5.5 Splines

Wir kehren wieder zurück zur Interpolation.

Bis jetzt

- #Stützstellen = Polynomgrad + 1
- Großer Polynomgrad = viele Stützstellen $\Rightarrow$ evtl. starke Abweichung zwischen den Stützstellen.

Idee: Stückweise Polynome niedrigen Grades.

Definition 5.16

Sei $X = (x_0, x_1, \dots, x_n)$ mit $a = x_0 < x_1 < \dots < x_n = b$ eine Zerlegung des Intervalles $[a,b]$ und sei $m \in \mathbb{N}$. Die Menge

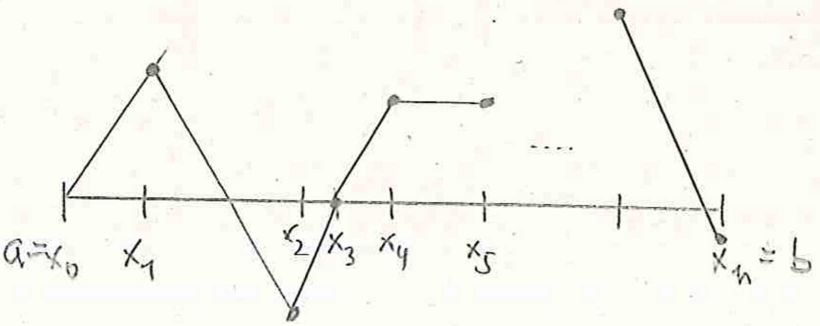
$$S^m(X) = \left\{ s \in C^{m-1}([a,b]) : \; s\big|_{[x_i, x_{i+1}]} \in P_m, \; 0 \le i < n \right\}$$

heißt Spline-Raum vom Grad m über der Zerlegung X.

Beispiel 5.17

$S^1(X)$ bedeutet:

- $s \in S^1(X)$ ist Polynom vom Grad 1 auf jedem Teilintervall $[x_i, x_{i+1}]$
- $S^1(X) \subset C^0([a,b])$, als $s \in S^1(X)$ stetig.



Stetigkeit $\Rightarrow$ $s \in S^1(X)$ ist eindeutig durch Werte in den Stützstellen beschrieben.

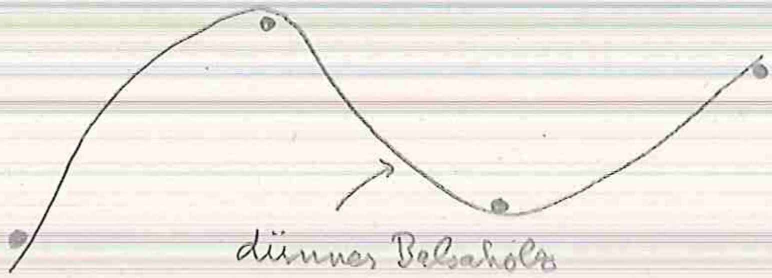
$\dim S^1(X) = |X|$.

Kubische Splines

In der Praxis ist $S^3(X)$ sehr beliebt.

$S^3(X)$ heißt Raum der kubischen Splines.

Geschichte: „Straklatte" zur Konstruktion glatter Kurven im Schiffs- und Flugzeugbau.



dünnes Balsaholz

biegt sich unter Energieminimierung

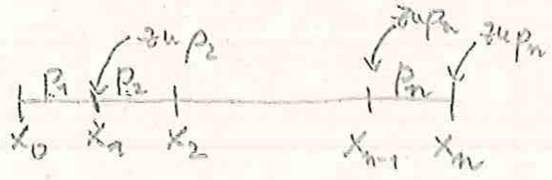
$$\int_a^b \frac{|y''(t)|^2}{1+|y'(t)|^2}\,dt \approx \int_a^b |y''(t)|^2\,dt \to \min.$$

Krümmung. $\qquad |y'(t)| \ll 1$

Konstruktion von $s \in S^3(X)$. Die Funktion setzt sich stückweise aus n Polynomen zusammen:

$$s(x) = \begin{cases} p_i(x) & x \in [x_{i-1}, x_i) \quad i \in \{1,\dots,n\} \\ p_n(x_n) & x = x_n \end{cases}$$

Für die Polynome p_i gelten folg. Bed.:



(a) Interpolationsbedingung (Stetigkeit)

$$i = 1,\dots,n: \quad \left.\begin{aligned} p_i(x_{i-1}) &= y_{i-1} \\ p_i(x_i) &= y_i \end{aligned}\right\} \ 2n \text{ Bedingungen}$$

(b) Stetigkeit der ersten und zweiten Ableitung an inneren Punkten:

$$i = 1,\dots,n-1 \quad \left.\begin{aligned} p_i'(x_i) &= p_{i+1}'(x_i) \\ p_i''(x_i) &= p_{i+1}''(x_i) \end{aligned}\right\} \ 2(n-1) = 2n-2 \text{ Bedingungen.}$$

$\Rightarrow$ zusammen $4n-2$ Bedingungen.

Pro Polynom p_i (vom Grad 3) hat man 4, also insgesamt $4n$ Freiheitsgrade.

Die fehlenden 2 Bedingungen erhält man durch Randbedingungen an den Stellen x_0 und x_n. Dabei gibt es verschiedene Varianten:

(c) Randbedingungen, Eine der folgenden Varianten:

i) Natürliche Randbedingungen:

$$p_1''(x_0) = 0$$
$$p_n''(x_n) = 0$$

„Krümmung 0"

ii) Hermite-Randbedingungen

$$p_1'(x_0) = f'(x_0)$$
$$p_n'(x_n) = f'(x_n)$$

f ist die zu interpolierende Funktion

iii) Periodische Randbedingungen

$$p_1'(x_0) = p_n'(x_n)$$
$$p_1''(x_0) = p_n''(x_n)$$

Wir behandeln im folgenden nur die natürliche RB (c)(i).

Neue Tafel! | 26
8.12.09

Satz 5.18 (Berechnung kubischer Splines)

Wir schreiben die Teilpolynome des Splines in der Form

$$p_i(x) = a_0^{(i)} + a_1^{(i)}(x-x_i) + a_2^{(i)}(x-x_i)^2 + a_3^{(i)}(x-x_i)^3 \quad i=1,\dots,n.$$

Die $a_2^{(i)}$ sind dann die Lösung des linearen Gleichungssystems der Dimension $n-1$:

$$h_i a_2^{(i-1)} + 2(h_i + h_{i+1}) a_2^{(i)} + h_{i+1} a_2^{(i+1)} = 3\left(\frac{y_{i+1}-y_i}{h_{i+1}} - \frac{y_i - y_{i-1}}{h_i}\right) \quad i=1,\dots,n-1, \tag{5.2}$$

wobei $a_2^{(0)} = a_2^{(n)} = 0$ (natürliche Randbedingung!) und $h_i := x_i - x_{i-1}$.

Die restlichen Koeffizienten ergeben sich zu:

$$a_0^{(i)} = y_i \tag{5.3}$$

$$a_1^{(i)} = \frac{y_i - y_{i-1}}{h_i} + \frac{h_i}{3}\left(2a_2^{(i)} + a_2^{(i-1)}\right) \tag{5.4}$$

$$a_3^{(i)} = \frac{a_2^{(i)} - a_2^{(i-1)}}{3h_i} \tag{5.5}$$

Beweis:

(i) Berechne Ableitungen der Teilpolynome

$$p_i'(x) = a_1^{(i)} + 2a_2^{(i)}(x-x_i) + 3a_3^{(i)}(x-x_i)^2 \tag{5.6}$$

$$p_i''(x) = 2a_2^{(i)} + 6a_3^{(i)}(x-x_i) \tag{5.7}$$

(ii) Interpolationsbed. nutzen. Einsetzen von x_i:

$$y_i = p_i(x_i) = a_0^{(i)} \Rightarrow \boxed{a_0^{(i)} = y_i} \quad i=1,\dots,n. \tag{5.8}$$

Das ist (5.3).

Einsetzen von x_{i-1}:

$$y_{i-1} = p_i(x_{i-1}) = \underbrace{a_0^{(i)}}_{=y_i} + a_1^{(i)}(-h_i) + a_2^{(i)} h_i^2 + a_3^{(i)}(-h_i^3) \quad i=1,\dots,n$$

$$\Longleftrightarrow \quad y_{i-1} - y_i = -h_i a_1^{(i)} + h_i^2 a_2^{(i)} - h_i^3 a_3^{(i)} \tag{5.9}$$

(viii) Nun setze $a_1^{(i)}$ und $a_3^{(i)}$ in die verbleibende Gleichung (5.12) ein

$$\frac{y_i - y_{i-1}}{h_i} + \frac{h_i}{3}\left(2a_2^{(i)} + a_2^{(i-1)}\right) = \frac{y_{i+1}-y_i}{h_{i+1}} + \frac{h_{i+1}}{3}\left(2a_2^{(i+1)} + a_2^{(i)}\right)$$

$$-2a_2^{(i+1)}h_{i+1} + 3h_{i+1}^2\left(\frac{a_2^{(i+1)} - a_2^{(i)}}{3h_{i+1}}\right) \qquad i = 1,\dots,n-1$$

$\Longrightarrow$

$$a_2^{(i-1)}\left(+\frac{h_i}{3}\right) + a_2^{(i)}\left(\frac{2h_i}{3} - \frac{h_{i+1}}{3} + h_{i+1}\right) + a_2^{(i+1)}\left(-\frac{2h_{i+1}}{3} + 2h_{i+1} - h_{i+1}\right)$$

$$= \frac{y_{i+1}-y_i}{h_{i+1}} - \frac{y_i - y_{i-1}}{h_i}$$

mal 3

$$\Longleftrightarrow \quad h_i a_2^{(i-1)} + 2(h_i + h_{i+1})a_2^{(i)} + h_{i+1}a_2^{(i+1)} = 3\left(\frac{y_{i+1}-y_i}{h_{i+1}} - \frac{y_i - y_{i-1}}{h_i}\right)$$

Und das ist (5.2).

Beachte, dass in (iii) und (vi) $a_2^{(0)} = a_2^{(n)} = 0$ gesetzt wurde. ∎

Zur Lösung des Tridiagonalsystem:

- GEM / LR-Zerlegung hat in diesem Fall $O(n)$ Aufwand!
  → sehr schnell.
- Das Gleichungssystem ist symmetrisch und strikt diagonal dominant

$$\sum_{\substack{j=1\\ j\neq i}}^{n-1} |a_{ij}| < |a_{ii}|$$

$\Rightarrow$ Regularität, stabile LR-Zerlegung ohne Pivotisierung

<u>Beispiel 5.19</u> Aus Numerik Beispiel 6.2.

(iii) Randbedingungen einsetzen. Wir behandeln nur natürliche.

$$0 = p_1''(x_0) = 2a_2^{(1)} - 6a_3^{(1)}h_1 \qquad (5.10)$$

$$0 = p_n''(x_n) = 2a_2^{(n)} \quad \Rightarrow \boxed{a_2^{(n)} = 0} \qquad (5.11)$$

(iv) Stetigkeit der ersten Ableitung

$$p_i'(x_i) = p_{i+1}'(x_i) \qquad i = 1, \dots, n-1$$

$$\Leftrightarrow \quad a_1^{(i)} = a_1^{(i+1)} - 2a_2^{(i+1)}h_{i+1} + 3a_3^{(i+1)}h_{i+1}^2 \qquad (5.12)$$

(v) Stetigkeit der zweiten Ableitung

$$p_i''(x_i) = p_{i+1}''(x_i) \qquad i = 1, \dots, n-1$$

$$\Leftrightarrow \quad 2a_2^{(i)} = 2a_2^{(i+1)} - 6a_3^{(i+1)}h_{i+1} \qquad (5.13)$$

(vi) Drücke $a_3^{(i)}$ durch $a_2^{(i)}$ aus; d.h. löse (5.13) nach $a_3^{(i+1)}$ auf.

$$a_3^{(i+1)} = \frac{a_2^{(i+1)} - a_2^{(i)}}{3h_{i+1}} \qquad i = 1, \dots, n-1$$

umnummerieren

$$\Leftrightarrow \boxed{a_3^{(i)} = \frac{a_2^{(i)} - a_2^{(i-1)}}{3h_i}}$$

$i = 2, \dots, n$ aus (v) (5.14)

$i = 1$ aus (5.10) wenn man $\boxed{a_2^{(0)} := 0}$ setzt.

also $i = 1, \dots, n$

Das ist (5.5)

(vii) $a_1^{(i)}$ durch $a_2^{(i)}$ ausdrücken. Dazu löse (5.8) nach a_1 auf:

$$a_1^{(i)} = \frac{y_i - y_{i-1}}{h_i} + h_i a_2^{(i)} - h_i^2 a_3^{(i)} \qquad i = 1, \dots, n$$

$$\overset{(5.13)\text{ eins.}}{=} \frac{y_i - y_{i-1}}{h_i} + h_i a_2^{(i)} - h_i^2 \left(\frac{a_2^{(i)} - a_2^{(i-1)}}{3h_i} \right) \qquad i = 1, \dots, n$$

$$\boxed{a_1^{(i)} = \frac{y_i - y_{i-1}}{h_i} + \frac{h_i}{3}\left(2a_2^{(i)} + a_2^{(i-1)}\right)} \qquad i = 1, \dots, n$$

Das ist (5.4)

Satz 5.20 (Fehlerabschätzung)

Sei $f \in C^4([a,b])$. Erfüllt der kubische Spline

$$s''(a) = f''(a) \quad \text{und} \quad s''(b) = f''(b)$$ (hatten wir oben nicht)

so gilt

$$\max_{a \le x \le b} |f(x) - s(x)| \le \frac{1}{2} h^4 \max_{a \le x \le b} |f^{(4)}(x)|$$

für $h := \max_{1 \le i \le n} |x_i - x_{i-1}|$.

Beweis: Werner/Schaback II, siehe [Ramacher]. ☒

Selbst unter noch (wesentlich) schwächeren Voraussetzungen konvergiert die Spline-Interpolierende gleichmäßig gegen f.

Beispiel 5.21 Zur Konvergenzordnung Beispiel 6.5 aus Num Stoch.

5.6 Trigonometrische Interpolation

<u>Problem</u>: Interpolation „periodischer" Funktionen, d.h. es gebe ein $\omega \in \mathbb{R}$, $\omega > 0$ so dass

$$f(x+\omega) = f(x) \qquad \forall x \in \mathbb{R}.$$

Es bietet sich die Interpolation mit „trigonometrischen Summen"

$$t_n(x) = \frac{a_0}{2} + \sum_{k=1}^{m} \left\{ a_k \cos\left(\frac{k x 2\pi}{\omega}\right) + b_k \sin\left(\frac{k x 2\pi}{\omega}\right) \right\} \qquad (5.15)$$

an, denn jeder Summand ist ω-periodisch: $\cos(k(x+\omega)2\pi/\omega) = \cos(kx2\pi/\omega + k2\pi) = \cos\left(\frac{kx2\pi}{\omega}\right)$

(5.15) hat $2m+1$ Parameter, deshalb setze

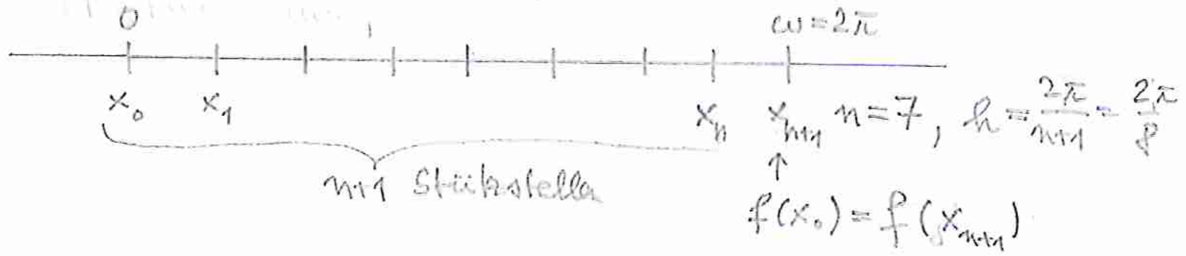
$$n := 2m$$

(bisher: Polynom vom Grad n hatte $n+1$ freie Parameter).

Ab jetzt: Nehme an, dass $\omega = 2\pi$ (sonst Transformation des Arguments).

<u>Außerdem</u>: Verwende wieder äquidistante Stützstellen

$$x_k = \frac{2\pi k}{n+1} \qquad k = 0, \dots, n.$$



<u>Es zeigt sich</u>: Die Interpolationsaufgabe $t_n(x_k) = f(x_k)$ $k = 0, \dots, n$ ist zunächst einfacher im Körper $\mathbb{C}$ zu lösen!

D.h. betrachte das <u>komplexe</u> trigonometrische Polynom

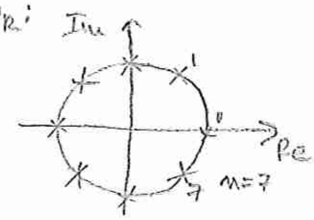
$$t_n^*(x) = \sum_{k=0}^{n} c_k e^{ikx}$$

mit $i = \sqrt{-1}$ imaginäre Einheit

$c_k \in \mathbb{C}$ komplexe Koeffizienten

Eulersche Identität $e^{i\varphi} = \cos\varphi + i \sin\varphi$ für $\varphi \in \mathbb{R}$.

Wir untersuchen zunächst die Eigenschaften der komplexen Exponentialfunktion:



Hilfssatz 5.22 (Komplexe Einheitswurzeln)

Setze $w_k := e^{ix_k} = e^{i\frac{2\pi k}{n+1}}$ für alle $k\in\mathbb{Z}$ und gegebenes $n\in\mathbb{N}$.
$w_k \in \mathbb{C}$ heißt „k-te Einheitswurzel" und hat folgende Eigenschaften

a) $w_k^{n+1} = 1$ für alle $k\in\mathbb{Z}$.

M.a.W. die w_k sind Lösungen der Gleichung $w^{n+1}-1=0$ in $\mathbb{C}$.

Bew: $w_k^{n+1} = \left(e^{i\frac{2\pi k}{n+1}}\right)^{n+1} = e^{i2\pi k} = \underbrace{\cos 2\pi k}_{=1} + i\underbrace{\sin 2\pi k}_{=0} = 1.$

b) $w_k^j = w_j^k \quad \forall j,k\in\mathbb{Z}$

Bew: $w_k^j = \left(e^{i\frac{2\pi k}{n+1}}\right)^j = e^{i\frac{2\pi kj}{n+1}} = \left(e^{i\frac{2\pi j}{n+1}}\right)^k = w_j^k$

c) $w_k^{-j} = w_j^{-k} \quad \forall j,k\in\mathbb{Z}$

$w_k^{-j} \overset{(b)}{=} w_{-j}^k \overset{z.z.}{=} w_j^{-k}$ sieht man hier.

Beweis genau wie b, ist aber nicht identisch zu b).

d) $w_k^j = w_k^{j \bmod (n+1)} = w_{k \bmod (n+1)}^j = w_{k \bmod (n+1)}^{j \bmod (n+1)} \quad \forall j,k\in\mathbb{Z}$.

Zeige nur erste Identität, Rest analog:
Sei $j = r(n+1)+s$ mit $0\le s<n+1$.

$$w_k^j = e^{i\frac{2\pi kj}{n+1}} = e^{i\frac{2\pi[kr(n+1)+ks]}{n+1}} = \underbrace{e^{2\pi kr}}_{=1}\cdot\underbrace{e^{i\frac{2\pi ks}{n+1}}}_{(w_k)^s} = w_k^{j \bmod (n+1)}$$

e) $$\sum_{j=0}^{n} w_k^j = \begin{cases} n+1 & k \bmod (n+1) = 0 \\ 0 & \text{sonst}\end{cases}$$

Sei $\underline{k=0}$: $w_0^j = e^0 = 1 \ \forall j$, also $\sum_{j=0}^{n} 1 = n+1$.

$\underline{k\neq 0}$: w_k ist nach a) Lösung von

„Teleskopsumme"

$$w^{n+1}-1 = (w-1)(w^n+w^{n-1}+\dots+1) = 0$$

Für $k\neq 0$ ist $w_k\neq 1$ also $w_k-1\neq 0$ somit muss der zweite Faktor, also $\sum_{j=0}^{n} w_k^j = 0$ sein.

Satz 5.23 (Komplexe trigonometrische Interpolation)

Zu gegebenen Zahlen $y_0, \dots, y_n \in \mathbb{C}$ gibt es genau eine Funktion der Gestalt

$$t_n^*(x) = \sum_{k=0}^{n} c_k e^{ikx},$$

die den Interpolationsbedingungen

$$t_n^*(x_j) = y_j \qquad j = 0, \dots, n, \quad x_j = \frac{2\pi j}{n+1}$$

genügt. Die komplexen Koeffizienten sind bestimmt durch

$$c_k = \frac{1}{n+1} \sum_{j=0}^{n} y_j \underbrace{e^{-ijx_k}}_{= e^{-i\frac{2\pi jk}{n+1}} = w_k^{-j}} \qquad \forall\, k = 0, \dots, n. \qquad (6.15)$$

Beweis: Mit der Abkürzung $w = e^{ix}$ gilt

$$t_n^*(x) = \sum_{k=0}^{n} c_k e^{ikx} = \sum_{k=0}^{n} c_k \big(\underbrace{e^{ix}}_{=w}\big)^k = \sum_{k=0}^{n} c_k w^k = p_n(w).$$

Transformation $w = e^{ix}$ eindeutig auf $0 \dots 2\pi$

Jedem t_n^* entspricht also genau ein komplexes Polynom p_n vom Grad n.
Übertragung der Interpolationsbedingungen für t_n^* auf p_n ergibt:

$$t_n^*(x_j) = p_n(\underbrace{e^{ix_j}}_{=w_j}) = y_j \qquad \forall\, j = 0, \dots, n.$$

Da die Polynominterpolation zu paarw. versch. Stützstellen (auch im Komplexen) eindeutig ist, gibt es genau ein solches p_n, also auch t_n^*.

Bleibt die Berechnung der Koeffizienten c_k. Diese ergeben sich durch das lineare Gleichungssystem

$$p_n(e^{ix_j}) = \sum_{\ell=0}^{n} c_\ell (e^{ix_j})^\ell = \sum_{\ell=0}^{n} c_\ell e^{i\frac{2\pi \ell j}{n+1}} = \underbrace{\sum_{\ell=0}^{n} c_\ell w_j^\ell \overset{!}{=} y_j}_{(n+1)\times(n+1) \text{ LGS für die Koeffizienten}} \qquad j = 0, \dots, n.$$

Das Gleichungssystem kann man explizit auflösen:

Für ein $k \in 0,\dots,n$:

Bilde Linear-Kombi.
$$\sum_{j=0}^{n} w_k^{-j} \underbrace{\left(\sum_{\ell=0}^{n} c_\ell w_j^{\ell}\right)}_{=y_j} \overset{1)\ \sum \text{ vertauschen},\ 2)\ w_k^{-j}=w_j^{-k}\ (5.22c)}{=} \sum_{\ell=0}^{n} c_\ell \underbrace{\left(\sum_{j=0}^{n} w_j^{\ell-k}\right)}_{=\sum_{j=0}^{n} w_{\ell-k}^{j} = \begin{cases} n+1 & \ell-k=0 \\ 0 & \ell-k\neq 0\end{cases}} \overset{!}{=} \sum_{j=0}^{n} w_k^{-j} y_j$$

Und damit

$$c_k\,(n+1) = \sum_{j=0}^{n} y_j w_k^{-j} \iff c_k = \frac{1}{n+1}\sum_{j=0}^{n} y_j e^{-ijx_k} \quad k=0,\dots,n$$
$\square$.

→ Aufwand $O(n^2)$ zur Berechnung aller c_k.

Damit ist auch die reelle Interpolationsaufgabe gelöst, indem man $y_j \in \mathbb{R}$ annimmt.

Zu zeigen ist nun, wie die Koeffizienten a_k, b_k berechnet werden.

Satz 5.24 (Diskrete Fourier-Analyse)

Für $n \in \mathbb{N}_0$ gibt es zu gegebenen reellen Zahlen $y_0,\dots,y_n$ genau ein trigonometrisches Polynom der Form

$$t_n(x) = \frac{a_0}{2} + \sum_{k=1}^{m}\{a_k\cos(kx) + b_k\sin(kx)\} + \frac{\theta}{2} a_{m+1}\cos((m+1)x)$$

mit $t_n(x_j) = y_j$, $j=0,\dots,n$, $x_j = \frac{2\pi j}{n+1}$, sowie

$\theta = 0$, $m = \frac{n}{2}$ $\quad$ n gerade $\to a_0 \dots a_m, b_1,\dots,b_m$

$\theta = 1$, $m = \frac{n-1}{2}$ $\quad$ n ungerade $\to a_0,\dots,a_{m+1}, b_1,\dots,b_m$

(n gerade: $2\cdot(n/2)+1 = n+1$, n ungerade: $2\cdot(n-1)/2+2 = n+1$).

Die Koeffizienten werden bestimmt durch

$$a_k = \frac{2}{n+1}\sum_{j=0}^{n} y_j\cos(jx_k),\quad b_k = \frac{2}{n+1}\sum_{j=0}^{n} y_j \sin(jx_k).$$

Beweisskizze: (ausführlich: siehe Ra-Skript)

Man bestimmt die Koeffizienten c_k des komplexen trigonometrischen Polynoms zu reellen Daten $y_j \in \mathbb{R}$ und setzt dann

$$a_0 = 2c_0$$

$$a_k = c_k + c_{n+1-k} \qquad k = 1, \dots, m$$

$$b_k = i(c_k - c_{n+1-k}) \qquad k = 1, \dots, m$$

$$a_{m+1} = 2c_{m+1} \qquad n = 2m+1 \text{ (n ungerade)}.$$

Dann rechnet man nach, dass $t_n^{tr}(x_j) = t_n(x_j) = y_j \in \mathbb{R}$.

Wir wollen noch nachrechnen, dass die a_k reell sind:

$$e^{-i\frac{2\pi j(n+1)-k)}{n+1}} = e^{i\frac{2\pi jk}{n+1}} = e^{ijx_k}$$

$$a_k = c_k + c_{n+1-k}$$

$$\overset{(6.15)}{=} \frac{1}{n+1} \sum_{j=0}^{n} y_j \left(e^{-ijx_k} + e^{-ijx_{n+1-k}}\right)$$

$$= \frac{1}{n+1} \sum_{j=0}^{n} y_j \left(e^{-ijx_k} + e^{ijx_k}\right)$$

$$= \frac{1}{n+1} \sum_{j=0}^{n} y_j \left(\underbrace{\cos(-jx_k)}_{=\cos(jx_k)} + i\underbrace{\sin(-jx_k)}_{=-\sin(jx_k)} + \cos(jx_k) + i\sin(jx_k)\right)$$

$$= \frac{1}{n+1} \sum_{j=0}^{n} y_j \, 2\cos(jx_k).$$

Schnelle Fouriertransformation

Einer der berühmtesten Algorithmen der angewandten Mathematik / Informatik!

Entwickelt von James Cooley und John Tukey 1965.

Zurück zur komplexen trigonometrischen Interpolation:

$$\boxed{\begin{aligned} c_k &= \frac{1}{N}\sum_{j=0}^{N-1} y_j e^{-i\frac{2\pi jk}{N}} \qquad k=0,\dots,N-1 \quad \text{(Hintransformation)} \qquad (6.15a) \\ y_j &= \sum_{k=0}^{N-1} c_k e^{i\underbrace{\frac{2\pi j}{N}}_{x_j}k} \qquad j=0,\dots,N-1 \quad \text{(Rücktransformation)} \qquad (6.15b) \end{aligned}}$$

- Hier wurde $N = n+1$ gesetzt.
- Die Berechnung der y_j aus den c_k ist schlicht die Definition des komplexen trig. Polynoms
- 6.15 bezeichnet man als <u>diskrete Fouriertransformation</u>.

Aufwand: Betrachte

$$c := (c_0, \dots, c_{N-1})^T, \quad y := (y_0, \dots, y_{N-1})$$

Dann ist (6.15a) äquivalent zu

$$c = \frac{1}{N} W y$$

mit $(W)_{k,j} = e^{-i\frac{2\pi jk}{N}} = w_k^{-j}$ (Einheitswurzel)

Die Rücktransformation (6.15b) lautet

$y = Uc$ mit $(U)_{j,k} = w_j^k$.

Wegen $U \frac{1}{N} W = I$ ist $W^{-1} = \frac{1}{N} U$, man hat also explizite Darst. der Inversen.

Da W, U voll besetzt sind beträgt der Aufwand für Hin- und Rücktransformation jeweils $O(N^2)$.

Betrachte (6.15a) ohne den Vorfaktor $\frac{1}{N}$, nenne das $\tilde{c}_k$. | 12.01.10

Es sei N gerade, dann gilt:

$$\tilde{c}_k = \sum_{j=0}^{N-1} y_j e^{-i\frac{2\pi jk}{N}}$$

Aufspalten der Summe in gerade und unger. Indizes

$$= \underbrace{\sum_{j=0}^{N/2-1} y_{2j} e^{-i\frac{2\pi 2jk}{N}}}_{\text{gerader Teil}} + \underbrace{\sum_{j=0}^{N/2-1} y_{2j+1} e^{-i\frac{2\pi(2j+1)k}{N}}}_{\text{ungerader Teil}}$$

bringe N/2 in Nenner

$$= \underbrace{\sum_{j=0}^{N/2-1} y_{2j} e^{-i\frac{2\pi jk}{N/2}}}_{=:\tilde{c}_k^g} + \underbrace{e^{-i\frac{2\pi k}{N}}}_{\text{Ausgeklammert!}} \underbrace{\sum_{j=0}^{N/2-1} y_{2j+1} e^{-i\frac{2\pi jk}{N/2}}}_{=:\tilde{c}_k^u}$$

Wegen der N/2-Periodizität der Einheitswurzeln $e^{-i\frac{2\pi k}{N/2}}$ gilt

$$\tilde{c}^g_{k+N/2} = \tilde{c}^g_k \quad \text{und} \quad \tilde{c}^u_{k+N/2} = \tilde{c}^u_k \quad \forall k = 0,\dots,N/2-1.$$

Damit gilt:

- $\tilde{c}_k^g, \tilde{c}_k^u$, $k = 0,\dots,N/2-1$ berechnen sich jeweils durch eine DFT der Länge N/2
- Daraus berechnet man dann die ursprünglich gesuchten Koeffizienten

$$\tilde{c}_k = \tilde{c}_k^g + e^{-i\frac{2\pi k}{N}} \tilde{c}_k^u \qquad 0 \le k < N/2 \qquad (6.16a)$$

$$\tilde{c}_k = \tilde{c}_{k-N/2} + e^{-i\frac{2\pi k}{N}} \tilde{c}^u_{k-N/2} \qquad N/2 \le k < N. \qquad (6.16b)$$

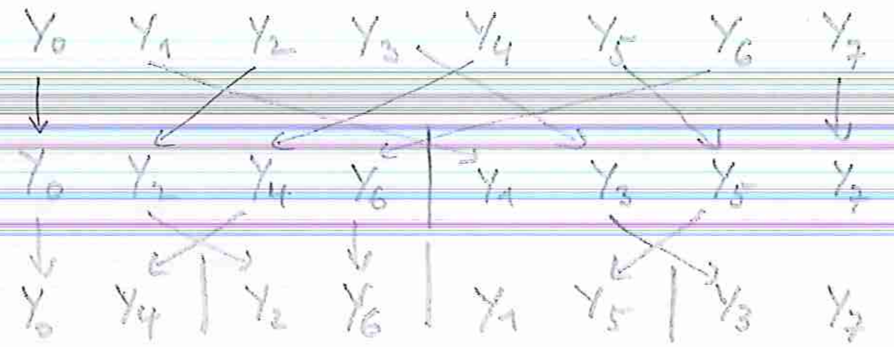
- Falls N/2 wieder gerade kann man das Prinzip rekursiv fortsetzen.
- Ist $N = 2^d$ eine Zweierpotenz erhält man schließlich eine DFT der Länge 2, die einfach durchführbar ist.

Beispiel $N=8$.

„Abstiegsphase" der Rekursion: Umsortieren der Eingabedaten.

Am besten schreibt man die Indizes zur Basis 2

$000_2 \quad 001_2 \quad 010_2 \quad 011_2 \quad 100_2 \quad 101_2 \quad 110_2 \quad 111_2$

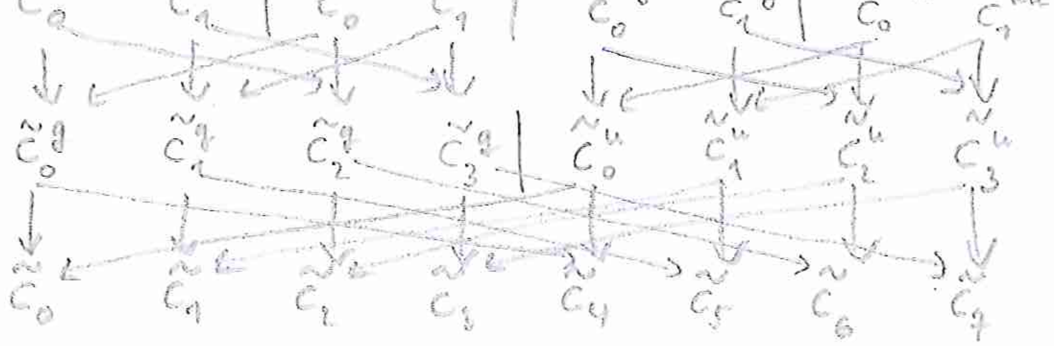


$000_2 \quad 100_2 \quad 010_2 \quad 110_2 \quad 001_2 \quad 101_2 \quad 011_2 \quad 111_2$

Permutation $(b_{d-1} \dots b_0)_2 \longrightarrow (b_0 \dots b_{d-1})_2$ heißt „bit reversal"

„Aufstiegsphase": Rekombination der Koeffizienten nach (6.16)

↑ von unten nach oben



Dieses Verknüpfungsmuster nennt man „perfect shuffle".

Aufwand der FFT: $N = 2^d$ Zweierpotenz $\Rightarrow d = \operatorname{ld} N$.

$$A(N) = \underbrace{2A\left(\tfrac{N}{2}\right)}_{\text{2 Transf. der Länge } N/2 \text{ Ber.}} + \underbrace{cN}_{\text{Aufwand für (6.16)}}$$

$A(N)$: # Gleitkommaoperationen für DFT der Länge N

$$= 2\left[2A\left(\tfrac{N}{4}\right) + c\tfrac{N}{2}\right] + cN$$

$$= 4A\left(\tfrac{N}{4}\right) + cN + cN$$

$$= 4\left(2A\left(\tfrac{N}{8}\right) + c\tfrac{N}{4}\right) + cN + cN$$

$$= 8A\left(\tfrac{N}{8}\right) + cN + cN + cN$$

d mal

$$= \underbrace{2^d}_{=N} \underbrace{A(1)}_{=c} + \underbrace{cN + \ldots + cN}_{d-1 \text{ mal}} = cdN$$

$$= O(N \operatorname{ld} N)$$

deutlich schneller für große N.

Praktisches zur DFT:

- Spektralanalyse
- Beispiele auf dem Computer.