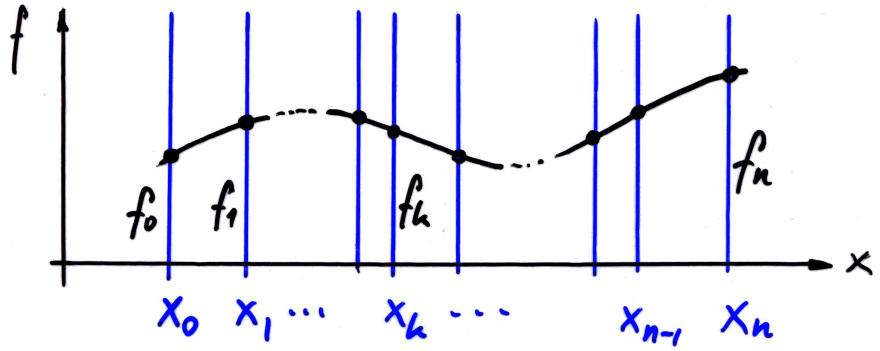
6. Interpolation

6.1. Problemstellung

Rekonstruktion einer Funktion aus diskreten Punkten ("sampled values")



Gegeben: $n+1$ Stützstellen $x_0, \cdots, x_n$, alle verschieden

$n+1$ Stützwerte $f_0, f_1, \cdots, f_n$

(diskrete Werte einer Funktion $f(x)$ in $x_0 \le x \le x_1$)

Gesucht: Approximation $p(x)$ für $f(x)$ in $x_0 \le x \le x_1$, z.B. zur Gewinnung von Zwischenwerten.

Interpolationsbedingung

$$p(x_k) = f_k \,, \quad k = 0, 1, \cdots, n \tag{1}$$

Man braucht einen Ansatz für $p(x)$,

z.B. $p(x)$ periodisch, $p(x)$ = Exponentialsumme, ...

Einfachster Fall:

$p(x)$ = Polynom vom Grad $\le n$

$n+1$ Parameter für $n+1$ Bedingg. (1). 1-deut. Lösung ?

6.2. Das Lagrange-Polynom

Wie man es **nicht** machen soll:

(2) $$p(x) = a_0 + a_1 x + a_2 x^2 + \dots + a_n x^n$$

Interpolationsbedingung (1) ergibt

$$\underbrace{\begin{bmatrix} 1 & x_0 & x_0^2 & \dots & x_0^n \\ 1 & x_1 & x_1^2 & \dots & x_1^n \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_n & x_n^2 & \dots & x_n^n \end{bmatrix}}_{V \in \mathbb{R}^{(n+1)\times(n+1)},\ \text{Vandermonde-Matrix}} \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{bmatrix} = \begin{bmatrix} f_0 \\ f_1 \\ \vdots \\ f_n \end{bmatrix}$$

- Ist das lineare Gleichungssystem lösbar ?
- Wie löst man es effizient und genau ?

Bemerkungen:

- $\text{cond}(V)$ ist häufig gross
- Gauss-Elimination, Aufwand $= \frac{n^3}{3}$ Operationen ist viel zu teuer; es geht schneller !

Gute Methode:

Basisfunktionen von Lagrange

- einfach
- liefert Existenz und Eindeutigkeit
- liefert effiziente Algorithmen

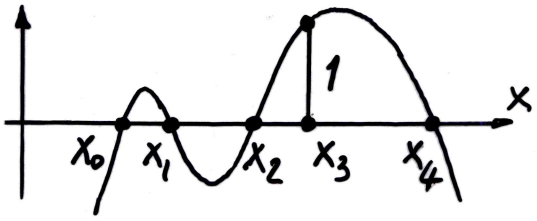
Bestimme Basisfunktionen $B_k(x)$ mit

(3) $$B_k(x_\ell) = \delta_{k\ell} = \begin{cases} 1 & \text{für } k=\ell \\ 0 & \text{für } k\neq\ell \end{cases}$$

(4) $$\Rightarrow \quad B_k(x) = \frac{\prod_{i\neq k} (x - x_i)}{\prod_{\substack{i=0 \\ i\neq k}}^{n} (x_k - x_i)}, \qquad k = 0, 1, \dots, n$$

($n+1$ Polynome vom Grad $= n$)

Beispiel: B_3 für $n = 4$



Dann erfüllt

(5) $$p(x) := \sum_{k=0}^{n} f_k \, B_k(x)$$

die Interpolationsbedingung (1)

Bemerkung: – $p(x)$ hat Grad $\leq n$

– Dies funktioniert immer, wenn alle x_k verschieden. Existenzbeweis!

Eindeutig? Ja:

Seien $p(x)$, $q(x)$ zwei Lösungen; $\Delta(x) := p(x) - q(x)$.
Δ vom Grad $\leq n$ mit $n+1$ Nullstellen x_k. $\Rightarrow \Delta(x) \equiv 0$.

SATZ. Für $n+1$ verschiedene Stützstellen x_k $(k=0,1,\dots,n)$ und beliebige Stützwerte f_k existiert genau ein Interpol. polynom p vom Grad $\leq n$ mit $p(x_k) = f_k$, $k=0,\dots,n$

6.3. Die baryzentrische Formel

Effiziente Auswertung von p an gegebener "Neustelle" x.

Def. $L(x) := \prod_{i=0}^{n} (x - x_i)$ Produkt aller Linearfaktoren

$$w_k := \frac{1}{\prod_{\substack{i=0 \\ i \neq k}}^{n} (x_k - x_i)}, \quad k = 0, 1, \cdots, n$$

"Stützkoeffizienten"

Damit wird (4):

$$B_k(x) = L(x) \underbrace{\frac{w_k}{x - x_k}}_{c_k(x)} = L(x) \cdot c_k(x)$$

(5) ergibt:

(6) $$p(x) = L(x) \cdot \sum_{k=0}^{n} f_k \, c_k(x) \quad \text{mit} \quad c_k(x) := \frac{w_k}{x - x_k}$$

unabhängig von f_k

(6) muss f. beliebige Wahl von $\{f_k\}$ gelten,
z. B. für $f_k = 1$ $(k = 0, \cdots, n)$ $\Rightarrow$ $f(x) \equiv 1$.

Damit folgt aus (6):

$$1 = L(x) \cdot \sum_{k=0}^{n} c_k(x) \quad \Rightarrow \quad L(x) = \frac{1}{\sum_{k=0}^{n} c_k(x)}$$

Zusammenfassung

Seien $x_0, x_1, \cdots, x_n$ $n+1$ verschiedene Stützst.

$f_0, f_1, \cdots, f_n$ $n+1$ zugehörige Funktionswerte

Gegeben: "Neustelle" x.

Algorithmus : Auswertung $p(x)$

(i) Berechne Stützkoeff., $w_k = \dfrac{1}{\prod\limits_{\substack{i=0 \\ i\neq k}}^{n} (x_k - x_i)}$, $k=0,\dots,n$

Aufwand: $\mathcal{O}(n^2)$

allein die Stst. gehen ein!

(ii) Für Neustelle x:

$$c_k := \frac{w_k}{x - x_k}, \qquad k = 0, 1, \dots, n$$

Damit

$$p(x) = \frac{\sum\limits_{k=0}^{n} c_k f_k}{\sum\limits_{k=0}^{n} c_k}$$

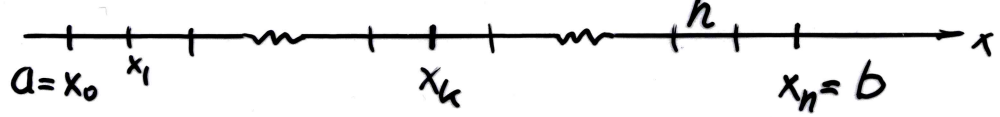
Aufwand: $\mathcal{O}(n)$; die f_k gehen erst zuletzt ein!

Bemerkung: Falls $x = x_k$, setze z.B. $c_k = 10^{20}$ $\left(\geq \frac{1}{eps}\right)$.

Die Stützkoeffizienten

Zwei wichtige Spezialfälle

(a) Äquidistante Stützstellen, Schritt $= h$.



$$w_k = \frac{1}{h^n \cdot k(k-1)\cdots 1 \cdot (-1)^{n-k} \cdot 1\cdot 2 \cdots (n-k)}$$

ein Vielfaches von w_k: $w_k^* = (-h)^n n!\, w_k = (-1)^k \binom{n}{k}$, $k=0,\dots,n$

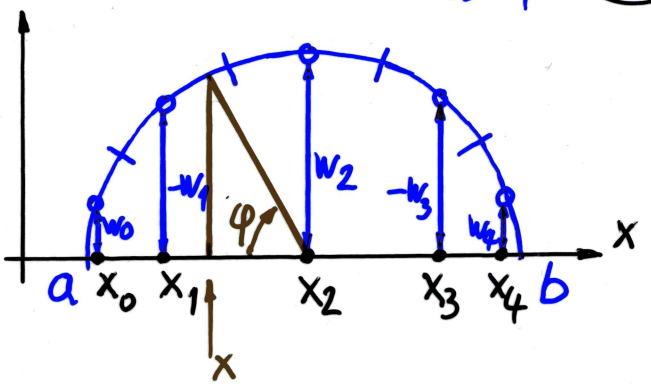
die Binomialkoeffizienten mit alternierendem Vorz.

Beispiel: $n = 3$

$w_k^* = \{1, -3, 3, -1\}$

(b) Tschebyschev-Punkte

$n=4$



$$x_k := \frac{a+b}{2} + \frac{a-b}{2}\cos\left(\frac{(k+\frac{1}{2})\pi}{n+1}\right)$$

$k = 0, 1, \cdots, n$

Allgemein gilt

$$w_k = \lim_{x \to x_k} \frac{x - x_k}{\prod_{i \neq k}(x - x_i) \cdot (x - x_k)} = \lim_{x \to x_k} \frac{x - x_k}{L(x) - \underbrace{L(x_k)}_{0}}$$

(7) Also:

$$\boxed{w_k = \frac{1}{L'(x_k)}} \qquad k = 0, 1, \cdots, n$$

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

Berechnung von $L(x)$ für die T-Punkte:

Setze $x := \frac{a+b}{2} + \frac{a-b}{2}\cos\varphi, \quad 0 \le \varphi \le \pi$

Wegen $L(x_k) = 0$ für $k = 0, 1, \cdots, n$ folgt

(8) $$L(x) = c \cdot \cos(n+1)\varphi \quad \text{mit } c = \text{const.}$$

Differentiation nach φ ergibt:

(9) $$L'(x) \cdot \frac{b-a}{2}\sin\varphi = -(n+1) \cdot c \cdot \sin(n+1)\varphi$$

Für w_k brauchen wir $L'(x_k)$; also setze

$x = x_k$, d.h. $\varphi = \varphi_k := \frac{(k+\frac{1}{2})\pi}{n+1}$

(10) $$\sin(n+1)\varphi_k = (-1)^k$$

Resultat aus (7), (9), (10):

$$\boxed{w_k = \text{const} \cdot (-1)^k \frac{b-a}{2} \sin\frac{(k+\frac{1}{2})\pi}{n+1}} \qquad k = 0, \cdots, n$$

6.4. Problematik der Polynominterpolation

SATZ: Sei f auf dem Intervall $I := [a, b]$ $(n+1)$ mal stetig differenzierbar, und sei p das zu den paarweise verschiedenen Stützstellen $x_k \in I$, $k = 0, \dots, n$ gehörige Interpolationspolynom vom Grad $\leq n$, d.h. $p(x_k) = f(x_k)$, $k = 0, \dots, n$.

Dann existiert zu einer beliebigen Neustelle $x \in I$ ein $\xi_x \in I$, so dass

$$f(x) - p(x) = \frac{1}{(n+1)!} L(x) \cdot f^{(n+1)}(\xi_x),$$

wobei

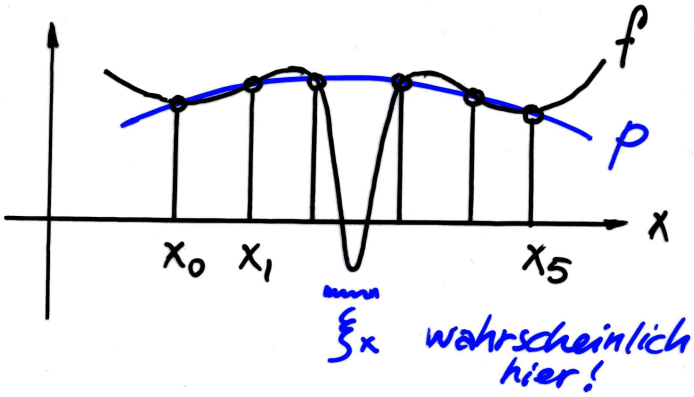
$$L(x) = (x - x_0)(x - x_1) \cdots (x - x_n).$$

Beweis: Siehe z.B. Schwarz

Folgerungen: Grosse Interpolationsfehler können 2 Ursachen haben:

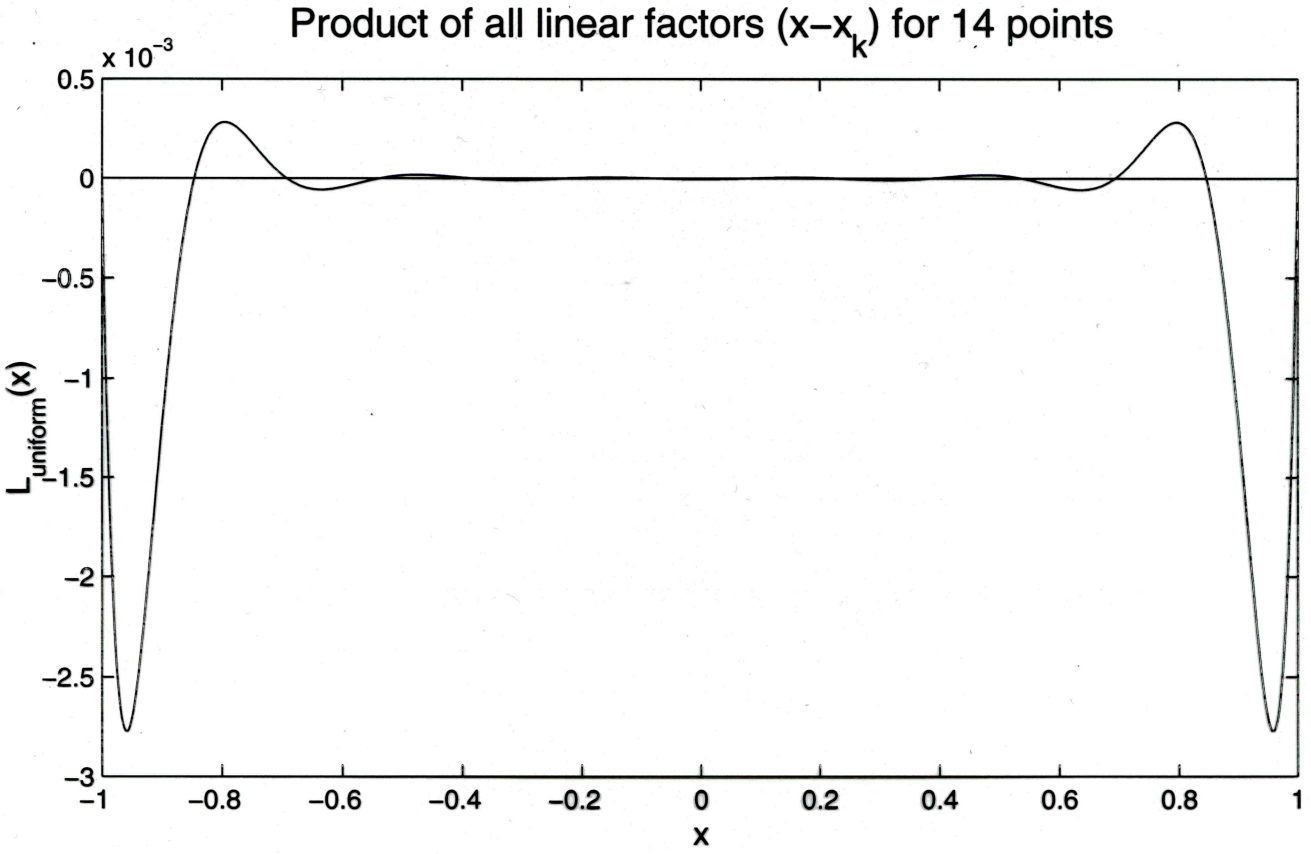
(i) $|f^{(n+1)}(\xi_x)|$ gross

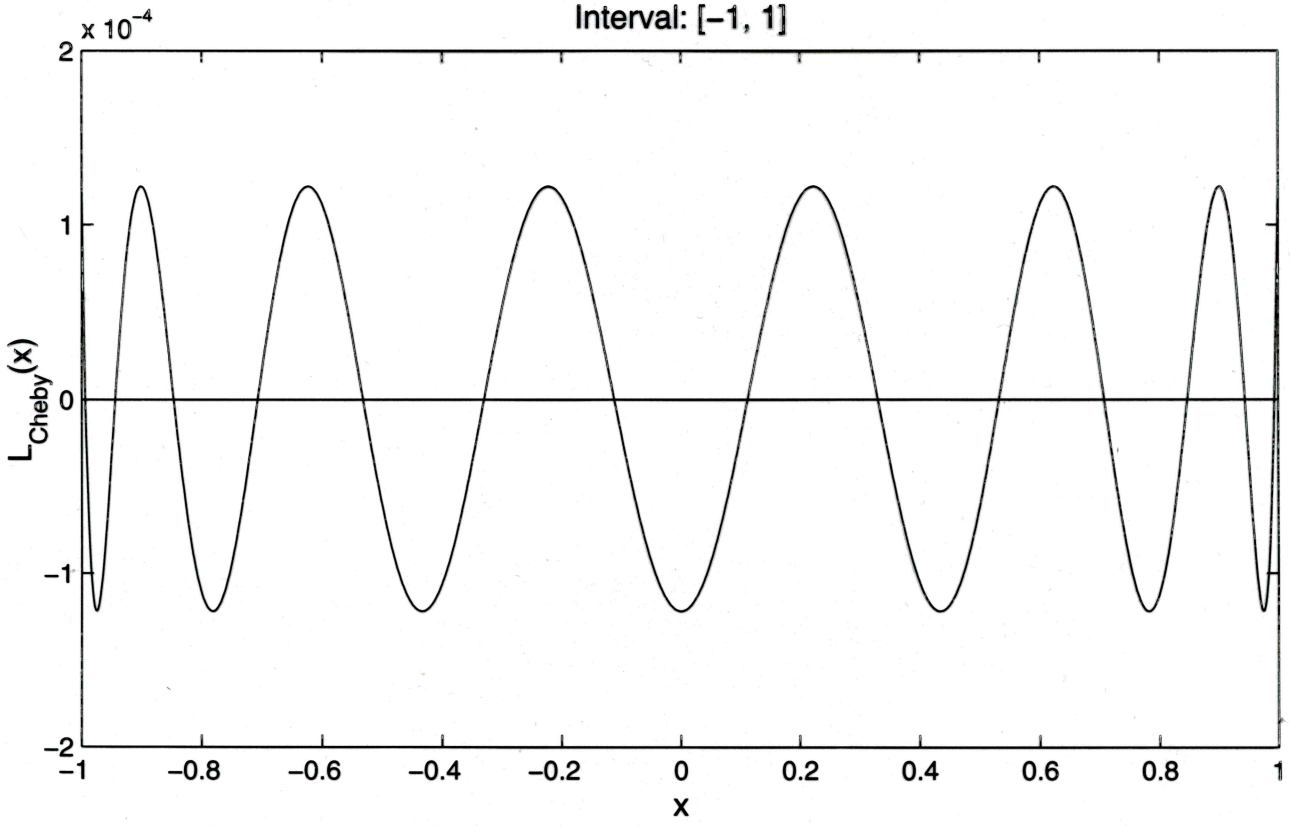
Es liegt an f!
Stützstellen dichter!

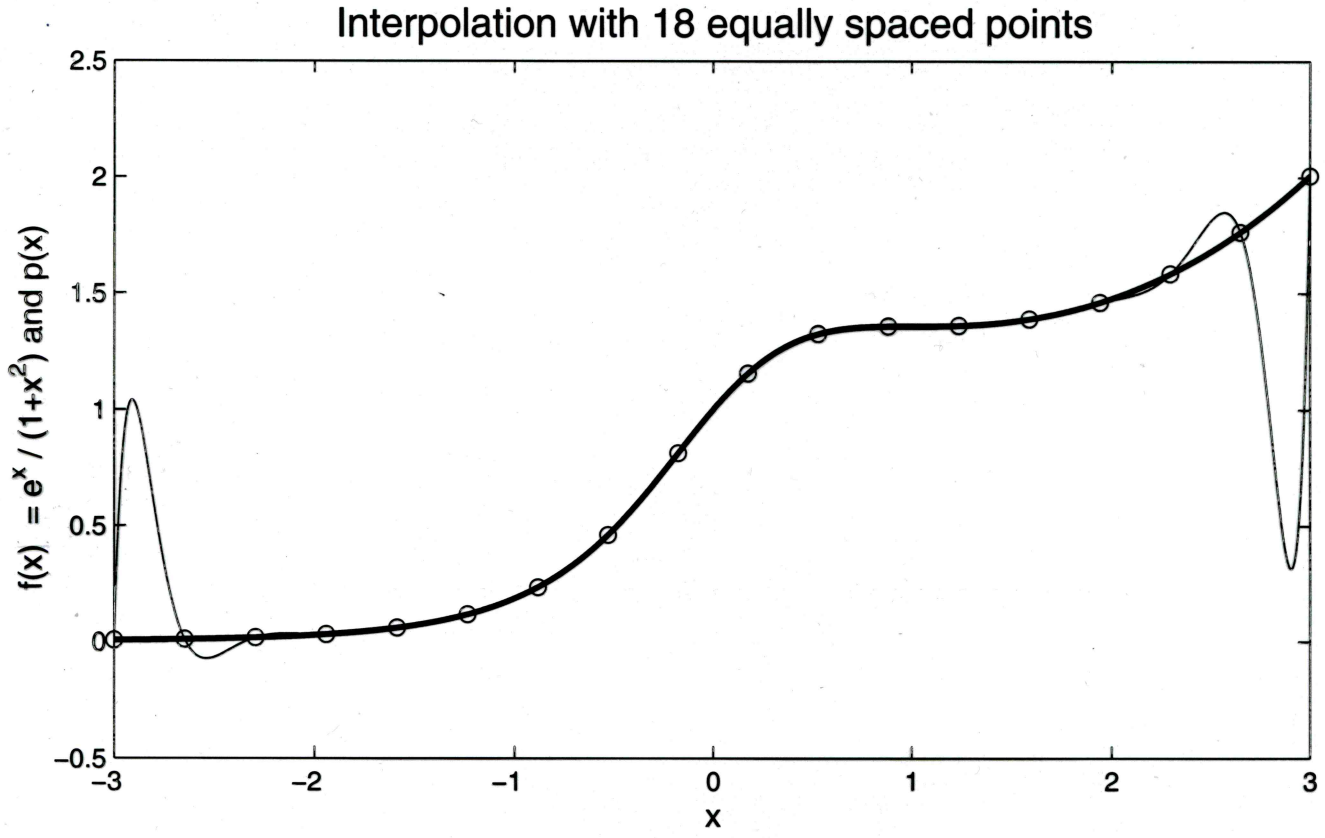


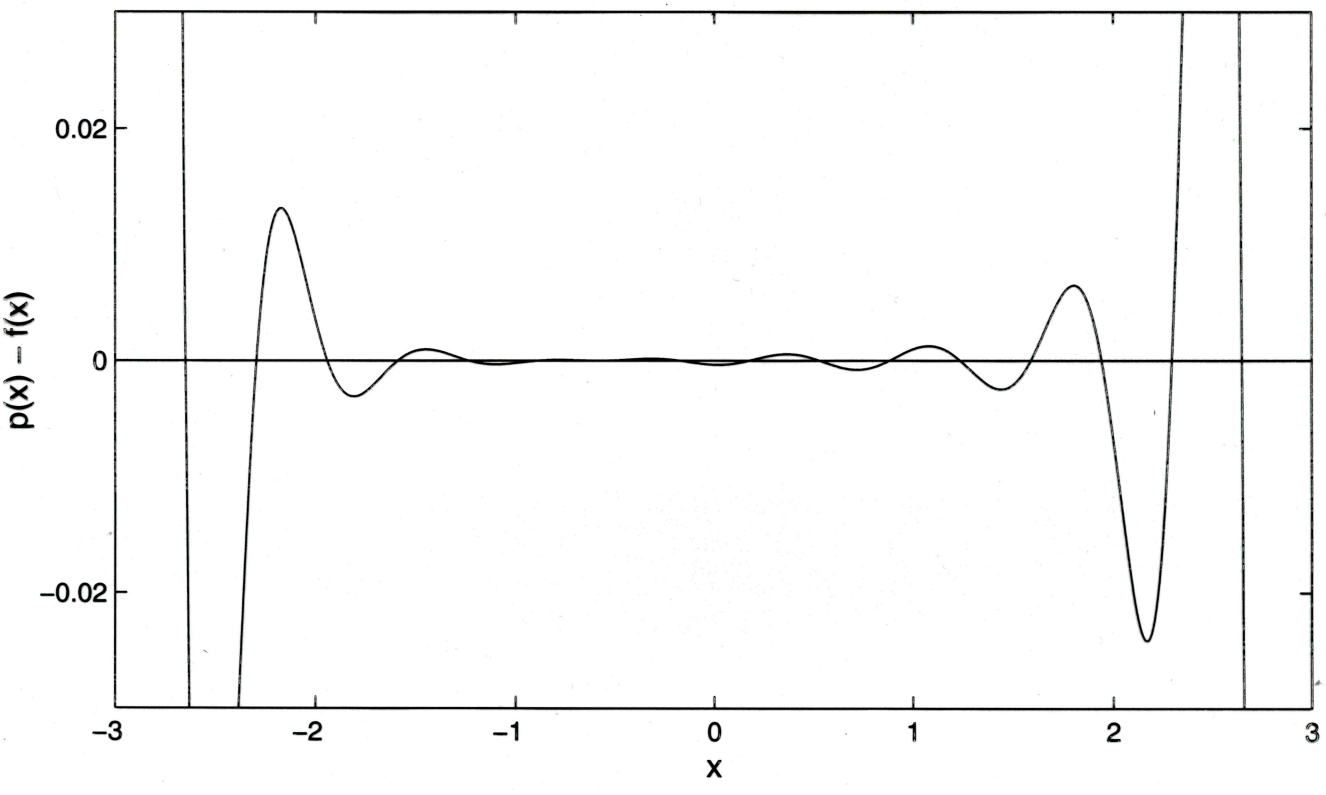
(ii) $|L(x)|$ gross

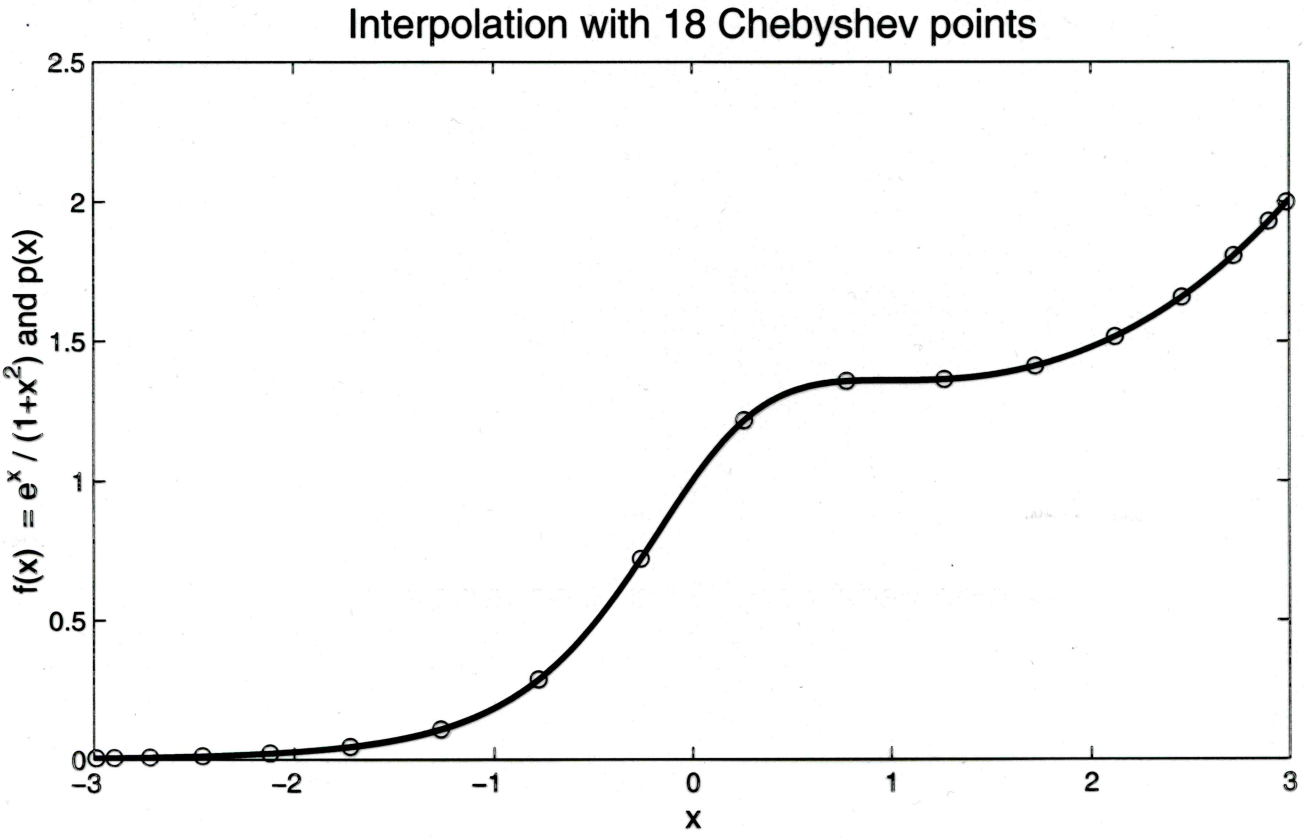
- Viele äquidistante Stützstellen ungünstig!
- Besser: Verdichtung der Stützstellen am Intervallrand, z.B. Tschebyschev-Punkte.
- Siehe Figuren S. 99 – 101

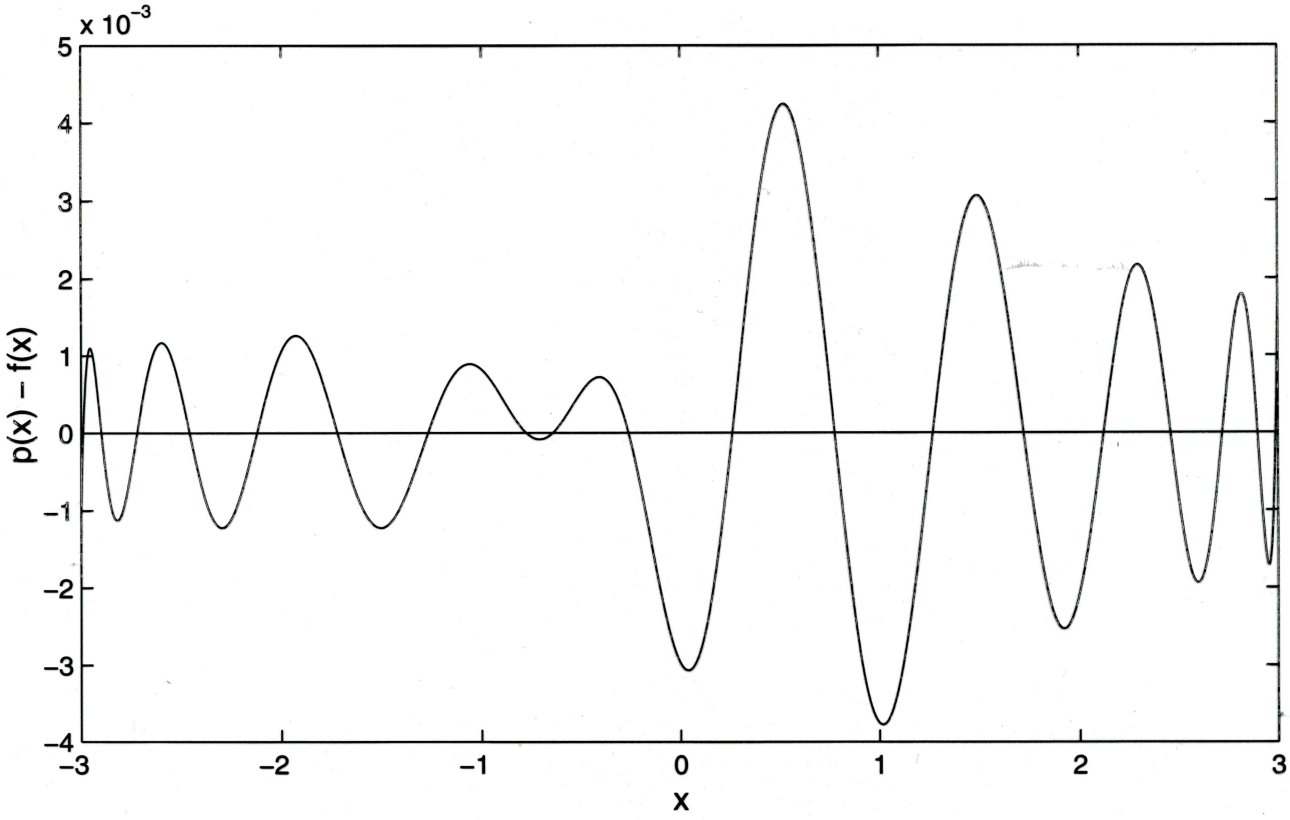

Product of all linear factors (x–x_k) for 14 points
x 10^-3
0.5
0
−0.5
−1
−1.5
−2
−2.5
−3
L_uniform(x)
−1
−0.8
−0.6
−0.4
−0.2
0
0.2
0.4
0.6
0.8
1
x


Interval: [−1, 1]
x 10^-4
2
1
0
−1
−2
L_Cheby(x)
−1
−0.8
−0.6
−0.4
−0.2
0
0.2
0.4
0.6
0.8
1
x


Interpolation with 18 equally spaced points
f(x) = e^x / (1+x^2) and p(x)
x
2.5
2
1.5
1
0.5
0
−0.5
−3
−2
−1
0
1
2
3


p(x) − f(x)
x
0.02
0
−0.02
−3
−2
−1
0
1
2
3

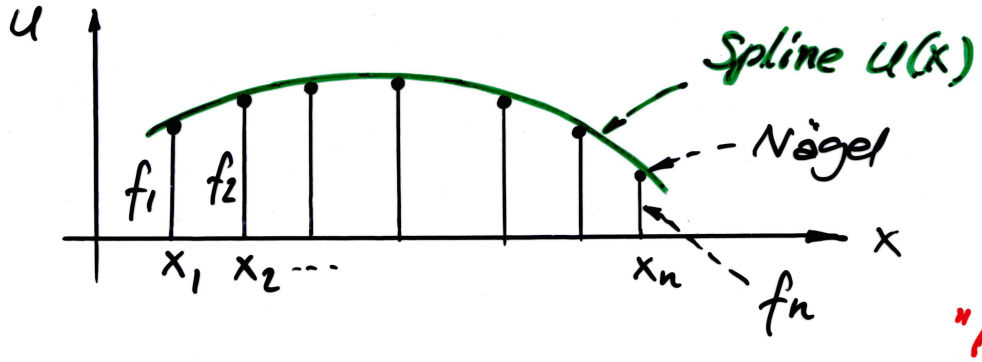

Interpolation with 18 Chebyshev points
f(x) = e^x / (1+x^2) and p(x)
x
2.5
2
1.5
1
0.5
0
−3
−2
−1
0
1
2
3


x 10^-3
p(x) − f(x)
x
5
4
3
2
1
0
−1
−2
−3
−4
−3
−2
−1
0
1
2
3

6.5. Kubische Splines

Neuer Ansatz für die interpolierenden Funktionen: Idee aus dem Schiffbau

Engl. "spline" = biegsame Holzlatte



x_i: "Knoten"

Gesucht: $u = u(x)$, so dass

(i) u 2 mal stetig differenzierbar, d.h. $u''(x)$ stetig, stetige Krümmung

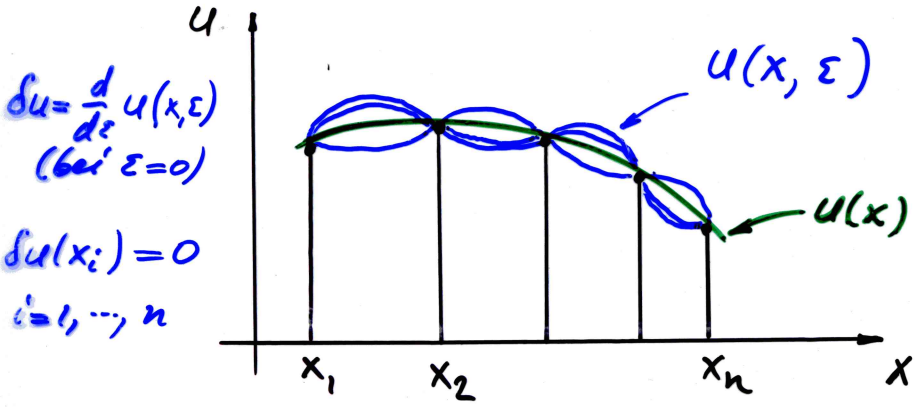
(ii) $u(x_i) = f_i$, $i = 1, 2, \cdots, n$

(iii') Elastische Deformationsenergie = min:

$$E(u) := \frac{1}{2}\int_{x_1}^{x_n} [u''(x)]^2\, dx \stackrel{!}{=} \min$$

Sorgt für möglichst kleine Auslenkungen zwischen den x_i

Lösung mit Variationsrechnung:



Konkurrenzschar:

$u(x, \varepsilon)$, $u(x, 0) = u(x)$

$u(x_i, \varepsilon) = f_i$

fixiert an den x_i

$$\delta E = \int_{x_1}^{x_n} u''(x) \cdot \big(\delta u(x)\big)'' \, dx$$

$$= u''(\delta u)'\Big|_{x_1}^{x_n} - \int_{x_1}^{x_n} u'''(x) \cdot (\delta u(x))' \, dx$$

$$= \underbrace{u''(\delta u)'\Big|_{x_1}^{x_n}}_{=0} - \underbrace{u'''\,\delta u\Big|_{x_1}^{x_n}}_{=0} + \underbrace{\int_{x_1}^{x_n} u^{(4)} \cdot \delta u \cdot dx}_{=0} = 0$$

Lemma der Var'rechn'g ⇒ jeder Term für sich = 0 !

⇓

natürliche Randbed.

$$\begin{cases} u''(x_1) = 0, \\ u''(x_n) = 0, \\ u'' \text{ stetig in } x_2, x_3, \dots, x_{n-1} \end{cases}$$

automatisch erfüllt wegen $\delta u(x_i) = 0$, $i = 1, \dots, n$

⇓

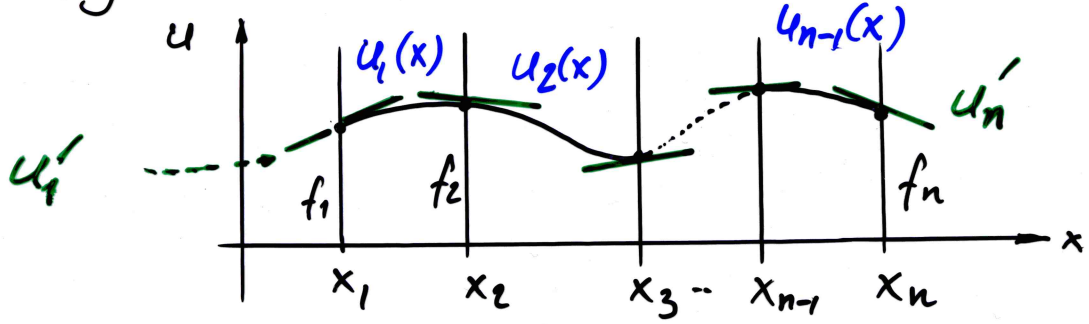
$$u^{(4)}(x) = 0 \quad \forall\, x \neq x_i$$

⇓

u ist stückweise kubisches Polynom

Zusammengefasst:

Algebraisches Problem:



$u_i(x)$: kubische Polynome, $i = 1, 2, \dots, n-1$

n Unbekannte : u_i', $i = 1, 2, \dots, n$

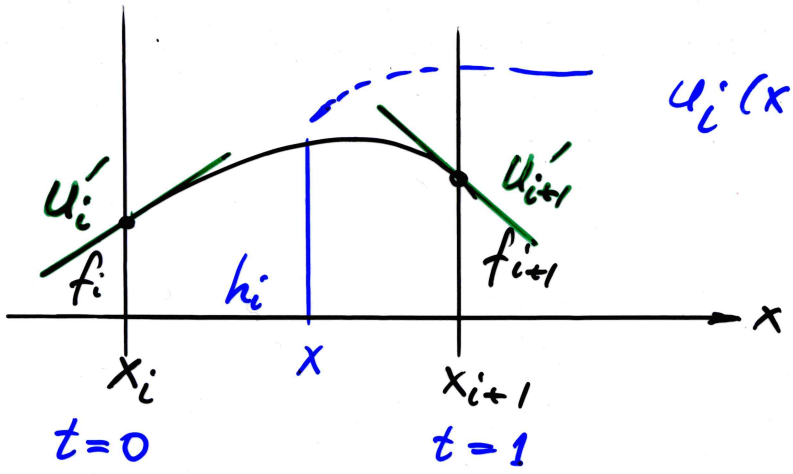
n Bedingungen : $u_1''(x_1) = 0$

$u_i''(x_{i+1}) = u_{i+1}''(x_{i+1})$, $i = 1, \dots, n-2$

$u_{n-1}''(x_n) = 0$

(1)

Hermite - Interpolation



$u_i(x) = Q_i(t)$

Lokale Variable: $t := \dfrac{x - x_i}{h_i}$, $0 \le t \le 1$

mit $h_i := x_{i+1} - x_i$, $i = 1, \dots, n-1$

Differentiationsregel: $\dfrac{d}{dt} = h_i \dfrac{d}{dx}$

oder $\dot{(\,)} = h_i (\,)'$

Bedingungen:

(2)
$$Q_i(0) = f_i \qquad Q_i(1) = f_{i+1}$$
$$\dot{Q}_i(0) = h_i u_i' \qquad \dot{Q}_i(1) = h_i u_{i+1}'$$

Behauptung

Das Hermite-Interpolationspolynom $u_i(x)$ schreibt sich in der lokalen Variablen t als

(3)
$$Q_i(t) = a_0 + \big(b_0 + (c_0 + d_0 t)(t-1)\big)\, t,$$

wobei die Koeffizienten a_0, b_0, c_0, d_0 dem folgenden Differenzenschema entstammen:

$a_{-1} = f_i$

$a_0 = f_i$

$a_1 = f_{i+1}$

$a_2 = f_{i+1}$

$b_{-1} = h_i u_i'$

$b_0 = f_{i+1} - f_i$

$b_1 = h_i u_{i+1}'$

$c_0 = b_0 - b_{-1}$

$c_1 = b_1 - b_0$

$d_0 = c_1 - c_0$

Bemerkung: Sehr effizienter Auswertungsalgorithmus: Alg. von Casteljau

Beweis: Bedingungen (2) nachrechnen!

Zur Formulierung der Bedingungen (1) brauchen wir die zweiten Ableitungen:

$$u_i''(x) = \frac{1}{h_i^2} \ddot{Q}_i(t)$$

(3) $\Rightarrow$

$$\ddot{Q}_i(t) = 2c_0 + d_0(6t-2)$$

$$\dddot{Q}_i(t) = 6d_0$$

$$\ddot{Q}_i(0) = 4c_0 - 2c_1 = 6b_0 - 4b_{-1} - 2b_1$$

$$\ddot{Q}_i(1) = -2c_0 + 4c_1 = -(6b_0 - 2b_{-1} - 4b_1)$$

Bedingungen (1) $\Rightarrow$

$$\frac{\ddot{Q}_i(1)}{h_i^2} = \frac{\ddot{Q}_{i+1}(0)}{h_{i+1}^2}, \qquad i = 1, \dots, n-2$$

Dazu für natürliche Splines (aus Variationsproblem: Wendepunkte am Rand):

$$\ddot{Q}_1(0) = 0, \qquad \ddot{Q}_{n-1}(1) = 0.$$

Gleichungssystem für die Unbekannten $u_1', u_2', \dots, u_n'$:

$$-\frac{1}{2h_1^2}\ddot{Q}_1(0) = -3h_1^{-2}(f_2 - f_1) + 2h_1^{-1}u_1' + h_1^{-1}u_2' = 0$$

$$-3h_i^{-2}(f_{i+1} - f_i) + h_i^{-1}u_i' + 2h_i^{-1}u_{i+1}' =$$

(4) $i = 2, \dots, n-2$

$$= 3h_{i+1}^{-2}(f_{i+2} - f_{i+1}) - 2h_{i+1}^{-1}u_{i+1}' - h_{i+1}^{-1}u_{i+2}'$$

$$\frac{1}{2h_{n-1}^2}\ddot{Q}_{n-1}(1) = -3h_{n-1}^{-2}(f_n - f_{n-1}) + 2h_{n-1}^{-1}u_n' + h_{n-1}^{-1}u_{n-1}' = 0$$

Abkürzungen:

(5)
$$r_i := \frac{1}{h_i}, \quad i = 1, 2, \dots, n-1$$
$$c_i := \frac{1}{h_i^2}(f_{i+1} - f_i), \quad i = 1, 2, \dots, n-1$$

Damit lautet das lineare Gleichungssystem (4):

(6) $$A \cdot \underline{u}' = \underline{b}, \quad \underline{u}' = (u_1', \dots, u_n')^T \in \mathbb{R}^n$$

mit

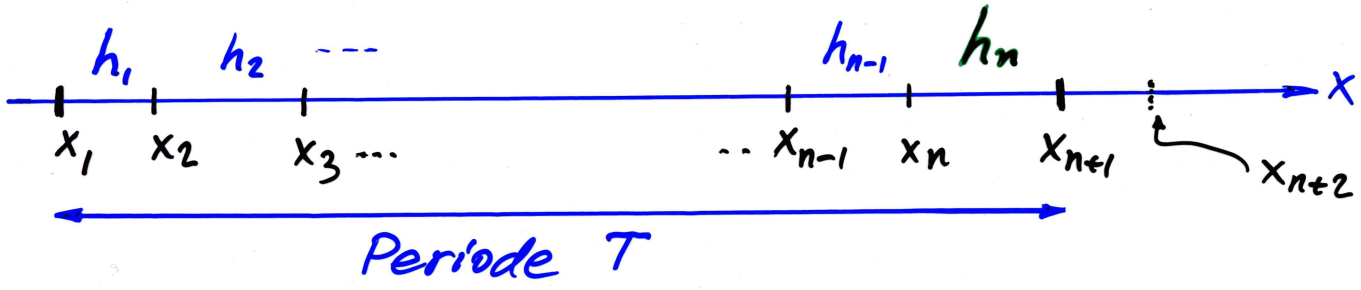
(7)
$$A = \begin{bmatrix} 2r_1 & r_1 & & & 0 \\ r_1 & 2(r_1+r_2) & r_2 & & \\ & r_2 & 2(r_2+r_3) & r_3 & \\ & & \ddots & \ddots & \ddots \\ & & r_{n-2} & 2(r_{n-2}+r_{n-1}) & r_{n-1} \\ 0 & & & r_{n-1} & 2r_{n-1} \end{bmatrix}, \quad \underline{b} = 3\begin{bmatrix} c_1 \\ c_1 + c_2 \\ c_2 + c_3 \\ \vdots \\ c_{n-2} + c_{n-1} \\ c_{n-1} \end{bmatrix}$$

für natürliche Splines

Lösung von (6): LR-Zerlegung mit Diagonal-Pivots
Aufwand: $\mathcal{O}(n)$, billig!

Zwei wichtige Varianten:

(i) Periodische Splines



Periodizität: $x_{n+i} = x_i + T, \quad i = 1, 2, \ldots$

$f_{n+i} = f_i$

Definiere Abkürzungen (5) für $i = 1, 2, \ldots, n$

Gleichungssystem (6) wie oben, aber mit

$$A = \begin{bmatrix} 2(r_n+r_1) & r_1 & & & & r_n \\ r_1 & 2(r_1+r_2) & r_2 & & 0 & \\ & r_2 & 2(r_2+r_3) & r_3 & & \\ & & \ddots & \ddots & \ddots & \\ & 0 & & r_{n-2} & 2(r_{n-2}+r_{n-1}) & r_{n-1} \\ r_n & & & & r_{n-1} & 2(r_{n-1}+r_n) \end{bmatrix}, \quad \underline{b} = 3 \begin{bmatrix} c_n + c_1 \\ c_1 + c_2 \\ c_2 + c_3 \\ \vdots \\ c_{n-2} + c_{n-1} \\ c_{n-1} + c_n \end{bmatrix}$$

Matrix A ist tridiagonal und "zyklisch"

Lösungsaufwand: ebenfalls $O(n)$.

(ii) de Boor - Splines

Ersetze die natürlichen Randbed. $\ddot{Q}_1(0) = 0$, $\ddot{Q}_{n-1}(1) = 0$ durch de Boors

"not a knot condition": (nicht aus Variationsrechnung ableitbar)

u''' stetig in $x = x_2$, $x = x_{n-1}$

Geometrisch: Spline im ersten und zweiten Teilintervall ist dasselbe Polynom
Ebenso im letzten und zweitletzten Teilintervall.

Formelmässig:

$$h_1^{-3}\dddot{Q}_1(1) = h_2^{-3}\dddot{Q}_2(0); \quad h_{n-2}^{-3}\dddot{Q}_{n-2}(1) = h_{n-1}^{-3}\dddot{Q}_{n-1}(0)$$

Modifiziere erste und letzte Gleichung von (7):

$$A = \begin{array}{|l|}\hline r_1 \quad r_1+r_2 \\ \hline \text{Zeilen \# 2 bis } n-1 \\ \text{von (7)} \\ \hline r_{n-2}+r_{n-1} \quad r_{n-1} \\ \hline \end{array}, \quad \underline{b} = \begin{array}{|l|}\hline 2c_1+(c_1+c_2)h_1/(h_1+h_2) \\ \hline \text{Zeilen \# 2 bis } n-1 \\ \text{von (7)} \\ \hline 2c_{n-1}+(c_{n-2}+c_{n-1})h_{n-1}/(h_{n-2}+h_{n-1}) \\ \hline \end{array}$$

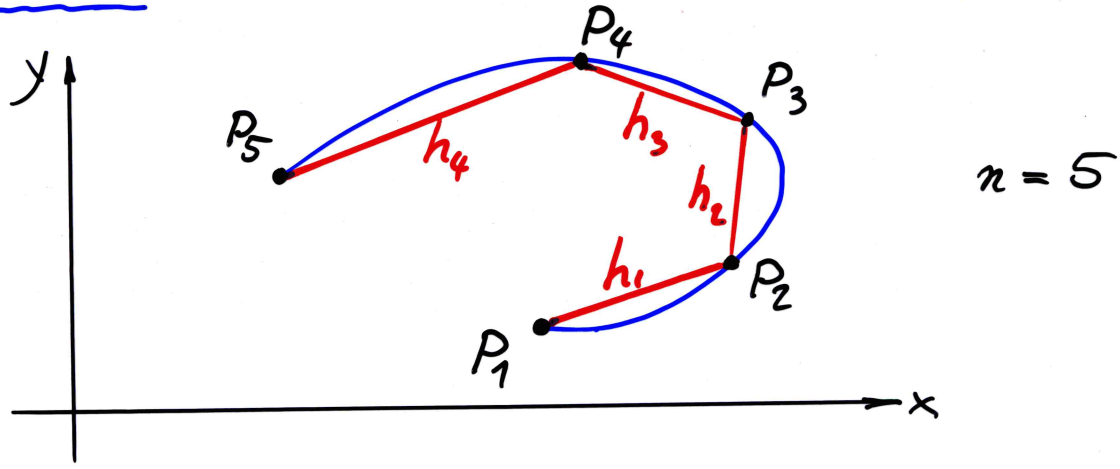
Bemerkung:

- Natürliche Splines verwenden, falls sich nat. RB (Wendepunkte am Rand) aufdrängen.
- Periodische Splines vor allem für geschlossene Kurven (siehe unten).
- de Boor-Splines ergeben i.a. etwas "schönere" Kurven als natürliche Splines.

Parametrische Spline-Kurven

Gegeben: Punktefolge $P_1, P_2, \dots, P_n$ der Ebene

Gesucht: "Schöne" Kurve durch $P_1, \dots, P_n$



Bemerkung: y ist i.a. **keine eindeutige** Funktion von x !

Idee: Suche Parameterdarstellung

$$x = f(t), \quad y = g(t),$$

mit einem geeigneten Kurvenparameter t.

Einfache, gut funktionierende Wahl:

t = Bogenlänge auf dem die Punkte $P_1, \dots, P_n$ verbindenden Polygonzug:

Sei $h_i := \text{dist}(P_i, P_{i+1})$, $i = 1, \dots, n-1$

Spline-Knoten in der Variablen t:

$$t_i = \sum_{j=1}^{i-1} h_j$$