi) \\ &= f^{(n+1)}(\xi) - \underbrace{\frac{\mathrm{d}^{n+1}\Phi(x)}{\mathrm{d}x^{n+1}}\bigg|_{x=\xi}}_{=\,0,\ \text{da } \Phi \in \Pi_n} - K \cdot \frac{\mathrm{d}^{n+1}}{\mathrm{d}x^{n+1}}\Big((x - x_0)(x - x_1)\cdots(x - x_n)\Big)\bigg|_{x=\xi} \\ &= f^{(n+1)}(\xi) - K \cdot (n + 1)! \end{aligned}$$

folgt schließlich $K = \dfrac{f^{(n+1)}(\xi)}{(n+1)!}$ und hieraus die Behauptung. ∎

Bemerkung 1.14 (Wahl der Stützstellen)

a) Nach Satz 1.13 sollte man die Stützstellen möglichst so wählen, dass

$$\max_{x \in [a,b]} |(x - x_0)(x - x_1)\cdots(x - x_n)| \to \min$$

Typische Stützstellen:

- äquidistant: $x_j := a + jh\,,$ Schrittweite $h := \dfrac{b - a}{n}$
- Tschebyscheff–Stützstellen: $[a, b] = [-1, 1]$

$$x_j := \cos\frac{(2j + 1)\pi}{2n + 2}\,, \quad (\, j = 0, 1, \ldots, n\,)$$

b) Zu jeder Folge von Stützstellen lässt sich ein $f \in C[a, b]$ angeben, so dass die zugehörige Folge der Interpolationspolynome nicht gleichmäßig konvergiert (Satz von Faber).

c) Praktische Erfahrung: Polynome hohen Grades neigen zu Oszillationen und sollten vermieden werden.

Beispiel 1.15 (Funktion von Runge)
Äquidistante Stützstellen sind ungeeignet zur Interpolation der *Funktion von Runge*

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

(vgl. Abb. 1.9, 1.10 und 1.11). Bessere Ergebnisse für Tschebyscheff–Stützstellen (vgl. Abb. 1.12 und 1.13).

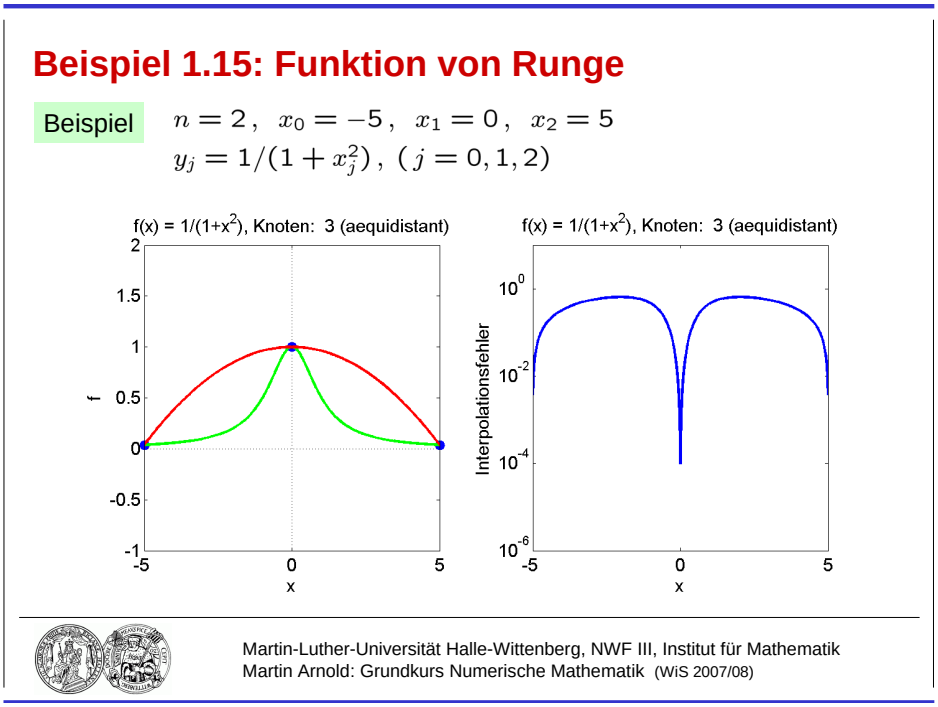


Abbildung 1.9: Interpolation der Funktion von Runge: $\Phi^{(2)}(x)$, Stützstellen äquidistant.

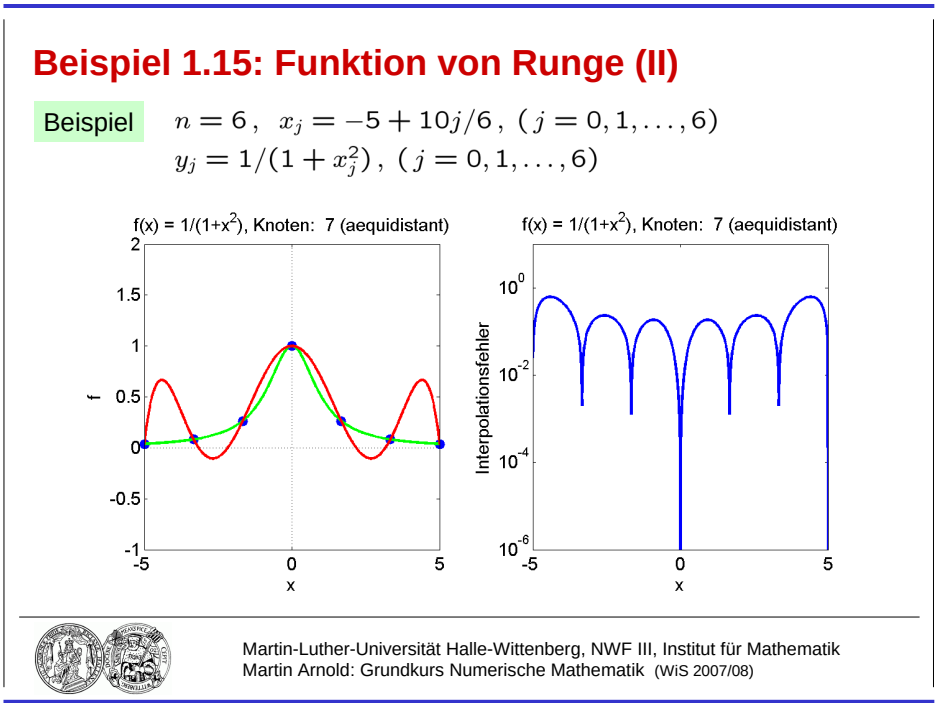


Abbildung 1.10: Interpolation der Funktion von Runge: $\Phi^{(6)}(x)$, Stützstellen äquidistant.

Beispiel 1.15: Funktion von Runge (III)

Beispiel $n = 8\,,\ x_j = -5 + 10j/8\,,\ (j = 0, 1, \ldots, 8)$
$y_j = 1/(1 + x_j^2)\,,\ (j = 0, 1, \ldots, 8)$

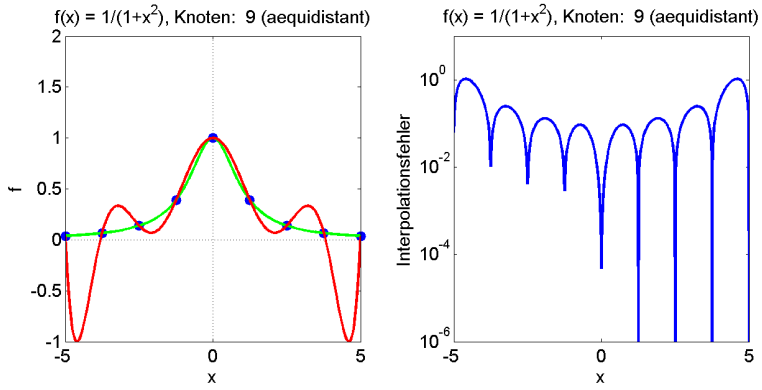


Abbildung 1.11: Interpolation der Funktion von Runge: $\Phi^{(8)}(x)$, Stützstellen äquidistant.

Beispiel 1.15: Funktion von Runge (IV)

Beispiel $n = 6\,,\ x_j = 5\cos\frac{(2j+1)\pi}{14}\,,\ (j = 0, 1, \ldots, 6)$
$y_j = 1/(1 + x_j^2)\,,\ (j = 0, 1, \ldots, 6)$

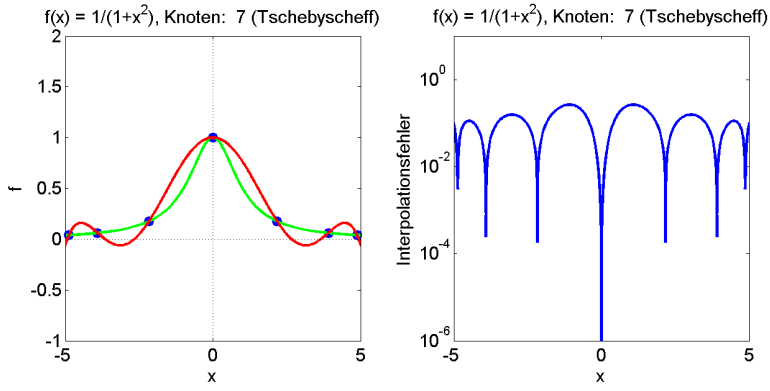


Abbildung 1.12: Interpolation der Funktion von Runge: $\Phi^{(6)}(x)$, Tschebyscheff–Stützstellen.

Beispiel 1.15: Funktion von Runge (V)

Beispiel $n = 8\,,\quad x_j = 5\cos\frac{(2j+1)\pi}{18}\,,\ (j = 0, 1, \ldots, 8)$

$y_j = 1/(1 + x_j^2)\,,\ (j = 0, 1, \ldots, 8)$

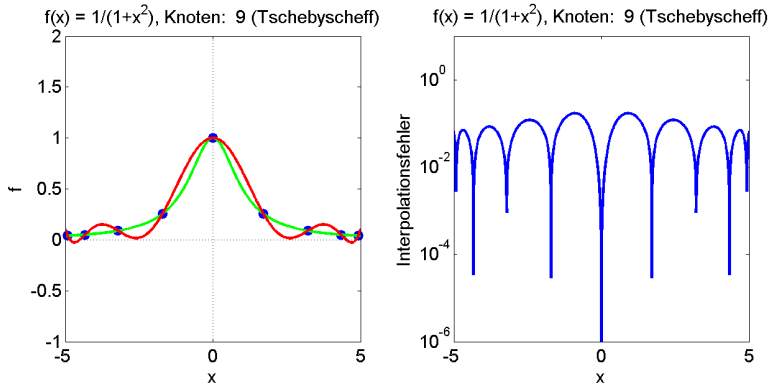


Abbildung 1.13: Interpolation der Funktion von Runge: $\Phi^{(8)}(x)$, Tschebyscheff–Stützstellen.

2 Lineare Gleichungssysteme

2.1 Gaußscher Algorithmus

Bemerkung 2.1 (Aufgabenstellung)

geg.: $A \in \mathbb{R}^{n\times n}$, $b \in \mathbb{R}^n$, $n = 10^1 \ldots 10^7$
ges.: Lösung x des linearen Gleichungssystems $Ax = b$

formal $x = A^{-1}b$, numerisch ungeeignet (Rechenaufwand zur Auswertung von A^{-1})

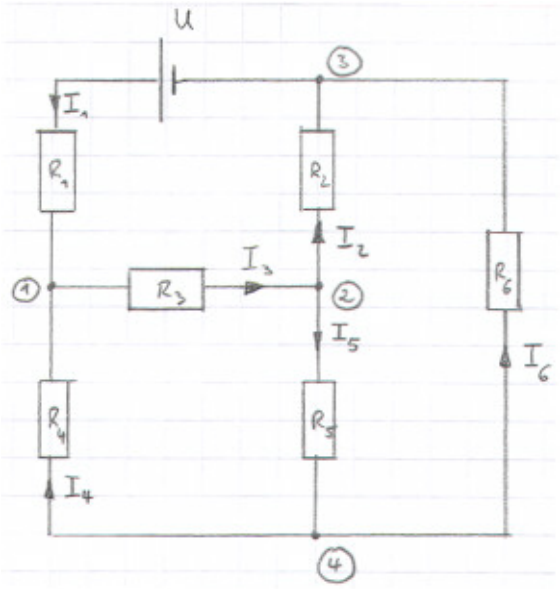
Lösbarkeit $x \in \mathbb{R}^n$ genau dann eindeutig bestimmt, wenn A regulär

$n = 2$, $n = 3$ Lösung mittels Cramerscher Regel

Bemerkung 2.2 (Lineare Netzwerkmodelle)

Modellbildung komplexer Systeme (Technik, Umwelt)
Basisbausteine, verknüpft durch Ein-/Ausgangsbeziehungen und Erhaltungsgrößen

Beispiel elektrische Schaltungen, Chip-Design



Basisbaustein: Widerstand
Ohmsches Gesetz

$$U_i = R_i \cdot I_i\,, \quad (\,i = 1, 2, \ldots, 6\,)$$

1. Kirchhoffsche Regel
In jedem Knoten: Summe der zufließenden gleich Summe der abfließenden Ströme

2. Kirchhoffsche Regel
In jeder Masche: Summe der Spannungsabfälle gleich Summe der Quellspannungen

Ergebnis Lineares Gleichungssystem

$$\begin{array}{c} ① \\ ② \\ ③ \\ ④ \\ [\,①,④,③\,] \\ [\,②,③,④\,] \\ [\,①,②,④\,] \\ [\,①,②,③\,] \end{array} \begin{pmatrix} 1 & 0 & -1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 & -1 & 0 \\ -1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 & 1 & -1 \\ R_1 & 0 & 0 & -R_4 & 0 & R_6 \\ 0 & R_2 & 0 & 0 & -R_5 & -R_6 \\ 0 & 0 & R_3 & R_4 & R_5 & 0 \\ R_1 & R_2 & R_3 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} I_1 \\ I_2 \\ I_3 \\ I_4 \\ I_5 \\ I_6 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ U \\ 0 \\ 0 \\ U \end{pmatrix}$$

Streicht man die redundanten Gleichungen $④ = -① - ② - ③$ und

$$[\,①,②,③\,] = [\,①,④,③\,] + [\,②,③,④\,] + [\,①,②,④\,]\,,$$

so ergibt sich $Ax = b$ mit $A \in \mathbb{R}^{6\times 6}$, $x = (I_1, I_2, \ldots, I_6)^\top \in \mathbb{R}^6$, $b = (0,0,0,U,0,0)^\top \in \mathbb{R}^6$.

Beachte

- Anteil der Nichtnullelemente in A gering $\Rightarrow$ „schwach besetzte“ Matrizen
- Position der Nichtnullelemente a_{ij} in A ergibt sich aus der sog. Topologie der Schaltung

Bemerkung 2.3 (Gestaffelte Gleichungssysteme)

Spezialfall $Rx = z$ mit regulärer oberer Dreiecksmatrix $R = (r_{ij}) \in \mathbb{R}^{n\times n}$, d. h., $r_{ii} \neq 0$, $r_{ij} = 0$, $(\, i = 1, \ldots, n;\ j = 1, \ldots, i-1\,)$.

$$\left.\begin{array}{rcl} r_{11}x_1 + r_{12}x_2 + \ \ldots\ + r_{1n}x_n &=& z_1 \\ r_{22}x_2 + \ \ldots\ + r_{2n}x_n &=& z_2 \\ \ddots \qquad \vdots &=& \vdots \\ r_{nn}x_n &=& z_n \end{array}\right\} \ Rx = z$$

Beispiel

$$\left.\begin{array}{rcl} 10x_1 - \ 7x_2 + \ 0\cdot x_3 &=& 7 \\ 2.5x_2 + \ 5\cdot x_3 &=& 2.5 \\ 6.2\cdot x_3 &=& 6.2 \end{array}\right\}$$

$$\Rightarrow \quad x_3 = \frac{6.2}{6.2} = 1\,, \quad x_2 = \frac{2.5 - 5\cdot 1}{2.5} = -1\,, \quad x_1 = \frac{7 + 7\cdot(-1)}{10} = 0$$

$$x = (\,0,\, -1,\, 1\,)^\top \quad \ldots \quad \text{Rückwärtssubstitution}$$

allgemein

$$x_n = \frac{z_n}{r_{nn}}\,, \qquad x_i = \frac{z_i - \displaystyle\sum_{j=i+1}^{n} r_{ij}x_j}{r_{ii}}\,, \quad (\, i = n-1, n-2, \ldots, 1\,)$$

Algorithmus

```
for i = n : −1 : 1
    | s := z_i
    | for j = (i + 1) : n
    |     | s := s − r_ij x_j
    | x_i := s/r_ii
```

Matlab–Code

```
x = zeros(size(z));
x(n) = z(n)/r(n,n);
for i=n-1:-1:1,
  x(i) = ( z(i) - r(i,i+1:n) * x(i+1:n) ) / r(i,i);
end;
```

analog $Lx = z$ mit regulärer unterer Dreiecksmatrix $L = (l_{ij}) \in \mathbb{R}^{n\times n}$,
d. h., $l_{ii} \neq 0$, $l_{ij} = 0$, $(\, i = 1, \ldots, n;\ j = i+1, \ldots, n\,) \ \Rightarrow$ „Vorwärtssubstitution“.

Bemerkung 2.4 (Gaußscher Algorithmus)

Idee Gleichungssystem $Ax = b$ in äquivalentes gestaffeltes Gleichungssystem umformen durch

- Multiplikation einer Gleichung mit einer von Null verschiedenen Zahl,
- Addition des Vielfachen einer Gleichung zu einer anderen Gleichung und / oder
- Vertauschung von Gleichungen.

Beispiel

$$\left.\begin{aligned} 10x_1 - 7x_2 &= 7 \\ -3x_1 + 2x_2 + 6x_3 &= 4 \\ 5x_1 - x_2 + 5x_3 &= 6 \end{aligned}\right\}$$

Schritt k Addiere Vielfache der k-ten Gleichung zu den Gleichungen $k+1, \ldots, n$, so dass in der k-ten Spalte unterhalb der Hauptdiagonale die Nichtnullelemente eliminiert werden $\rightsquigarrow$ **engl.:** Gaussian elimination

$k = 1$

$$\left.\begin{aligned} & 10x_1 - 7x_2 = 7 \\ \mathrm{II}' = +\tfrac{3}{10} * \mathrm{I} + \mathrm{II}\ : \quad & -0.1x_2 + 6x_3 = 6.1 \\ \mathrm{III}' = -\tfrac{5}{10} * \mathrm{I} + \mathrm{III}\,: \quad & 2.5x_2 + 5x_3 = 2.5 \end{aligned}\right\}$$

$k = 2$

$$\left.\begin{aligned} & 10x_1 - 7x_2 = 7 \\ & -0.1x_2 + 6x_3 = 6.1 \\ \mathrm{III}'' = +25 * \mathrm{II}' + \mathrm{III}'\,: \quad & 155x_3 = 155 \end{aligned}\right\}$$

Rücksubstitution $\Rightarrow$ $x = (\,0, -1, 1\,)^\top$.

Problem Hauptdiagonalelement $a_{kk}^{(k)} = 0$ im k-ten Eliminationsschritt

Lösung Vertauschung der k-ten Gleichung mit einer der Gleichungen $k+1, \ldots, n$ so, dass *Pivotelement* $a_{kk}^{(k)} \neq 0$ (stets möglich, falls A regulär).

Strategie Bestimme im k-ten Eliminationsschritt $p \in \{\, k, k+1, \ldots, n\,\}$ so, dass

$$|a_{pk}^{(k)}| = \max\{\, |a_{lk}^{(k)}| \,:\, l = k, k+1, \ldots, n\,\}$$

und vertausche k-te und p-te Gleichung $\Rightarrow$ (Spalten)–*Pivotisierung*
$\rightsquigarrow$ auch vorteilhaft zur Verringerung des Einflusses von Rundungsfehlern

Beispiel $k = 2$, tausche Gleichungen $\mathrm{II}' \rightleftharpoons \mathrm{III}'$

$$\left.\begin{array}{rrrrrl} & 10x_1 - & 7x_2 & & = & 7 \\ \widetilde{\mathrm{II}}' = \mathrm{III}' : & & 2.5x_2 + & 5x_3 & = & 2.5 \\ \widetilde{\mathrm{III}}' = \mathrm{II}' \ : & & -0.1x_2 + & 6x_3 & = & 6.1 \end{array}\right\}$$

$$\left.\begin{array}{rrrrrl} & 10x_1 - & 7x_2 & & = & 7 \\ & & 2.5x_2 + & 5x_3 & = & 2.5 \\ \widetilde{\mathrm{III}}'' = +\frac{1}{25} * \widetilde{\mathrm{II}}' + \widetilde{\mathrm{III}}' : & & & 6.2x_3 & = & 6.2 \end{array}\right\}$$

Algorithmus

```
for k = 1 : n − 1
  | p := k;  s := |a_kk|
  | for i = k + 1 : n
  |   | if |a_ik| > s then p := i;  s := |a_ik|
  | for j = k : n
  |   | s := a_kj;  a_kj := a_pj;  a_pj := s
  | s := b_k;  b_k := b_p;  b_p := s
  | for i = k + 1 : n
  |   | l_ik := a_ik/a_kk;
  |   | b_i := b_i − l_ik · b_k
  |   | for j = k + 1 : n
  |   |   | a_ij := a_ij − l_ik · a_kj
```

Bemerkung 2.5 (LU–Zerlegung)

***k*-ter Eliminationsschritt** in Matrixschreibweise (ohne Pivotisierung)
$A^{(k)} \rightsquigarrow A^{(k+1)}$ mit

$$A^{(k)} = \begin{pmatrix} \overset{k-1}{} & \overset{n-k+1}{} \\ \begin{array}{|cc} & \\ & \\ & \end{array} & \begin{array}{l} \\ \\ \end{array} \end{pmatrix} \begin{array}{l} k-1 \\ n-k+1 \end{array}, \qquad A^{(k+1)} = \begin{pmatrix} \begin{array}{|cc} \diagdown & \\ & \end{array} & \\ \begin{array}{c} * \\ 0 \\ \vdots \\ 0 \end{array} & \begin{array}{|c} \\ \\ \end{array} \end{pmatrix} \begin{array}{l} \\ n-k \end{array}$$

$$\begin{array}{c} \\ n-k \end{array}$$

$$A^{(k+1)} = \begin{pmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & & 1 & & \\ & & & -l_{k+1,k} & 1 & \\ & & & \vdots & & \ddots \\ & & & -l_{n,k} & & 1 \end{pmatrix} \cdot A^{(k)} = (I - L^{(k)})A^{(k)}$$

Bemerkung 2.4: Gaußscher Algorithmus

```
for k = 1 : n − 1
  | p := k;  s := |a_kk|
  | for i = k + 1 : n
  |   | if |a_ik| > s then p := i;  s := |a_ik|
  | for j = k : n
  |   | s := a_kj;  a_kj := a_pj;  a_pj := s
  | s := b_k;  b_k := b_p;  b_p := s
  | for i = k + 1 : n
  |   | l_ik := a_ik/a_kk;
  |   | b_i := b_i − l_ik · b_k
  |   | for j = k + 1 : n
  |   |   | a_ij := a_ij − l_ik · a_kj
```

Abbildung 2.1: Gaußscher Algorithmus.

mit

$$L^{(k)} = \begin{pmatrix} 0 & & & & & & \\ & \ddots & & & & & \\ & & 0 & & & & \\ & & & 0 & & & \\ & & & l_{k+1,k} & 0 & & \\ & & & \vdots & & \ddots & \\ & & & l_{n,k} & & & 0 \end{pmatrix}, \quad A^{(1)} := A\,, \quad A^{(n)} =: U \quad \text{(obere Dreiecksmatrix)},$$

$$(I - L^{(n-1)}) \cdots (I - L^{(1)})A = U\,.$$

Auflösen nach A

$$A = LU \quad \text{mit} \quad L := (I - L^{(1)})^{-1} \cdots (I - L^{(n-1)})^{-1}\,.$$

Wegen $L^{(i)}L^{(j)} = 0\,,\ (i \le j\,)$, ist

$$(I - L^{(i)})(I + L^{(i)}) = I - L^{(i)} + L^{(i)} = I \quad \Rightarrow \quad (I - L^{(i)})^{-1} = (I + L^{(i)})\,.$$

Außerdem erhält man $(I + L^{(i)})(I + L^{(j)}) = I + L^{(i)} + L^{(j)}\,,\ (i \le j\,)$, also insgesamt

$L = (I + L^{(1)}) \cdots (I + L^{(n-1)}) = I + L^{(1)} + L^{(2)} + \ldots + L^{(n-1)}$ ... untere Dreiecksmatrix.

Ergebnis Gauß–Algorithmus berechnet *LU–Zerlegung* von A: $A = L \cdot U$ mit der oberen Dreiecksmatrix U aus dem gestaffelten linearen Gleichungssystem und der unteren Dreiecksmatrix L, die die Eliminationskoeffizienten l_{ik} enthält und deren Hauptdiagonalelemente $= 1$ sind.

praktisch Abspeicherung der Nicht–Diagonalelemente von L unterhalb der Hauptdiagonalen von A.

Pivotisierung Spaltenpivotisierung $\Rightarrow$ $LU = PA$ mit *Permutationsmatrix* P

Beispiel wie oben

$$\underset{P}{\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}} \underset{A}{\begin{pmatrix} 10 & -7 & 0 \\ -3 & 2 & 6 \\ 5 & -1 & 5 \end{pmatrix}} = \underset{L}{\begin{pmatrix} 1 & 0 & 0 \\ 1/2 & 1 & 0 \\ -3/10 & -1/25 & 1 \end{pmatrix}} \underset{U}{\begin{pmatrix} 10 & -7 & 0 \\ 0 & 2.5 & 5 \\ 0 & 0 & 6.2 \end{pmatrix}}$$

Lösung linearer Gleichungssysteme

Schritt 1 Berechne mittels Gauß–Algorithmus mit Spaltenpivotisierung die LU–Zerlegung $PA = LU$.

Für jede rechte Seite: **Schritt 2** Vorwärtssubstitution

$$Ly = Pb \rightsquigarrow y$$

Schritt 3 Rückwärtssubstitution

$$Ux = y \rightsquigarrow x$$

Software
- Matlab–Kommando `lu`
- `LAPACK` (`FORTRAN`, `C`), über `http://www.netlib.org`
  `DGETRF`, `DGETRS`

Rechenaufwand Gemessen in flops ... Floating point operations
(1 Gleitpunktaddition, 1 Gleitpunktmultiplikation)

Schritt 1 $\frac{n^3}{3} + \mathcal{O}(n^2)$ Rechenoperationen, vgl. Bemerkung 1.7

Schritt 2+3 jeweils $\frac{n^2}{2} + \mathcal{O}(n)$ Rechenoperationen $\Rightarrow$ $n^2 + \mathcal{O}(n)$ Rechenoperationen pro linearem Gleichungssystem

Bemerkung 2.6 (Symmetrische Koeffizientenmatrizen, Cholesky–Zerlegung)
Wichtiger Spezialfall: $Ax = b$ mit $A = A^\top$, insbesondere auch symmetrische, positiv definite Koeffizientenmatrizen A, d. h.

$$x^\top Ax > 0\,, \quad (\, x \in \mathbb{R}^n \setminus \{0\}\,)\,.$$

$\boldsymbol{A = A^\top}$ Gauß–Algorithmus ohne Pivotisierung $\rightsquigarrow$ $A = L \cdot U$

Sei $D := \operatorname{diag}_{1 \le i \le n} u_{ii}$ $\Rightarrow$ $D^{-1}U$ ist obere Dreiecksmatrix mit Hauptdiagonalelementen $= 1$.

$$A^\top = (L \cdot D \cdot D^{-1}U)^\top = (D^{-1}U)^\top DL^\top$$

Aus der Eindeutigkeit der LU–Zerlegung (!) folgt $D^{-1}U = L^\top$, also $A = LDL^\top$.

Berechnung mit $\frac{n^3}{6} + \mathcal{O}(n^2)$ Rechenoperationen möglich.

Pivotisierung: gleichzeitiger Zeilen- und Spaltentausch, um Symmetrie zu erhalten.

A symmetrisch, positiv definit Man zeigt, dass der Gauß–Algorithmus ohne Pivotisierung stets durchführbar ist. Wegen $0 < y^\top Ay$ für $y := (L^\top)^{-1}x$ ist

$$0 < y^\top Ay = ((L^\top)^{-1}x)^\top (LDL^\top)((L^\top)^{-1}x) = (L^\top (L^\top)^{-1}x)^\top D(L^\top (L^\top)^{-1}x) = x^\top Dx\,,$$

also ist auch D positiv definit: $D = \operatorname*{diag}_{1\le i\le n} d_i$ mit $d_i > 0$.

Cholesky–Zerlegung von A

$$A = \hat{L}\hat{L}^\top \quad \text{mit} \quad \hat{L} := L \cdot D^{1/2}\,,\ D^{1/2} = \operatorname{diag}_i \sqrt{d_i}$$

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix} = \begin{pmatrix} \hat{l}_{11} & & & \\ \hat{l}_{21} & \hat{l}_{22} & & \\ \vdots & \vdots & \ddots & \\ \hat{l}_{n1} & \hat{l}_{n2} & \cdots & \hat{l}_{nn} \end{pmatrix} \begin{pmatrix} \hat{l}_{11} & \hat{l}_{21} & \cdots & \hat{l}_{n1} \\ & \hat{l}_{22} & \cdots & \hat{l}_{n2} \\ & & \ddots & \vdots \\ & & & \hat{l}_{nn} \end{pmatrix}$$

Algorithmus

```
for k = 1 : n
    l̂_kk := ( a_kk − Σ_{j=1}^{k−1} l̂_kj^2 )^{1/2}
    for i = (k + 1) : n
        l̂_ik := ( a_ik − Σ_{j=1}^{k−1} l̂_ij l̂_kj ) / l̂_kk
```

Vorwärts- / Rückwärtssubstitution
wie im allgemeinen Fall, beachte jedoch, dass nur $\hat{L}$, nicht $\hat{L}^\top$ abgespeichert wird.

Rechenaufwand $\frac{n^3}{6} + \mathcal{O}(n^2)$ Rechenoperationen, n Quadratwurzeln

2.2 Lineare Ausgleichsrechnung

Bemerkung 2.7 (Methode der kleinsten Quadrate)

geg.: $A \in \mathbb{R}^{m\times n},\ m > n,\ \operatorname{rank}(A) = n,\ b \in \mathbb{R}^m$
ges.: „Lösung“ $x \in \mathbb{R}^n$ von $Ax = b$

Methode der kleinsten Quadrate (Gauß)
Da i. Allg. keine *klassische Lösung* $x \in \mathbb{R}^n$ mit $Ax - b = 0$ existiert, sucht man als *verallgemeinerte Lösung* ein $x \in \mathbb{R}^n$ so, dass

$$\|Ax - b\|_2 \to \min\,. \qquad (*)$$

Hierbei bezeichnet $\|v\|_2 := \sqrt{v^\top v} = \Big(\sum_{i=1}^{m} v_i^2\Big)^{1/2}$ die *euklidische Vektornorm* des Vektors $v = (v_i)_{i=1}^m \in \mathbb{R}^m$, vgl. Abschnitt 3.2.

Eigenschaft: Für Vektoren $y = (y_i) \in \mathbb{R}^n,\ z = (z_i) \in \mathbb{R}^{m-n}$ gilt

$$\|\begin{pmatrix} y \\ z \end{pmatrix}\|_2^2 = \sum_{i=1}^{n} y_i^2 + \sum_{i=1}^{m-n} z_i^2 = \|y\|_2^2 + \|z\|_2^2 \,.$$

Das *Kleinste–Quadrate–Problem* $(*)$ ist äquivalent zu

$$\|Ax - b\|_2^2 = \sum_{i=1}^{m} \Big(\sum_{j=1}^{n} a_{ij}x_j - b_i\Big)^2 \to \min \,.$$

Notwendige Bedingung für Minimum

$$0 \stackrel{!}{=} \frac{\partial}{\partial x_k}\|Ax - b\|_2^2 = 2\sum_{i=1}^{m}\Big(\sum_{j=1}^{n} a_{ij}x_j - b_i\Big)a_{ik} \,, \quad (\,k = 1, \ldots, n\,) \,,$$

$$\sum_{i=1}^{m} a_{ik}\Big(\sum_{j=1}^{n} a_{ij}x_j\Big) = \sum_{i=1}^{m} a_{ik}b_i \,, \quad (\,k = 1, \ldots, n\,) \,,$$

Gaußsche Normalgleichungen $\boxed{A^\top Ax = A^\top b}$

Wegen $A^\top A = (A^\top A)^\top \in \mathbb{R}^{n\times n}$ und $\operatorname{rank}(A) = n$ ist $A^\top A$ symmetrisch und positiv definit:

$$\xi \in \mathbb{R}^n \setminus \{0\} \ \Rightarrow\ \xi^\top(A^\top A)\xi = (\xi^\top A^\top)(A\xi) = (A\xi)^\top(A\xi) = \|A\xi\|_2^2 > 0 \,,$$

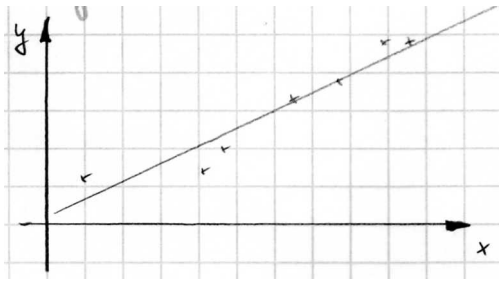
denn $A\xi \neq 0$ wegen $\xi \neq 0$ und $\operatorname{rank}(A) = n$.

Lösung der Normalgleichungen mittels Cholesky–Zerlegung

Problem Bei Verwendung der Gaußschen Normalgleichungen reagiert die numerische Lösung oft sehr empfindlich auf Rundungsfehler.

Alternative Orthogonalisierungsverfahren, vgl. Bemerkung 2.10.

Beispiel 2.8 (Lineare Regression)



geg.: Messdaten (x_i, y_i), $(\, i = 1, \ldots, m\,)$, mit Messfehlern behaftet

ges.: Gerade $y = a + bx$, die die Messwerte „möglichst gut“ approximiert:
$y_i \approx a + bx_i$, $(\, i = 1, \ldots, m\,)$

Ansatz $\displaystyle\sum_{i=1}^{m} (a + bx_i - y_i)^2 \to \min$

Matrixschreibweise

$$A \begin{pmatrix} a \\ b \end{pmatrix} = y \quad \text{mit} \quad y = (y_1, \ldots, y_m)^\top \in \mathbb{R}^m\,, \quad A = \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_m \end{pmatrix} \in \mathbb{R}^{m \times 2}\,, \quad n = 2$$

Normalgleichungen $A^\top A \cdot \begin{pmatrix} a \\ b \end{pmatrix} = A^\top y$

$$\begin{pmatrix} m & \displaystyle\sum_{j=1}^{m} x_j \\ \displaystyle\sum_{j=1}^{m} x_j & \displaystyle\sum_{j=1}^{m} x_j^2 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} \displaystyle\sum_{j=1}^{m} y_j \\ \displaystyle\sum_{j=1}^{m} x_j y_j \end{pmatrix} \quad \Rightarrow \quad a, b \in \mathbb{R}$$

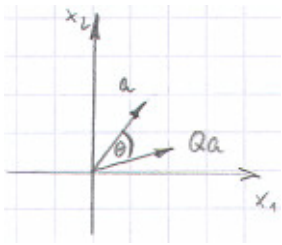
Bemerkung 2.9 (Orthogonale Transformationen)

Idee Verwende orthogonale Transformationen, um lineare Gleichungssysteme und lineare Ausgleichsprobleme in äquivalente Probleme einfacherer Gestalt umzuformen.

***n* = 2** Drehungen, Spiegelungen

hier Drehmatrizen

$$Q = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}$$



Literatur zu Spiegelungsmatrizen:
Stoer, Deuflhard/Hohmann

allgemein Givens–Drehungen, Householder–Spiegelungen

hier Givens–Drehungen

$$G_{kl} = \begin{pmatrix} 1 &&&&&&&&&& \\ & \ddots &&&&&&&&& \\ && 1 &&&&&&&& \\ &&& c &&&& s &&& \\ &&&& 1 &&&&&& \\ &&&&& \ddots &&&&& \\ &&&&&& 1 &&&& \\ &&& -s &&&& c &&& \\ &&&&&&&& 1 && \\ &&&&&&&&& \ddots & \\ &&&&&&&&&& 1 \end{pmatrix} \begin{matrix} \\ \\ \\ l \\ \\ \\ \\ k \\ \\ \\ \\ \end{matrix} \in \mathbb{R}^{m \times m}$$

$$\qquad\qquad\qquad l \qquad\qquad\qquad k$$

mit $c = \cos\theta\,,\quad s = \sin\theta\,,\quad c^2 + s^2 = 1\,.$

Bestimme θ so, dass $\big(G_{kl}A\big)_{kl} = 0$:

$$a_{kl} \;\mapsto\; -sa_{ll} + ca_{kl} \;\overset{!}{=}\; 0 \quad \text{und} \quad c^2 + s^2 = 1\,.$$

$$|a_{kl}| > |a_{ll}| \;: \qquad \tau := \frac{a_{ll}}{a_{kl}}\,, \quad s := \frac{1}{\sqrt{1+\tau^2}}\,, \quad c := s \cdot \tau$$

$$|a_{kl}| \le |a_{ll}| \;: \qquad \tau := \frac{a_{kl}}{a_{ll}}\,, \quad c := \frac{1}{\sqrt{1+\tau^2}}\,, \quad s := c \cdot \tau$$

Bemerkung 2.10 (QR–Zerlegung)

geg.: $A \in \mathbb{R}^{m\times n}\,,\;\; m \ge n\,,\;\; \operatorname{rank}(A) = n$

Schrittweise Elimination der Nichtnullelemente a_{ij}, $(\,j < i\,)$ mittels Givens–Drehungen:

$$A \to G_{21}A \to G_{31}G_{21}A \to \;\ldots\; \to G_{m1}\cdots G_{31}G_{21}A \to G_{32}G_{m1}\cdots G_{31}G_{21}A \to \;\ldots\; \to R$$

mit

$$G_{mn}G_{m-1,n}\,\cdots\,G_{n+1,n}G_{m,n-1}\,\cdots\,G_{32}G_{m1}\,\cdots\,G_{31}G_{21}A = R = \begin{pmatrix} \widetilde{R} \\ \cdots \\ 0 \end{pmatrix} \begin{matrix} {\scriptstyle n} \\ \\ {\scriptstyle m-n} \end{matrix}$$

$$\scriptstyle n$$

QR–Zerlegung $A = QR$ mit der orthogonalen Matrix

$$Q := G_{21}^\top G_{31}^\top\,\cdots\,G_{m,n-1}^\top G_{n+1,n}^\top\,\cdots\,G_{m,n}^\top\,.$$

Wegen $n = \operatorname{rank}(A) = \operatorname{rank}(R) = \operatorname{rank}(\widetilde{R})$ ist $\widetilde{R} \in \mathbb{R}^{n\times n}$ regulär.

Lösung regulärer linearer Gleichungssysteme

1. QR–Zerlegung mittels Givens–Drehungen, $\frac{2}{3}n^3 + \mathcal{O}(n^2)$ Rechenoperationen
2. $Ax = b \;\Leftrightarrow\; QRx = b \;\Leftrightarrow\; Rx = z\,,\;\; Qz = b$

   2a) $z = Q^\top b = G_{n,n-1}\,(G_{n,n-2}\,(G_{n-1,n-2}\,(\,\cdots\; G_{31}(G_{21}b)\;\cdots\,)))$
   Berechnung durch sukzessive Auswertung von Matrix–Vektor–Produkten $G_{kl}w$, keine explizite Berechnung von Q

   2b) Löse $Rx = z$ mittels Rücksubstitution.

Besonders geeignet für Gleichungssysteme $Ax = b$, deren Lösung sehr empfindlich gegenüber Rundungsfehlern ist, und zur numerischen Rangbestimmung von A.

Lösung von Kleinste–Quadrate–Problemen
Wegen $\|w\|_2^2 = w^\top w$ ist $\|w\|_2$ invariant gegenüber orthogonalen Transformationen:

$$\|Qw\|_2^2 = (Qw)^\top(Qw) = w^\top(Q^\top Q)w = w^\top w = \|w\|_2^2$$

$$\Rightarrow\;\; \|Ax - b\|_2^2 = \|QRx - b\|_2^2 = \|Q(Rx - Q^\top b)\|_2^2 = \|Rx - Q^\top b\|_2^2 = \|\widetilde{R}x - z_1\|_2^2 + \|z_2\|_2^2$$

mit $\;Q^\top b = \begin{pmatrix} z_1 \\ \cdots \\ z_2 \end{pmatrix} \begin{matrix} n \\ \\ m-n \end{matrix} \in \mathbb{R}^m\,.$

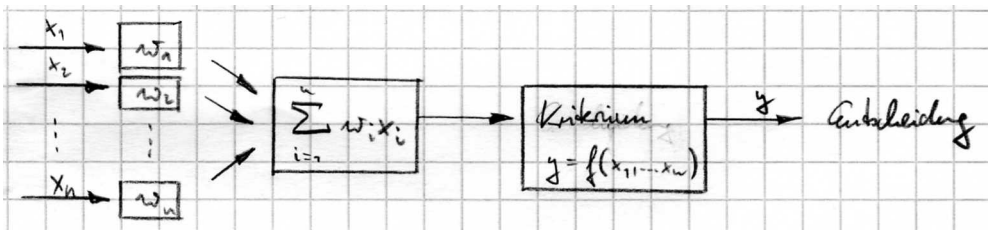
Lösung: Berechne $Q^\top b$ und löse $\widetilde{R}x = z_1$ mittels Rücksubstitution.

Beispiel 2.11 (Neuronale Netze)

Idee Entscheidungen fällen auf Grundlage einer Vielzahl von Einzelinformationen in Anlehnung an Entscheidungsprozesse im menschlichen Hirn.

Beispiel Technik Wäsche in Waschmaschine „stark verschmutzt“
Messung der Wassertemperatur und Wassertrübung an verschiedenen Punkten
Grundlage der Entscheidung: Erfahrung
Konsequenz einer Fehlentscheidung: Lerneffekt

Neuronales Netz (Ein–Schicht–Modell, linear)



Verhalten des Netzes bestimmt durch *Gewichte* $w_1, \ldots, w_n$

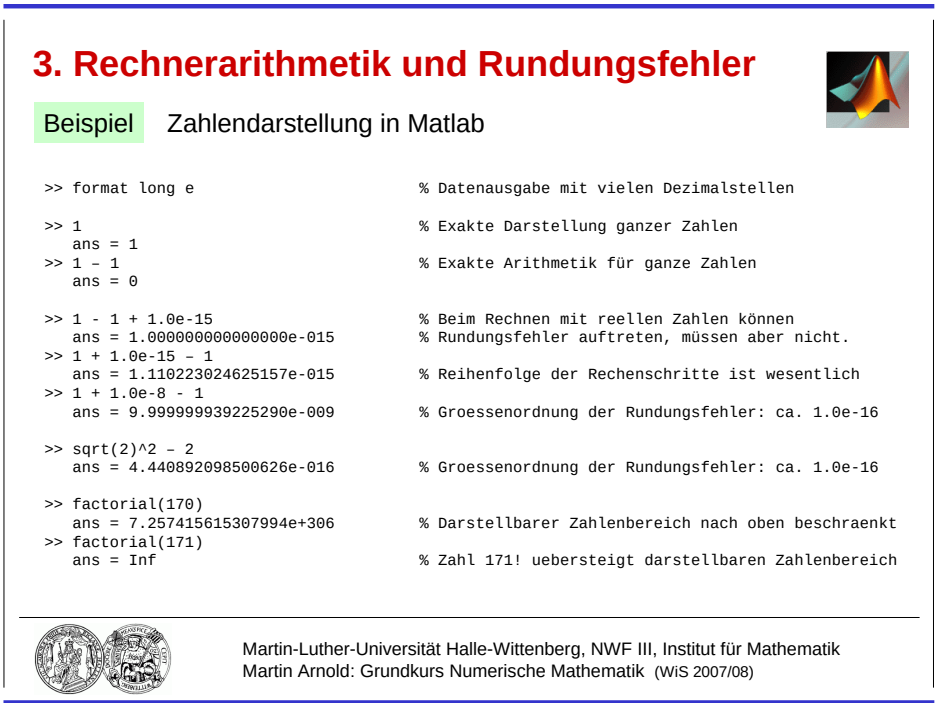


Abbildung 3.1: Rechnen in Gleitpunktarithmetik: Beispiel Matlab.

Training des Netzes Wähle $w_1, \ldots, w_n$ so, dass eine große Zahl von Tests mit vorgegebenen Eingangsdaten $(x_1^{(j)}, \ldots, x_n^{(j)})^\top$ und bekannten Resultaten $y^{(j)}$ möglichst gut wiedergegeben wird:

$$\sum_{i=1}^{n} x_i^{(j)} w_i \approx y^{(j)}\,, \quad (\,j = 1, \ldots, m\,)$$

$\rightsquigarrow$ überbestimmtes lineares Gleichungssystem, Bestimmung von $(w_1, \ldots, w_n)$ als Kleinste–Quadrate–Lösung

praktisch Lösung der Normalgleichungen oder Lösung mittels QR–Zerlegung.

3 Rechnerarithmetik und Rundungsfehler

3.1 Gleitpunktarithmetik

Bemerkung 3.1 (Gleitpunktzahlen)

a) Ganzzahlige Datentypen (INTEGER) mit exakter Arithmetik

$$-\text{MaxInt} - 1\,, \ldots\,, -1,\, 0,\, 1,\, \ldots\,, \text{MaxInt}\,,$$

z. B. für Indizes in Laufanweisungen.

b) *Normalisierte Gleitpunktdarstellung* (engl.: floating point numbers) zur Darstellung reeller Zahlen

$$F := \{\, y \,:\, y = \pm\, m * \beta^{e-t} \,\} \subset \mathbb{R}$$

3. Rechnerarithmetik und Rundungsfehler (II)

Beispiel Schlecht konditioniertes Gleichungssystem

```
>> n = 8;                                % Dimension des linearen Gleichungssystems
>> a = hilb ( n );                       % n-reihige Hilbert-Matrix definieren
>> format short e, 1./a, format long e   % a_{ij} = 1/(i+j)

>> xsol = ones ( n, 1 );                 % Loesung des Gleichungssystems vorgeben
>> b = a * xsol;                         % Rechte Seite des Gleichungssystems vorgeben

>> xnum = a \ b;                         % Numerische Lösung (Gauss mit Pivotisierung)

>> xnum - xsol                           % Differenz von analytischer und numerischer Lösung
```

Ergebnisse

n	$\|x_{\text{num}} - x_{\text{sol}}\|_2$
2	$8.95_{\text{E}} - 16$
4	$3.74_{\text{E}} - 13$
6	$7.55_{\text{E}} - 11$
8	$2.87_{\text{E}} - 07$
10	$8.72_{\text{E}} - 04$
12	$2.86_{\text{E}} - 01$

Abbildung 3.2: Rechnen in Gleitpunktarithmetik: Beispiel Hilbert–Matrix.

mit β ... Basis (meist 2, 8 oder 16),
t ... Mantissenlänge,
e ... Exponent, $e_{\min} \le e \le e_{\max}$
m ... Mantisse (ganzzahlig), $m = 0$ oder $\beta^{t-1} \le m < \beta^t$

Schreibweise $y = \pm\, \beta^e * [0.d_1 d_2 \cdots d_t]_\beta := \pm\, \beta^e \Big(\frac{d_1}{\beta} + \frac{d_2}{\beta^2} + \ldots + \frac{d_t}{\beta^t} \Big)$

mit Mantisse $m = d_1 d_2 \cdots d_t$ oder $m = 0$.

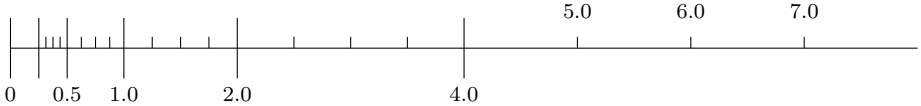
Beispiel $\beta = 2$, $t = 3$, $e_{\min} = -1$, $e_{\max} = 3$

$e{=}{-1}$ $e{=}0$ $e{=}3$

$[0.100]_2$ $[0.101]_2$ $[0.110]_2$ $[0.111]_2$ $[0.100]_2$ $[0.111]_2$

$$F \cap [0, \infty) = \{\, 0,\ 0.25,\ 0.3125,\ 0.3750,\ 0.4375,\ 0.5,\ 0.625,\ 0.750,\ 0.875,$$
$$1.0,\ 1.25,\ 1.50,\ 1.75,\ 2.0,\ 2.5,\ 3.0,\ 3.5,\ 4.0,\ 5.0,\ 6.0,\ 7.0 \,\}$$

Beachte: Gleitpunktzahlen sind auf der reellen Achse *nicht* gleichverteilt.



IEEE–Standard 754 [1985] Binäre Gleitpunktarithmetik, Quasi–Standard

einfache Genauigkeit (single precision)

4 Byte, $\beta = 2$, $t = 23$, $e_{\min} = -126$, $e_{\max} = 127$

Zahlenbereich: $[1.2_{\mathrm{E}} - 38, 3.4_{\mathrm{E}} + 38]$

doppelte Genauigkeit (double precision)

8 Byte, $\beta = 2$, $t = 52$, $e_{\min} = -1022$, $e_{\max} = 1023$

Zahlenbereich: $[2.2_{\mathrm{E}} - 308, 1.8_{\mathrm{E}} + 308]$

Abstand zweier positiver Gleitpunktzahlen x, $\tilde{x}$:

Maschinenepsilon $\texttt{eps} = \beta^{1-t}$ ... kleinste Maschinenzahl, die zu $1 = [0.10\cdots0]_\beta * \beta^1$ addiert einen von 1 verschiedenen Wert ergibt

Sind x, $\tilde{x}$ unmittelbar benachbart, so gilt

$$\frac{1}{\beta}\,\texttt{eps}\,|x| \le |x - \tilde{x}| \le \texttt{eps}\,|x|\,.$$

Bemerkung 3.2 (Rundung, Rundungsfehler)

a) Sei G die Menge aller y wie in Bemerkung 3.1, jedoch für beliebiges $e \in \mathbb{Z}$, und $\texttt{fl} : \mathbb{R} \to G$ eine Abbildung mit

$$|x - \texttt{fl}(x)| = \min_{\tilde{x}\in G} |x - \tilde{x}|\,, \quad (\,x \in \mathbb{R}\,).$$

Der Übergang $x \mapsto \texttt{fl}(x)$ heißt *runden*. fl ist nicht eindeutig, praktisch meist: *Gerade–Zahl–Regel*, d. h., für $\tilde{x}_1, \tilde{x}_2 \in G$ mit $\tilde{x}_1 \neq \tilde{x}_2$ und

$$|x - \tilde{x}_1| = |x - \tilde{x}_2| = \min_{\tilde{x}\in G} |x - \tilde{x}|$$

wählt man fl so, dass d_t geradzahlig.

b) praktisch $\texttt{fl}(x) \overset{!}{\in} F$

Exponentenüberlauf (engl.: overflow): $|\texttt{fl}(x)| > \max\{\,|y| \,:\, y \in F\,\}$

Exponentenunterlauf (engl.: underflow): $0 < |\texttt{fl}(x)| < \min\{\,|y| \,:\, y \in F\,,\ y \neq 0\,\}$

c) Zu jedem $x \in \mathbb{R}$ mit $\texttt{fl}(x) \in F$ ist

$\texttt{fl}(x) = x(1+\delta)$ mit einem δ mit $|\delta| < \varepsilon$

und

$\texttt{fl}(x) = x/(1+\bar{\delta})$ mit einem $\bar{\delta}$ mit $|\bar{\delta}| \le \varepsilon\,,$

wobei $\varepsilon := \frac{1}{2}\beta^{1-t}$ die *Maschinengenauigkeit* (engl.: *unit round-off*) bezeichnet:

$\varepsilon \approx 5.96_{\mathrm{E}} - 8$ (single), $\varepsilon \approx 1.11_{\mathrm{E}} - 16$ (double).

Begründung

Für $x > 0$ ist $x = \mu * \beta^{e-t}$ mit einem $\mu \in [\beta^{t-1}, \beta^t - 1]$ und $e \in [e_{\min}, e_{\max}]$. Unmittelbar benachbarte Gleitpunktzahlen: $\underline{\mu} * \beta^{e-t}$, $\bar{\mu} * \beta^{e-t}$ mit $\underline{\mu} \le \mu \le \bar{\mu}$. Es gilt

$$\begin{aligned} |x - \texttt{fl}(x)| &= \min\{\,|\mu - \underline{\mu}|, |\mu - \bar{\mu}|\,\} * \beta^{e-t} \\ &\le \frac{1}{2}\,|\bar{\mu} - \underline{\mu}| * \beta^{e-t} \le \frac{1}{2} * \beta^{e-t} = x \cdot \frac{1}{2\mu} \le x \cdot \varepsilon \end{aligned}$$

Bemerkung 3.3 (Absoluter und relativer Fehler)
a) Der *absolute Fehler* einer Größe mit Soll–Wert $\bar{\xi}$ und Ist–Wert ξ ist

$$\delta\xi := |\xi - \bar{\xi}|\,,$$

für $\bar{\xi} \neq 0$ ist der zugehörige *relative Fehler* $f_{\text{rel}}(\bar{\xi}) := \dfrac{|\xi - \bar{\xi}|}{|\bar{\xi}|}$.

b) Der in Bemerkung 3.2 betrachtete Rundungsfehler erfüllt

$$f_{\text{rel}}(x) = \frac{|\mathtt{fl}(x) - x|}{|x|} \leq \varepsilon\,.$$

Bemerkung 3.4 (Gleitpunktarithmetik)

Grundrechenarten $\mathtt{op} \in \{+, -, *, /\}$, $\mathtt{op} : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$

Problem F nicht abgeschlossen bez. $\mathtt{op}$

Anforderung („Standardmodell“)

$$\left.\begin{array}{rcl} x\ \widetilde{\mathtt{op}}\ y & = & (x\ \mathtt{op}\ y)(1+\delta) \\ x\ \widetilde{\mathtt{op}}\ y & = & \dfrac{x\ \mathtt{op}\ y}{1+\bar{\delta}} \end{array}\right\},\quad (x, y \in F)\,,$$

mit $\delta = \delta(x, y; \mathtt{op})$, $\bar{\delta} = \bar{\delta}(x, y; \mathtt{op})$, $|\delta| \leq \varepsilon$, $|\bar{\delta}| \leq \varepsilon$.

praktisch $x\ \widetilde{\mathtt{op}}\ y$ „unendlich genau“ (praktisch: „mit größerer Mantissenlänge“) auswerten, anschließend runden auf nächstgelegene Gleitpunktzahl (Gerade–Zahl–Regel).

Alternativen
- Runden auf nächst kleinere bzw. nächst größere Maschinenzahl $\rightsquigarrow$ Intervallarithmetik
- „Abschneiden“ überzähliger Ziffern („chopping“)

Beispiel $\beta = 2$, $t = 3$, $x = \dfrac{7}{4} = [0.111]_2 * 2^1$, $y = \dfrac{3}{8} = [0.110]_2 * 2^{-1}$

$$\begin{array}{rcl} x + y & = & [0.111]_2 * 2^1 + [0.110]_2 * 2^{-1} = [11.100]_2 * 2^{-1} + [0.110]_2 * 2^{-1} \\ & = & [100.010]_2 * 2^{-1} = [0.100010]_2 * 2^2 \\ x \mathbin{\widetilde{+}} y & = & [0.100]_2 * 2^2 = 2.00 \end{array}$$

absoluter Fehler: $\quad |\,2.00 - \dfrac{17}{8}\,| = \dfrac{1}{8}$

relativer Fehler: $\quad \dfrac{1}{8} : \dfrac{17}{8} \approx 6\%$

Maschinengenauigkeit: $\quad \varepsilon = 2^{-3} = \dfrac{1}{8} = 12.5\%$

Bemerkung 3.5 (Rundungsfehleranalyse: Beispiel Addition)

geg.: $a, b, c \in F$
ges.: $s = a + b + c$ in Gleitpunktarithmetik

$\underline{\widetilde{s} := (a \mathbin{\widetilde{+}} b) \mathbin{\widetilde{+}} c}$

$$\begin{aligned} \widetilde{s} &= ((a \mathbin{\widetilde{+}} b) + c)(1+\delta_2) = ((a+b)(1+\delta_1)+c)(1+\delta_2) \\ &= s + (a+b)\delta_1 + (a+b+c)\delta_2 + (a+b)\delta_1\delta_2 \doteq s + (a+b)\delta_1 + s\cdot\delta_2 \end{aligned}$$

mit $|\delta_1|, |\delta_2| \le \varepsilon$. Terme höherer Ordnung werden vernachlässigt („ $\doteq$ ").

$$f_{\text{rel}}(s) \doteq \Big| \delta_2 + \frac{a+b}{a+b+c}\,\delta_1 \Big| \le (\,1 + \Big|\frac{a+b}{a+b+c}\Big|)\,\varepsilon$$

Beachte Fehler in Zwischenergebnissen (δ_1) können verstärkt werden, ebenso auch Fehler in Ausgangsdaten.

kritisch $|a+b+c| \ll |a+b|$

$\underline{\widetilde{\widetilde{s}} := a \mathbin{\widetilde{+}} (b \mathbin{\widetilde{+}} c)}$

$$\widetilde{\widetilde{s}} \doteq s + (b+c)\delta_3 + s\cdot\delta_4 \quad \text{mit} \quad |\delta_3|, |\delta_4| \le \varepsilon\,,$$

$$f_{\text{rel}}(s) \doteq \Big| \delta_4 + \frac{b+c}{a+b+c}\,\delta_3 \Big| \le (\,1 + \Big|\frac{b+c}{a+b+c}\Big|)\,\varepsilon$$

Beachte Gleitpunktoperationen sind in der Regel weder assoziativ noch kommutativ.

Beispiel 3.6 (Addition in Gleitpunktarithmetik)

geg.: $\beta = 2\,,\ t = 3$

$$a = [0.111]_2 * 2^0 = \frac{7}{8}\,,\ \ b = -[0.110]_2 * 2^0 = -\frac{6}{8}\,,\ \ c = [0.110]_2 * 2^{-2} = \frac{3}{16} \in F$$

$$\begin{aligned} \widetilde{s} &= (\,[0.111]_2 * 2^0 \ \widetilde{-}\ [0.110]_2 * 2^0\,) \ \widetilde{+}\ [0.110]_2 * 2^{-2} \\ &= [0.001]_2 * 2^0 \ \widetilde{+}\ [0.110]_2 * 2^{-2} = [0.100]_2 * 2^{-2} \ \widetilde{+}\ [0.110]_2 * 2^{-2} \\ &= [1.01]_2 * 2^{-2} = [0.101]_2 * 2^{-1} = \frac{5}{16} = s\,, \ \text{exaktes Ergebnis} \end{aligned}$$

$$\begin{aligned} \widetilde{\widetilde{s}} &= [0.111]_2 * 2^0 \ \widetilde{+}\ (\,-[0.110]_2 * 2^0 \ \widetilde{+}\ [0.110]_2 * 2^{-2}\,) \\ &= [0.111]_2 * 2^0 \ \widetilde{-}\ [0.100]_2 * 2^0 = [0.011]_2 * 2^0 = [0.110]_2 * 2^{-1} = \frac{3}{8} \end{aligned}$$

Ergebnis: $|\widetilde{\widetilde{s}} - s| = \dfrac{1}{16}$, relativer Fehler 20%

Relativer Fehler in $b \mathbin{\widetilde{+}} c$: $\dfrac{1}{8} = 12.5\%$

Verstärkung im Endergebnis $\widetilde{\widetilde{s}}$ wegen $\left|\dfrac{b+c}{a+b+c}\right| = \dfrac{9}{5} = 1.8$

Bemerkung 3.7 (Auslöschung)

Problem Subtraktion annähernd gleich großer Zahlen in Gleitpunktarithmetik
geg.: $a, b \in \mathbb{R}$, ges.: $a - b$

$$\widetilde{a} := \mathtt{fl}(a) = a\,(1+\delta_a)\,, \quad \widetilde{b} := \mathtt{fl}(b) = b\,(1+\delta_b) \quad \text{mit } |\delta_a|,\, |\delta_b| \le \varepsilon$$

$$\begin{aligned} \mathtt{fl}(a-b) &= \widetilde{a} \mathbin{\widetilde{-}} \widetilde{b} = (\widetilde{a} - \widetilde{b})\,(1+\delta_-) \quad \text{mit } |\delta_-| \le \varepsilon \\ &\doteq a - b + \delta_- \cdot (a-b) + \delta_a \cdot a - \delta_b \cdot b \end{aligned}$$

$$f_{\mathrm{rel}}(a-b) \doteq \left| \frac{a}{a-b}\,\delta_a - \frac{b}{a-b}\,\delta_b + \delta_- \right| \le \left(1 + \frac{|a|+|b|}{|a-b|}\right)\varepsilon$$

Relative Fehler δ_a, δ_b in den Ausgangsdaten können drastisch verstärkt werden, falls $|a-b| \ll |a|,\, |b|$, insbesondere für $a \approx b$.

Beispiel $\beta = 2$, $a = \dfrac{3}{5}$, $b = \dfrac{4}{7}$

$t = 5$ $\widetilde{a} = [0.10011]_2 * 2^0$, $\widetilde{b} = [0.10010]_2 * 2^0$

$\delta_a \approx 0.010$, $\delta_b \approx 0.016$, $\varepsilon \approx 0.031$

$\widetilde{a} \mathbin{\widetilde{-}} \widetilde{b} = [0.10000]_2 * 2^{-4} = \dfrac{1}{32}$

absoluter Fehler: $\dfrac{1}{32} - \dfrac{1}{35}$, relativer Fehler: 8.6%

$t = 3$ $\widetilde{a} = \widetilde{b} = [0.100]_2 * 2^0$

$\delta_a \approx 0.042$, $\delta_b \approx 0.094$, $\varepsilon = 0.125$

$\widetilde{a} \mathbin{\widetilde{-}} \widetilde{b} = 0$, relativer Fehler: 100%

Führende Ziffern $[0.1001\cdots]_2$ in a und b sind gleich und werden bei Subtraktion „ausgelöscht“ $\Rightarrow$ „Auslöschung“.

Faustregel Vermeide – falls möglich – die Subtraktion annähernd gleich großer Zahlen in numerischen Algorithmen.

Strategien und Tricks

a) Unvermeidbare Subtraktionen annähernd gleich großer Zahlen möglichst an den Anfang des Algorithmus stellen.

b) Konjugierte Wurzelausdrücke

- $$\begin{aligned}\sqrt{1+x}-\sqrt{1-x} &= \frac{(\sqrt{1+x}-\sqrt{1-x})(\sqrt{1+x}+\sqrt{1-x})}{\sqrt{1+x}+\sqrt{1-x}} \\ &= \frac{2x}{\sqrt{1+x}+\sqrt{1-x}}\,, \quad |x| \ll 1\end{aligned}$$

- $x^2+px+q=0 \;\Rightarrow\; x_{1,2} = -\frac{p}{2} \pm \sqrt{\frac{p^2}{4}-q}$

  Auslöschung für $|q| \ll 1$, deshalb für $p \neq 0$

$$x_1 := -\frac{p}{2} - \operatorname{sgn}(p)\sqrt{\frac{p^2}{4}-q} \quad \text{mit} \quad \operatorname{sgn}(p) := \begin{cases} 1 & \text{für } p>0\,, \\ 0 & \text{für } p=0\,, \\ -1 & \text{für } p<0\,, \end{cases}$$

$$x_2 := \frac{q}{x_1} \qquad \text{(VIETAscher Wurzelsatz)}$$

c) Analytische Umformungen, z. B. Reihenentwicklungen (vgl. Abb. 3.3)

$$\frac{1-\cos x}{x} = \frac{1}{x}\Big(1-(1-\frac{x^2}{2}+\frac{x^4}{24}\mp\ \dots)\Big) = \frac{x}{2}(1-\frac{x^2}{12}\pm\ \dots)$$

Fehler der Approximation $\frac{1-\cos x}{x} \approx \frac{x}{2}$ betragsmäßig beschränkt durch $\frac{|x|}{2}\cdot\frac{x^2}{12}$

(Reihenrest alternierender Reihen, Satz von Leibniz)

3.2 Vektor- und Matrixnormen

Definition 3.8 (Vektornorm)

Eine Abbildung $\|.\| : \mathbb{R}^n \to \mathbb{R}$ *heißt* Vektornorm auf $\mathbb{R}^n$, *falls*

1. $\|x\| \geq 0\,,\ (x \in \mathbb{R}^n)$ *und* $\big(\|x\| = 0 \;\Leftrightarrow\; x = 0\big)$ *(Positivität),*
2. $\|\alpha x\| = |\alpha|\,\|x\|\,,\ (\alpha \in \mathbb{R},\ x \in \mathbb{R}^n)$ *(Homogenität),*
3. $\|x+y\| \leq \|x\| + \|y\|\,,\ (x, y \in \mathbb{R}^n)$ *(Dreiecksungleichung).*

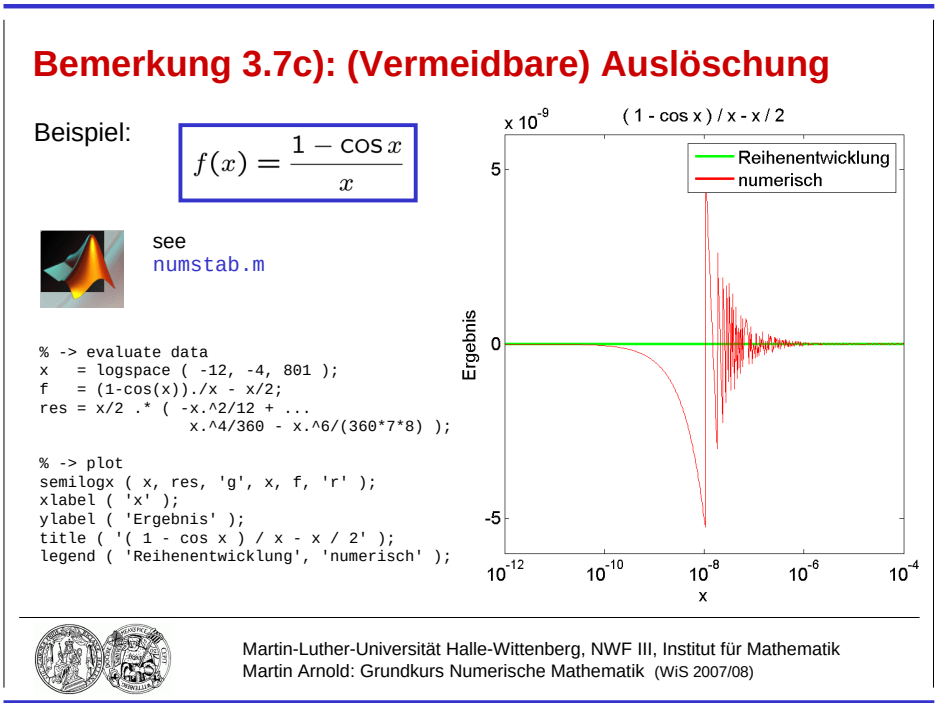


Abbildung 3.3: Analytische Umformungen zur Vermeidung von Auslöschung.

Beispiel 3.9 (Vektornorm)

a) $\|x\|_2 := \sqrt{\sum_{i=1}^{n} x_i^2}$ ... Euklidische Vektornorm

$\|x\|_1 := \sum_{i=1}^{n} |x_i|$ ... 1–Norm

$\|x\|_\infty := \max_{i=1,\ldots,n} |x_i|$ ... Maximumnorm, ∞–Norm

b) Kugeln im $\mathbb{R}^n$: $\{ x : \|x\| \leq 1 \}$, vgl. Abb. 3.4.

Bemerkung 3.10 (Eigenschaften von Vektornormen)

a) Jedes Skalarprodukt $\langle .,. \rangle$ in $\mathbb{R}^n$ erzeugt eine Vektornorm in $\mathbb{R}^n$:

$$\|x\| := \sqrt{\langle x, x \rangle}$$

mit $|\langle x, y \rangle| \leq \|x\| \cdot \|y\|$, $(x, y \in \mathbb{R}^n)$ ... Cauchy–Schwarzsche Ungleichung.

b) Auf $\mathbb{R}^n$ sind sämtliche Vektornormen *äquivalent*, d. h., zu beliebig vorgegebenen Vektornormen $\|.\|_p$, $\|.\|_q$ gibt es Konstanten $\underline{c}, \overline{c} > 0$ mit

$$\underline{c}\|x\|_q \leq \|x\|_p \leq \overline{c}\|x\|_q, \quad (x \in \mathbb{R}^n).$$

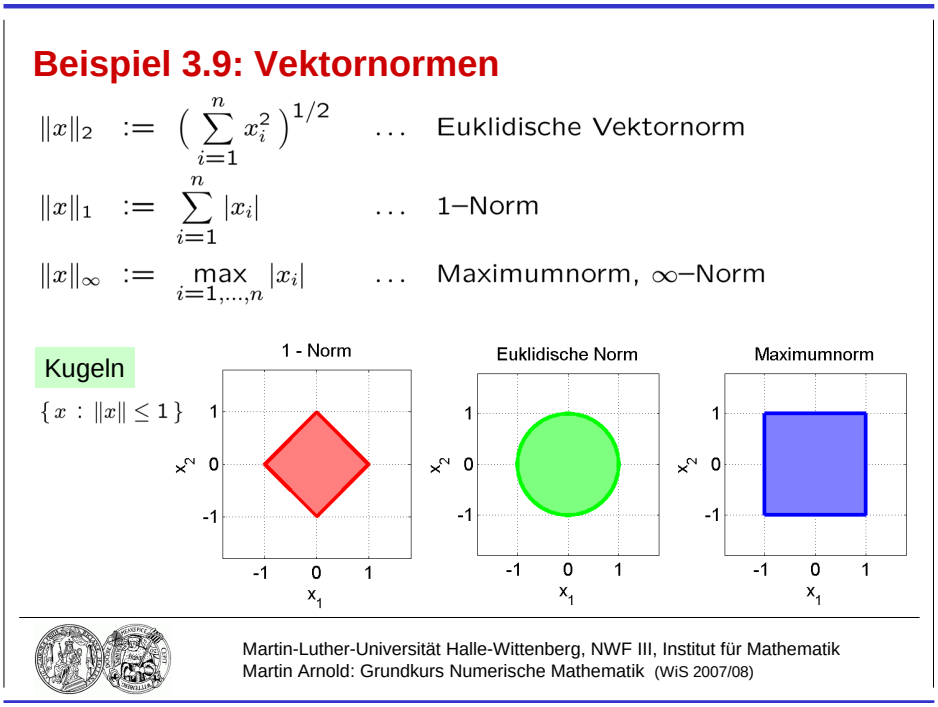


Abbildung 3.4: Einheitskugeln im $\mathbb{R}^2$.

Definition 3.11 (Matrixnorm)

a) Eine Abbildung $\|.\| : \mathbb{R}^{m\times n} \to \mathbb{R}$ *heißt* Matrixnorm, *falls*

1. $\|A\| \geq 0\,,\ (A \in \mathbb{R}^{m\times n})$ *und* $\big(\|A\| = 0 \ \Leftrightarrow\ A = 0\big)$ *(Positivität),*
2. $\|\alpha A\| = |\alpha| \cdot \|A\|\,,\ (\alpha \in \mathbb{R},\ A \in \mathbb{R}^{m\times n})$ *(Homogenität),*
3. $\|A + B\| \leq \|A\| + \|B\|\,,\ (A, B \in \mathbb{R}^{m\times n})$ *(Dreiecksungleichung).*

b) Eine Matrixnorm $\|.\|$ *heißt* submultiplikativ, *falls*

$$\|AB\| \leq \|A\| \cdot \|B\|\,,\quad (A \in \mathbb{R}^{m\times n}\,,\ B \in \mathbb{R}^{n\times p})\,.$$

c) Eine submultiplikative Matrixnorm $\|.\|$ *heißt* verträglich *(auch:* konsistent*) mit einer vorgegebenen Vektornorm* $\|.\|$*, falls*

$$\|Ax\| \leq \|A\| \cdot \|x\|\,,\quad (A \in \mathbb{R}^{m\times n}\,,\ x \in \mathbb{R}^n)\,.$$

Beispiel 3.12 (Frobeniusnorm)

$$\|A\|_{\mathrm{F}} := \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}^2} \qquad \ldots \text{ Frobeniusnorm}$$

- Submultiplikative Matrixnorm, verträglich mit $\|.\|_2$:

$$\|Ax\|_2^2 = \sum_{i=1}^{m}\Big(\sum_{j=1}^{n} a_{ij}x_j\Big)^2 \leq \sum_{i=1}^{m}\Big(\sum_{j=1}^{n} a_{ij}^2\Big)\Big(\sum_{j=1}^{n} x_j^2\Big) = \|A\|_{\mathrm{F}}^2 \cdot \|x\|_2^2$$

- $\|I_n\| = \sqrt{n}$

Satz 3.13 (Zugeordnete Matrixnorm)
Zu einer vorgegebenen Vektornorm $\|.\|$ wird durch

$$A \mapsto \|A\| := \sup_{x \neq 0} \frac{\|Ax\|}{\|x\|} = \sup_{\|x\|=1} \|Ax\|$$

eine submultiplikative, mit $\|.\|$ verträgliche Matrixnorm definiert, die der Vektornorm $\|.\|$ zugeordnete Matrixnorm. *Es gilt* $\|I_n\| = 1$.

Beweis vgl. Huckle/Schneider, Anhang B.2, z. B.

$$\|I_n\| = \sup_{x \neq 0} \frac{\|I_n x\|}{\|x\|} = 1 \,. \quad \blacksquare$$

Beispiel 3.14 (Zugeordnete Matrixnorm)
a) *Zeilensummennorm*

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

ist $\|x\|_\infty$ zugeordnet, denn

- $\|Ax\|_\infty = \max\limits_{i=1,\ldots,m} \left| \sum\limits_{j=1}^{n} a_{ij} x_j \right| \leq \max\limits_{i=1,\ldots,m} \sum\limits_{j=1}^{n} |a_{ij}| \, |x_j| \leq \|A\|_\infty \cdot \|x\|_\infty \,,$

- zu einem $i_0 \in \{\, 1, \ldots, m \,\}$ mit $\sum\limits_{j=1}^{n} |a_{i_0,j}| = \max\limits_{i=1,\ldots,m} \sum\limits_{j=1}^{n} |a_{ij}|$ wählt man $x \in \mathbb{R}^n$ so, dass $\|x\|_\infty = 1$ und $x_j = 1$, falls $a_{i_0,j} > 0$, $x_j = -1$, falls $a_{i_0,j} < 0$

$$\Rightarrow \quad \|Ax\|_\infty \geq \sum_{j=1}^{n} |a_{i_0,j} x_j| = \sum_{j=1}^{n} |a_{i_0,j}| = \|A\|_\infty \,.$$

b) *Spaltensummennorm*

$$\|A\|_1 := \max_{j=1,\ldots,n} \sum_{i=1}^{m} |a_{ij}| \quad \text{ist } \|x\|_1 \text{ zugeordnet.}$$

c) *Spektralnorm*

$$\|A\|_2 := \max_{i=1,\ldots,n} \sqrt{\lambda_i(A^\top A)}$$

Für orthogonale Matrizen $U \in \mathbb{R}^{n \times n}$, also $U^\top U = I_n$, ist $\lambda_i(U^\top U) = 1$ und $\|U\|_2 = 1$.

3.3 Kondition und Stabilität

Bemerkung 3.15 (Eingabefehler und Fehler im Ergebnis)

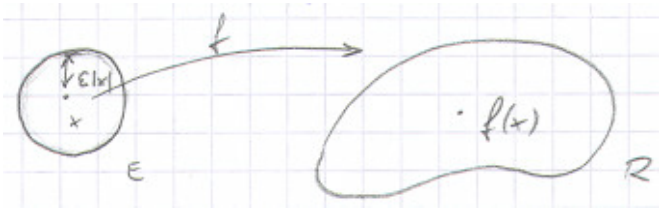
analytisch Eingabe x $\rightarrow$ Algorithmus / Berechnungsvorschrift
$\rightarrow$ Resultat $f(x)$

numerisch Fehler im Resultat entstehen durch
– Eingabefehler
– Fehler im Algorithmus

Numerische Eingabe $x \in F$ repräsentiert

Eingabemenge $E = \{\, \tilde{x} \in \mathbb{R} : \mathtt{fl}(\tilde{x}) = x \,\}$

Resultatmenge $R = f(E) := \{\, f(\tilde{x}) : \tilde{x} \in E \,\}$



„Kondition“ Maß für das Verhältnis von R zu E

Ziel Fehler in der Berechnungsvorschrift sollen Menge R nicht deutlich vergrößern

Fehler im Ergebnis $y = f(x)$:

$$\delta_y := f(x + \delta_x) - f(x) \doteq f'(x)\,\delta_x$$

Relativer Fehler:

$$\frac{\|\delta_y\|}{\|y\|} \doteq \frac{\|f'(x)\,\delta_x\|}{\|y\|} \leq \frac{\|x\|\,\|f'(x)\|}{\|y\|} \cdot \frac{\|\delta_x\|}{\|x\|}$$

Definition 3.16 (Konditionszahl)
Zu einem Problem $x \mapsto f(x)$ *heißt*

$$\mathrm{cond}_x := \frac{\|x\|\,\|f'(x)\|}{\|f(x)\|}$$

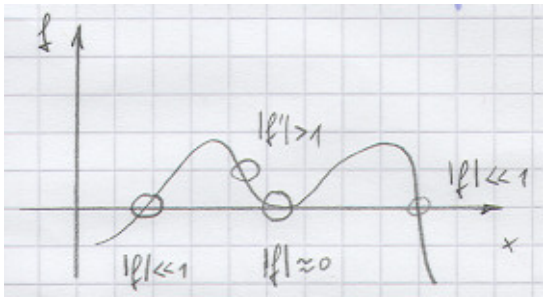
Konditionszahl. *Das Problem ist* gut konditioniert, *wenn* cond_x *klein ist, und* schlecht konditioniert *für große Konditionszahlen* cond_x.

Beispiel 3.17 (Kondition)

Exponentialfunktion $x \mapsto \mathrm{e}^x$, $\mathrm{cond}_x = \left|\frac{x\,\mathrm{e}^x}{\mathrm{e}^x}\right| = |x|$, gut konditioniert für $|x| \overset{<}{\approx} 1$.

Logarithmus $x \mapsto \ln x$, $\mathrm{cond}_x = \left|\frac{x \cdot \frac{1}{x}}{\ln x}\right| = \frac{1}{|\ln x|}$, sehr schlecht konditioniert für $x \approx 1$.

Gute / schlechte Kondition



Polynomnullstellen häufig schlecht konditioniertes Problem

Beispiel: $\pi(t) = t^4 - 8t^3 + 24t^2 - 32t + 15.999\,999\,99 = (t-2)^4 - 10^{-8}$

$t_{1,2} = 2 \pm 0.01\,, \quad t_{3,4} = 2 \pm 0.01i$

Relativer Fehler bei Darstellung von $15.999\,999\,99$ durch 16.0: $\varepsilon_x = 6.25 \cdot 10^{-10}$, z. B. bei Maschinengenauigkeit $\varepsilon = 8 \cdot 10^{-10} \;\Rightarrow\; t_{1,2,3,4} = 2.0$

$$\mathrm{cond}_{t_{1,2}} = \frac{0.01/2.01}{6.25 \cdot 10^{-10}} \approx 8.0 \cdot 10^6$$

Bemerkung 3.18 (Berechnungsvorschrift)

Zum mathematischen Problem $x \mapsto f(x)$ sei die Abbildung $x \mapsto \tilde{f}(x)$ gegeben zur Berechnung von $f(x)$ in Gleitpunktarithmetik (u. a. auch Reihenfolge der Rechenoperationen festgelegt) $\rightsquigarrow$ *Berechnungsvorschrift*

Beispiel $f(x) = 1 - \sqrt{1-x^2}\,,$ für $|x| \ll 1$ gut konditioniert, $\mathrm{cond}_x \approx 2\,.$

Berechnungsvorschrift 1: $\tilde{f}(x) := 1 - \left(\sqrt{1-(x^2)}\right)$

Berechnungsvorschrift 2: $\tilde{f}(x) := \dfrac{(x^2)}{\left(1 + \left(\sqrt{1-(x^2)}\right)\right)}$

Definition 3.19 (Numerische Stabilität)

Zu einem gut konditionierten Problem $x \mapsto f(x)$ heißt eine Berechnungsvorschrift $x \mapsto \tilde{f}(x)$ numerisch stabil*, wenn die relativen Eingabefehler durch die Berechnungsvorschrift nicht vergrößert werden, und* numerisch instabil *sonst.*

Bemerkung 3.20 (Lineare Gleichungssysteme: Kondition und Stabilität)

a) Betrachte zu gegebener regulärer Matrix $A \in \mathbb{R}^{n\times n}$ und gegebenem $b \in \mathbb{R}^n$ die Lösung

des linearen Gleichungssystems $Ax = b$ als Abbildung $b \mapsto x := A^{-1}b$ mit gestörten Eingangsdaten b und exakten Matrizen A, A^{-1} $\Rightarrow$

$$A(x + \delta_x) = b + \delta_b\,, \quad Ax = b$$
$$\|\delta_x\| = \|A^{-1}\delta_b\| \le \|A^{-1}\|\,\|\delta_b\|$$
$$\|b\| = \|Ax\| \le \|A\|\,\|x\|$$

Ergebnis $\dfrac{\|\delta_x\|}{\|x\|} \le \|A\| \cdot \|A^{-1}\| \cdot \dfrac{\|\delta_b\|}{\|b\|}$

- Empfindlichkeit der Lösung gegenüber Störungen in den Eingangsdaten wird beschrieben durch die *Konditionszahl* $\mathrm{cond}(A) := \|A\| \cdot \|A^{-1}\|$.
- Ergebnis lässt sich übertragen auf Empfindlichkeit gegenüber Störungen in A.
- Wegen $1 \le \|I_n\| = \|A \cdot A^{-1}\| \le \|A\| \cdot \|A^{-1}\|$ gilt stets $\mathrm{cond}(A) \ge 1$.

  Gut konditioniert: $\mathrm{cond}(A) \lessapprox 10^3$

  Schlecht konditioniert: $\mathrm{cond}(A) \gg 10^6$

Beispiel 1

$$A = \begin{pmatrix} 1 & 0 \\ 0 & \varepsilon \end{pmatrix}, \quad A^{-1} = \begin{pmatrix} 1 & 0 \\ 0 & 1/\varepsilon \end{pmatrix} \quad \text{mit} \quad 0 < \varepsilon \ll 1$$

$\Rightarrow$ $\mathrm{cond}_2(A) = 1 \cdot \dfrac{1}{\varepsilon} = \dfrac{1}{\varepsilon} \to \infty$ für $\varepsilon \to 0$, Fehlerverstärkung um Faktor $1/\varepsilon$ möglich

Beispiel 2

Hilbert–Matrizen $H^{(n)} = (h_{ij}^{(n)})_{i,j} \in \mathbb{R}^{n\times n}$ mit $h_{ij}^{(n)} := 1/(i+j-1)$, $(i, j = 1, \ldots, n)$.

Zu $b^{(n)} \in \mathbb{R}^n$, $b^{(n)} = (b_i^{(n)})_i$ mit $b_i^{(n)} := \displaystyle\sum_{j=1}^{n} \frac{1}{i+j-1}$ ist die Lösung $x^{(n)}$ des linearen Gleichungssystems $H^{(n)}x^{(n)} = b^{(n)}$ gegeben durch $x^{(n)} = (1, 1, \ldots, 1)^\top$.

Fehler der mit Matlab ($\varepsilon = 1.1_{\mathrm{E}} - 16$) berechneten Lösung $\tilde{x}^{(n)}$:

n	$\|\tilde{x}^{(n)} - x^{(n)}\|_2$	$\mathrm{cond}_2(H^{(n)})$
2	9.0e-16	1.9e+01
4	4.6e-13	1.6e+04
6	3.5e-10	1.5e+07
8	1.3e-08	1.5e+10
10	3.0e-04	1.6e+13
12	1.6e+00	1.7e+16

b) Für orthogonale Matrizen $Q \in \mathbb{R}^{n\times n}$ ist $Q^{-1} = Q^\top$ und $\|Q\|_2 = \|Q^\top\|_2 = 1$

$\Rightarrow$ $\mathrm{cond}_2(Q) = 1$. Operationen mit orthogonalen Matrizen lassen die Kondition einer Matrix unverändert: $A = QR$ $\Rightarrow$ $\mathrm{cond}_2(R) = \mathrm{cond}_2(A)$.

c) Realisierung des Gauß–Algorithmus in Gleitpunktarithmetik:

Fehlerschranke hängt linear ab von $\max\limits_{i,k} |l_{ik}|$.

Spaltenpivotisierung: $|l_{ik}| \leq 1 \;\rightsquigarrow$ kleine Fehlerschranke

Numerische Stabilität: numerische Lösung $\tilde{x}$ erfüllt

$$(A + \delta_A)\tilde{x} = b$$

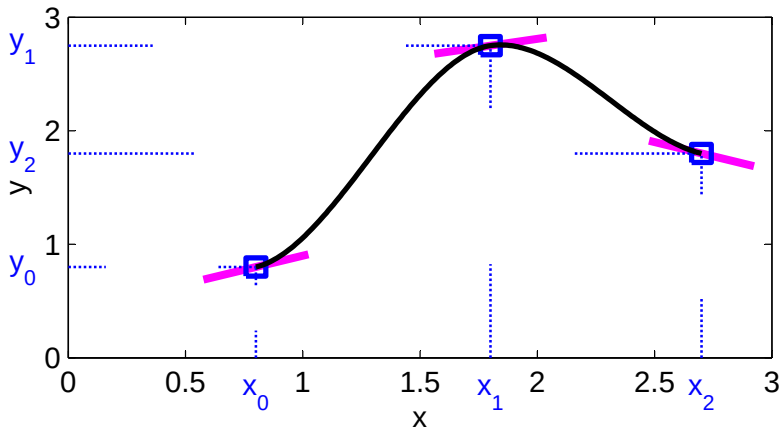
mit

$$\frac{\|\delta_A\|_\infty}{\|A\|_\infty} \lesssim 8n^3 \cdot \frac{\max_{i,j,k} |a_{ij}^{(k)}|}{\max_{i,j} |a_{ij}|} \varepsilon \,.$$

4 Interpolation (II)

Bemerkung 4.1 (Stückweise Hermite–Interpolation)

geg.: $r+1$ Stützstellen $x_0, x_1, \ldots, x_r$
Stützwerte $(y_k, y_k')\,,\ (k = 0, 1, \ldots, r\,)$



Definiert man die interpolierende Funktion Φ stückweise durch Hermite–Interpolationspolynome $\Phi\big|_{[x_{i-1},x_i]}$, $(i = 1, \ldots, r\,)$ mit Interpolationsbedingungen

$$\Phi(x_{i-1}) = y_{i-1}\,,\quad \Phi'(x_{i-1}) = y'_{i-1}\,,\quad \Phi(x_i) = y_i\,,\quad \Phi'(x_i) = y'_i\,,\quad (i = 1, \ldots, r\,)\,,$$

so ist $\Phi \in C^1[a,b]$, aber $\deg \Phi\big|_{[x_{i-1},x_i]} \leq 3$.

4.1 Spline–Interpolation

Bemerkung 4.2 (Kubische Spline–Interpolation)

Kubische Splines erreichen ähnlich wie zusammengesetzte Hermite–Interpolierende eine hohe globale Glattheit, jedoch mit deutlich niedrigerem Polynomgrad:

$$s \in C^2[a,b]\,,\quad s\big|_{[x_i,x_{i+1}]} \in \Pi_3\,.$$

Splines der *Ordnung* k: $s \in C^{k-2}[a,b]\,,\quad s\big|_{[x_i,x_{i+1}]} \in \Pi_{k-1}\,.$

Splinegitter $a = x_0 < x_1 < \ \ldots \ < x_n = b\,.$

Zum kubischen Spline s ist

$$s\big|_{[x_i,x_{i+1}]} = a_i + b_i(x-x_i) + \frac{c_i}{2}(x-x_i)^2 + \frac{d_i}{6}(x-x_i)^3\,, \quad (\,i = 0,1,\ldots,n-1\,)$$

$\Rightarrow$ insgesamt $4n$ Parameter (a_i, b_i, c_i, d_i), $(\,i = 0,1,\ldots,n-1\,)$.

$n+1$ Interpolationsbedingungen $\quad s(x_i) = y_i \quad \Rightarrow \quad a_i = y_i\,,\ (\,i = 0,1,\ldots,n-1\,)\,,$

$$s(x_n) = y_n \quad \Rightarrow \quad a_{n-1} + b_{n-1}(x_n - x_{n-1}) + \frac{c_{n-1}}{2}(x_n - x_{n-1})^2 + \frac{d_{n-1}}{6}(x_n - x_{n-1})^3 = y_n =: a_n\,.$$

$3(n-1)$ Stetigkeitsbedingungen

$$s(x_{i+1}-0) = s(x_{i+1}+0)\ : \qquad a_i + b_i h_i + \frac{c_i}{2}h_i^2 + \frac{d_i}{6}h_i^3 \ = \ a_{i+1}\,,\ (\,i = 0,1,\ldots,n-2\,)$$

$$s'(x_{i+1}-0) = s'(x_{i+1}+0)\ : \qquad b_i + c_i h_i + \frac{d_i}{2}h_i^2 \ = \ b_{i+1}\,,\ (\,i = 0,1,\ldots,n-2\,)$$

$$s''(x_{i+1}-0) = s''(x_{i+1}+0)\colon \qquad c_i + d_i h_i \ = \ c_{i+1}\,,\ (\,i = 0,1,\ldots,n-2\,)$$

mit *Schrittweiten* $h_i := x_{i+1} - x_i\,,\ (\,i = 0,1,\ldots,n-1\,)\,.$

$\Rightarrow$ insgesamt $4n-2$ lineare Bedingungen an $4n$ Parameter

Zusatzbedingungen

(i) $\quad s''(x_0) = s''(x_n) = 0 \quad \ldots$ natürlicher kubischer Spline

oder

(ii) $\quad s'(x_0) = y_0'\,,\ s'(x_n) = y_n \quad \ldots$ vollständiger kubischer Spline

oder

(iii) $\quad s'(x_0) = s'(x_n)\,,\ s''(x_0) = s''(x_n) \quad \ldots$ periodischer kubischer Spline, für periodische Daten $(\,y_0 = y_n\,) \ \Rightarrow \ \ s(x_0) = s(x_n)\,.$

In jedem der drei Fälle ist die Splinefunktion eindeutig bestimmt.

Berechnung der Koeffizienten

$$d_i \ = \ \frac{c_{i+1} - c_i}{h_i}\,,$$

$$b_i \ = \ \frac{a_{i+1} - a_i}{h_i} - \frac{c_i}{2}h_i - \frac{d_i}{6}h_i^2 \ = \ \frac{y_{i+1} - y_i}{h_i} - \frac{2c_i + c_{i+1}}{6}h_i$$

Stetigkeitsbedingung für $s'(x) \ \Rightarrow \ (\,i = 0,1,\ldots,n-2\,)$

$$\frac{y_{i+1} - y_i}{h_i} - \frac{2c_i + c_{i+1}}{6}h_i + c_i h_i + \frac{c_{i+1} - c_i}{2}h_i \ = \ \frac{y_{i+2} - y_{i+1}}{h_{i+1}} - \frac{2c_{i+1} + c_{i+2}}{6}h_{i+1}$$

$$\frac{h_i}{6}c_i + \frac{h_i + h_{i+1}}{3}c_{i+1} + \frac{h_{i+1}}{6}c_{i+2} \ = \ \frac{y_{i+2} - y_{i+1}}{h_{i+1}} - \frac{y_{i+1} - y_i}{h_i}$$

Zusammen mit $c_0 = c_n = 0$ ergibt sich für den natürlichen kubischen Spline ein tridiagonales lineares Gleichungssystem der Dimension $n-1$ zur Bestimmung von $c_1, \ldots, c_{n-1}$
$\Rightarrow$ Gaußscher Algorithmus erfordert $\mathcal{O}(n)$ Rechenoperationen. Analoges Vorgehen für vollständigen und periodischen Spline.

Algorithmus 1 Bestimmung der Splinekoeffizienten.

1. $a_i := y_i\,,\ (\,i = 0, 1, \ldots, n-1\,)$
2. Berechne $c_0, c_1, \ldots, c_n$ als Lösung eines tridiagonalen Gleichungssystems.
3. Bestimme $b_i, d_i\,,\ (\,i = 0, 1, \ldots, n-1\,)\,.$

Algorithmus 2 Auswertung der Splinefunktion.

1. Bestimme Teilintervall (binäre Suche)

$\underline{i} := 0\,,\ \overline{i} := n$
`repeat`
$\quad i_* := \left[\ \dfrac{\underline{i} + \overline{i}}{2}\ \right]$
$\quad$ `if` $x \geq x_{i_*}$ `then` $\underline{i} := i_*$ `else` $\overline{i} := i_*$
`until` $\overline{i} - \underline{i} \leq 1$
$i := \underline{i}$

Einfachster Spezialfall: äquidistantes Gitter $x_i = a + ih$ mit $h = \dfrac{b-a}{n}$

$\Rightarrow \quad i := \left[\ \dfrac{x-a}{h}\ \right]$

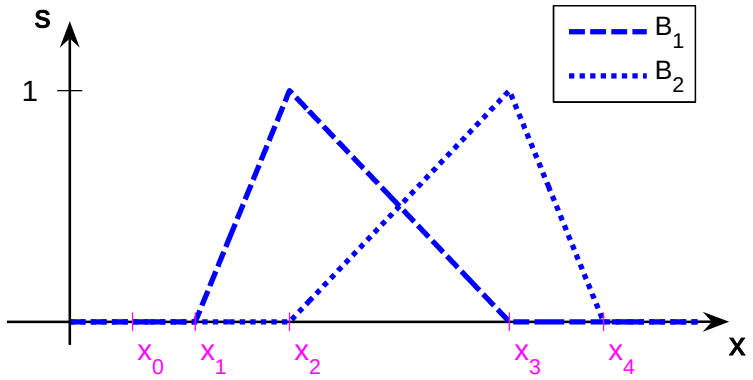
2. Splineauswertung $s(x) = a_i + (x - x_i)\big(b_i + (x - x_i)\big(\,\dfrac{1}{2}c_i + \dfrac{1}{6}d_i(x - x_i)\,\big)\big)$

Bemerkung 4.3 (B–Splines)

Idee Darstellung der Splinefunktion als Linearkombination „einfacher“ Basisfunktionen des Vektorraums der Splinefunktionen $\rightsquigarrow$ B–Splines.

Beispiel $k = 2:$ stetige, stückweise lineare Funktion

$$B_j(x) = \begin{cases} \dfrac{x - x_j}{x_{j+1} - x_j}\,, & (\,x \in [x_j, x_{j+1}]\,), \\ \dfrac{x - x_{j+2}}{x_{j+1} - x_{j+2}}\,, & (\,x \in [x_{j+1}, x_{j+2}]\,), \\ 0 & \text{sonst.} \end{cases}$$

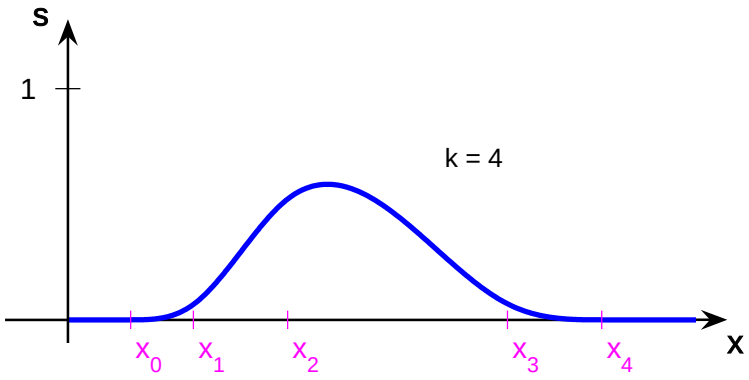


Interpolierender linearer Spline $s(x) = \sum_{j=0}^{n} y_j B_{j-1}(x)$.

allgemein

- $B_j\big|_{[x_i,x_{i+1}]} \in \Pi_{k-1}$, $(i = 0, 1, \ldots, n-1)$
- $B_j \in C^{k-2}[a, b]$
- $\sum_j B_j(x) = 1$, $(x \in [a, b])$
- $\operatorname{supp} B_j = [x_j, x_{j+k}]$, d. h. $B_j(x) = 0$, $(x \le x_j$ oder $x \ge x_{j+k})$

Beispiel Kubischer B–Spline



Bestimmung der Koeffizienten α_j des interpolierenden Splines $\sum_j \alpha_j B_j(x)$ als Lösung eines linearen Gleichungssystems der Bandbreite $k - 1$.

Satz 4.4 (Approximationseigenschaften kubischer Splines)
Gegeben sei eine Funktion $f \in C^4[a, b]$ *mit* $\max_{a \le x \le b} |f^{(4)}(x)| \le M$ *sowie ein Gitter*

$$\Delta = \{\, a = x_0 < x_1 < \ldots < x_n = b \,\}$$

mit Schrittweiten $h_i := x_{i+1} - x_i$, $(i = 0, 1, \ldots, n-1)$ *und einer Konstanten*

$$K \ge \max_{0 \le i \le n-1} h_i \;/\; \min_{0 \le i \le n-1} h_i \,.$$

Dann gibt es zum vollständigen interpolierenden kubischen Spline s_Δ Konstanten C_0, C_1, C_2 und C_3, die von Δ und K unabhängig sind und für die gilt

$$|f^{(k)}(x) - s_\Delta^{(k)}(x)| \le C_k M K \big(\max_{0 \le i \le n-1} h_i \big)^{4-k}, \quad (x \in [a,b],\ k = 0,1,2,3)$$

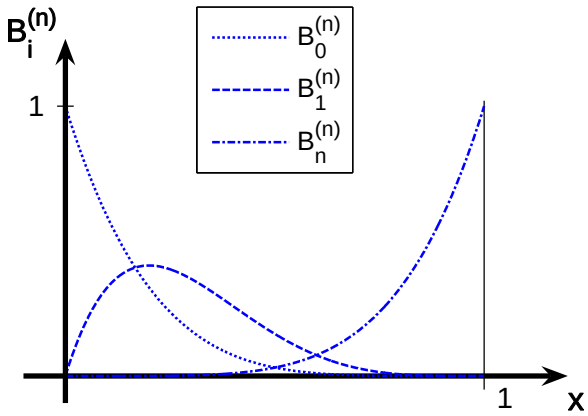
in jedem Punkt x, in dem $s_\Delta^{(k)}(x)$ definiert ist.

Beweisidee (i) $f(x_i) = s_\Delta(x_i)$ für Stützstellen x_i
(ii) Abschätzung von $|f''(x_i) - s''_\Delta(x_i)|$ durch Einsetzen von f in das Gleichungssystem aus Bemerkung 4.2
(iii) Hieraus Abschätzungen für $x \neq x_i$. ∎

Bemerkung 4.5 (Bernstein–Polynome und Bezier–Kurven)

a) Bernstein–Polynome

$$B_i^{(n)}(x) = \binom{n}{i} (1-x)^{n-i} x^i, \quad (i = 0,1,\ldots,n)$$



Eigenschaften

a) i-fache Nullstelle $x = 0$, $(n-i)$-fache Nullstelle $x = 1$

b) $B_i^{(n)}\Big|_{[0,1]} \ge 0$

c) $\displaystyle\sum_{i=0}^{n} B_i^{(n)}(x) = \sum_{i=0}^{n} \binom{n}{i} (1-x)^{n-i} x^i = \big((1-x)+x\big)^n = 1$

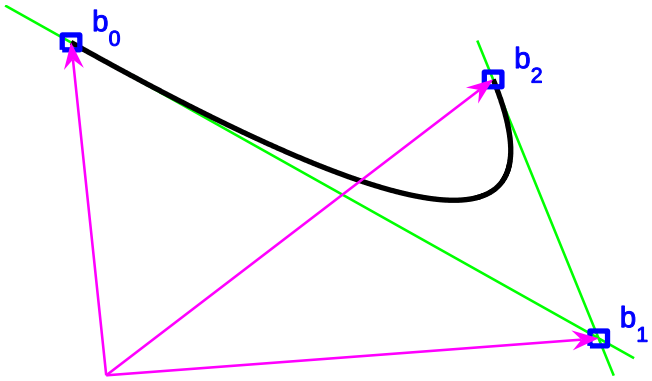
d) $B_i^{(n)}(x) = x \cdot B_{i-1}^{(n-1)}(x) - B_i^{(n-1)}(x)$

b) Bezier–Kurve

geg.: *Kontrollpunkte* $b_0, b_1, \ldots, b_n \in \mathbb{R}^k$

Bezierkurve im $\mathbb{R}^k$: $\displaystyle\sum_{i=0}^{n} b_i B_i^{(n)}(x)\,,\ (\,x \in [0,1]\,)\,.$

- Anfangspunkt b_0, Endpunkt b_n
- Tangente in b_0 verläuft durch b_1, denn
$$\frac{\mathrm{d}}{\mathrm{d}x} B_0^{(n)}(0) = -n\,,\quad \frac{\mathrm{d}}{\mathrm{d}x} B_1^{(n)}(0) = n\,,\quad \frac{\mathrm{d}}{\mathrm{d}x} B_i^{(n)}(0) = 0\,,\ (\,i > 1\,)$$
- $n = 2$: Tangenten in b_0 und b_2 schneiden sich in b_1
- Kurve verläuft in der konvexen Hülle des von den b_i gebildeten Polygons
  $\Rightarrow$ keine unerwünschten Oszillationen



4.2 Trigonometrische Interpolation – Schnelle Fouriertransformation

Bemerkung 4.6 (Problemstellung)
Interpolation periodischer Daten durch trigonometrische Polynome.

***N* = 2*n* + 1 ungerade**

$$T_R^N := \{\,\phi_{2n+1}(t) := \frac{a_0}{2} + \sum_{j=1}^{n}(a_j \cos jt + b_j \sin jt) \ \text{ mit } \ a_j, b_j \in \mathbb{R}\,,\ (\,j = 0, \ldots, n\,)\,\}$$

***N* = 2*n* gerade**

$$T_R^N := \{\,\phi_{2n}(t) := \frac{a_0}{2} + \sum_{j=1}^{n-1}(a_j \cos jt + b_j \sin jt) + \frac{a_n}{2}\cos nt \ \text{ mit } \ a_j, b_j \in \mathbb{R}\,,\ (\,j = 0, \ldots, n\,)\,\}$$

Komplexe Darstellung

$$T_C^N := \{\, \phi_N(t) := \sum_{j=0}^{N-1} c_j \mathrm{e}^{ijt} \,:\, c_j \in \mathbb{C}\,,\ (\, j = 0, \ldots, N-1\,)\,\}$$

Ansatzfunktionen $\{\, 1, \cos jt, \sin jt\,\}$ bzw. $\{\, \mathrm{e}^{ijt} \,:\, 0 \le j \le N-1\,\}$ sind linear unabhängig $\rightsquigarrow$ Interpolationsaufgabe zu N Stützpunkten (t_k, f_k), $(\, k = 0, 1, \ldots, N-1\,)$ eindeutig lösbar.

Typische Aufgabenstellung Digitale Signalverarbeitung, Datenerfassung im festen Takt $\Rightarrow$ äquidistante Knoten, normiert auf $t_k = k \cdot 2\pi/N$.

Bemerkung 4.7 (Trigonometrische Interpolation auf äquidistantem Gitter)
Zu $t_k = k \cdot 2\pi/N$, $(\, k = 0, 1, \ldots, N-1\,)$ sind $\omega_k := \mathrm{e}^{it_k} = \mathrm{e}^{ik\cdot 2\pi/N}$ die N-ten Einheitswurzeln.

Interpolationsbedingungen $\phi_N(t_k) = f_k$, $(\, k = 0, 1, \ldots, N-1\,)$ bestimmen ϕ_N mit

$$\begin{pmatrix} 1 & \omega_0 & \omega_0^2 & \cdots & \omega_0^{N-1} \\ 1 & \omega_1 & \omega_1^2 & \cdots & \omega_1^{N-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & \omega_{N-1} & \omega_{N-1}^2 & \cdots & \omega_{N-1}^{N-1} \end{pmatrix} \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_{N-1} \end{pmatrix} = \begin{pmatrix} f_0 \\ f_1 \\ \vdots \\ f_{N-1} \end{pmatrix}.$$

Bemerkung 4.8 (Komplexe und reelle trigonometrische Polynome)
Gilt für das (komplexe) trigonometrische Polynom

$$\phi_N(t) = \sum_{j=0}^{N-1} c_j \mathrm{e}^{ijt}$$

$\phi_N(t) \in \mathbb{R}$, $(\, t \in \mathbb{R}\,)$, so gilt $\phi_N \in T_R^N$ mit

$$a_j = 2\,\mathrm{Re}\, c_j = c_j + c_{N-j}\,, \quad b_j = -2\,\mathrm{Im}\, c_j = i\,(c_j - c_{N-j})\,.$$

Beweis Für die N äquidistanten Knoten $t_k = k \cdot 2\pi/N$, $(\, k = 0, 1, \ldots, N-1\,)$ gilt

$$\mathrm{e}^{-ijt_k} = \underbrace{\mathrm{e}^{iNk\cdot 2\pi/N}}_{=1} \cdot \mathrm{e}^{-ijt_k} = \mathrm{e}^{i(N-j)t_k}$$

und

$$\sum_{j=0}^{N-1} c_j \mathrm{e}^{ijt_k} = \phi_N(t_k) = \overline{\phi_N(t_k)} = \sum_{j=0}^{N-1} \overline{c_j}\, \mathrm{e}^{-ijt_k} = \sum_{j=0}^{N-1} \overline{c_j}\, \mathrm{e}^{i(N-j)t_k} = \sum_{l=1}^{N} \overline{c_{N-l}}\, \mathrm{e}^{ilt_k}$$

Bemerkung 4.8: Trigonometrische Polynome

Gilt $\phi_N(t) \in \mathbb{R}$, $(t \in \mathbb{R})$, für das (komplexe) trigonometrische Polynom

$$\phi_N(t) = \sum_{j=0}^{N-1} c_j e^{ijt}$$

so gilt $\phi_N \in T_R^N$ mit $a_j = 2\,\mathrm{Re}\, c_j = c_j + c_{N-j}$, $b_j = -2\,\mathrm{Im}\, c_j = i\,(c_j - c_{N-j})$.

Beweis

Für die N äquidistanten Knoten $t_k = k \cdot 2\pi/N$, $(k = 0, 1, \ldots, N-1)$ gilt

$$e^{-ijt_k} = \underbrace{e^{iNk\cdot 2\pi/N}}_{=1} \cdot e^{-ijt_k} = e^{i(N-j)t_k}$$

und

$$\sum_{j=0}^{N-1} c_j e^{ijt_k} = \phi_N(t_k) = \overline{\phi_N(t_k)} = \sum_{j=0}^{N-1} \overline{c_j}\, e^{-ijt_k} = \sum_{j=0}^{N-1} \overline{c_j}\, e^{i(N-j)t_k} = \sum_{l=1}^{N} \overline{c_{N-l}}\, e^{ilt_k}$$

$\Rightarrow$ $c_j = \overline{c_{N-j}}$, da Interpolationsaufgabe eindeutig lösbar.

Martin-Luther-Universität Halle-Wittenberg, NWF III, Institut für Mathematik
Martin Arnold: Grundkurs Numerische Mathematik (WiS 2007/08)

Abbildung 4.1: Zusammenhang zwischen komplexer und reeller Darstellung des trigonometrischen Interpolationspolynoms.

$\Rightarrow$ $c_j = \overline{c_{N-j}}$, da Interpolationsaufgabe eindeutig lösbar.

Wegen $e^{i\cdot 0\cdot t_k} = e^{i\cdot N\cdot t_k} = 1$ ist insbesondere $c_0 = \overline{c_0}$, also $c_0 \in \mathbb{R}$.

Ebenso folgt $c_n \in \mathbb{R}$ für gerade $N = 2n$.

Sei nun $N = 2n + 1$, so ist

$$\begin{aligned}
\phi_N(t_k) &= c_0 + \sum_{j=1}^{2n} c_j e^{ijt_k} = c_0 + \sum_{j=1}^{n} c_j e^{ijt_k} + \sum_{l=1}^{n} c_{N-l} e^{i(N-l)t_k} \\
&= c_0 + \sum_{j=1}^{n} (c_j e^{ijt_k} + \overline{c_j} e^{-ijt_k}) \\
&= c_0 + 2\sum_{j=1}^{n} \mathrm{Re}\,(c_j e^{ijt_k}) = c_0 + 2\sum_{j=1}^{n} (\mathrm{Re}\, c_j \cdot \cos jt_k - \mathrm{Im}\, c_j \cdot \sin jt_k)\,.
\end{aligned}$$

Wegen der Eindeutigkeit des trigonometrischen Interpolationspolynoms folgt die Behauptung durch Koeffizientenvergleich. Analog für gerade $N = 2n$. ∎

Satz 4.9 (Lösung der trigonometrischen Interpolationsaufgabe)
Für äquidistante Stützstellen $t_k = k \cdot 2\pi/N$, $(k = 0, 1, \ldots, N-1)$ ist die Lösung der trigonometrischen Interpolationsaufgabe

$$\phi_N(t_k) = f_k\,, \quad (k = 0, 1, \ldots, N-1)$$

gegeben durch

$$\phi_N(t) = \sum_{j=0}^{N-1} c_j \mathrm{e}^{ijt} \quad mit \quad c_j := \frac{1}{N} \sum_{k=0}^{N-1} f_k \omega_k^{-j}\,, \quad (\,j = 0, 1, \ldots, N-1\,)\,.$$

Beweis Wegen der Eindeutigkeit des trigonometrischen Interpolationspolynoms reicht es aus, für $l = 0, 1, \ldots, N-1$ zu zeigen

$$f_l \stackrel{!}{=} \sum_{j=0}^{N-1} \underbrace{\left(\frac{1}{N} \sum_{k=0}^{N-1} f_k \omega_k^{-j}\right)}_{= c_j} \mathrm{e}^{ijt_l} = \sum_{k=0}^{N-1} f_k \cdot \frac{1}{N} \sum_{j=0}^{N-1} \omega_k^{-j} \omega_l^j\,.$$

Nun ist

$$\sum_{j=0}^{N-1} \omega_k^{-j} \omega_l^j = \sum_{j=0}^{N-1} \omega_{l-k}^j = \begin{cases} N & \text{falls } k = l\,, \\ 0 & \text{sonst}\,, \end{cases}$$

denn

$$\frac{\omega_{l-k} - 1}{\omega_{l-k} - 1} \cdot \sum_{j=0}^{N-1} \omega_{l-k}^j = \frac{1}{\omega_{l-k} - 1} \sum_{j=0}^{N-1} (\omega_{l-k}^{j+1} - \omega_{l-k}^j) = \frac{\omega_{l-k}^N - 1}{\omega_{l-k} - 1} = 0\,,$$

falls $l \neq k$, denn ω_{l-k} ist N-te Einheitswurzel. Hieraus folgt die Behauptung. ∎

Bemerkung 4.10 (Diskrete Fourier–Transformation)
Für 2π–periodische Funktionen $f \in L^2(\mathbb{R})$ erhält man mit den Fourierkoeffizienten

$$\hat{f}(j) := \frac{1}{2\pi} \int_0^{2\pi} f(t) \mathrm{e}^{-ijt}\, \mathrm{d}t\,, \quad (\,j \in \mathbb{Z}\,)$$

die abgebrochenen Fourier–Reihen

$$f_n(t) := \sum_{j=-n}^{n} \hat{f}(j) \mathrm{e}^{ijt}\,.$$

Setzt man für $N = 2n + 1$

$$c_{-j} := c_{N-j}\,, \quad (\,j = 1, \ldots, n\,)\,,$$

so ist

$$\begin{aligned} \phi_N(t_k) &= \sum_{j=0}^{N-1} c_j \mathrm{e}^{ijt_k} = \sum_{j=0}^{n} c_j \mathrm{e}^{ijt_k} + \sum_{j=n+1}^{N-1} c_j \mathrm{e}^{ijt_k} \\ &= \sum_{j=0}^{n} c_j \mathrm{e}^{ijt_k} + \sum_{l=1}^{n} c_{N-l} \mathrm{e}^{i(N-l)t_k} = \sum_{j=0}^{n} c_j \mathrm{e}^{ijt_k} + \sum_{l=1}^{n} c_{-l} \mathrm{e}^{i\cdot(-l)\cdot t_k} = \sum_{j=-n}^{n} c_j \mathrm{e}^{ijt_k}\,. \end{aligned}$$

Bemerkung 4.10: Diskrete Fourier-Transformation

Trigonometrische Interpolation ($N = 2n+1$)

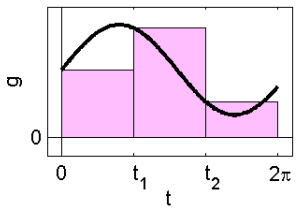
$$\phi_N(t_k) = \sum_{j=0}^{N-1} c_j e^{ijt_k} = \sum_{j=-n}^{n} c_j e^{ijt_k}$$

2π–periodische Funktionen $f \in L^2(\mathbb{R})$ $\Rightarrow$ (abgebrochene) Fourier–Reihe

$$f_n(t) := \sum_{j=-n}^{n} \hat{f}(j)\, e^{ijt} \quad \text{mit Fourierkoeffizienten} \quad \hat{f}(j) := \frac{1}{2\pi}\int_0^{2\pi} f(t) e^{-ijt}\, dt$$

Approximation

$$\int_0^{2\pi} g(t)\, dt \approx \frac{2\pi}{N} \sum_{k=0}^{N-1} g(t_k)$$

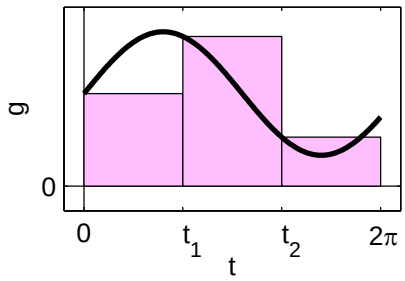


$$\hat{f}(j) = \frac{1}{2\pi}\int_0^{2\pi} f(t) e^{-ijt}\, dt \approx \frac{1}{N} \sum_{k=0}^{N-1} f(t_k) e^{-ijt_k} = \frac{1}{N} \sum_{k=0}^{N-1} f(t_k) \omega_k^{-j} = c_j$$

Abbildung 4.2: Zusammenhang zwischen diskreter und kontinuierlicher Fouriertransformation.

Approximiert man

$$\int_0^{2\pi} g(t)\, dt \approx \frac{2\pi}{N} \sum_{k=0}^{N-1} g(t_k)\,,$$



so ergibt sich

$$\hat{f}(j) \approx \frac{1}{N} \sum_{k=0}^{N-1} f(t_k) e^{-ijt_k} = \frac{1}{N} \sum_{k=0}^{N-1} f_k \omega_k^{-j} = c_j\,.$$

Wegen der Analogie zur klassischen kontinuierlichen Fourier–Transformation heißt die Abbildung

$$\mathcal{F}_N : \mathbb{C}^N \to \mathbb{C}^N\,, \quad \left(f_k\right)_{k=0}^{N-1} \mapsto \left(c_j\right)_{j=0}^{N-1}$$

aus Satz 4.9 *Diskrete Fourier–Transformation* (DFT) und die Umkehrabbildung

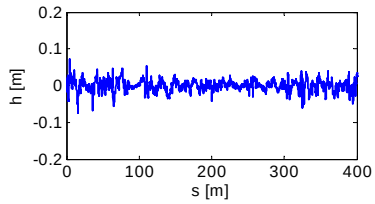
$$\mathcal{F}_N^{-1} : (c_j)_j \mapsto (f_k)_k\,, \quad f_k := \sum_{j=0}^{N-1} c_j \omega_j^k\,,\ (\,k = 0, 1, \ldots, N-1\,)$$

(Fourier–)Synthese oder *Inverse diskrete Fourier–Transformation* (IDFT).

Beispiel 4.11: Tiefpassfilter

Beispiel (Prof. Dr.-Ing. P. Pickel, U Halle, Landwirtschaftliche Fakultät)

Fahrweg eines landwirtschaftlichen Nutzfahrzeugs (vertikale Auslenkung)



Messdaten im Abstand $\Delta_s = 0.05\,\mathrm{m}$

Geschwindigkeit $v = 2.0\,\mathrm{m/s}$

$\Rightarrow \quad \Delta_t = 0.025\,\mathrm{s} \quad \Rightarrow \quad \bar{f}_{\mathrm{input}} = 40\,\mathrm{Hz}$

Abbildung 4.3: Anwendung der DFT in der Signalverarbeitung: Ausgangsdaten.

Beispiel 4.11: Tiefpassfilter (II)

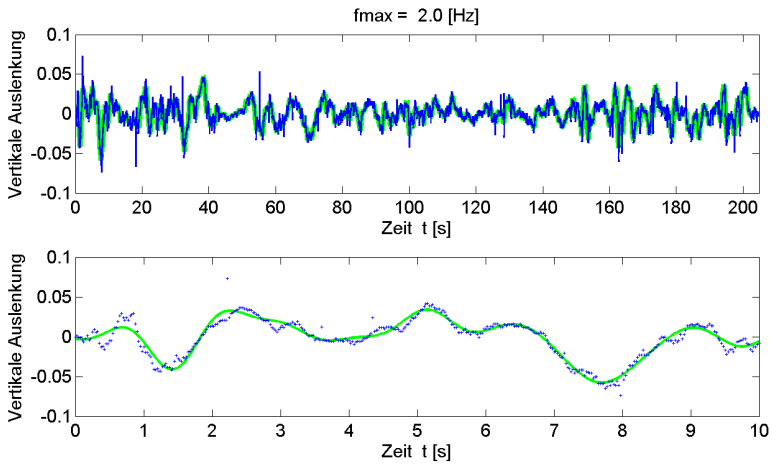


Abbildung 4.4: Anwendung der DFT in der Signalverarbeitung: $f_{\max} = 2.0\,\mathrm{Hz}$.

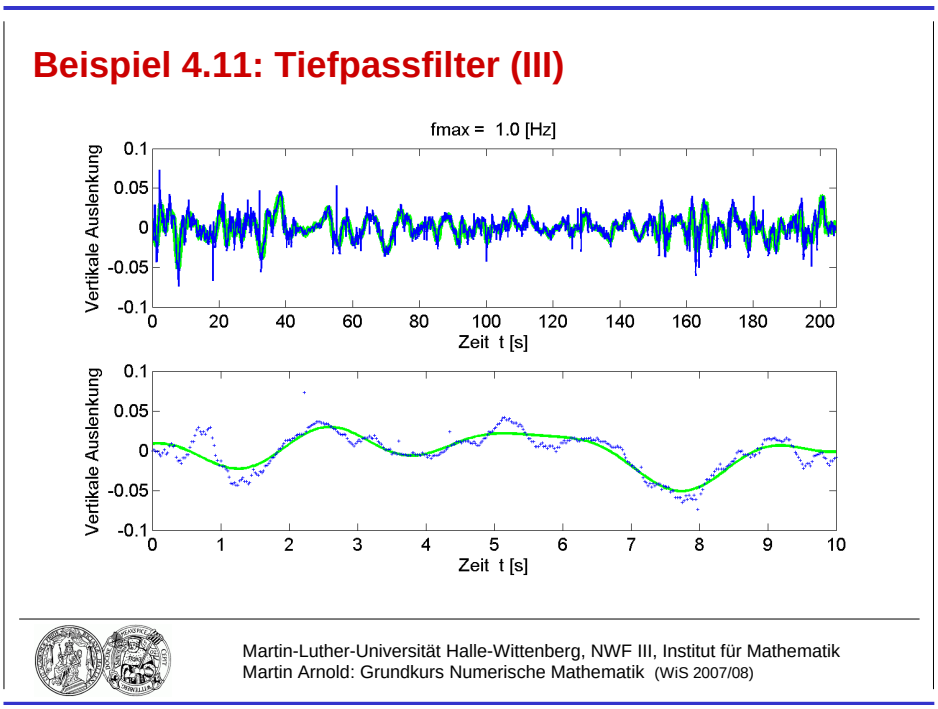


Abbildung 4.5: Anwendung der DFT in der Signalverarbeitung: $f_{\max} = 1.0\,\mathrm{Hz}$.

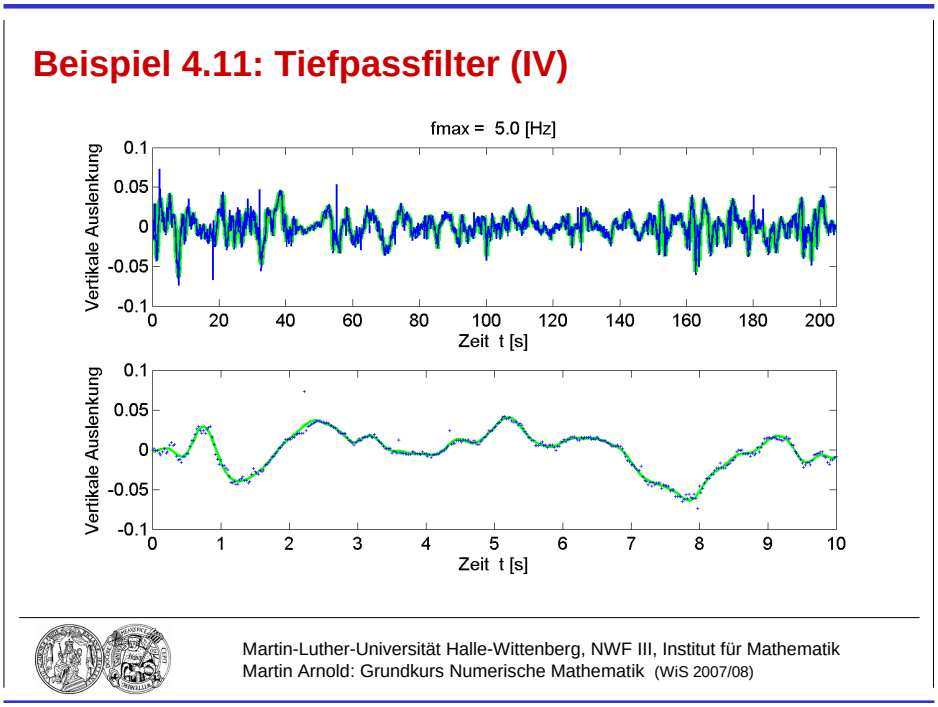


Abbildung 4.6: Anwendung der DFT in der Signalverarbeitung: $f_{\max} = 5.0\,\mathrm{Hz}$.

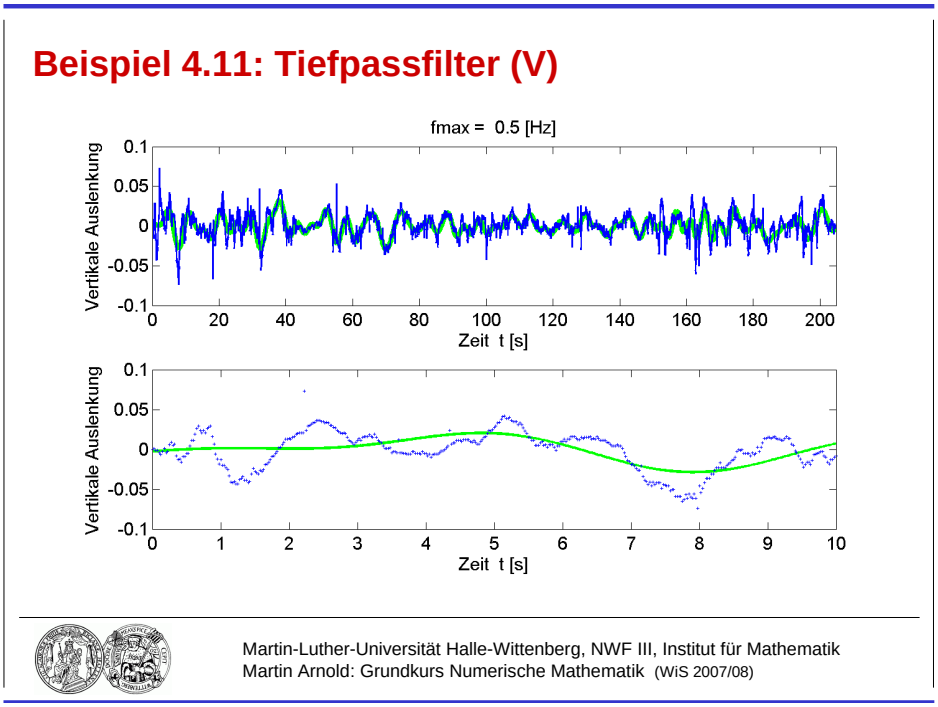


Abbildung 4.7: Anwendung der DFT in der Signalverarbeitung: $f_{\max} = 0.5\,\mathrm{Hz}$.

Beispiel 4.11 (Tiefpassfilter)

Anwendung der DFT in der Signalverarbeitung

Elimination hochfrequenter Anteile im Signal, die häufig durch Messfehler verfälscht („Messrauschen“) und darüberhinaus für die praktische Anwendung oft nicht relevant sind $\Rightarrow$ „*Tiefpassfilter*“.

praktisch Berücksichtige in $\phi_N(t)$ nur die Terme $a_j \cos jt$ und $b_j \sin jt$, die zu den ersten $j_{\max} \ll n$ Frequenzen gehören ($j_{\max}$ ist vom Anwender vorzugeben). Für $N = 2n+1$ erhält man

$$\tilde{f}_k := c_0 + \sum_{j=1}^{j_{\max}} c_j \omega_j^k + \sum_{j=1}^{j_{\max}} c_{N-j} \omega_{N-j}^k = c_0 + \sum_{j=1}^{j_{\max}} (c_j \omega_j^k + c_{N-j} \overline{\omega_j}^k), \quad (k = 0, 1, \ldots, N-1).$$

Beispiel Rauigkeit einer Feldoberfläche

(Originaldaten von Prof. Dr.-Ing. P. Pickel, Institut für Agrartechnik und Landeskultur).

Die Messdaten für die Höhe $u(x)$ einer Feldoberfläche am Ort x liegen im Abstand von $\Delta_x = 0.05\,\mathrm{m}$ vor (Abb. 4.3). Bei konstanter Fahrgeschwindigkeit $v = 2.0\,\mathrm{m/s}$ entspricht dies einer Frequenz von $1\,/\,(\,0.05\,\mathrm{m}\,/\,2.0\,\mathrm{m/s}\,) = 40\,\mathrm{Hz}$. Abb. 4.4 – Abb. 4.7 zeigen für verschiedene maximale Frequenzen $f_{\max}$ den Vergleich zwischen gefilterten Daten und Originaldaten. Die Elimination der hochfrequenten Anteile in $u(t)$ ist deutlich erkennbar.

Bemerkung 4.12 (Schnelle Fourier–Transformation)
engl.: Fast Fourier Transform(ation) (FFT)

Problem Standard–Algorithmus zur Auswertung von $\mathcal{F}_N$ oder $\mathcal{F}_N^{-1}$ würde $\mathcal{O}(N^2)$ Rechenoperationen erfordern (Matrix–Vektor–Multiplikation).

Cooley–Tuckey (1965) Sei $N = 2M$ gerade und $\omega = \mathrm{e}^{i\frac{2\pi}{N}}$ oder $\omega = \mathrm{e}^{-i\frac{2\pi}{N}}$.
Dann gilt für

$$\alpha_j = \sum_{k=0}^{N-1} f_k \omega^{kj}\,, \quad (\, j = 0, 1, \ldots, N-1\,)$$

$$\alpha_{2l} = \sum_{k=0}^{M-1} g_k \xi^{kl}\,, \qquad \alpha_{2l+1} = \sum_{k=0}^{M-1} h_k \xi^{kl}\,, \quad (\, l = 0, 1, \ldots, M-1\,)$$

mit $M := N/2$, $\xi := \omega^2$ und

$$g_k := f_k + f_{k+M}\,, \qquad h_k := (f_k - f_{k+M})\,\omega^k\,.$$

Mit $2M$ Additionen und $2M$ Multiplikationen ($\omega^k \to \omega^{k+1} = \omega \cdot \omega^k$, $h_k = (\ldots) \cdot \omega^k$) wird die Berechnung von N Summen der Länge N zurückgeführt auf $2M = N$ Summen der Länge $M = N/2$.

Beweis

$$\alpha_{2l} = \sum_{k=0}^{N-1} f_k \omega^{k\cdot 2l} = \sum_{k=0}^{\frac{N}{2}-1} \left(f_k \omega^{2kl} + f_{k+\frac{N}{2}} \omega^{2(k+\frac{N}{2})l} \right) = \sum_{k=0}^{M-1} (f_k + f_{k+M})\,\omega^{2kl}$$

$$\alpha_{2l+1} = \sum_{k=0}^{N-1} f_k \omega^{(2l+1)k} = \sum_{k=0}^{\frac{N}{2}-1} \left(f_k \omega^{2kl+k} + f_{k+\frac{N}{2}} \omega^{(2l+1)(k+\frac{N}{2})} \right) = \sum_{k=0}^{M-1} (f_k - f_{k+\frac{N}{2}})\,\omega^k \cdot \omega^{2kl}$$

Rekursive Anwendung besonders einfach für $N = 2^p$ $\Rightarrow$ Aufwand zur Berechnung von $\alpha_0, \alpha_1, \ldots, \alpha_{N-1}$ (Analyse oder Synthese) beträgt $2N \log_2 N$ Multiplikationen.

Algorithmus 4.13 (Schnelle Fourier–Transformation)
Sei $N = 2^p$ und $\omega = \mathrm{e}^{i\frac{2\pi}{N}}$ oder $\omega = \mathrm{e}^{-i\frac{2\pi}{N}}$.

Eingabe: $f_0, f_1, \ldots, f_{N-1} \in \mathbb{C}$

Ausgabe: $\alpha_0, \alpha_1, \ldots, \alpha_{N-1} \in \mathbb{C}$ mit $\alpha_j := \sum_{k=0}^{N-1} f_k \omega^{kj}$.

```
N_red := N ;  z := ω
while N_red > 1 do
    M_red := N_red/2
    for j = 0 : (N/N_red − 1)
        l := jN_red
        for k = 0 : M_red − 1
            a := f_{l+k} + f_{l+k+M_red}
            f_{l+k+M_red} := (f_{l+k} − f_{l+k+M_red}) z^k
            f_{l+k} := a
    N_red := M_red ;  z := z^2
for k = 0 : N − 1
    α_{σ(k)} := f_k
```

Vertauschung der Komponenten von α bestimmt durch Permutation $\sigma(k)$:

$$\sigma\big(\sum_{j=0}^{N-1} a_j 2^j\big) := \sum_{j=0}^{N-1} a_{N-j} 2^j \quad \text{mit} \quad a_0, a_1, \ldots, a_{N-1} \in \{\,0,\,1\,\}\,.$$

$\rightsquigarrow$ „bit reversal“, einfache Implementierung durch Bitmanipulationen

- Typisches Beispiel eines „divide–and–conquer“–Algorithmus,
- gut parallelisierbar,
- in Signalprozessoren hardwaremäßig verfügbar.

5 Quadratur

Bemerkung 5.1 (Problemstellung)

geg.: stückweise stetige Funktion $f : [a,b] \to \mathbb{R}$

ges.: $I(f) := I_a^b(f) := \displaystyle\int_a^b f(x)\,\mathrm{d}x$

Eigenschaften

a) Linearität: $I(\alpha f + \beta g) = \alpha I(f) + \beta I(g)$

b) Positivität: $f(x) \geq 0\,,\ (\,x \in [a,b]\,) \ \Rightarrow\ I(f) \geq 0$

c) Additivität: $I_a^b(f) = I_a^c(f) + I_c^b(f)\,,\ (\,c \in (a,b)\,)$

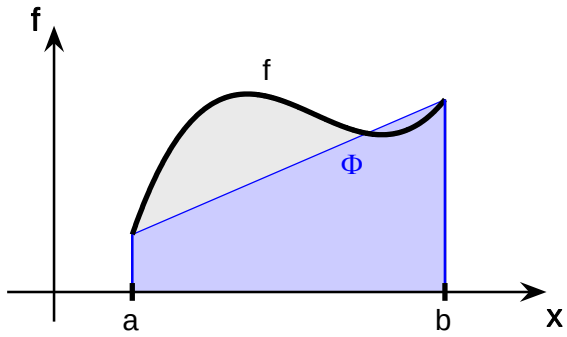
Spezialfall $\tilde{f}(x) = x^k \;\Rightarrow\; I_a^b(\tilde{f}) = \displaystyle\int_a^b x^k \,\mathrm{d}x = \frac{1}{k+1}\, x^{k+1}\Big|_a^b$

Idee Approximiere $f : [a,b] \to \mathbb{R}$ durch ein (Interpolations–)Polynom $\tilde{f}$ und verwende $I(\tilde{f})$ als Näherungswert für $I(f)$.

Beispiel 5.2 (Trapezregel)

a) Approximation von $f : [a,b] \to \mathbb{R}$ durch lineares Interpolationspolynom Φ mit Stützstellen a und b:

$$\Phi(x) = f(a) + \frac{x-a}{b-a}\big(f(b) - f(a)\big)$$



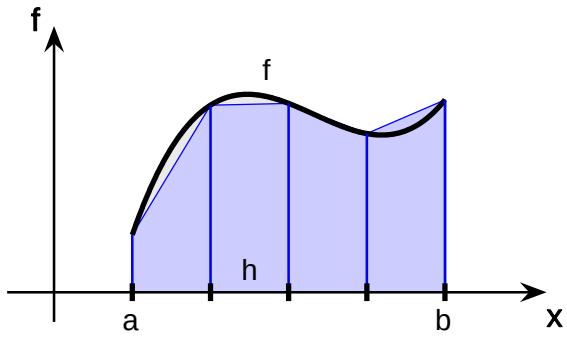
$$I(\Phi) = (b-a)f(a) + \frac{f(b)-f(a)}{b-a} \cdot \frac{1}{2}(x-a)^2\Big|_a^b = \frac{b-a}{2}\big(f(a)+f(b)\big)$$

b) Zusammengesetzte Trapezregel, *Trapezsumme*. Wähle ein Gitter

$$\Delta = \{\, a = x_0 < x_1 < x_2 < \ldots < x_N = b \,\}$$

mit Schrittweiten

$$h_i := x_{i+1} - x_i\,,\ (\,i = 0, 1, \ldots, N-1\,)\,.$$



$$I_a^b(f) = \sum_{i=0}^{N-1} I_{x_i}^{x_{i+1}}(f)\,, \qquad \text{Trapezregel: } I_{x_i}^{x_{i+1}}(f) \approx \frac{h_i}{2}\big(f(x_i) + f(x_{i+1})\big)$$

Aufsummieren $\Rightarrow$ *zusammengesetzte Trapezregel*:

$$I_a^b(f) \approx \tilde{I}_a^b(f) := \frac{h_0}{2} f(x_0) + \sum_{i=1}^{N-1} \frac{h_{i-1}+h_i}{2} f(x_i) + \frac{h_{N-1}}{2} f(x_N)$$

Spezialfall äquidistantes Gitter $x_i := a + ih$ mit $i = 0, 1, \ldots, N$ und $h := (b-a)/N$:

$$I_a^b(f) \approx \tilde{I}_a^b(f) = \frac{h}{2}\Big(f(a) + 2\sum_{i=1}^{N-1} f(a+ih) + f(b)\Big)\,.$$

Bemerkung 5.3 (Newton–Cotes–Formeln)
Wähle Interpolationspolynom Φ zu $n+1$ äquidistanten Stützstellen $x_i := a + ih$ mit $h := (b-a)/n\,,\ (\,i = 0, \ldots, n\,)$:

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

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

mit *Knoten* $x_0, x_1, \ldots, x_n$ und *Gewichten* $w_i := \displaystyle\int_a^b L_i^{(n)}(x)\,\mathrm{d}x\,.$

Allgemeine Struktur einer *Quadraturformel*: $\tilde{I}(f) = \displaystyle\sum_{i=0}^{n} w_i f(x_i)\,.$

Positivität, falls $w_i > 0\,,\ (\,i = 0, 1, \ldots, n\,)$

Gewichte der *Newton–Cotes–Quadraturformeln* liegen für $[a,b] = [0,1]$ tabelliert vor:

n	w_i	Name	Fehler
1	$1/2,\ 1/2$	Trapezregel	$h^3/12\ f''(\xi_1)$
2	$1/6,\ 2/3,\ 1/6$	Keplersche Fassregel	$h^5/90\ f^{IV}(\xi_2)$
3	$1/8,\ 3/8,\ 3/8,\ 1/8$	Newtonsche 3/8–Regel	$3h^5/80\ f^{IV}(\xi_3)$

mit geeigneten $\xi_1, \xi_2, \xi_3 \in (0,1)\,.$
Positivität nur bis $n \le 7\,.$
Für $n \ge 8$ treten auch negative Gewichte auf $\rightsquigarrow$ praktisch unbrauchbar.

Analog zu Beispiel 5.2 definiert man *zusammengesetzte Newton–Cotes–Formeln*:

Simpson–Regel Zusammengesetzte Keplersche Fassregel.

$$S(h) := \frac{h}{3}\Big(f(a) + 4f(a+h) + 2f(a+2h) + 4f(a+3h) + 2f(a+4h) + \ldots + \\ + 4f(a+(2N-1)h) + f(b) \Big) \quad \text{mit} \quad h := \frac{b-a}{2N}\,.$$

Bemerkung 5.4 (Fehlerabschätzungen für Newton–Cotes–Formeln)

Trapezregel Restglied der Polynominterpolation, vgl. Satz 1.13 $\Rightarrow$

$$f(x) - \Phi(x) = \frac{f''(\xi)}{2}(x-a)(x-b) \quad \text{mit einem } \xi \in [a,b]$$

$$\Big|\underbrace{\int_a^b f(x)\,\mathrm{d}x}_{I(f)} - \underbrace{\int_a^b \Phi(x)\,\mathrm{d}x}_{\tilde{I}(f)}\Big| = \Big|\int_a^b \frac{f''(\xi)}{2}\underbrace{(x-a)(x-b)}_{<0}\,\mathrm{d}x\Big|$$

$$\leq \max_{\xi\in[a,b]}|f''(\xi)|\cdot\frac{1}{2}\int_a^b (x-a)(b-x)\,\mathrm{d}x$$

$$= \max_{\xi\in[a,b]}|f''(\xi)|\cdot\frac{(b-a)^3}{12}$$

Polynome ersten Grades werden durch die Trapezregel exakt integriert.

Keplersche Fassregel $n=2\,,\ \ h=\dfrac{b-a}{2}\,,\ \ x_0=a\,,\ \ x_1=\dfrac{a+b}{2}\,,\ \ x_2=b$

$$f(x)-\Phi(x)=\frac{f'''(\xi)}{3!}(x-a)(x-\frac{a+b}{2})(x-b)$$

Ähnlich wie bei der Trapezregel zeigt man

$$|I(f)-\tilde{I}(f)|\leq\frac{h^5}{90}\max_{\xi\in[a,b]}|f^{IV}(\xi)|\,.$$

Polynome bis zum Grad 3 (!) werden exakt integriert.

Bemerkung 5.5 (Gauß–Christoffel–Quadraturformeln)

Idee Wähle Gewichte w_i *und* Knoten x_i so, dass Polynome möglichst hohen Grades exakt integriert werden:

$$\sum_{i=0}^{n} w_i x_i^j = \tilde{I}(x^j) \stackrel{!}{=} I(x^j) = \int_a^b x^j\,\mathrm{d}x = \frac{b^{j+1}-a^{j+1}}{j+1}\,,\ (\,j=0,1,\ldots,m\,)$$

$m+1$ nichtlineare Gleichungen für $2n+2$ Unbekannte $x_0,x_1,\ldots,x_n,w_0,w_1,\ldots,w_n$.

Ergebnisse

- $m=2n+1$ ist stets erreichbar,
- Knoten x_i und Gewichte w_i sind für $m=2n+1$ eindeutig bestimmt,
- Knoten x_i sind Nullstellen der sog. Legendre–Polynome.

Beispiel $n=1 \quad\Rightarrow\quad x_{0,1}=\dfrac{a+b}{2}\pm\dfrac{b-a}{2\sqrt{3}}\,,\ \ w_{0,1}=\dfrac{b-a}{2}$

$$\tilde{I}(f)=\frac{b-a}{2}\big(f(x_0)+f(x_1)\big)$$

integriert Polynome bis zum Grad $2n+1=3$ exakt.

Erweiterung $\int_a^b \omega(x) f(x)\,\mathrm{d}x$ mit Gewichtsfunktion $\omega(x) \geq 0\,,\ (\,x \in [a,b]\,)\,.$

Die optimalen Knoten $x_0, x_1, \ldots, x_n$ ergeben sich auch hier als Nullstellen orthogonaler Polynome.

Beispiel Tschebyscheff–Quadratur

$$\int_{-1}^{1} \frac{f(x)}{\sqrt{1-x^2}}\,\mathrm{d}x \approx \sum_{i=0}^{n} w_i f(x_i) \ \text{ mit } \ x_i = \cos\frac{2i+1}{2n+2}\pi\,,\ w_i = \frac{\pi}{n+1}\,,\ (\,i = 0,1,\ldots,n\,)$$

Literatur Deuflhard/Hohmann, Kapitel 9.3.

Bemerkung 5.6 (Romberg–Quadratur)
Die zusammengesetzte Trapezregel $T(h)$ aus Beispiel 5.2b) ergibt (theoretisch) für $h \to 0$ den exakten Wert $I_a^b(f)$.

Idee Berechne $T(h_i)$ für einige endliche Schrittweiten $h_i > 0$, interpoliere die Stützpunkte $(h_1, T(h_1)), \ldots, (h_k, T(h_k))$ durch ein Polynom $\pi(h)$ mit $\deg \pi \leq k-1$ und setze $I_a^b(f) \approx \tilde{I}_a^b(f) := \pi(0)\,.$

Theorie Asymptotische Entwicklung des Fehlers der Trapezsumme (Euler–Maclaurinsche Summenformel):

$$T(h) = \int_a^b f(x)\,\mathrm{d}x + \sum_{k=1}^{M} c_k h^{2k} + \mathcal{O}(h^{2M+2}) \quad \text{mit Konstanten } c_k\,.$$

praktisch Wähle $h_i = H/n_i$ mit Grundschrittweite H und $n_i = 2^{i-1}$ und bestimme Interpolationspolynom π mit $\pi(h_i^2) = T(h_i)\,,\ (\,i = 1,\ldots,k\,)$. Berechnung von $\pi(0)$ mittels Neville–Schema, vgl. Bemerkung 1.10:

$$T_{i,\ldots,i+l} = T_{i+1,\ldots,i+l} - \frac{T_{i+1,\ldots,i+l} - T_{i,\ldots,i+l-1}}{\left(\frac{h_i}{h_{i+l}}\right)^2 - 1} \quad \text{mit} \quad \left(\frac{h_i}{h_{i+l}}\right)^2 = \left(\frac{n_{i+l}}{n_i}\right)^2$$

und $T_i := T(h_i)\,.$ Ergebnis: $I_a^b(f) \approx T_{1,\ldots,k}$ (*Romberg–Quadratur, Romberg–Schema*).

wichtig Für $n_i = 2^{i-1}$ ist $h_i = 2h_{i+1}$ und

$$\begin{aligned} T(h_{i+1}) &= \frac{h_{i+1}}{2}\big(f(a) + 2\sum_{j=1}^{n_{i+1}-1} f(a + jh_{i+1}) + f(b)\big) \\ &= \frac{1}{2}\,T(h_i) + h_{i+1}\sum_{k=0}^{n_i-1} f(a + (2k+1)h_{i+1}) \end{aligned}$$

allgemein Romberg–Quadratur ist Spezialfall von *Extrapolationsverfahren*, die immer dann angewendet werden können, wenn der Fehler eines numerischen Verfahrens eine asymptotische Entwicklung in Potenzen von h hat.

Beispiel 5.7 (Extrapolation und Differenzenquotient)
a) Rechtsseitiger Differenzenquotient erster Ordnung

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$

Taylorentwicklung von $f(x+h)$ ergibt asymptotische h–Entwicklung des Fehlers.

b) Zentraler Differenzenquotient zur Approximation der ersten Ableitung

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h}$$

Fehler hat h^2–Entwicklung (Taylorentwicklung von $f(x+h)$ und $f(x-h)$).

6 Iterationsverfahren für lineare und nichtlineare Gleichungssysteme

6.1 Nullstellen reeller Funktionen

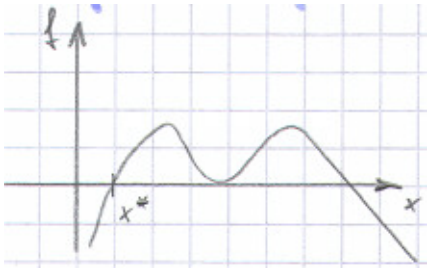
Bemerkung 6.1 (Problemstellung)

geg.: $f \in C[a,b]$

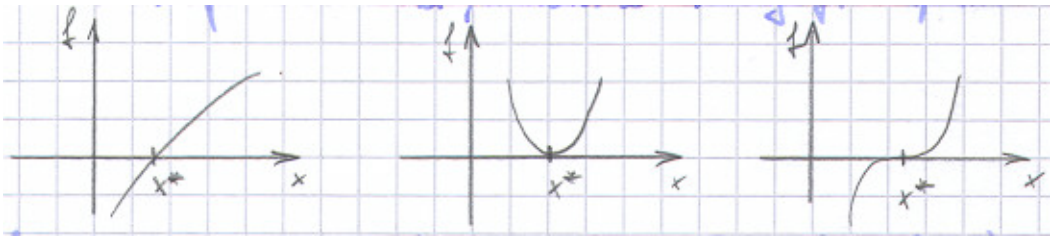
ges.: $x^* \in [a,b]$ mit $f(x^*) = 0$

Lösungstheorie

- f linear $\Rightarrow$ $f(x) = 0$ genau dann eindeutig lösbar in $\mathbb{R}$, falls $f' \neq 0$.
- f nichtlinear $\Rightarrow$ i. Allg. nur Aussagen über lokale Eindeutigkeit der Lösung



- Satz über die implizite Funktion: Ist $f(x^*) = 0$, $f \in C^1[a,b]$ und $f'(x^*) \neq 0$, so ist $y = f(x)$ in einer Umgebung von x^* eindeutig nach x auflösbar: $x = x(y)$.
- $f'(x^*) = 0$ $\Rightarrow$ mehrfache Nullstelle, numerische Bestimmung oft kompliziert



- $f(a) \cdot f(b) < 0$ $\Rightarrow$ es existiert ein x^* mit $f(x^*) = 0$ (Zwischenwertsatz)

Bemerkung 6.2 (Bisektionsverfahren)
geg.: $f \in C[a,b]$, Intervallenden a, b mit $f(a) \cdot f(b) < 0$, Abbruchschranke TOL

Initialisierung: $f_a := f(a)$, $f_b := f(b)$.
`repeat`
$c := \dfrac{a+b}{2}$, $f_c := f(c)$
`if` $f_a \cdot f_c < 0$ `then` $b := c$, $f_b := f_c$
`else` $a := c$, $f_a := f_c$
`until` $|b-a| \leq \mathrm{TOL}$
Ergebnis: $x^* \approx x_{\mathrm{bi}} := \dfrac{a+b}{2}$

Konvergenz stets gesichert: Werden mindestens $1 + \log_2 \dfrac{b-a}{\mathrm{TOL}}$ Iterationsschritte ausgeführt, so gilt

$$|x_{\mathrm{bi}} - x^*| \leq \mathrm{TOL}\,.$$

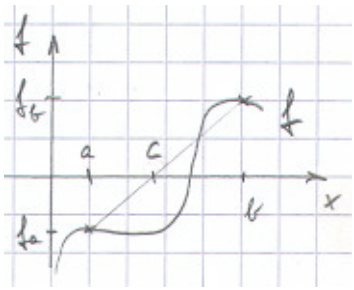
beachte Wegen der Rundungsfehler bei der Auswertung von f kann x^* außerhalb des numerisch bestimmten Intervalls $[a,b]$ liegen.

praktisch sehr robust, einfach zu implementieren, aber sehr langsame Konvergenz

Bemerkung 6.3 (Regula falsi)

Idee Bestimme wie im Bisektionsverfahren immer kleinere Intervalle, die x^* enthalten, berücksichtige bei der Wahl von c jedoch den Lösungsverlauf.

praktisch Wähle c als Nullstelle des (linearen) Interpolationspolynoms zu den Stützpunkten (a, f_a) und (b, f_b).



geg.: $f \in C[a,b]$, Intervallenden a, b mit $f(a) \cdot f(b) < 0$, Abbruchschranke TOL

Initialisierung: $f_a := f(a)$, $f_b := f(b)$.
`repeat`
$c := \dfrac{af_b - bf_a}{f_b - f_a}$, $f_c := f(c)$
`if` $f_a \cdot f_c < 0$ `then` $b := c$, $f_b := f_c$
`else` $a := c$, $f_a := f_c$
`until` $|b-a| \leq \mathrm{TOL}$
Ergebnis: $x^* \approx x_{\mathrm{rf}} := \dfrac{a+b}{2}$

Regula falsi konvergiert i. Allg. deutlich schneller als das Bisektionsverfahren.

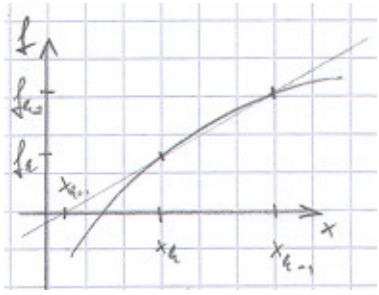
Problem Langsame Konvergenz, wenn eines der beiden Intervallenden stets unverändert bleibt wie z. B. für $f(x) := x^{10} - 1/2\,,\ (\,x \in [0,1]\,)\,.$

Alternative Bleibt eines der beiden Intervallenden a bzw. b über mehr als einen Iterationsschritt unverändert, so ersetze $f_a := \frac{1}{2} f_a$ bzw. $f_b := \frac{1}{2} f_b\,.$

Bemerkung 6.4 (Sekantenverfahren)
Verzichtet man auf eine Einschließung der Nullstelle x^*, so ergibt sich ausgehend von $(x_{k-1}, f(x_{k-1}))$, $(x_k, f(x_k))$ in der Regel eine wesentlich bessere Näherung für f:

Betrachte das (lineare) Interpolationspolynom

$$f(x) \approx f_k(x) := f_{k-1} + \frac{x - x_{k-1}}{x_k - x_{k-1}}\,(f_k - f_{k-1})$$



und bestimme aus $f_k(x) \stackrel{!}{=} 0$ die neue Näherung

$$x_{k+1} := x_k - f_k\,\frac{x_k - x_{k-1}}{f_k - f_{k-1}}\,, \qquad f_{k+1} := f(x_{k+1})\,.$$

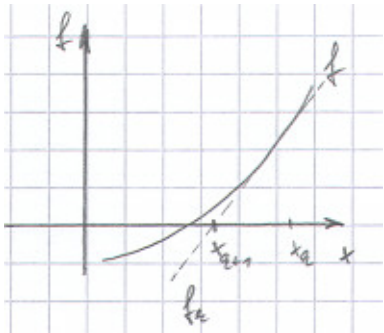
Konvergenzordnung des Sekantenverfahrens $q := (1 + \sqrt{5})/2\,,$ d. h.

$$0 \le \lim_{k\to\infty} \frac{|x_{k+1} - x^*|}{|x_k - x^*|^q} < \infty$$

praktisch Schnelle Konvergenz für gute Startwerte, jedoch Gefahr der Divergenz für schlechte Startwerte

Verallgemeinerung Inverse Interpolation.
Hinzunahme weiterer Stützpunkte $(x_i, f(x_i))$, Bestimmung des Interpolationspolynoms $\pi(y)$ zu den Stützstellen $y = f_k,\ f_{k-1},\ f_{k-2},\ \ldots$ und Stützwerten $\pi = x_k,\ x_{k-1},\ x_{k-2},\ \ldots$ und Wahl von c als $c := \pi(0)\,.$

Bemerkung 6.5 (Newtonverfahren)
geg.: $f \in C^1[a,b]$



Linearisierung von f in x_k:

$$f(x) \approx f_k(x) := f(x_k) + f'(x_k)(x - x_k)$$

Bestimmung von x_{k+1} als Nullstelle von $f_k(x)$:

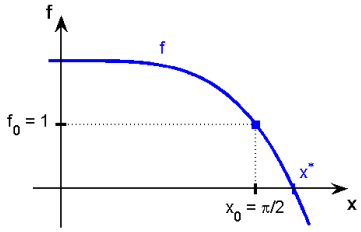
$$x_{k+1} := x_k - \frac{f(x_k)}{f'(x_k)}$$

Beispiel 6.6: Newtonverfahren

gesucht

Kleinste positive Lösung von

$$f(x) = \cos x \cosh x + 1 = 0$$



Newtonverfahren $\quad x_{k+1} = x_k - \dfrac{f(x_k)}{f'(x_k)}, \quad$ Startwert $\; x_0 := \dfrac{\pi}{2}$

k	x_k	$f(x_k)$	$f'(x_k)$	$\lvert x_k - x^*\rvert$
0	1.570796326794897	$1.0000_E + 00$	−2.5092	$3.0431_E - 01$
1	1.969333142133283	$-4.1751_E - 01$	−4.7298	$9.4229_E - 02$
2	1.881060554590512	$-2.4757_E - 02$	−4.1744	$5.9565_E - 03$
3	1.875129963043149	$-1.0716_E - 04$	−4.1383	$2.5894_E - 05$
4	1.875104069204172	$-2.0368_E - 09$	−4.1381	$4.9221_E - 10$
5	1.875104068711961	$2.2204_E - 16$	−4.1381	$< 1.0_E - 16$
6	1.875104068711961			

Martin-Luther-Universität Halle-Wittenberg, NWF III, Institut für Mathematik
Martin Arnold: Grundkurs Numerische Mathematik (WiS 2007/08)

Abbildung 6.1: Quadratische Konvergenz des Newtonverfahrens.

Quadratisch konvergent für einfache Nullstellen, d. h.

$$0 \le \lim_{k\to\infty} \frac{|x_{k+1} - x^*|}{|x_k - x^*|^2} < \infty\,,$$

linear konvergent für mehrfache Nullstellen.

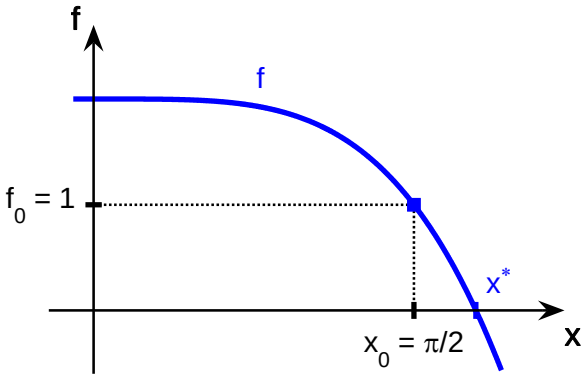
Beispiel 6.6 (Newtonverfahren)
Berechnung der kleinsten positiven Lösung von $f(x) = \cos x \cosh x + 1 = 0\,.$

Newtonverfahren

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

mit Startwert $x_0 := \pi/2$ und

$$f'(x) = \cos x \sinh x - \sin x \cosh x\,.$$



k	x_k	$f(x_k)$	$f'(x_k)$	$\lvert x_k - x^* \rvert$
0	1.570796326794897	$1.0000_{\mathrm{E}} + 00$	-2.5092	$3.0431_{\mathrm{E}} - 01$
1	1.969333142133283	$-4.1751_{\mathrm{E}} - 01$	-4.7298	$9.4229_{\mathrm{E}} - 02$
2	1.881060554590512	$-2.4757_{\mathrm{E}} - 02$	-4.1744	$5.9565_{\mathrm{E}} - 03$
3	1.875129963043149	$-1.0716_{\mathrm{E}} - 04$	-4.1383	$2.5894_{\mathrm{E}} - 05$
4	1.875104069204172	$-2.0368_{\mathrm{E}} - 09$	-4.1381	$4.9221_{\mathrm{E}} - 10$
5	1.875104068711961	$2.2204_{\mathrm{E}} - 16$	-4.1381	$< 1.0_{\mathrm{E}} - 16$
6	1.875104068711961			

Ergebnis Folge (x_k) konvergiert quadratisch gegen $x^* = 1.875104068711961\ldots$.

6.2 Das Newtonverfahren

Bemerkung 6.7 (Newton–Raphson–Verfahren)

geg.: $F : D \to \mathbb{R}^n$ mit $D \subset \mathbb{R}^n$, F stetig differenzierbar
Anfangsnäherung $x_0 \in D$

ges.: $x^* \in D$ mit $F(x^*) = 0$

Algorithmus

Schritt 0 $k := 0$.

Schritt 1 Berechne $J := F_x(x_k)$ und LU–Zerlegung von J .

Schritt 2 Berechne $F := F(x_k)$.

Schritt 3 Berechne p_k mit $Jp_k = -F$ mittels Vorwärts- und Rückwärtssubstitution.

Schritt 4 $x_{k+1} := x_k + p_k$, $k := k + 1$

```
if Konvergenz then stop
              else goto Schritt 1
```

x_{k+1} ist Nullstelle der linearisierten Funktion $F_k(x) := F(x_k) + F_x(x_k)(x - x_k) \approx F(x)$.

praktisch Berechne Ableitungen $F_x(x_k)$ analytisch unter Verwendung mathematischer Hilfsprogramme (Maple, Mathematica) oder numerisch mittels Differenzenquotienten, vgl. Beispiel 5.7.

Vereinfachtes Newtonverfahren Vermeide die häufige Neuberechnung und LU–Zerlegung der Jacobimatrix, indem J über mehrere Iterationsschritte konstant gehalten wird.

? Auswirkungen auf das Konvergenzverhalten und auf die Konvergenzgeschwindigkeit

? Kriterium für die Neuberechnung der Jacobimatrix

Beispiel 6.8: Vereinfachtes Newtonverfahren

gesucht Kleinste positive Lösung von $f(x) = \cos x \cosh x + 1 = 0$.

Vereinfachtes Newtonverfahren $x_{k+1} = x_k - \dfrac{f(x_k)}{f'(x_0)}$

Startwert $x_0 = 2.0$: (x_k) konvergiert linear mit $\alpha \approx 0.16$.

k	x_k	$f(x_k)$	$f'(x_k)$	$\lvert x_k - x^* \rvert$
0	2.000000000000000	-5.6563_E-01	-4.9303	1.2490_E-01
1	1.885274674997890	-4.2402_E-02		1.0171_E-02
2	1.876674249774155	-6.5051_E-03		1.5702_E-03
3	1.875354824372530	-1.0379_E-03		2.5076_E-04
4	1.875144317977280	-1.6656_E-04		4.0249_E-05
5	1.875110534418510	-2.6756_E-05		6.4657_E-06
6	1.875105107508024	-4.2987_E-06		1.0388_E-06
7	1.875104235610924	-6.9065_E-07		1.6690_E-07
8	1.875104095527000	-1.1096_E-07		2.6815_E-08
9	1.875104073020237	-1.7828_E-08		4.3083_E-09
10	1.875104069404156	-2.8644_E-09		6.9220_E-10

Abbildung 6.2: Lineare Konvergenz des vereinfachten Newtonverfahrens, einmalige Auswertung der Ableitung $f'(x_k)$, Startwert $x_0 = 2.0$.

Beispiel 6.8: Vereinfachtes Newtonverfahren (II)

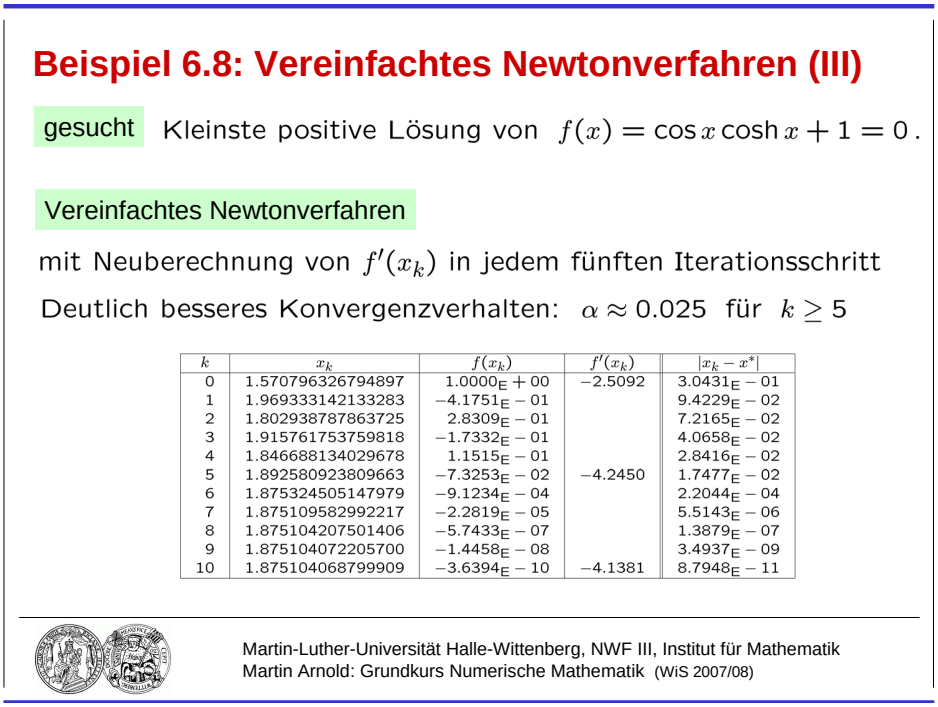
gesucht Kleinste positive Lösung von $f(x) = \cos x \cosh x + 1 = 0$.

Vereinfachtes Newtonverfahren $x_{k+1} = x_k - \dfrac{f(x_k)}{f'(x_0)}$

Startwert $x_0 = \pi/2$: (x_k) konvergiert linear mit $\alpha \approx 0.65$.

k	x_k	$f(x_k)$	$f'(x_k)$	$\lvert x_k - x^* \rvert$
0	1.570796326794897	1.0000_E+00	-2.5092	3.0431_E-01
1	1.969333142133283	-4.1751_E-01		9.4229_E-02
2	1.802938787863725	2.8309_E-01		7.2165_E-02
3	1.915761753759818	-1.7332_E-01		4.0658_E-02
4	1.846688134029678	1.1515_E-01		2.8416_E-02
5	1.892580923809663	-7.3253_E-02		1.7477_E-02
6	1.863386753854851	4.8072_E-02		1.1717_E-02
7	1.882545193155160	-3.0961_E-02		7.4411_E-03
8	1.870206143078939	2.0195_E-02		4.8979_E-03
9	1.878254787141379	-1.3068_E-02		3.1507_E-03
10	1.873046598671781	8.5012_E-03		2.0575_E-03

Abbildung 6.3: Lineare Konvergenz des vereinfachten Newtonverfahrens, einmalige Auswertung der Ableitung $f'(x_k)$, Startwert $x_0 = \pi/2$.



k	x_k	$f(x_k)$	$f'(x_k)$	$\lvert x_k - x^*\rvert$
0	1.570796326794897	$1.0000_{\mathrm{E}} + 00$	−2.5092	$3.0431_{\mathrm{E}} - 01$
1	1.969333142133283	$-4.1751_{\mathrm{E}} - 01$		$9.4229_{\mathrm{E}} - 02$
2	1.802938787863725	$2.8309_{\mathrm{E}} - 01$		$7.2165_{\mathrm{E}} - 02$
3	1.915761753759818	$-1.7332_{\mathrm{E}} - 01$		$4.0658_{\mathrm{E}} - 02$
4	1.846688134029678	$1.1515_{\mathrm{E}} - 01$		$2.8416_{\mathrm{E}} - 02$
5	1.892580923809663	$-7.3253_{\mathrm{E}} - 02$	−4.2450	$1.7477_{\mathrm{E}} - 02$
6	1.875324505147979	$-9.1234_{\mathrm{E}} - 04$		$2.2044_{\mathrm{E}} - 04$
7	1.875109582992217	$-2.2819_{\mathrm{E}} - 05$		$5.5143_{\mathrm{E}} - 06$
8	1.875104207501406	$-5.7433_{\mathrm{E}} - 07$		$1.3879_{\mathrm{E}} - 07$
9	1.875104072205700	$-1.4458_{\mathrm{E}} - 08$		$3.4937_{\mathrm{E}} - 09$
10	1.875104068799909	$-3.6394_{\mathrm{E}} - 10$	−4.1381	$8.7948_{\mathrm{E}} - 11$

Abbildung 6.4: Lineare Konvergenz des vereinfachten Newtonverfahrens, Auswertung der Ableitung $f'(x_k)$ in jedem fünften Iterationsschritt, Startwert $x_0 = \pi/2$.

Beispiel 6.8 (Vereinfachtes Newtonverfahren)
Betrachte zur Funktion $f(x)$ aus Beispiel 6.6 das vereinfachte Newtonverfahren

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_0)}\,.$$

Startwert $x_0 = 2.0$ (x_k) konvergiert linear mit $\alpha \approx 0.16$: $|x_{k+1} - x^*| \approx 0.16|x_k - x^*|$.

k	x_k	$f(x_k)$	$f'(x_k)$	$\lvert x_k - x^*\rvert$
0	2.000000000000000	$-5.6563_{\mathrm{E}} - 01$	−4.9303	$1.2490_{\mathrm{E}} - 01$
1	1.885274674997890	$-4.2402_{\mathrm{E}} - 02$		$1.0171_{\mathrm{E}} - 02$
2	1.876674249774155	$-6.5051_{\mathrm{E}} - 03$		$1.5702_{\mathrm{E}} - 03$
3	1.875354824372530	$-1.0379_{\mathrm{E}} - 03$		$2.5076_{\mathrm{E}} - 04$
4	1.875144317977280	$-1.6656_{\mathrm{E}} - 04$		$4.0249_{\mathrm{E}} - 05$
5	1.875110534418510	$-2.6756_{\mathrm{E}} - 05$		$6.4657_{\mathrm{E}} - 06$
6	1.875105107508024	$-4.2987_{\mathrm{E}} - 06$		$1.0388_{\mathrm{E}} - 06$
7	1.875104235610924	$-6.9065_{\mathrm{E}} - 07$		$1.6690_{\mathrm{E}} - 07$
8	1.875104095527000	$-1.1096_{\mathrm{E}} - 07$		$2.6815_{\mathrm{E}} - 08$
9	1.875104073020237	$-1.7828_{\mathrm{E}} - 08$		$4.3083_{\mathrm{E}} - 09$
10	1.875104069404156	$-2.8644_{\mathrm{E}} - 09$		$6.9220_{\mathrm{E}} - 10$

Startwert $x_0 = \pi/2$ (x_k) konvergiert linear mit $\alpha \approx 0.65$: $|x_{k+1} - x^*| \approx 0.65|x_k - x^*|$.

k	x_k	$f(x_k)$	$f'(x_k)$	$\lvert x_k - x^*\rvert$
0	1.570796326794897	$1.0000_{\mathrm{E}} + 00$	-2.5092	$3.0431_{\mathrm{E}} - 01$
1	1.969333142133283	$-4.1751_{\mathrm{E}} - 01$		$9.4229_{\mathrm{E}} - 02$
2	1.802938787863725	$2.8309_{\mathrm{E}} - 01$		$7.2165_{\mathrm{E}} - 02$
3	1.915761753759818	$-1.7332_{\mathrm{E}} - 01$		$4.0658_{\mathrm{E}} - 02$
4	1.846688134029678	$1.1515_{\mathrm{E}} - 01$		$2.8416_{\mathrm{E}} - 02$
5	1.892580923809663	$-7.3253_{\mathrm{E}} - 02$		$1.7477_{\mathrm{E}} - 02$
6	1.863386753854851	$4.8072_{\mathrm{E}} - 02$		$1.1717_{\mathrm{E}} - 02$
7	1.882545193155160	$-3.0961_{\mathrm{E}} - 02$		$7.4411_{\mathrm{E}} - 03$
8	1.870206143078939	$2.0195_{\mathrm{E}} - 02$		$4.8979_{\mathrm{E}} - 03$
9	1.878254787141379	$-1.3068_{\mathrm{E}} - 02$		$3.1507_{\mathrm{E}} - 03$
10	1.873046598671781	$8.5012_{\mathrm{E}} - 03$		$2.0575_{\mathrm{E}} - 03$

Startwert $x_0 = \pi/2$, Neuberechnung von $f'(x_k)$ in jedem 5. Iterationsschritt
Deutliche Verbesserung des Konvergenzverhaltens: $\alpha \approx 0.025$ für $k \geq 5$.

k	x_k	$f(x_k)$	$f'(x_k)$	$\lvert x_k - x^*\rvert$
0	1.570796326794897	$1.0000_{\mathrm{E}} + 00$	-2.5092	$3.0431_{\mathrm{E}} - 01$
1	1.969333142133283	$-4.1751_{\mathrm{E}} - 01$		$9.4229_{\mathrm{E}} - 02$
2	1.802938787863725	$2.8309_{\mathrm{E}} - 01$		$7.2165_{\mathrm{E}} - 02$
3	1.915761753759818	$-1.7332_{\mathrm{E}} - 01$		$4.0658_{\mathrm{E}} - 02$
4	1.846688134029678	$1.1515_{\mathrm{E}} - 01$		$2.8416_{\mathrm{E}} - 02$
5	1.892580923809663	$-7.3253_{\mathrm{E}} - 02$	-4.2450	$1.7477_{\mathrm{E}} - 02$
6	1.875324505147979	$-9.1234_{\mathrm{E}} - 04$		$2.2044_{\mathrm{E}} - 04$
7	1.875109582992217	$-2.2819_{\mathrm{E}} - 05$		$5.5143_{\mathrm{E}} - 06$
8	1.875104207501406	$-5.7433_{\mathrm{E}} - 07$		$1.3879_{\mathrm{E}} - 07$
9	1.875104072205700	$-1.4458_{\mathrm{E}} - 08$		$3.4937_{\mathrm{E}} - 09$
10	1.875104068799909	$-3.6394_{\mathrm{E}} - 10$	-4.1381	$8.7948_{\mathrm{E}} - 11$

Lemma 6.9 (Kontrahierende Abbildungen)
Sei $D \subset \mathbb{R}^n$ eine konvexe offene Menge und $\Phi : \bar{D} \to \mathbb{R}^n$ stetig differenzierbar. Existiert $\bar{\alpha} := \sup_{x \in \bar{D}} \|\Phi_x(x)\|$ und ist $\bar{\alpha} < 1$, so ist Φ kontrahierend*, d. h., es gibt ein $\alpha \in [0, 1)$ so, dass*

$$\|\Phi(y) - \Phi(x)\| \leq \alpha \|y - x\|\,, \ (\, x, y \in \bar{D}\,).$$

Beweis Für $\varphi(\vartheta) := \Phi(x + \vartheta(y - x))\,, \ (\, \vartheta \in [0, 1]\,)$ ist

$$\varphi'(\vartheta) = \Phi_x(x + \vartheta(y - x)) \cdot (y - x)$$

und

$$\varphi(1) - \varphi(0) = \int_0^1 \varphi'(\vartheta)\, \mathrm{d}\vartheta\,,$$

also

$$\|\Phi(y) - \Phi(x)\| \leq \int_0^1 \|\Phi_x(x + \vartheta(y - x))\| \cdot \|y - x\|\, \mathrm{d}\vartheta \leq \bar{\alpha} \|y - x\|\,. \quad \blacksquare$$

Satz 6.10 (Banachscher Fixpunktsatz)
Sei $E \subset \mathbb{R}^n$ kompakt und $\Phi : E \to E$ kontrahierend. Dann gilt:

a) Φ hat genau einen Fixpunkt x^* in E : $\Phi(x^*) = x^*$.

b) Für jeden Startwert $x_0 \in E$ konvergiert die Fixpunktiteration $x_{k+1} = \Phi(x_k)$ gegen x^* und es gilt:

(i) $\|x_k - x^*\| \leq \dfrac{\alpha^k}{1-\alpha} \|x_1 - x_0\|$,

(ii) $\|x_{k+1} - x^*\| \leq \dfrac{\alpha}{1-\alpha} \|x_{k+1} - x_k\|$.

Beweis Wegen $\Phi : E \to E$ folgt mittels vollständiger Induktion $x_k \in E$ und

$$\|x_{k+1} - x_k\| = \|\Phi(x_k) - \Phi(x_{k-1})\| \leq \alpha \|x_k - x_{k-1}\| \leq \ldots \leq \alpha^k \|x_1 - x_0\| , \ (k \geq 0) .$$

Aus der Dreiecksungleichung folgt

$$\begin{aligned} \|x_{k+m} - x_k\| &= \|x_{k+m} - x_{k+m-1} + x_{k+m-1} - x_{k+m-2} + x_{k+m-2} - + \ldots - x_k\| \\ &\leq \|x_{k+m} - x_{k+m-1}\| + \|x_{k+m-1} - x_{k+m-2}\| + \ldots + \|x_{k+1} - x_k\| \\ &\leq (\alpha^{k+m-1} + \alpha^{k+m-2} + \ldots + \alpha^k) \|x_1 - x_0\| \\ &\leq \alpha^k \sum_{i=0}^{\infty} \alpha^i \cdot \|x_1 - x_0\| = \frac{\alpha^k}{1-\alpha} \|x_1 - x_0\| , \end{aligned}$$

also ist $(x_k)_k$ eine Cauchy–Folge, denn wählt man zu vorgegebenem $\varepsilon > 0$ ein k_0 mit

$$\frac{\alpha^{k_0}}{1-\alpha} \|x_1 - x_0\| \leq \varepsilon ,$$

so gilt für alle $k \geq k_0$ und alle $m \geq 0$ die Abschätzung $\|x_{k+m} - x_k\| \leq \varepsilon$. Als Cauchy–Folge im Kompaktum E hat $(x_k)_k$ einen Häufungspunkt $x^* = \lim\limits_{k\to\infty} x_k \in E$ mit

$$\|x^* - x_k\| = \lim_{m\to\infty} \|x_{k+m} - x_k\| \leq \frac{\alpha^k}{1-\alpha} \|x_1 - x_0\| .$$

Dieser Häufungspunkt ist Fixpunkt von Φ, denn

$$\begin{aligned} \|x^* - \Phi(x^*)\| &= \|x^* - x_{k+1} + \Phi(x_k) - \Phi(x^*)\| \\ &\leq \|x^* - x_{k+1}\| + \|\Phi(x_k) - \Phi(x^*)\| \leq \|x^* - x_{k+1}\| + \alpha \|x_k - x^*\| \to 0 . \end{aligned}$$

Der Fixpunkt ist eindeutig bestimmt, denn aus $x_1^* = \Phi(x_1^*)$ und $x_2^* = \Phi(x_2^*)$ folgt

$$\|x_2^* - x_1^*\| = \|\Phi(x_2^*) - \Phi(x_1^*)\| \leq \alpha \|x_2^* - x_1^*\| ,$$

also $\underbrace{(1-\alpha)}_{>0} \|x_2^* - x_1^*\| \leq 0$ und $x_2^* = x_1^*$.

Zum Beweis von (ii) verwendet man

$$\begin{aligned} \|x_{k+1+m} - x_{k+m}\| &= \|\Phi(x_{k+m}) - \Phi(x_{k+m-1})\| \\ &\leq \alpha\|x_{k+m} - x_{k+m-1}\| \leq \ldots \leq \alpha^m\|x_{k+1} - x_k\|\,,\ (\,m \geq 1\,)\,, \end{aligned}$$

um wie oben unter Verwendung der Dreiecksungleichung die Abschätzung

$$\|x_{k+1+m} - x_{k+1}\| \leq (\alpha^m + \alpha^{m-1} + \ldots + \alpha)\,\|x_{k+1} - x_k\| \leq \frac{\alpha}{1-\alpha}\,\|x_{k+1} - x_k\|\,,\ (\,m \geq 1\,)\,,$$

zu zeigen, aus der die Behauptung durch Grenzübergang $m \to \infty$ folgt. ∎