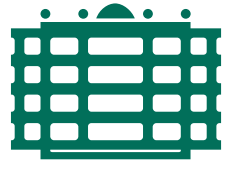
Mathematik II/2
Numerik

Oliver Ernst

Professur Numerische Mathematik

Sommersemester 2013



TECHNISCHE UNIVERSITÄT
CHEMNITZ

Inhalt

1. Einleitung
2. Computer-Arithmetik und Fehleranalyse
   - 2.1 Ein Beispiel
   - 2.2 Gleitpunktzahlen
   - 2.3 Rundung
   - 2.4 Gleitpunktarithmetik
   - 2.5 Numerische Stabilität und Fehleranalyse
3. Schnelle Fourier-Transformation
   - 3.1 Vorbemerkungen
   - 3.2 Überblick Fourier-Analyse
   - 3.3 Die diskrete Fourier-Transformation
   - 3.4 Interpolation
   - 3.5 Trigonometrische Interpolation
   - 3.6 Die FFT
4. Abschließende Bemerkungen

Inhalt

1 Einleitung

2 Computer-Arithmetik und Fehleranalyse
- 2.1 Ein Beispiel
- 2.2 Gleitpunktzahlen
- 2.3 Rundung
- 2.4 Gleitpunktarithmetik
- 2.5 Numerische Stabilität und Fehleranalyse

3 Schnelle Fourier-Transformation
- 3.1 Vorbemerkungen
- 3.2 Überblick Fourier-Analyse
- 3.3 Die diskrete Fourier-Transformation
- 3.4 Interpolation
- 3.5 Trigonometrische Interpolation
- 3.6 Die FFT

4 Abschließende Bemerkungen

Numerik

Begriffliches

- Numerik, numerische Mathematik, numerical analysis
- Teilgebiet der angewandten Mathematik
- Aufgabe der Numerik ist die Konstruktion und Analyse von Algorithmen zur Lösung mathematischer Probleme.
- Diese Probleme stammen ursprünglich aus Technik, Naturwissenschaften, Sozial- oder Wirtschaftswissenschaften, liegen aber in mathematischer Form, z.B. als Gleichungssystem, Differentialgleichung oder Optimierungsproblem vor.
- In den kommenden 3 Vorlesungen: drei Beispiele für numerische Verfahren für mathematische Grundaufgaben.
- Zuvor: Versuch einer Definition

  "... the theory of constructive methods of mathematical analysis."

Peter Henrici, Elements of Numerical Analysis

Numerik

Trefethen's Essay

- Essay von L. N. Trefethen (Oxford University)
- Computer spielen wichtige Rolle, aber keine bloße Formelauswertung.
- Algorithmen: Berechnungsmethoden zur Lösung mathematischer Aufgaben.
- Endlich vielen Schritte
  - lineare Gleichungssysteme
  - lineare Programmierung
  - Problem des Handlungsreisenden
- Unendlich viele Schritte, in jedem Schritt genauer
  - Eigenwerte einer Matrix
  - Minimierung multivariater Funktionen
  - Bestimmtes Integral
  - Lösung von Differentialgleichungen
- Fehleranalyse erforderlich, macht aber (entgegen landläufige Vorurteile) geringen Teil der Numerik aus.

Inhalt

1 Einleitung

2 Computer-Arithmetik und Fehleranalyse
- 2.1 Ein Beispiel
- 2.2 Gleitpunktzahlen
- 2.3 Rundung
- 2.4 Gleitpunktarithmetik
- 2.5 Numerische Stabilität und Fehleranalyse

3 Schnelle Fourier-Transformation
- 3.1 Vorbemerkungen
- 3.2 Überblick Fourier-Analyse
- 3.3 Die diskrete Fourier-Transformation
- 3.4 Interpolation
- 3.5 Trigonometrische Interpolation
- 3.6 Die FFT

4 Abschließende Bemerkungen

Inhalt

1 Einleitung

2 Computer-Arithmetik und Fehleranalyse

2.1 Ein Beispiel

2.2 Gleitpunktzahlen

2.3 Rundung

2.4 Gleitpunktarithmetik

2.5 Numerische Stabilität und Fehleranalyse

3 Schnelle Fourier-Transformation

3.1 Vorbemerkungen

3.2 Überblick Fourier-Analyse

3.3 Die diskrete Fourier-Transformation

3.4 Interpolation

3.5 Trigonometrische Interpolation

3.6 Die FFT

4 Abschließende Bemerkungen

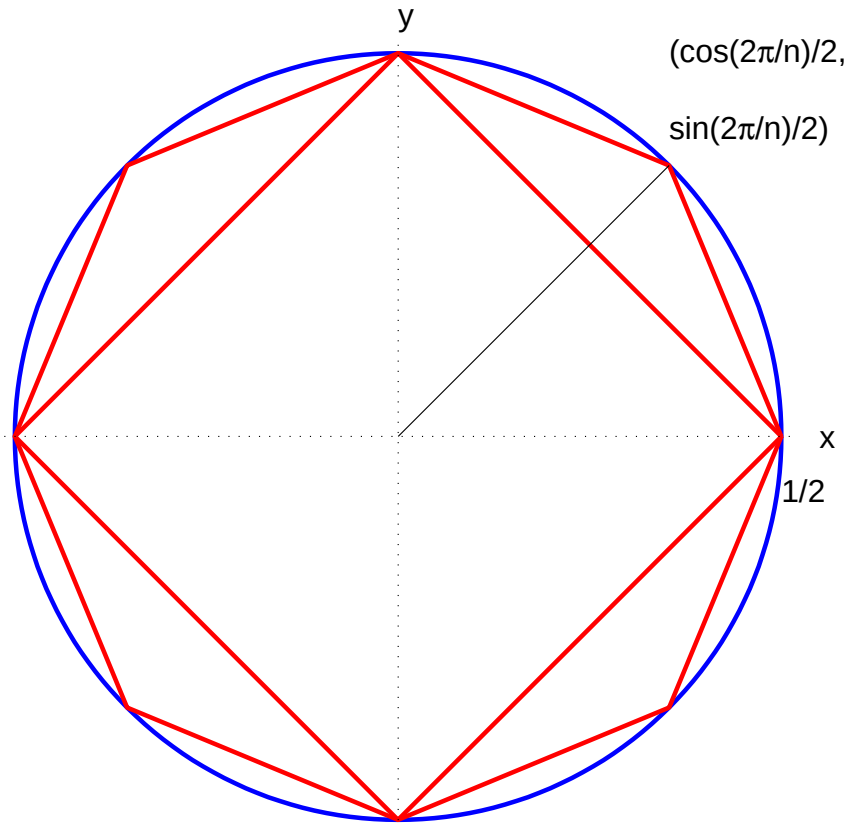
Einführendes Beispiel

Berechnung von π

π = Umfang eines Kreises mit Radius $r = \frac{1}{2}$,

U_n = Umfang eines einbeschriebenen regelmäßigen n-Ecks

$= n \sin(\pi/n)$.



Klar:

$$\lim_{n\to\infty} U_n = \lim_{n\to\infty} n \sin\frac{\pi}{n} = \pi \qquad \text{(unbrauchbar!)}$$

Einführendes Beispiel

Berechnung von π: Ein Algorithmus

Setze

$$A_n := U_{2^n} \qquad \text{(Umfang des regelmäßigen } 2^n\text{-Ecks)}.$$

Dann gelten:

$$A_2 = U_4 = 4\sqrt{\left(\tfrac{1}{2}\right)^2 + \left(\tfrac{1}{2}\right)^2} = 2\sqrt{2},$$

$$A_{n+1} = 2^n \sqrt{2\left(1 - \sqrt{1 - \left[\frac{A_n}{2^n}\right]^2}\right)}, \qquad n = 2, 3, \ldots\ ,$$

d.h. wir können A_{n+1} aus A_n rekursiv berechnen.

[Archimedes von Syrakus, 287–212 v. Chr.]: $A_3 = 3.06\ldots$,
$A_4 = 3.12\ldots$,
$A_5 = 3.14\ldots$.

Einführendes Beispiel

Berechnung von π: Eine Fehlerabschätzung

Zunächst gilt für $h > 0$:

$$|\sin(h) - h| \leq \frac{h^3}{6} \qquad \text{(Taylorformel)}.$$

Setze $h = \pi/N$ (und multipliziere mit N):

$$|N \sin(\pi/N) - \pi| \leq \frac{\pi^3}{6 \cdot N^2}.$$

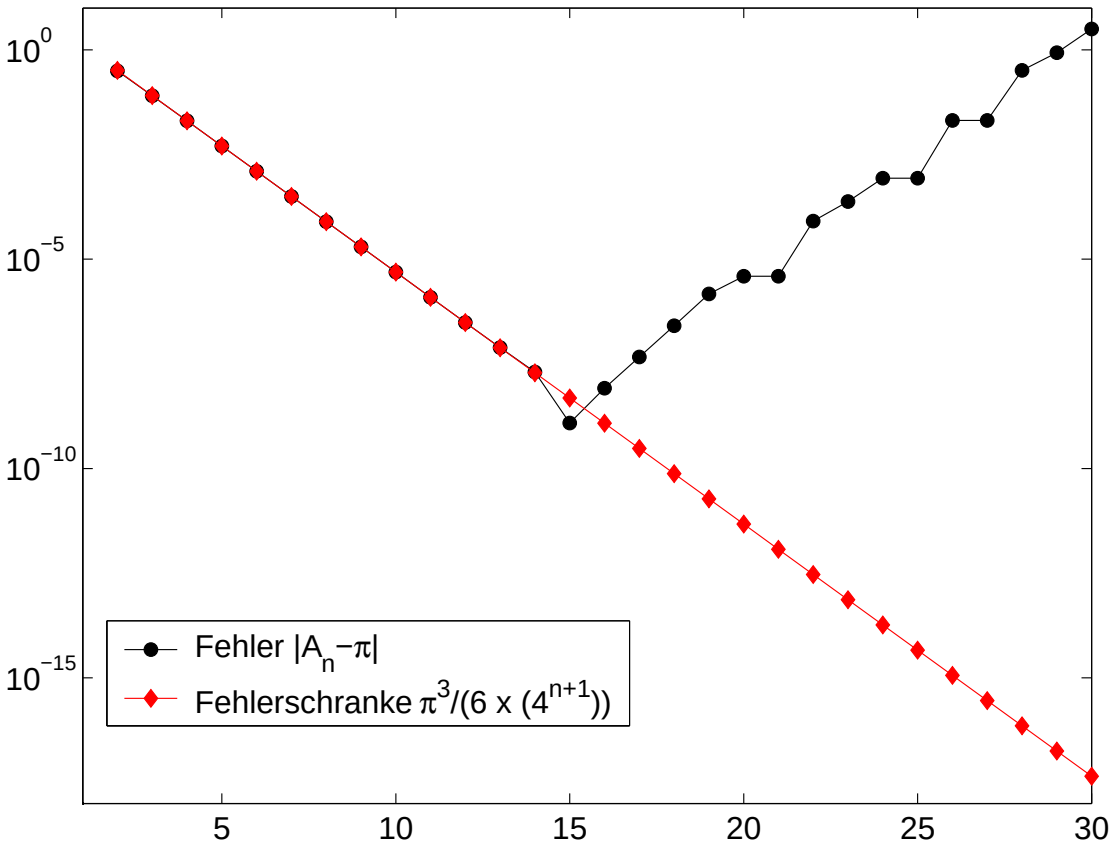
D.h. (mit $N = 2^n$):

$$|A_n - \pi| \leq \frac{\pi^3}{6 \cdot 4^n} \quad (< 10^{-10} \text{ für } n \geq 18).$$

Auf zum Rechner ...

Einführendes Beispiel

Berechnung von π: Ernüchterung



Die berechnete Folge $\{A_n\}$ verhält sich völlig anders als die „wirkliche“ Folge $\{A_n\}$! Wie ist das möglich?

Inhalt

1 Einleitung

2 Computer-Arithmetik und Fehleranalyse
- 2.1 Ein Beispiel
- 2.2 Gleitpunktzahlen
- 2.3 Rundung
- 2.4 Gleitpunktarithmetik
- 2.5 Numerische Stabilität und Fehleranalyse

3 Schnelle Fourier-Transformation
- 3.1 Vorbemerkungen
- 3.2 Überblick Fourier-Analyse
- 3.3 Die diskrete Fourier-Transformation
- 3.4 Interpolation
- 3.5 Trigonometrische Interpolation
- 3.6 Die FFT

4 Abschließende Bemerkungen

Gleitpunktzahlen

Gleitpunktzahlen sind rationale Zahlen der Form

$$\pm \, . \, d_1 d_2 d_3 \ldots d_t \cdot b^e \, , \qquad \text{wobei}$$

- $b \in \mathbb{N}$ $(b > 1)$ Basis,
- $. \, d_1 d_2 d_3 \ldots d_t$ Mantisse (zur Basis b) und
- $t \in \mathbb{N}$ Mantissenlänge genannt werden.

Die Ziffern $d_1, d_2, d_3, \ldots, d_t$ sind ganze Zahlen, die zwischen 0 und $b-1$ liegen. Der Exponent e ist eine ganze Zahl, die zwischen m und M liegt. Z.B. ist $x = . \, 10101 \cdot 2^{-1}$ eine Gleitpunktzahl zur Basis 2 mit dem Wert

$$\left(1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} + 0 \cdot 2^{-4} + 1 \cdot 2^{-5}\right) \cdot 2^{-1} = \frac{21}{64} = 0.328125.$$

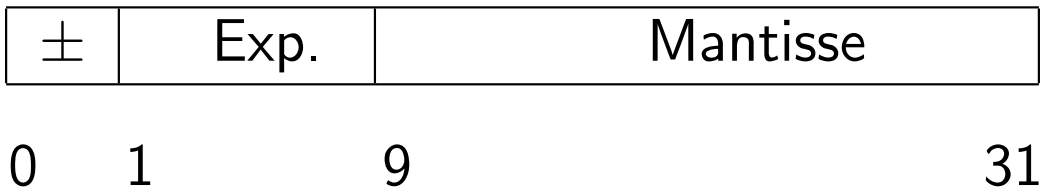
Gebräuchliche Basiswerte sind $b = 2$ (fast alle Computer), $b = 10$ (die Standardbasis des „täglichen Lebens“) und $b = 16$ (IBM-Mainframes).

Gleitpunktzahlen

IEEE Formate

Der IEEE-Standard (IEEE = Institute of Electrical and Electronics Engineers), der auf fast allen Maschinen realisiert ist, rechnet mit der Basis $b = 2$ und erlaubt i.W. zwei Datenformate:

Single (FORTRAN: REAL*4, C: float) = 1 Wort = 32 bits,

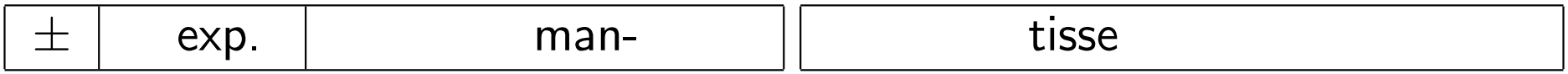


also 1 bit für das Vorzeichen der Mantisse, 8 bits für den Exponenten, der zwischen $1 - 2^7 = -127$ und $2^7 = 128$ liegt, und 23 bits für die Mantisse (ohne Vorzeichen).
Die Potenzen 2^e decken also den Bereich von $2^{-127} \sim 5 \cdot 10^{-39}$ bis $2^{128} \sim 3 \cdot 10^{38}$ ab. Der kleinste positive Wert der Mantisse ist $2^{-23} \sim 1.2 \cdot 10^{-7}$.

Gleitpunktzahlen

IEEE Formate

Double (FORTRAN: REAL*8, C: double) = 2 Worte = 64 bits,



0 1 12 32 63

1 bit für das Vorzeichen der Mantisse, 11 bits für den Exponenten und 52 bits für die Mantisse.
Die Potenzen 2^e bewegen sich folglich zwischen $2^{1-2^{10}} \sim 1.1 \cdot 10^{-308}$ und $2^{2^{10}} \sim 1.7 \cdot 10^{308}$. Der kleinste positive Mantissenwert ist $2^{-52} \sim 2.2 \cdot 10^{-16}$.

Drei weitere IEEE-„Zahlen“:

NAN (not a number), z.B. $1/0 = \text{INF}$, $-1/0 = -\text{INF}$

$\pm$ INF ($\pm\infty$), z.B. $0/0 = \text{NAN}$.

Gleitpunktzahlen

Beispielsystem

Normalisierte Gleitpunktzahl = erste Ziffer der Mantisse ungleich 0.

Beispiel: Basis = 2, Mantissenlänge = 4, Exponentenlänge = 3.
Das bedeutet:
Acht Exponenten: -3, -2, -1, 0, 1, 2, 3, 4.
Acht (normalisierte) Mantissen:

$$
\begin{array}{llll}
0.1000 & \hat{=} & 0.5, & 0.1001 \;\hat{=}\; 0.5625, \\
0.1010 & \hat{=} & 0.6250, & 0.1011 \;\hat{=}\; 0.6875, \\
0.1100 & \hat{=} & 0.7500, & 0.1101 \;\hat{=}\; 0.8125, \\
0.1110 & \hat{=} & 0.8750, & 0.1111 \;\hat{=}\; 0.9375.
\end{array}
$$

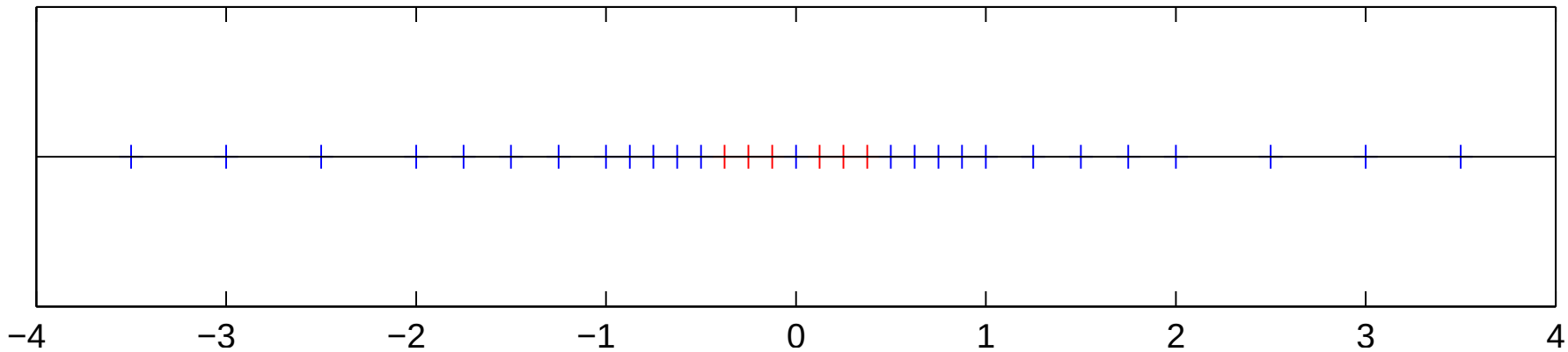
Damit enthält unser System $2 \cdot 8 \cdot 8 + 1 = 129$ normalisierte Gleitpunktzahlen.

Die kleinste positive ist $0.1000 \cdot 2^{-3} = 0.0625$, die größte ist $0.1111 \cdot 2^{4} = 15$.

Gleitpunktzahlen

Beispielsystem

Natürlich sind diese Gleitpunktzahlen nicht gleichmäßig verteilt:
so liegen etwa die 16 kleinsten, d.h. ein Viertel der positiven Gleitpunktzahlen, im Inyervall (0, 0.234375], während die größten 16, also wieder ein Viertel aller positiven Gleitpunktzahlen, das etwa vierzigmal längere Intervall [4,15] überdecken.



Gleitpunktzahlen

Über- und Unterlauf

Es gibt nur endlich viele zulässige Exponenten, was zu Under- und Overflow (Exponentenunterlauf bzw. -überlauf) führen kann.
Overflow (im Beispiel: $(.1 \cdot 2^3)(.1 \cdot 2^4)$) bedeutet i.a. Abbruch des Programms mit einer Fehlermeldung (IEEE-Arithmetik: +INF, $-$INF oder auch NAN). Beim Underflow (im Beispiel: $(.1 \cdot 2^{-3})(.1 \cdot 2^{-3})$) wird das Ergebnis normalerweise auf Null gesetzt und ohne Fehlermeldung weitergerechnet.
Overflow kann durch geeignete Skalierung oft — auf Kosten eines harmlosen Underflows — vermieden werden.
$c = \sqrt{a^2 + b^2}$ mit $a = 10^{60}$ und $b = 1$ (Rechnung mit vier Dezimalstellen in Mantisse und zwei Dezimalstellen im Exponent). Standardauswertung verursacht Overflow. Besser: $c = s\,\sqrt{(a/s)^2 + (b/s)^2}$ mit $s = \max\,(|a|,\ |b|)$.
Im weiteren: Over- und Underflow werden vernachlässigt, d.h. der Exponent darf beliebige (ganzzahlige) Werte annehmen.

Inhalt

1 Einleitung

2 Computer-Arithmetik und Fehleranalyse
- 2.1 Ein Beispiel
- 2.2 Gleitpunktzahlen
- 2.3 Rundung
- 2.4 Gleitpunktarithmetik
- 2.5 Numerische Stabilität und Fehleranalyse

3 Schnelle Fourier-Transformation
- 3.1 Vorbemerkungen
- 3.2 Überblick Fourier-Analyse
- 3.3 Die diskrete Fourier-Transformation
- 3.4 Interpolation
- 3.5 Trigonometrische Interpolation
- 3.6 Die FFT

4 Abschließende Bemerkungen

Rundung

Rundung zur nächstgelegenen Maschinenzahl

$\mathbb{M}$ = Menge der Maschinenzahlen.

Runden zur nächstgelegenen Maschinenzahl

$$\mathsf{rd} : \mathbb{R} \to \mathbb{M}, \qquad x \mapsto \mathsf{rd}\,(x),$$

wird wie folgt realisiert: Schreibe

$$x = \pm.d_1 d_2 \ldots d_{t-1} d_t d_{t+1} \ldots \cdot 10^e$$

mit möglicherweise unendlich langer Mantisse (der Einfachheit halber arbeiten wir mit der Basis 10) und setze

$$\mathsf{rd}\,(x) := \begin{cases} \pm.d_1 d_2 \ldots d_{t-1} d_t \cdot 10^e, & \text{falls } d_{t+1} \leq 4, \\ \pm.d_1 d_2 \ldots d_{t-1} (d_t + 1) \cdot 10^e, & \text{falls } d_{t+1} \geq 5. \end{cases}$$

Rundung

Rundung zur nächstgelegenen Maschinenzahl

Ist $d_t = 9$, so entsteht ein Übertrag und d_{t-1}, möglicherweise auch $d_{t-2}, \ldots$ sowie e, müssen modifiziert werden.

Für $t = 4$ gilt etwa

$$\mathsf{rd}\,(0.44499) = +.4450 \cdot 10^0 \quad \text{und} \quad \mathsf{rd}\,(0.99999) = +.1000 \cdot 10^1.$$

Relativer Fehler der Rundung von $x = \pm a \cdot 10^e \in \mathbb{R} \setminus \{0\}$ $(0.1 \le a < 1)$:

$$\left| \frac{\mathsf{rd}(x) - x}{x} \right| \le \left| \frac{(0.5 \cdot 10^{-t})10^e}{a \cdot 10^e} \right| \le 5 \cdot 10^{-t} =: \mathsf{eps},$$

eps heißt **Maschinengenauigkeit**. Anders formuliert

$$\mathsf{rd}\,(x) = (1+\varepsilon)x \quad \text{mit } |\varepsilon| \le \mathsf{eps}.$$

Frage: Was ist die größte reelle Zahl, die auf 0 gerundet wird?

Inhalt

1 Einleitung

2 Computer-Arithmetik und Fehleranalyse
- 2.1 Ein Beispiel
- 2.2 Gleitpunktzahlen
- 2.3 Rundung
- 2.4 Gleitpunktarithmetik
- 2.5 Numerische Stabilität und Fehleranalyse

3 Schnelle Fourier-Transformation
- 3.1 Vorbemerkungen
- 3.2 Überblick Fourier-Analyse
- 3.3 Die diskrete Fourier-Transformation
- 3.4 Interpolation
- 3.5 Trigonometrische Interpolation
- 3.6 Die FFT

4 Abschließende Bemerkungen

Gleitpunktarithmetik

Die Machinenzahlen $\mathbb{M}$ sind bez. der Grundrechenarten (Addition, Subtraktion, Multiplikation und Division) nicht abgeschlossen (selbst wenn wir für die Exponenten beliebige Werte erlauben).

Z.B. ist

$$x = .11 \cdot 10^0$$

eine Gleitpunktzahl zur Basis 10 mit Mantissenlänge 2, während

$$x \cdot x = 0.121 \cdot 10^{-1}$$

eine dreistellige Mantisse besitzt.

Für jede der Operationen $\circ \in \{+, -, \cdot, /\}$ wird für die Implementierung der entsprechenden Gleitpunktoperation gefordert, dass

$$\mathsf{fl}\,(x \circ y) := \mathsf{rd}\,(x \circ y), \qquad x, y \in \mathbb{M}.$$

Gleitpunktarithmetik

Für alle $x, y \in \mathbb{M}$ gilt somit, falls weder Unter- noch Überlauf auftritt,

$$\mathsf{fl}\,(x \circ y) = (1+\varepsilon)(x \circ y) \text{ mit } \ |\varepsilon| \leq \text{eps}.$$

Das Hauptproblem dieser Semantik ist, dass die neuen Operationen den klassischen Gesetzen der Arithmetik (wie etwa den Kommutativ-, Assoziativ- und Distributivgesetzen) nicht mehr genügen.

Z.B. in vierstelliger Gleitpunktarithmetik zur Basis 10:

$$\begin{aligned} x &= 0.1234 \cdot 10^4 \in \mathbb{M}, \\ y &= 0.1234 \cdot 10^0 \in \mathbb{M}, \\ x+y &= 0.12341234 \cdot 10^4, \quad \text{d.h.} \quad \mathsf{fl}\,(x+y) = 0.1234 \cdot 10^4, \end{aligned}$$

was $\mathsf{fl}\,(x+y) = x$ bedeutet, obwohl $\mathsf{rd}\,(y) \neq 0$.

Gleitpunktarithmetik

Frage:
Welchen Wert `x` liefert das Programm

```
x = 1;
while 1+x ~= 1,
      x=x/2;
end while
output x
```

in t-stelliger Gleitpunktarithmetik (zur Basis 2) ?

Inhalt

1 Einleitung

2 Computer-Arithmetik und Fehleranalyse
- 2.1 Ein Beispiel
- 2.2 Gleitpunktzahlen
- 2.3 Rundung
- 2.4 Gleitpunktarithmetik
- 2.5 Numerische Stabilität und Fehleranalyse

3 Schnelle Fourier-Transformation
- 3.1 Vorbemerkungen
- 3.2 Überblick Fourier-Analyse
- 3.3 Die diskrete Fourier-Transformation
- 3.4 Interpolation
- 3.5 Trigonometrische Interpolation
- 3.6 Die FFT

4 Abschließende Bemerkungen

Es sei

$$\hat{y} = \mathrm{fl}(f(x))$$

das in Gleitpunktarithmetik berechnete Ergebnis einer Funktion $y = f(x)$.

Wie beurteilt man die Qualität von $\hat{y}$?

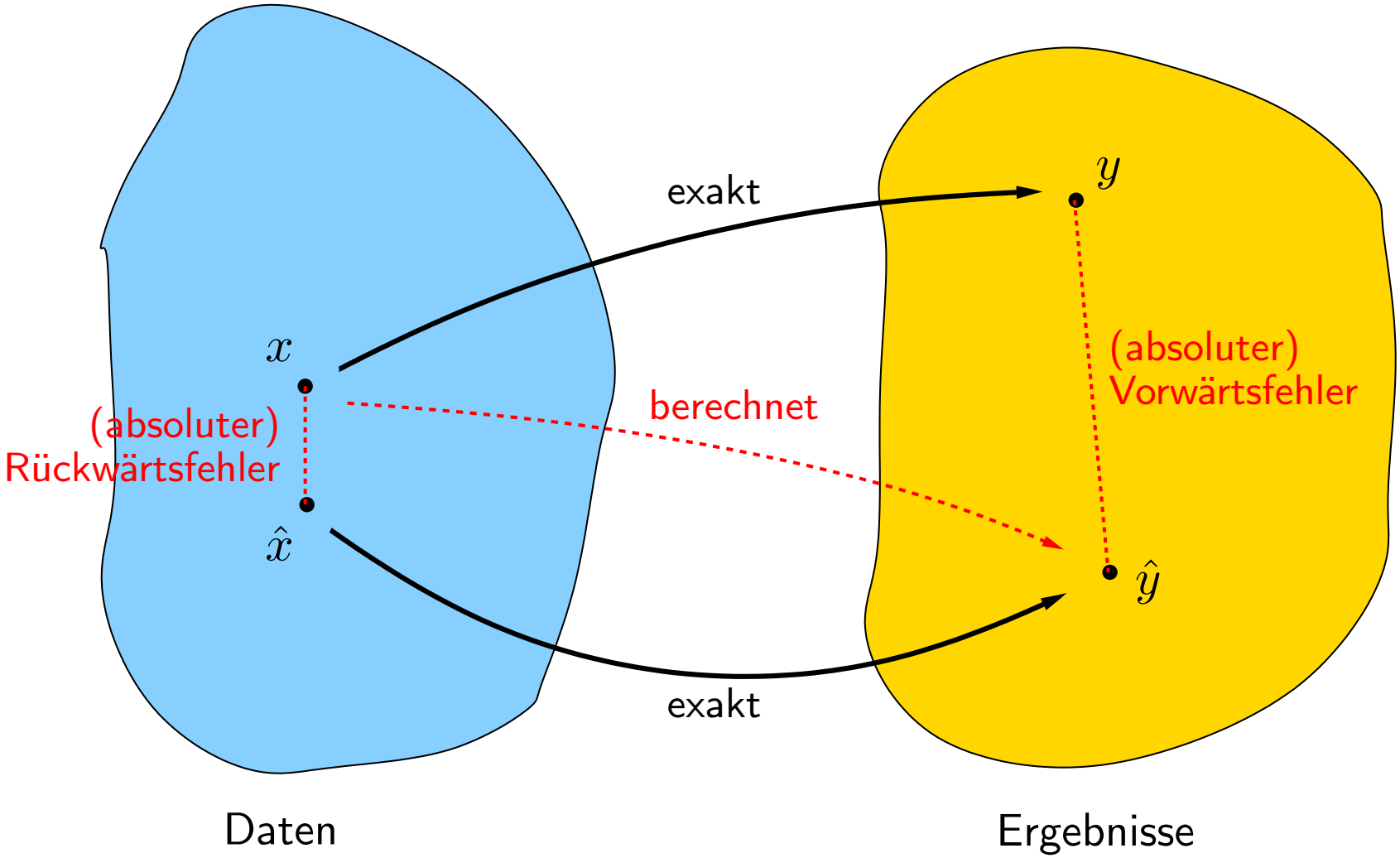
- (Relativer) Vorwärtsfehler: $|(y - \hat{y})/y|$.
- (Relativer) Rückwärtsfehler: $|(x - \hat{x})/x|$, dabei ist $\hat{x}$ das (ein) Eingabedatum, das bei rundungsfreier Rechnung zu $\hat{y}$ führt: $f(\hat{x}) = \hat{y}$ (Rundungsfehler werden als Datenfehler interpretiert).

Mit Störungstheorie kann man Vorwärtsfehler durch Rückwärtsfehler abschätzen.

Faustregel: Vorwärtsfehler $\lessapprox$ Konditionszahl $\times$ Rückwärtsfehler.

Numerische Stabilität und Fehleranalyse

Vorwärts- und Rückwärtsfehler



Ein Algorihmus heißt

- vorwärtsstabil, wenn der Vorwärtsfehler „klein“ ist,
- rückwärtsstabil, wenn der Rückwärtsfehler „klein“ ist.

Was „klein“ bedeutet, hängt vom Problem und der Maschinengenauigkeit ab.

Die Kondition(szahl) eines Problems (hat nichts mit Gleitpunktarithmetik zu tun!!) ist ein Maß dafür, wie empfindlich das Ergebnis auf Störungen der Daten reagiert.

Ein Problem ist gut (schlecht) konditioniert, wenn seine Konditionszahl klein (groß) ist.

Bestimme

$$y = f(x).$$

Störung der Daten: Δx

$$\hat{y} = f(x + \Delta x) = f(x) + f'(x)\Delta x + \frac{1}{2} f''(\xi)(\Delta x)^2.$$

Δx klein: $\hat{y} = f(x + \Delta x) \approx f(x) + f'(x)\Delta x = y + f'(x)\Delta x$ oder

$$\left| \frac{f(x + \Delta x) - f(x)}{f(x)} \right| = \left| \frac{\hat{y} - y}{y} \right| \approx \left| \frac{x f'(x)}{f(x)} \right| \left| \frac{\Delta x}{x} \right|.$$

(Relative) Konditionszahl von f an der Stelle x:

$$c_f(x) = \left| \frac{x f'(x)}{f(x)} \right|.$$

Beispiel. $f(x) = \log(x)$, d.h.: $c_f(x) = \left|\frac{x/x}{\log(x)}\right| = \left|\frac{1}{\log(x)}\right|$ moderat für sehr kleine und sehr große (positive) x, riesig für $x \approx 1$.

$x_1 = 0.01$: $\quad c_f(x_1) = 0.21715,$

$x_2 = 0.99$: $\quad c_f(x_2) = 99.4992,$

$x_3 = 100.$: $\quad c_f(x_3) = 0.21715.$

Wie wirkt sich eine relative Störung von $\varepsilon_x = (\Delta x)/x = 0.001$ aus?

Prognose:

$$\left|\frac{f(x_k + 0.001x_k) - f(x_k)}{f(x_k)}\right| \approx 0.001\, c_f(x_k) = 0.001 \left|\frac{1}{\log(x_k)}\right|.$$

k	rel. Fehler	Prognose
1	$2.1704 \cdot 10^{-4}$	$2.1715 \cdot 10^{-4}$
2	$9.9945 \cdot 10^{-2}$	$9.9499 \cdot 10^{-2}$
3	$2.1704 \cdot 10^{-4}$	$2.1715 \cdot 10^{-4}$

Allgemeiner: $y = f(x_1, x_2, \ldots, x_n)$.
Absolute Störungen der Daten, Δx_k $(k = 1, 2, \ldots, n)$, verursachen **absoluten Fehler** im Ergebnis:

$$\begin{aligned} \Delta y &= f(x_1 + \Delta x_1, \ldots, x_n + \Delta x_n) - f(x_1, \ldots, x_n) \sim \sum_{k=1}^{n} d_k \Delta x_k, \\ d_k &= \frac{\partial f(x_1, x_2, \ldots, x_n)}{\partial x_k} \quad \text{(absolute Konditionszahlen von } f\text{)}. \end{aligned}$$

Relative Störungen der Daten, $\varepsilon_k = \Delta x_k / x_k$ $(k = 1, 2, \ldots, n)$, verursachen **relativen Fehler** im Ergebnis:

$$\begin{aligned} \varepsilon_y &= \frac{f(x_1 + \Delta x_1, \ldots, x_n + \Delta x_n) - f(x_1, \ldots, x_n)}{f(x_1, \ldots, x_n)} \sim \sum_{k=1}^{n} c_k \varepsilon_k, \\ c_k &= \frac{x_k}{f(x_1, x_2, \ldots, x_n)} \frac{\partial f(x_1, x_2, \ldots, x_n)}{\partial x_k} \end{aligned}$$

((relative) Konditionszahlen von f).

Beispiele. (Grundoperationen)

- $y = f(x_1, x_2) = x_1 \cdot x_2$. D.h. $c_1 = 1$ und $c_2 = 1$ (unproblematisch).
- $y = f(x_1, x_2) = x_1/x_2$. D.h. $c_1 = 1$ und $c_2 = -1$ (unproblematisch).
- $y = f(x_1, x_2) = x_1 + x_2$. D.h. $c_1 = x_1/(x_1 + x_2)$ und $c_2 = x_2/(x_1 + x_2)$.
- $y = f(x_1, x_2) = x_1 - x_2$. D.h. $c_1 = x_1/(x_1 - x_2)$ und $c_2 = -x_2/(x_1 - x_2)$.

Bei den Operationen $\pm$ können die Konditionszahlen riesig werden:

$x_1 \approx -x_2$: Addition schlecht konditioniert.

$x_1 \approx x_2$: Subtraktion schlecht konditioniert.

Man spricht hier von Auslöschung.

Etwa:

$$x_1 = 3.14159, \qquad \Delta x_1 = 10^{-5}, \qquad \text{d.h. } \varepsilon_1 \approx 3.18 \cdot 10^{-6},$$
$$x_2 = 3.14140, \qquad \Delta x_2 = 2 \cdot 10^{-5}, \qquad \text{d.h. } \varepsilon_2 \approx 6.36 \cdot 10^{-6}.$$

Dann ist

$$y = x_1 - x_2 = 0.00019 \quad \text{(Auslöschung führender Ziffern).}$$

$$(x_1 + \Delta x_1) - (x_2 + \Delta x_2) = 0.00018, \quad \text{also} \quad \varepsilon_y = 5.26 \cdot 10^{-2}.$$

Prognose:

$$c_1 = 1.65 \cdot 10^4, \quad c_2 = -1.65 \cdot 10^4, \quad |\varepsilon_y| \approx |c_1 10^{-5} + c_2 2 \cdot 10^{-5}| = 1.65 \cdot 10^{-1}.$$

Die quadratische Gleichung

$$x^2 - bx + c = 0$$

besitzt die Lösungen

$$x_{1/2} = \frac{b \pm \sqrt{b^2 - 4c}}{2}.$$

Für

$$b = 3.6678 \quad \text{und} \quad c = 2.0798 \cdot 10^{-2}$$

erhält man nach Rechnung mit fünfstelliger Dezimalmantisse

$$\tilde{x}_1 = 3.6673 \qquad \text{(rel. Fehler: } 4.7 \cdot 10^{-6}\text{)},$$
$$\tilde{x}_2 = 5.5 \cdot 10^{-4} \qquad \text{(rel. Fehler: } 3.0 \cdot 10^{-2}\text{)}.$$

Wiseo?

Schritt	Ergebnis	rel. Fehler
1. b^2	$1.3453 \cdot 10^{+1}$	$1.8 \cdot 10^{-5}$
2. $4c$	$8.3192 \cdot 10^{-3}$	0.0
3. $b^2 - 4c$	$1.3445 \cdot 10^{+1}$	$4.1 \cdot 10^{-5}$
4. $\sqrt{b^2 - 4c}$	$3.6667 \cdot 10^{+0}$	$9.3 \cdot 10^{-6}$
5. $b - \sqrt{b^2 - 4c}$	$1.1000 \cdot 10^{-3}$	$3.0 \cdot 10^{-2}$
6. $(b - \sqrt{b^2 - 4c})/2$	$5.5000 \cdot 10^{-4}$	$3.0 \cdot 10^{-2}$
5'. $b + \sqrt{b^2 - 4c}$	$7.3345 \cdot 10^{+0}$	$4.7 \cdot 10^{-6}$
6'. $(b + \sqrt{b^2 - 4c})/2$	$3.6673 \cdot 10^{+0}$	$4.7 \cdot 10^{-6}$
7. $x_2 = c/x_1$	$5.6713 \cdot 10^{-4}$	$1.1 \cdot 10^{-6}$

Beachte: Nach dem Vietaschen Wurzelsatz ist $x_1 x_2 = c$.

Einführendes Beispiel

Berechnung von π

Beispiel aus der Einleitung dieses Kapitels:

$$A_{n+1} = 2^n \left[2 \underbrace{\left(1 - \sqrt{1 - (A_n/2^n)^2}\right)}_{\text{Auslöschung!}}\right]^{1/2}.$$

Setze

$$R_n := 4\,\frac{1 - \sqrt{1 - (A_n/2^n)^2}}{2}, \text{ d.h. } A_{n+1} = 2^n\sqrt{R_n}.$$

Beachte: $R_n = 4Z_n$ und Z_n ist (die kleinere) Lösung von

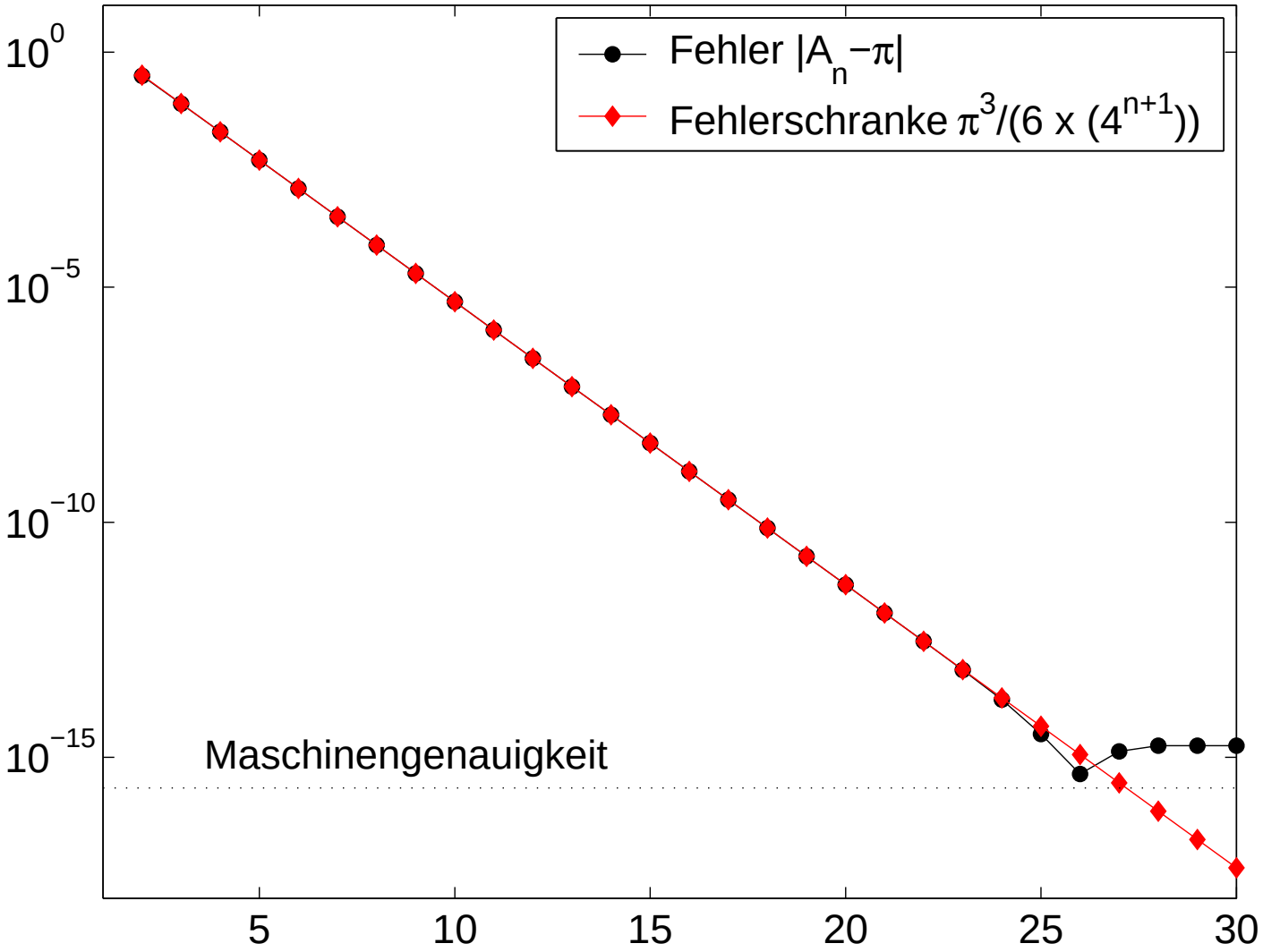
$$X^2 - X + \frac{1}{4}\,(A_n/2^n)^2 = X^2 - X + (A_n/2^{n+1})^2 = 0.$$

Stabile Berechnung der kleineren Lösung (wie oben):

$$Z_n = \frac{2(A_n/2^{n+1})^2}{1 + \sqrt{1 - (A_n/2^n)^2}}, \quad R_n = 4Z_n, \quad A_{n+1} = 2^n\sqrt{R_n}.$$

Einführendes Beispiel

Berechnung von π



Einführendes Beispiel

Berechnung von π

Schneller, weiter, höher?

Zur Berechnung von π gibt Verfahren, die noch schneller konvergieren und dabei beliebig viele (Millionen!) korrekte Nachkommastellen liefern. Wer darüber mehr erfahren oder mit solchen Verfahren in Matlab experimentieren möchte, dem sei dieser kleine Artikel des „Vaters“ von Matlab empfohlen.

Inhalt

1 Einleitung

2 Computer-Arithmetik und Fehleranalyse
- 2.1 Ein Beispiel
- 2.2 Gleitpunktzahlen
- 2.3 Rundung
- 2.4 Gleitpunktarithmetik
- 2.5 Numerische Stabilität und Fehleranalyse

3 Schnelle Fourier-Transformation
- 3.1 Vorbemerkungen
- 3.2 Überblick Fourier-Analyse
- 3.3 Die diskrete Fourier-Transformation
- 3.4 Interpolation
- 3.5 Trigonometrische Interpolation
- 3.6 Die FFT

4 Abschließende Bemerkungen

Inhalt

1 Einleitung

2 Computer-Arithmetik und Fehleranalyse
- 2.1 Ein Beispiel
- 2.2 Gleitpunktzahlen
- 2.3 Rundung
- 2.4 Gleitpunktarithmetik
- 2.5 Numerische Stabilität und Fehleranalyse

3 Schnelle Fourier-Transformation
- 3.1 Vorbemerkungen
- 3.2 Überblick Fourier-Analyse
- 3.3 Die diskrete Fourier-Transformation
- 3.4 Interpolation
- 3.5 Trigonometrische Interpolation
- 3.6 Die FFT

4 Abschließende Bemerkungen

Die schnelle Fourier-Transformation

Relevanz

The FFT is one of the truly great computational developments of [the 20th] century. It has changed the face of science and engineering so much that it is not an exaggeration to say that life as we know it would be very different without the FFT.

Charles van Loan, Computational Frameworks of the FFT, 1992

The Fast Fourier Transform—the most valuable numerical algorithm in our lifetime.

Gil Strang, Introduction to Linear Algebra, 2003

FFT voted one of the top 10 algorithms of the 20th century..

IEEE Computing in Science & Engineering, 1-2/2000

Die diskrete Fourier-Transformation

Geschichtliches

- Fourier-Reihen: Lagrange (1759), Fourier (1807)
  Problem der schwingenden Saite, Bewegung von Himmelskörpern
- Trigonometrische Interpolation, Gauss (1805)
- Harmonische Analysis: Clairaut, Bernoulli, d'Alembert, ...
- Cooley & Tukey, 1965
- Anwendungen in digitaler Signalverarbeitung, Bildverarbeitung, Numerik partieller Differentialgleichungen, inverse Probleme.
- Weiterentwicklung: Wavelets

Inhalt

1 Einleitung

2 Computer-Arithmetik und Fehleranalyse
- 2.1 Ein Beispiel
- 2.2 Gleitpunktzahlen
- 2.3 Rundung
- 2.4 Gleitpunktarithmetik
- 2.5 Numerische Stabilität und Fehleranalyse

3 Schnelle Fourier-Transformation
- 3.1 Vorbemerkungen
- 3.2 Überblick Fourier-Analyse
- 3.3 Die diskrete Fourier-Transformation
- 3.4 Interpolation
- 3.5 Trigonometrische Interpolation
- 3.6 Die FFT

4 Abschließende Bemerkungen

Kontinuierliche Fourier-Transformation

Für eine Funktion $u : \mathbb{R} \to \mathbb{R}$ ist die Fourier-Transformierte $\hat{u} : \mathbb{R} \to \mathbb{R}$ definiert durch

$$\hat{u}(\omega) := \frac{1}{\sqrt{2\pi}} \int_{\mathbb{R}} u(x) e^{-i\omega x}\, dx.$$

Die inverse Fourier-Transformation

$$u(x) = \frac{1}{\sqrt{2\pi}} \int_{\mathbb{R}} \hat{u}(\omega) e^{i\omega x}\, d\omega$$

stellt die ursprüngliche Funktion u dar als Überlagerung von Wellen $e^{i\omega x}$ der Amplitude $\hat{u}(\omega)$.

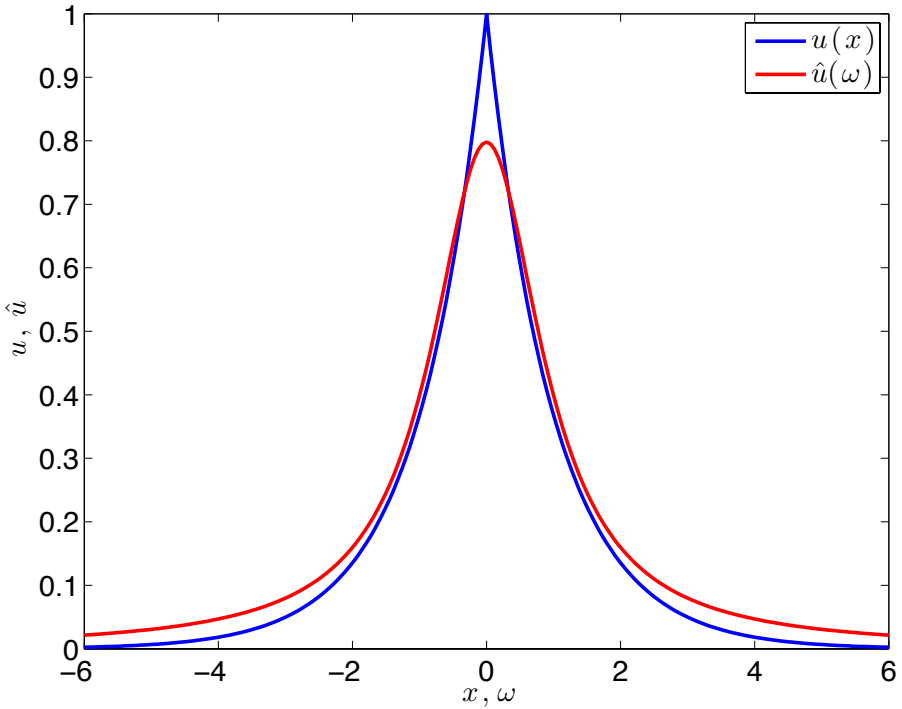
Bekannt: (Paley-Wiener Theorem) Abklingen von $\hat{u}(\omega)$ für $|\omega| \to \infty$ umso stärker, je glatter die Funktion u.

Beispiel:

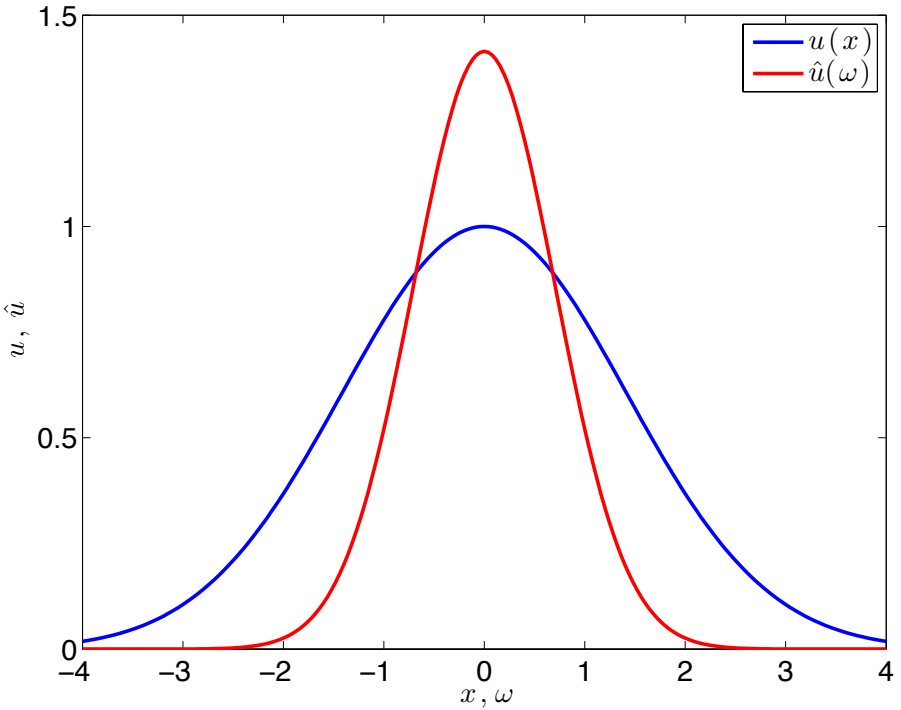
$$u(x) = \begin{cases} e^{-x}, & x \geq 0, \\ 0, & x < 0, \end{cases} \qquad \hat{u}(\omega) = \frac{1}{\sqrt{2\pi}} \frac{1}{1 + i\omega}.$$

Kontinuierliche Fourier-Transformation

Beispiele



Die Funktion $u(x) = e^{-|x|}$ und ihre Fourier-Transformierte.

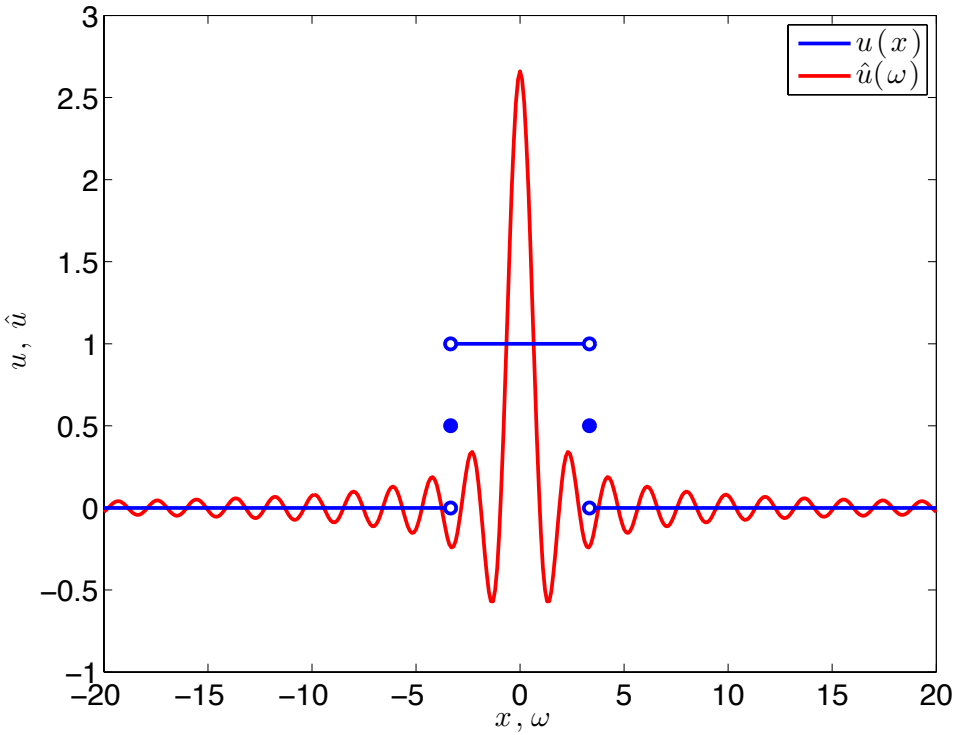


$$u(x) = e^{-\alpha x^2}, \quad \alpha = 1/4,$$

$$\hat{u}(\omega) = \frac{1}{\sqrt{2\alpha}} e^{-\omega^2/4\alpha}$$

Kontinuierliche Fourier-Transformation

Beispiele



$$u(x) = \operatorname{rect}(\alpha x), \quad \alpha = 1/4,$$

$$\hat{u}(\omega) = \frac{1}{\sqrt{2\alpha^2}} \operatorname{sinc} \frac{\omega}{2\alpha},$$

$$\operatorname{rect}(x) = \begin{cases} 0, & |x| > \frac{1}{2}, \\ \frac{1}{2}, & |x| = \frac{1}{2}, \\ 1, & |x| < \frac{1}{2}, \end{cases}$$

$$\operatorname{sinc}(x) = \frac{\sin(\pi x)}{\pi x}.$$

Fourier-Reihen

Für eine periodische Funktion $f : [0, 2\pi] \to \mathbb{R}$ ist ihre Fourier-Reihe gegeben durch

$$f(x) = \sum_{k=-\infty}^{\infty} c_k e^{ikx} = \frac{a_0}{2} + \sum_{k=1}^{\infty} [a_k \cos(kx) + b_k \sin(kx)]$$

mit komplexen bzw. reellen Fourier-Koeffizienten

$$a_0 = \frac{1}{\pi} \int_0^{2\pi} f(x)\, dx,$$

$$a_j = \frac{1}{\pi} \int_0^{2\pi} f(x) \cos(kx)\, dx, \quad b_j = \frac{1}{\pi} \int_0^{2\pi} f(x) \sin(kx)\, dx, \qquad k \in \mathbb{N},$$

$$c_k = \frac{1}{2\pi} \int_0^{2\pi} f(x) e^{-ikx}\, dx, \qquad k \in \mathbb{Z}.$$

Beispiel 1:

$$f(x) = \cos^2 x, \qquad x \in [0, 2\pi].$$

Wegen $\cos^2 x = \frac{1}{2}(1 + \cos(2x))$ und der Eindeutigkeit der Fourier-Koeffizienten folgt $a_0 = 1$, $a_2 = \frac{1}{2}$ und alle übrigen Koeffizienten sind Null.

Beispiel 2: Für die (ungerade) Funktion

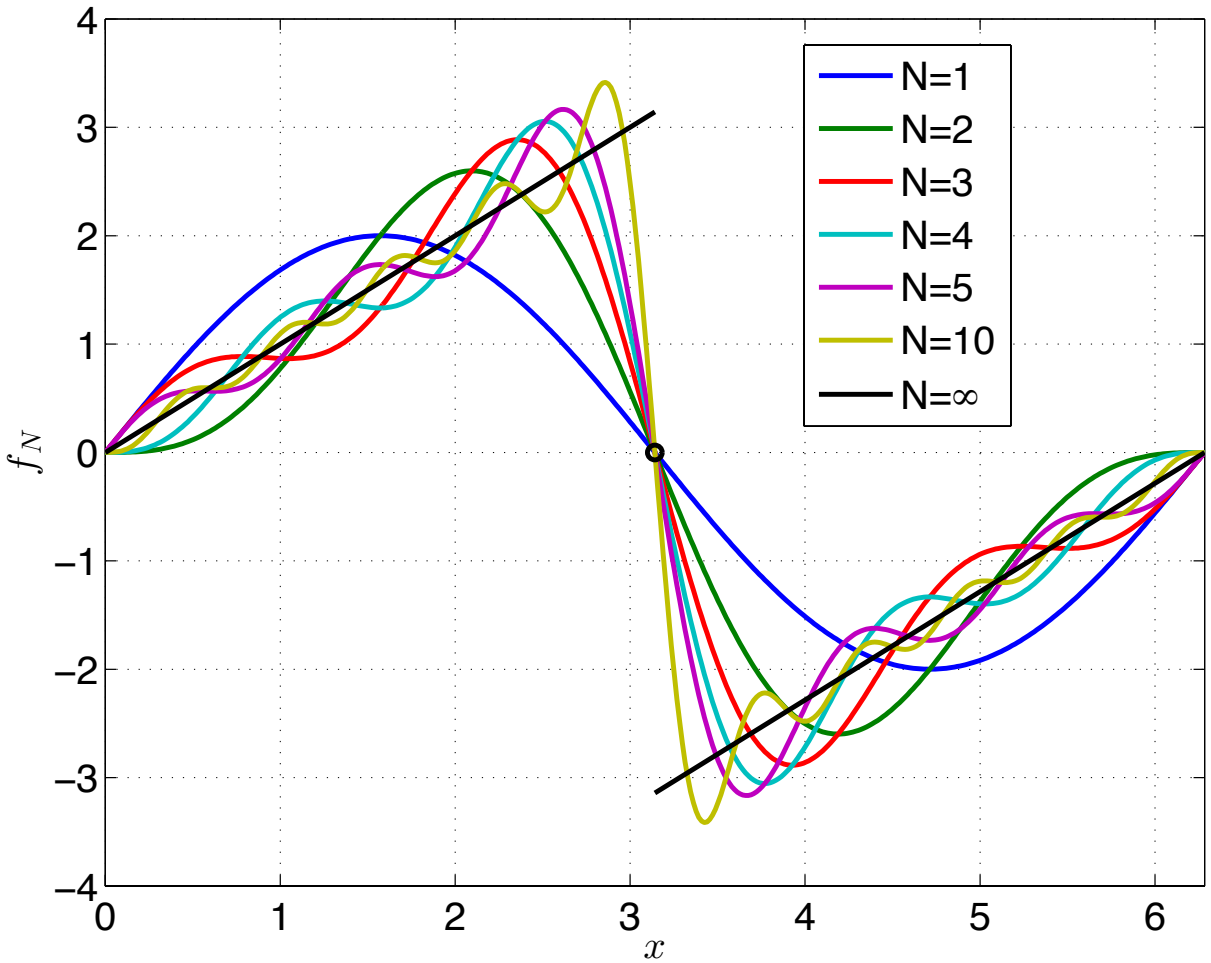
$$f(x) = x, \qquad x \in [-\pi, \pi]$$

ergibt sich $a_k = 0$ für alle k sowie $b_k = (-1)^{k+1} 2/k$, d.h.

$$x = 2\left(\sin(x) - \frac{1}{2}\sin(2x) + \frac{1}{3}\sin(3x) - + \ldots\right).$$

Fourier-Reihen

Beispiele



Inhalt

1 Einleitung

2 Computer-Arithmetik und Fehleranalyse
- 2.1 Ein Beispiel
- 2.2 Gleitpunktzahlen
- 2.3 Rundung
- 2.4 Gleitpunktarithmetik
- 2.5 Numerische Stabilität und Fehleranalyse

3 Schnelle Fourier-Transformation
- 3.1 Vorbemerkungen
- 3.2 Überblick Fourier-Analyse
- 3.3 Die diskrete Fourier-Transformation
- 3.4 Interpolation
- 3.5 Trigonometrische Interpolation
- 3.6 Die FFT

4 Abschließende Bemerkungen

Die diskrete Fourier-Transformation (DFT)

Komplex: Gegeben $\boldsymbol{f} = \{f_j\}_{j=0}^{N-1} \in \mathbb{C}^N$, gesucht sind die Fourier-Koeffizienten $\boldsymbol{c} = \{c_k\}_{k=0}^{N-1} \in \mathbb{C}^N$ mit

$$f_j = \sum_{k=0}^{N-1} c_k e^{2\pi i k j/N}, \qquad \text{d.h.} \qquad c_k = \frac{1}{N} \sum_{j=0}^{N-1} f_j e^{-2\pi i j k/N}.$$

Reell: Gegeben $\boldsymbol{f} = \{f_j\}_{j=0}^{N-1} \in \mathbb{R}^N$, gesucht sind die Fourier-Koeffizienten $\{a_k\}$ $\{b_k\}$ sodass

$$f_j = \frac{a_0}{2} + \sum_{k=1}^{M-1} \left[a_k \cos \frac{jk\pi}{M} + b_k \sin \frac{jk\pi}{M} \right] + \frac{(-1)^j a_M}{2},$$

$$k = 0, \ldots, N-1, \quad (N = 2M).$$

Inhalt

1 Einleitung

2 Computer-Arithmetik und Fehleranalyse
- 2.1 Ein Beispiel
- 2.2 Gleitpunktzahlen
- 2.3 Rundung
- 2.4 Gleitpunktarithmetik
- 2.5 Numerische Stabilität und Fehleranalyse

3 Schnelle Fourier-Transformation
- 3.1 Vorbemerkungen
- 3.2 Überblick Fourier-Analyse
- 3.3 Die diskrete Fourier-Transformation
- 3.4 Interpolation
- 3.5 Trigonometrische Interpolation
- 3.6 Die FFT

4 Abschließende Bemerkungen

Interpolation

Die allgemeine Interpolationsaufgabe

Zu gegebener Funktion $f : [a, b] \to \mathbb{C}$ und gegebenen Stützstellen (Knoten) $a \leq x_0 < x_1 < x_2 < \cdots < x_n \leq b$ soll eine „einfache“ Funktion $p : [a, b] \to \mathbb{C}$ konstruiert werden, die die Interpolationsbedingungen

$$p(x_i) = f(x_i), \qquad i = 0, 1, \ldots, n,$$

erfüllt.

Wozu?

- f nur an diskreten Punkten bekannt (Messwerte), aber geschlossene Formel für f auf ganz $[a, b]$ erwünscht (z.B. um f an Zwischenstellen $x \in [a, b] \setminus \{x_0, x_1, \ldots, x_n\}$ auszuwerten),
- f „kompliziert“ und soll durch „einfache“ Funktion angenähert werden (z.B. um die Ableitung $f'(x)$, $x \in [a, b]$, oder das Integral $\int_a^b f(x)dx$ näherungsweise zu bestimmen).

Zu gegebenen (paarweise verschiedenen) Knoten

$$a \le x_0 < x_1 < x_2 < \cdots < x_n \le b$$

und gegebenen Funktionswerten $f_0, f_1, \ldots, f_n \in \mathbb{C}$ soll ein Interpolationspolynom

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0 \in \mathscr{P}_n$$

(mit komplexen Koeffizienten $a_0, a_1, \ldots, a_n$, d.h. $n+1$ Freiheitsgrade) vom Grad n konstruiert werden, das die $n+1$ Interpolationsbedingungen

$$p(x_i) = f_i, \qquad i = 0, 1, \ldots, n,$$

erfüllt.

Satz 1

Die polynomiale Interpolationsaufgabe ist eindeutig lösbar. Mit den Lagrange-Grundpolynomen [Joseph Louis Lagrange, *1736–1813*]

$$\ell_i(x) := \prod_{\substack{j=0 \\ j\neq i}}^{n} \frac{x - x_j}{x_i - x_j} \in \mathscr{P}_n$$

(beachte $\ell_i(x_i) = 1$ *und* $\ell_i(x_j) = 0$ *für* $j \neq i$*) lässt sich das Interpolationspolynom in der Lagrange-Form darstellen:*

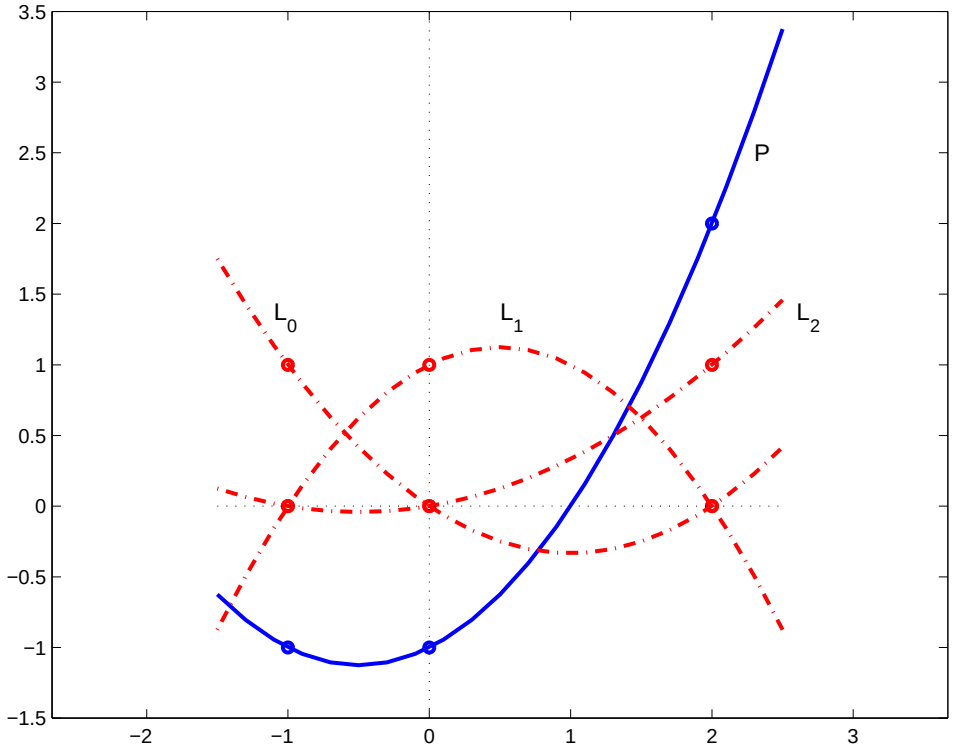
$$p(x) = \sum_{i=0}^{n} f_i \ell_i(x).$$

Polynominterpolation

Beispiel

Daten:

$$(x_0, f_0) = (-1, -1),$$
$$(x_1, f_1) = (0, -1),$$
$$(x_2, f_2) = (2, 2).$$



Lagrange-Grundpolynome:

$$\ell_0(x) = x(x-2)/3,$$
$$\ell_1(x) = (x+1)(x-2)/(-2),$$
$$\ell_2(x) = (x+1)x/6.$$

Interpolationspolynom:

$$p(x) = -\ell_0(x) - \ell_1(x) + 2\ell_2(x) = \frac{x^2}{2} + \frac{x}{2} - 1.$$

Die Auswertung der Lagrange-Formel ist aufwendig, wenn ein neues Datenpaar hinzukommt. Eine rekursive Berechnung ist ökonomischer:

Lemma 2

Für eine beliebige Indexmenge $0 \le i_0 < i_1 < \cdots < i_k \le n$ *bezeichne* $p_{i_0,i_1,\ldots,i_k}$ *das (nach Satz 1 eindeutig bestimmte) Polynom vom Grad* k*, das die Bedingungen*

$$p_{i_0,i_1,\ldots,i_k}(x_{i_j}) = f_{i_j}, \qquad j = 0, 1, \ldots, k,$$

erfüllt. Dann gilt die Rekursionsformel

$$\begin{aligned} p_i(x) &= f_i, \\ p_{i_0,i_1,\ldots,i_k}(x) &= \frac{(x - x_{i_0})p_{i_1,i_2,\ldots,i_k}(x) - (x - x_{i_k})p_{i_0,i_1,\ldots,i_{k-1}}(x)}{x_{i_k} - x_{i_0}}. \end{aligned}$$

Rechenschema Algorithmus von Neville-Aitken, [Charles William Neville, $*$ 1941]; [Alexander Craig Aitken, 1895–1967] :

x_i	$k=0$	$k=1$	$k=2$	$k=3$	$k=4$
x_0	$p_0(x)=f_0$				
		$p_{0,1}(x)$			
x_1	$p_1(x)=f_1$		$p_{0,1,2}(x)$		
		$p_{1,2}(x)$		$p_{0,1,2,3}(x)$	
x_2	$p_2(x)=f_2$		$p_{1,2,3}(x)$		$p_{0,1,2,3,4}(x)$
		$p_{2,3}(x)$		$p_{1,2,3,4}(x)$	
x_3	$p_3(x)=f_3$		$p_{2,3,4}(x)$		
		$p_{3,4}(x)$			
x_4	$p_4(x)=f_4$				

(Berechnungsreihenfolge : $p_0 \to p_1 \to p_{0,1} \to p_2 \to p_{1,2} \to p_{0,1,2} \to \cdots$)

Beispiel 2 (vgl. Beispiel 1).

x_i	$k=0$	$k=1$	$k=2$
-1	-1		
		$\frac{(x-(-1))(-1)-(x-0)(-1)}{0-(-1)} = -1$	
0	-1		$\frac{(x-(-1))(3x/2-1)-(x-2)(-1)}{2-(-1)}$
		$\frac{(x-0)2-(x-2)(-1)}{2-0} = \frac{3}{2}x - 1$	$= \frac{1}{2}x^2 + \frac{1}{2}x - 1$
2	2		

Aufwand des Neville-Aitken Schemas (für Auswertung des Interpolationspolynoms vom Grad n an einer Stelle x):

$$\frac{5}{2}n^2 + \frac{7}{2}n + 1 \quad \text{Gleitpunktoperationen}$$

(falls die Differenzen $x - x_i$ $(0 \le i \le n)$ vorab bestimmt werden).

Tableau der dividierten Differenzen von f:

x_i	$k=0$	$k=1$	$k=2$	$k=3$	$k=4$
x_0	f_0				
		$f_{0,1}$			
x_1	f_1		$f_{0,1,2}$		
		$f_{1,2}$		$f_{0,1,2,3}$	
x_2	f_2		$f_{1,2,3}$		$f_{0,1,2,3,4}$
		$f_{2,3}$		$f_{1,2,3,4}$	
x_3	f_3		$f_{2,3,4}$		
		$f_{3,4}$			
x_4	f_4				

mit

$$f_{i_0,i_1,\ldots,i_k} := \frac{f_{i_1,i_2,\ldots,i_k} - f_{i_0,i_1,\ldots,i_{k-1}}}{x_{i_k} - x_{i_0}} \qquad (k \geq 1).$$

Satz 3

Mit Hilfe der dividierten Differenzen lässt sich das (nach Satz 1 eindeutig bestimmte) Interpolationspolynom p in Newton-Form

$$\begin{aligned} p(x) &= f_0 + f_{0,1}(x - x_0) + f_{0,1,2}(x - x_0)(x - x_1) + \cdots \\ &\quad \cdots + f_{0,1,\ldots,n}(x - x_0)(x - x_1)\cdots(x - x_{n-1}) \end{aligned}$$

darstellen.

Rechenaufwand:

- Zur Bestimmung der Differenzentafel: $\frac{3}{2}(n^2 + n)$ Gleitpunktoperationen.
- Zur Auswertung des Newtonschen Interpolationspolynoms mit dem Horner-Schema [William George Horner, 1786–1837]: $3n$ Gleitpunktoperationen (pro Auswertungspunkt).

Beispiel 3 (vgl. Beispiele 1 und 2).
Dividierte Differenzen:

x_i	$k=0$	$k=1$	$k=2$
-1	$\boxed{-1}$		
		$f_{0,1} = \frac{(-1)-(-1)}{0-(-1)} = \boxed{0}$	
0	-1		$f_{0,1,2} = \frac{3/2-0}{2-(-1)} = \boxed{\frac{1}{2}}$
		$f_{1,2} = \frac{2-(-1)}{2-0} = \frac{3}{2}$	
2	2		

Das bedeutet:

$$p(x) = \boxed{(-1)} + \boxed{0}(x-(-1)) + \boxed{\frac{1}{2}}(x-(-1))(x-0) = \frac{1}{2}x^2 + \frac{1}{2}x - 1.$$

Satz 4 (Fehler der Polynominterpolation)

Die Funktion $f \in C^{n+1}[a,b]$ werde durch das Polynom $p \in \mathscr{P}_n$ interpoliert an den paarweise verschiedenen Knoten $\{x_0, x_1, \ldots, x_n\} \subset [a,b]$.
Dann gibt es zu jedem $x \in [a,b]$ ein $\xi = \xi(x) \in (a,b)$ mit

$$f(x) - p(x) = \frac{\omega_{n+1}(x)}{(n+1)!} f^{(n+1)}(\xi).$$

Mit $M_{n+1} := \max_{a \le t \le b} |f^{(n+1)}(t)|$ gilt für alle $x \in [a,b]$ die Fehlerabschätzung

$$|f(x) - p(x)| \le \frac{M_{n+1}}{(n+1)!} \max_{a \le t \le b} |\omega_{n+1}(t)|. \tag{1}$$

„Optimale“ Knoten: Idee (motiviert durch Fehlerabschätzung):
Wähle Knoten $a \le x_0 < x_1 < \cdots < x_n \le b$ so, dass

$$\max_{a\le t\le b} |\omega_{n+1}(t)| = \max_{a\le t\le b} \prod_{i=0}^{n} |t - x_i|$$

so klein wie möglich wird.
Lösung: **Tschebyscheff-Knoten** [Pafnutiĭ L'vovich Tschebyscheff, 1821–1894]

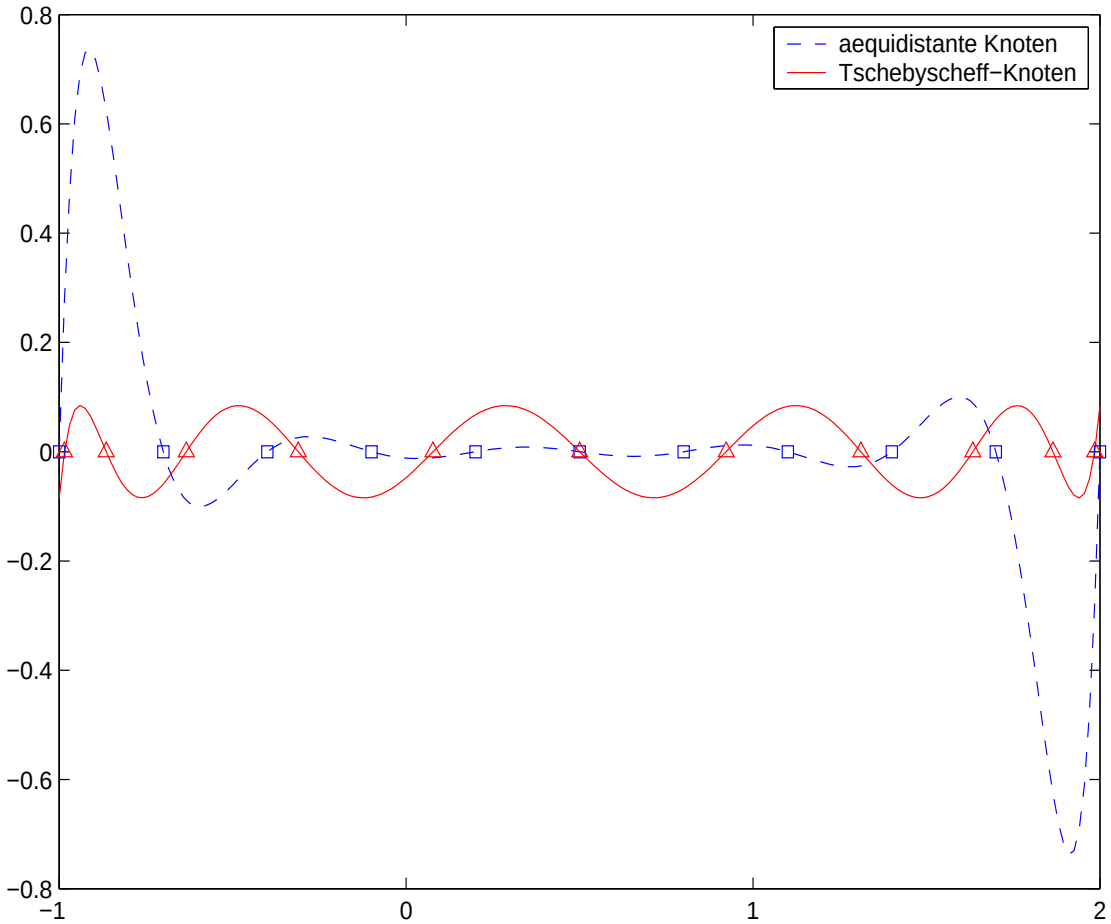
$$x_i^{(\mathsf{T})} = \frac{b-a}{2} \cos\left(\frac{2(n-i)+1}{2n+2}\,\pi\right) + \frac{a+b}{2} \qquad (i = 0, 1, \ldots, n)$$

mit

$$\max_{a\le t\le b} \prod_{i=0}^{n} |t - x_i^{(\mathsf{T})}| = 2\left(\frac{b-a}{4}\right)^n < \max_{a\le t\le b} \prod_{i=0}^{n} |t - x_i|$$

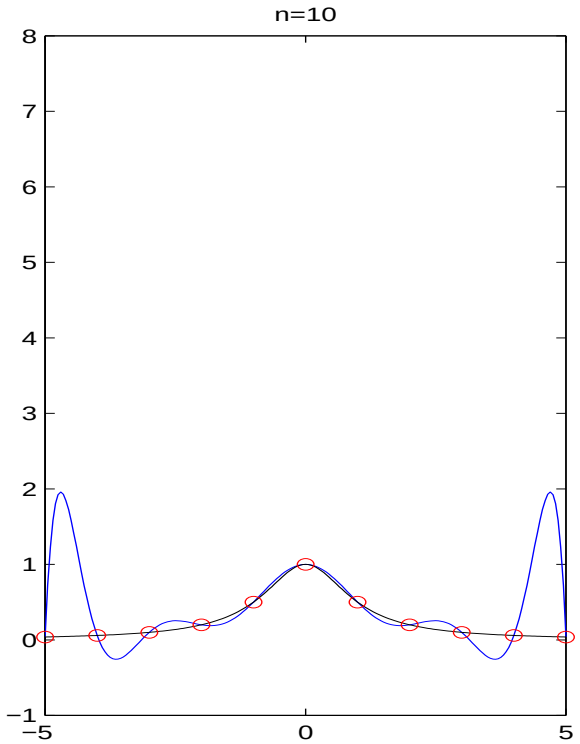
für jede andere Wahl $x_0, \ldots, x_n$ der Knoten.

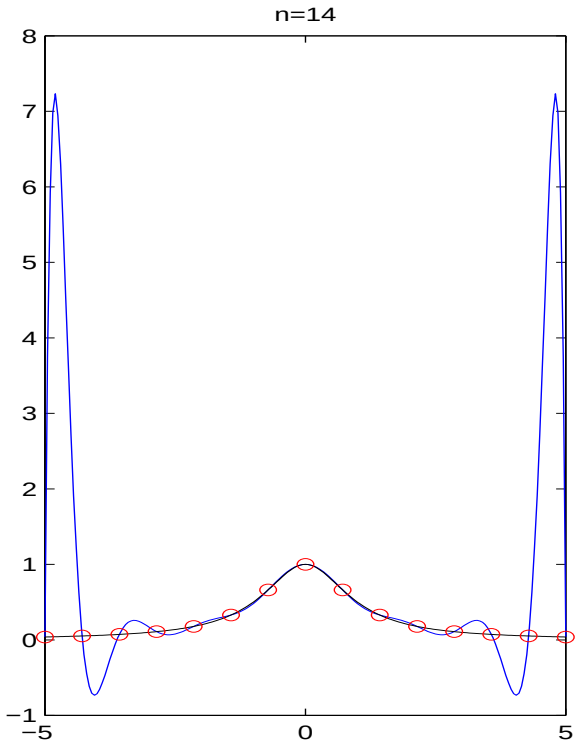
Knotenpolynome mit äquidistanten und Tschebyscheff-Knoten:



Beispiel 4.(Runge-Phänomen[1]) Interpoliere an $n+1$ äquidistanten Stützstellen

$$f(x) = \frac{1}{1+x^2}, \qquad -5 \le x \le 5, \qquad \text{(Runge-Funktion)}$$

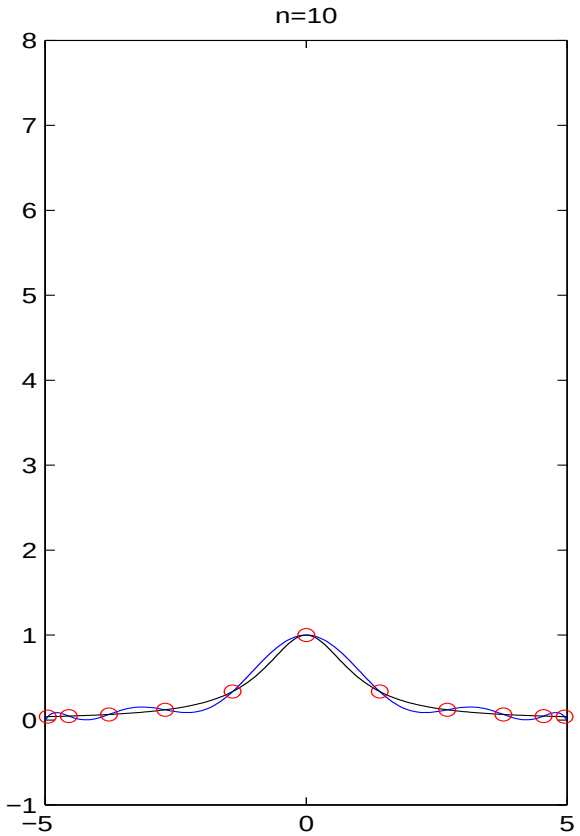


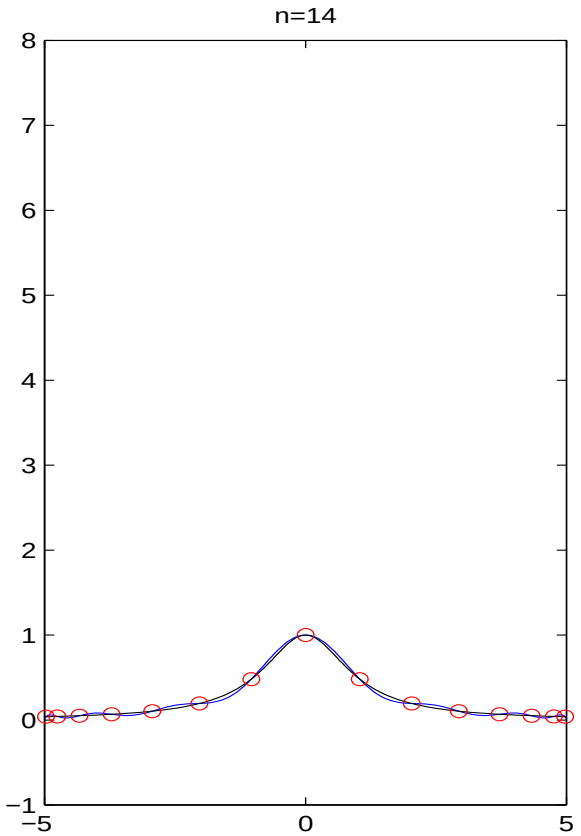


[1] C. Runge. *Über emprirische Funktionen und die Interpolation zwischen äquidistanten Ordinaten*. Zeitschrift für Mathematik und Physik 46 (1901) pp. 224–243

Beispiel 5. Interpoliere an $n+1$ Tschebyscheff-Knoten

$$f(x) = \frac{1}{1+x^2}, \qquad -5 \le x \le 5.$$





Inhalt

1 Einleitung

2 Computer-Arithmetik und Fehleranalyse
- 2.1 Ein Beispiel
- 2.2 Gleitpunktzahlen
- 2.3 Rundung
- 2.4 Gleitpunktarithmetik
- 2.5 Numerische Stabilität und Fehleranalyse

3 Schnelle Fourier-Transformation
- 3.1 Vorbemerkungen
- 3.2 Überblick Fourier-Analyse
- 3.3 Die diskrete Fourier-Transformation
- 3.4 Interpolation
- 3.5 Trigonometrische Interpolation
- 3.6 Die FFT

4 Abschließende Bemerkungen

Trigonometrische Interpolation

Aufgabenstellung

Seien $f_0, f_1, \ldots, f_{m-1} \in \mathbb{R}$ und $x_j := 2\pi j/m \quad (j = 0, 1, \ldots, m-1)$, d.h. $x_0 < x_1 < \cdots < x_{m-1}$ sind äquidistante Knoten aus $[0, 2\pi)$.

Gesucht ist ein reelles trigonometrisches Polynom vom Grad n,

$$t_n(x) = \frac{\alpha_0}{2} + \sum_{k=1}^{n} \big[\alpha_k \cos(kx) + \beta_k \sin(kx)\big],$$

das die m Interpolationsbedingungen

$$t_n(x_j) = f_j, \qquad j = 0, 1, \ldots, m-1, \tag{2}$$

erfüllt. Hierbei ist

$$n = \begin{cases} \frac{m}{2} & \text{falls } m \text{ gerade,} \\ \frac{m-1}{2} & \text{falls } m \text{ ungerade.} \end{cases}$$

Trigonometrische Interpolation

Transformation auf den (komplexen) Einheitskreis

$$\phi : [0, 2\pi) \longrightarrow \mathbb{T} := \{z \in \mathbb{C} : |z| = 1\}, \quad x \mapsto z = e^{ix} = \cos x + i \sin x.$$

Die Knoten x_j gehen über in die m-ten Einheitswurzeln:

$$\phi(x_j) = e^{2\pi ij/m} = [e^{2\pi i/m}]^j = \omega_m^j \qquad j = 0, 1, \ldots, m-1,$$

mit $\omega_m := e^{2\pi i/m} = \cos \frac{2\pi}{m} + i \sin \frac{2\pi}{m}$.

Setzt man $\beta_0 = 0$ und für $k = 0, 1, \ldots, n$

$$\mathbb{C}^2 \ni \begin{bmatrix} a_k \\ a_{-k} \end{bmatrix} := \begin{bmatrix} \frac{1}{2}(\alpha_k - i\beta_k) \\ \\ \frac{1}{2}(\alpha_k + i\beta_k) \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 & -i \\ 1 & i \end{bmatrix} \begin{bmatrix} \alpha_k \\ \beta_k \end{bmatrix}, \text{ d.h.}$$

Trigonometrische Interpolation

Transformation auf den (komplexen) Einheitskreis

$$\mathbb{R}^2 \ni \begin{bmatrix}\alpha_k \\ \beta_k\end{bmatrix} = \begin{bmatrix}1 & 1 \\ i & -i\end{bmatrix}\begin{bmatrix}a_k \\ a_{-k}\end{bmatrix} = \begin{bmatrix}a_k + a_{-k} \\ i(a_k - a_{-k})\end{bmatrix} = \begin{bmatrix}2\,\mathrm{Re}(a_k) \\ -2\,\mathrm{Im}(a_k)\end{bmatrix},$$

so folgt

$$t_n(x) = \sum_{k=-n}^{n} a_k e^{ikx} = \sum_{k=-n}^{n} a_k z^k = z^{-n}\sum_{k=-n}^{n} a_k z^{k+n} = z^{-n} p_{2n}(z)$$

mit $p_{2n}(z) = \sum_{k=-n}^{n} a_k z^{k+n} = \sum_{j=0}^{2n} a_{j-n} z^j \in \mathscr{P}_{2n}$.
Wegen

$$p_{2n}(\omega_m^j) = \omega_m^{jn}\; t_n(x_j)$$

ist die trigonometrische Interpolationsaufgabe hiermit zurückgeführt auf eine (gewöhnliche) Interpolationsaufgabe für (algebraische) Polynome.

Lemma 5

Für die m-ten Einheitswurzeln ω_m^k ($k \in \mathbb{Z}$, $m \in \mathbb{N}$) gelten:

(a) $[\omega_m^k]^j = \omega_m^{kj} = [\omega_m^j]^k \quad (j \in \mathbb{Z})$,

(b) $\omega_{m\ell}^{k\ell} = \omega_m^k \quad (\ell \in \mathbb{Z},\, \ell \neq 0)$,

(c) $\overline{\omega_m^k} = \omega_m^{-k}$,

(d) $$\sum_{j=0}^{m-1} \omega_m^{kj} = \begin{cases} m, & \text{falls} \quad k = 0 \pmod m, \\ 0, & \text{falls} \quad k \neq 0 \pmod m. \end{cases}$$

Komplexes algebraisches Interpolationspolynom

Satz 6

Das komplexe (algebraische) Interpolationspolynom

$$p_{m-1}(z) = \sum_{k=0}^{m-1} c_k z^k \in \mathscr{P}_{m-1}$$

mit $p_{m-1}(\omega_m^j) = f_j \in \mathbb{C} \quad (j = 0, 1, \ldots, m-1)$ *besitzt die Koeffizienten*

$$c_k = \frac{1}{m} \sum_{j=0}^{m-1} f_j \omega_m^{-kj}, \qquad k = 0, 1, \ldots, m-1. \tag{3}$$

Komplexes algebraisches Interpolationspolynom

In Matrix-Vektor-Schreibweise

$$\begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{m-1} \end{bmatrix} = \frac{1}{m} F_m \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_{m-1} \end{bmatrix}$$

mit der Fourier-Matrix

$$F_m := \left[\omega_m^{-kj}\right]_{0\le k,j\le m-1} = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & \omega_m^{-1} & \cdots & \omega_m^{-m+1} \\ \vdots & \vdots & & \vdots \\ 1 & \omega_m^{-m+1} & \cdots & \omega_m^{-(m-1)^2} \end{bmatrix}.$$

Trigonometrisches Interpolationspolynom

Theorem 7

Für $m = 2n$ oder $m = 2n+1$ gibt es zu beliebigen $f_0, f_1, \ldots, f_{m-1} \in \mathbb{R}$ ein reelles trigonometrisches Interpolationspolynom

$$t_n(x) = \frac{\alpha_0}{2} + \sum_{k=1}^{n} \big[\alpha_k \cos(kx) + \beta_k \sin(kx)\big] \in \mathscr{T}_n$$

vom Grad n, das die m Bedingungen

$$t_n(2\pi j/m) = f_j \qquad (j = 0, 1, \ldots, m-1)$$

erfüllt. Seine Koeffizienten sind gegeben durch

$$\alpha_k = \frac{2}{m} \sum_{j=0}^{m-1} f_j \cos \frac{2\pi jk}{m} \quad \textit{bzw.} \quad \beta_k = \frac{2}{m} \sum_{j=0}^{m-1} f_j \sin \frac{2\pi jk}{m}, \quad (k = 0, 1, \ldots, n).$$

Im Fall $m = 2n$ muss $\beta_n = 0$ gesetzt und α_n halbiert werden.

Schnelle Fourier-Transformation

Seien $\{\omega_m^j\}_{j=0}^{m-1}$ die m-ten Einheitswurzeln, ($\omega_m := e^{2\pi i/m}$).
Wir unterscheiden zwei grundlegende Aufgabenstellungen:
Diskrete Fourier-Analyse: Bestimme zu vorgegebenen Funktionswerten $f_0, \ldots, f_{m-1} \in \mathbb{C}$ die Koeffizienten $c_0, \ldots, c_{m-1}$ des Interpolationspolynoms

$$p(z) = \sum_{j=0}^{m-1} c_j z^j \quad \text{mit} \quad p(\omega_m^j) = f_j \qquad (j = 0, \ldots, m-1).$$

Wir wissen: Mit der Fourier-Matrix $F_m := [\omega_m^{-kj}]_{0 \le k,j \le m-1} \in \mathbb{C}^{m \times m}$ gilt

$$\begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{m-1} \end{bmatrix} = \frac{1}{m} F_m \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_{m-1} \end{bmatrix}.$$

Inhalt

1 Einleitung

2 Computer-Arithmetik und Fehleranalyse
- 2.1 Ein Beispiel
- 2.2 Gleitpunktzahlen
- 2.3 Rundung
- 2.4 Gleitpunktarithmetik
- 2.5 Numerische Stabilität und Fehleranalyse

3 Schnelle Fourier-Transformation
- 3.1 Vorbemerkungen
- 3.2 Überblick Fourier-Analyse
- 3.3 Die diskrete Fourier-Transformation
- 3.4 Interpolation
- 3.5 Trigonometrische Interpolation
- 3.6 Die FFT

4 Abschließende Bemerkungen

Schnelle Fourier-Transformation

Diskrete Fourier-Synthese (inverse Aufgabe): Bestimme zu vorgebenen Koeffizienten $c_0, \ldots, c_{m-1} \in \mathbb{C}$ die Funktionswerte $f_0, \ldots, f_{m-1}$ des Polynoms $p(z) = \sum_{j=0}^{m-1} c_j z^j$ an den m-ten Einheitswurzeln $\omega_m^0, \ldots \omega_m^{m-1}$. Offensichtlich:

$$\begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_{m-1} \end{bmatrix} = W_m \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{m-1} \end{bmatrix}$$

mit der Matrix $W_m := [\omega_m^{kj}]_{0 \le k,j \le m-1} = F_m^H (= \overline{F_m}) \in \mathbb{C}^{m \times m}$.

Schnelle Fourier-Transformation

Lemma 8

Für die Fourier-Matrix $F_m = [\omega_m^{-kj}]_{0\le k,j\le m-1} \in \mathbb{C}^{m\times m}$ *gelten:*

(a) $F_m^\top = F_m$ *(aber* $F_m^H \neq F_m$ *für* $m > 2$*!),*

(b) $F_m^H F_m = mI_m$*, d.h. die Spalten von* F_m *sind orthogonal und besitzen alle die Euklid-Norm* $\sqrt{m}$*.*

(c) $F_m^{-1} = \frac{1}{m}F_m^H = \frac{1}{m}\overline{F_m}$.

Die „naive“ Berechnung einer Fourier-Transformation (Matrix-Vektor Produkt mit F_m/m bzw. W_m) erfordert offenbar $O(m^2)$ komplexe Multiplikationen. Bei Anwendung der schnellen Fourier-Transformation (FFT) reduziert sich dieser Aufwand auf $O(m\log m)$[2]

[2] James William Cooley($*$1926) and John Wilder Tukey (1915–2000): An algorithm for the machine calculation of complex Fourier series, *Math. Comp.* **19**, 297–301 (1965).

Schnelle Fourier-Transformation

Wir setzen (aus schreibtechnischen Gründen) im Folgenden

$$\zeta_m := \overline{\omega_m} = e^{-2\pi i/m} = \cos\left(\frac{2\pi}{m}\right) - i\sin\left(\frac{2\pi}{m}\right),$$

so dass $F_m = [\zeta_m^{kj}]_{0\le k,j\le m-1}$. Außerdem sei m gerade.

Die Idee der FFT (für $m = 8$): Mit $\zeta := \zeta_8$ ist

$$F_8 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \zeta & \zeta^2 & \zeta^3 & \zeta^4 & \zeta^5 & \zeta^6 & \zeta^7 \\ 1 & \zeta^2 & \zeta^4 & \zeta^6 & \zeta^8 & \zeta^{10} & \zeta^{12} & \zeta^{14} \\ 1 & \zeta^3 & \zeta^6 & \zeta^9 & \zeta^{12} & \zeta^{15} & \zeta^{18} & \zeta^{21} \\ 1 & \zeta^4 & \zeta^8 & \zeta^{12} & \zeta^{16} & \zeta^{20} & \zeta^{24} & \zeta^{28} \\ 1 & \zeta^5 & \zeta^{10} & \zeta^{15} & \zeta^{20} & \zeta^{25} & \zeta^{30} & \zeta^{35} \\ 1 & \zeta^6 & \zeta^{12} & \zeta^{18} & \zeta^{24} & \zeta^{30} & \zeta^{36} & \zeta^{42} \\ 1 & \zeta^7 & \zeta^{14} & \zeta^{21} & \zeta^{28} & \zeta^{35} & \zeta^{42} & \zeta^{49} \end{bmatrix}.$$

Schnelle Fourier-Transformation

Wegen $\zeta^8 = 1$, d.h. $\zeta^j = \zeta^k$, wenn $j - k$ (ohne Rest) durch 8 teilbar ist, folgt

$$F_8 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \zeta & \zeta^2 & \zeta^3 & \zeta^4 & \zeta^5 & \zeta^6 & \zeta^7 \\ 1 & \zeta^2 & \zeta^4 & \zeta^6 & 1 & \zeta^2 & \zeta^4 & \zeta^6 \\ 1 & \zeta^3 & \zeta^6 & \zeta & \zeta^4 & \zeta^7 & \zeta^2 & \zeta^5 \\ 1 & \zeta^4 & 1 & \zeta^4 & 1 & \zeta^4 & 1 & \zeta^4 \\ 1 & \zeta^5 & \zeta^2 & \zeta^7 & \zeta^4 & \zeta & \zeta^6 & \zeta^3 \\ 1 & \zeta^6 & \zeta^4 & \zeta^2 & 1 & \zeta^6 & \zeta^4 & \zeta^2 \\ 1 & \zeta^7 & \zeta^6 & \zeta^5 & \zeta^4 & \zeta^3 & \zeta^2 & \zeta^1 \end{bmatrix}.$$

Schnelle Fourier-Transformation

Jetzt nummerieren wir die Zeilen von F_8 um: zuerst werden die mit geradem (0,2,4,6), danach die mit ungeradem Index (1,3,5,7) gezählt. Die zugehörige Pemutationsmatrix wird mit P bezeichnet.

$$PF_8 = \left[\begin{array}{cccc|cccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \zeta^2 & \zeta^4 & \zeta^6 & 1 & \zeta^2 & \zeta^4 & \zeta^6 \\ 1 & \zeta^4 & 1 & \zeta^4 & 1 & \zeta^4 & 1 & \zeta^4 \\ 1 & \zeta^6 & \zeta^4 & \zeta^2 & 1 & \zeta^6 & \zeta^4 & \zeta^2 \\ \hline 1 & \zeta & \zeta^2 & \zeta^3 & \zeta^4 & \zeta^5 & \zeta^6 & \zeta^7 \\ 1 & \zeta^3 & \zeta^6 & \zeta & \zeta^4 & \zeta^7 & \zeta^2 & \zeta^5 \\ 1 & \zeta^5 & \zeta^2 & \zeta^7 & \zeta^4 & \zeta & \zeta^6 & \zeta^3 \\ 1 & \zeta^7 & \zeta^6 & \zeta^5 & \zeta^4 & \zeta^3 & \zeta^2 & \zeta^1 \end{array}\right] =: \begin{bmatrix} B_{1,1} & B_{1,2} \\ B_{2,1} & B_{2,2} \end{bmatrix}$$

Schnelle Fourier-Transformation

Wir untersuchen die einzelnen Blöcke: Wegen $\zeta = \zeta_8$ ist $\zeta^2 = \zeta_4$, d.h.

$$B_{1,1} = B_{1,2} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & \zeta_4 & \zeta_4^2 & \zeta_4^3 \\ 1 & \zeta_4^2 & \zeta_4^4 & \zeta_4^6 \\ 1 & \zeta_4^3 & \zeta_4^6 & \zeta_4^9 \end{bmatrix} = F_4.$$

Aus den Spalten 0,1,2 bzw. 3 von $B_{2,1}$ „klammern“ wir ζ^0, ζ^1, ζ^2 bzw. ζ^3 „aus“:

$$B_{2,1} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & \zeta^2 & \zeta^4 & \zeta^6 \\ 1 & \zeta^4 & 1 & \zeta^4 \\ 1 & \zeta^6 & \zeta^4 & \zeta^2 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \zeta & 0 & 0 \\ 0 & 0 & \zeta^2 & 0 \\ 0 & 0 & 0 & \zeta^3 \end{bmatrix} = F_4 D_4.$$

Schnelle Fourier-Transformation

Analog:

$$B_{2,2} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & \zeta^2 & \zeta^4 & \zeta^6 \\ 1 & \zeta^4 & 1 & \zeta^4 \\ 1 & \zeta^6 & \zeta^4 & \zeta^2 \end{bmatrix} \begin{bmatrix} \zeta^4 & 0 & 0 & 0 \\ 0 & \zeta^5 & 0 & 0 \\ 0 & 0 & \zeta^6 & 0 \\ 0 & 0 & 0 & \zeta^7 \end{bmatrix} = F_4(\zeta^4 D_4) = -F_4 D_4.$$

Insgesamt erhalten wir

$$PF_8 = \begin{bmatrix} F_4 & F_4 \\ F_4 D_4 & -F_4 D_4 \end{bmatrix} = \begin{bmatrix} F_4 & O \\ O & F_4 \end{bmatrix} \begin{bmatrix} I_4 & I_4 \\ D_4 & -D_4 \end{bmatrix}.$$

Satz 9

Seien m gerade, σ die folgende (even/odd) Permutation

$$\sigma = [0,\, 2\, \ldots, m-2, 1, 3, \ldots, m-1]$$

und $P = P_\sigma$ die zugehörige Permutationsmatrix.
Dann besitzt die zeilenpermutierte Fourier-Matrix F_m die Zerlegung

$$PF_m = \begin{bmatrix} F_{m/2} & F_{m/2} \\ F_{m/2}D_{m/2} & -F_{m/2}D_{m/2} \end{bmatrix} = \begin{bmatrix} F_{m/2} & O \\ O & F_{m/2} \end{bmatrix} \begin{bmatrix} I_{m/2} & I_{m/2} \\ D_{m/2} & -D_{m/2} \end{bmatrix}.$$

Dabei bezeichnet $D_{m/2}$ die Diagonalmatrix

$$D_{m/2} = \operatorname{diag}\left(\zeta_m^0, \zeta_m^1, \ldots, \zeta_m^{m/2-1}\right) \in \mathbb{C}^{(m/2)\times(m/2)}$$

mit $\zeta_m = \overline{\omega_m} = e^{-2\pi i/m}$.

Schnelle Fourier-Transformation

Berechne jetzt $\boldsymbol{y} = F_m \boldsymbol{x}$ für ein $\boldsymbol{x} \in \mathbb{C}^m$ (m gerade). Gemäß der Zerlegung von F_m aus Satz 9 unterteilen wir dies in zwei Schritte:

1. Reduktionsschritt: Berechne

$$\boldsymbol{z} = \begin{bmatrix} I_{m/2} & I_{m/2} \\ D_{m/2} & -D_{m/2} \end{bmatrix} \boldsymbol{x}.$$

Im Fall $m = 8$ ergibt sich:

$$\begin{array}{llll} z_0 = x_0 + x_4, & z_1 = x_1 + x_5, & z_2 = x_2 + x_6, & z_3 = x_3 + x_7 \\ z_4 = (x_0 - x_4), & z_5 = (x_1 - x_5)\zeta_m, & z_6 = (x_2 - x_6)\zeta_m^2, & z_7 = (x_3 - x_7)\zeta_m^3 \end{array}$$

($m/2$ komplexe Multiplikationen und m komplexe Additionen).

2. Teilprobleme: Berechne

$$F_{m/2}\boldsymbol{z}(0 : m/2 - 1) \quad \text{und} \quad F_{m/2}\boldsymbol{z}(m/2 : m - 1)$$

(zwei Fourier-Transformationen der Dimension $m/2$).

Schnelle Fourier-Transformation

Ist $m = 2^p$ eine Zweierpotenz, so ist $m/2$ ebenfalls gerade und die beiden DFT der Dimension $m/2$ können auf vier DFT der Dimension $m/4$ reduziert werden.

Der Aufwand zur Reduktion beträgt $2 \cdot m/4 = m/2$ komplexe Multiplikationen (und $2 \cdot m/2 = m$ komplexe Additionen). Dieser Prozess wird solange fortgesetzt bis man eine Multiplikation mit F_m auf m Multiplikationen mit $F_1 = [1]$ reduziert hat (eine Multiplikation mit F_1 erfordert offenbar keinen Aufwand).

Dieses Reduktionsverfahren heißt schnelle Fourier-Transformation (FFT = Fast Fourier Transform).

Theorem 10

Zur Durchführung einer schnellen Fourier-Transformation der Ordnung $m = 2^p$ sind

$$\frac{m}{2}p = \frac{m}{2}\log_2(m) \qquad \textit{komplexe Multiplikationen}$$

und $m \log_2(m)$ komplexe Additionen erforderlich.

Schnelle Fourier-Transformation

Die naive Berechnung einer Fourier-Transformation der Länge $m = 2^p$ durch $F_m \boldsymbol{x}$ erfordert also $2^{p+1}/p$-mal mehr Multiplikationen als ihre Berechnung durch FFT. Wenn z.B. für $p = 20$ die FFT-Version eine Sekunde benötigt, so benötigt $F_m \boldsymbol{x}$ etwa 29 Stunden.

Verbleibendes Problem: Bestimmt man $\boldsymbol{y} = F_m \boldsymbol{x}$ durch FFT, so erhält man zunächst eine permutierte Version $\tilde{\boldsymbol{y}} = Q\boldsymbol{y}$ von $\boldsymbol{y}$ mit einer Permutationsmatrix $Q \in \mathbb{R}^{m\times m}$.

Es gilt: Besitzt für $m = 2^p$ der Index $i \in \{0, 1, \ldots, m-1\}$ die Binärdarstellung

$i = b_{p-1}2^{p-1} + \cdots + b_2 2^2 + b_1 2 + b_0 =: [b_{p-1} \ \ldots \ b_2\, b_1\, b_0]_2$, und ist

$$r(i) := [b_0\, b_1\, b_2 \ \ldots \ b_{p-1}]_2 = b_0 2^{p-1} + b_1 2^{p-2} + b_2 2^{p-3} + \cdots + b_{p-1}$$

(bit reversal), dann gelten

$$y_i = \tilde{y}_{r(i)} \quad \text{und} \quad \tilde{y}_i = y_{r(i)}.$$

Inhalt

1 Einleitung

2 Computer-Arithmetik und Fehleranalyse
- 2.1 Ein Beispiel
- 2.2 Gleitpunktzahlen
- 2.3 Rundung
- 2.4 Gleitpunktarithmetik
- 2.5 Numerische Stabilität und Fehleranalyse

3 Schnelle Fourier-Transformation
- 3.1 Vorbemerkungen
- 3.2 Überblick Fourier-Analyse
- 3.3 Die diskrete Fourier-Transformation
- 3.4 Interpolation
- 3.5 Trigonometrische Interpolation
- 3.6 Die FFT

4 Abschließende Bemerkungen