Wiederholung

Allgemeine Idee:
Approximiere die zu integrierende Funktion über Polynome. Diese lassen sich exakt integrieren.

$$\int f = \sum_{i=0}^{n} w_i f(x_i) + R_n \tag{1}$$

Integration mit n Stützstellen

- äquisdistant $\rightarrow$ Newton-Coates (exakt für Polynome vom Grad n-1) Beispiel Simpson, exakt bis Ordnung 2
- frei wählbar $\rightarrow$ Gauss (exakt für Polynome vom Grad 2n -1)

Gauss Stützstellen sind die Nullstellen der orthogonalen Polynome, für Legendre

$$L_n(x_i) = 0 \tag{2}$$

Ohne Herleitung:
Für die Integrationsgewichte findet man

$$w_i = \frac{2}{(1 - x_i^2)(L_n'(x_i))^2} \tag{3}$$

Restglied:

$$R_n = \frac{2^{2n+1}(n!)^4}{(2n+1)\left((2n)!\right)^3} f^{(2n)}(\xi) \tag{4}$$

Gaußfür beliebiges Intervall

$$\int_a^b f(x)dx = \int_{-1}^{1} f(h(x))h'(x)dx = \sum_{i=1}^{N} f(\tilde{x}_i)\tilde{w}_i + R_N \tag{5}$$

wobei

$$\tilde{x}_i = h(x_i) \quad \text{und} \quad \tilde{w}_i = h'(x_i)w_i \tag{6}$$

$$h(-1) = a \ , \ h(+1) = b \tag{7}$$

Linear Abb.

$$\tilde{x}_i = h(x_i) = \frac{b-a}{2}x_i + \frac{b+a}{2} \tag{8}$$

$$\tilde{w}_i = h'(x_i)w_i = \frac{b-a}{2}w_i \tag{9}$$

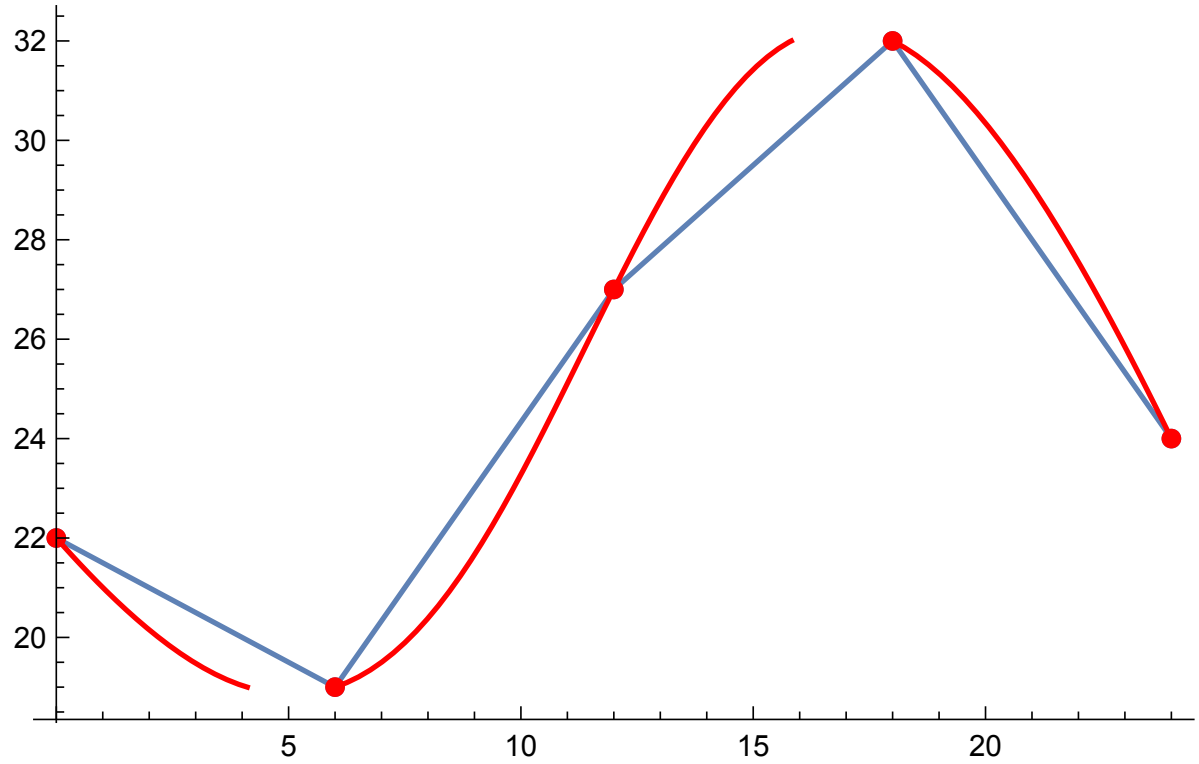
Interpolation

Zur Integration haben wir die Interpolation beliebiger Funktionen über Polynome genutzt.
Weiteres Beispiel:
Temperaturverlauf an einem Sommertag in Mainz

Uhr	0	6	12	18	24
T	22	19	27	32	24

Bei Interpolation werden Stützpunkte durch eine verbindende Kurve exakt beschrieben. Das ist nur dann sinnvoll, wenn die Daten-Punkte praktich fehlerfrei sind.

Lineare Interpolation



Für $t \in [t_i, t_{i+1}]$ mit $T_i = T(t_i)$ gilt

$$T(t) = T_i + \frac{t - t_i}{t_{i+1} - t_i}(T_{i+1} - T_i) \tag{10}$$

Polynominterpolation

Satz

Zu $n+1$ verschiedenen Stützstellen $x_0, \ldots, x_n$ mit den Stützwerten $f_0, \ldots, f_n$ gibt es genau ein Polynom $p_n(x)$ höchstens vom Grad n, für das

$$p_n(x_i) = f_i, \quad i = 0, 1, \ldots, n \tag{11}$$

Beweis

Angenommen $P_n(x)$, $Q_n(x)$ seien Polynome vom Grad n und an den $n+1$ Stützstellen gleich

$$D(x) = P_n(x) - Q_n(x) \tag{12}$$

ist auch ein Polynom vom Grad höchstens n, hat aber $n+1$ Nullstellen.

$$\Rightarrow D(x) = 0$$

Ansatz:

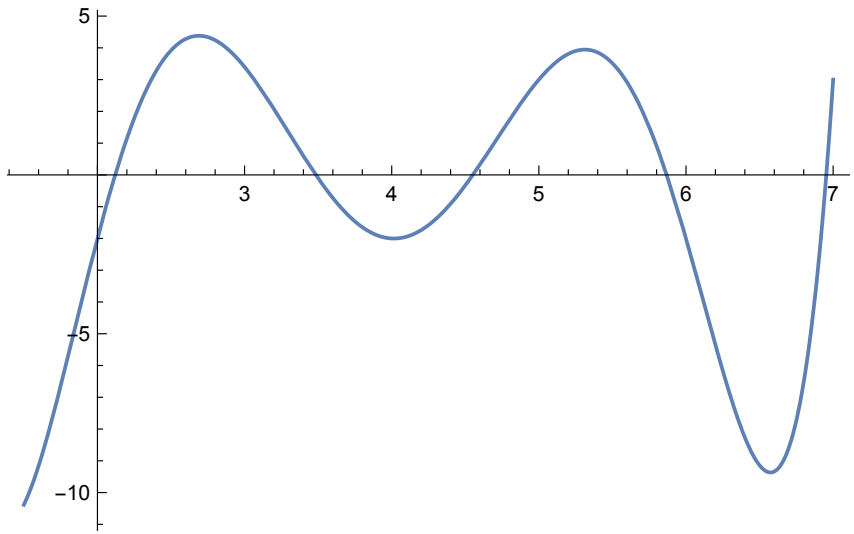
$$p_n(x) = c_0 + c_1 x + \cdots + c_n x^n \tag{13}$$

$$\begin{pmatrix} 1 & x_0 & \dots & x_0^n \\ \vdots & & & \\ 1 & x_n & \dots & x_n^n \end{pmatrix} \begin{pmatrix} c_0 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} f_0 \\ \vdots \\ f_n \end{pmatrix} \tag{14}$$

Invertieren dieser Matrix recht aufwendig, einfaches Verfahren über Lagrange-Interpolation (siehe Newtin-Coates Integration)
Anstelle der Basis-Polynome $1, x, x^2, \ldots x^n$ werden die Basispolynome $l_i^{(n)}(x)$ verwendet mit der Eigenschaft

$$l_i^{(n)}(x_k) = \delta_{ik} \tag{15}$$

d.h. $l_i(x) = 1$ am Stützpunkt x_i, an allen anderen Stellen 0.



Die $l_i(x)$ sind eindeutig bestimmt, außerdem lassen sie sich leicht konstruieren (n=2)

$$l_0^{(2)}(x) = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} \tag{16}$$

$$l_1^{(2)}(x) = \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} \tag{17}$$

$$l_2^{(2)}(x) = \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} \tag{18}$$

allgemein (19)

$$l_j(x) = \prod_{\substack{i=0 \\ j \neq i}}^{n} \frac{x - x_i}{x_j - x_i} \quad \text{, where } 0 \leq j \leq n \tag{20}$$

Damit lassen sich die Interpolationspolynome einfach schreiben als

$$f(x) \approx p_n(x) = \sum_{i=0}^{n} f_i l_i^{(n)}(x) \tag{21}$$

Einfach zu zeigen durch Einsetzen der Stützstellen.
Fehlerbetrachtung

$$f(x) = p_n(x) + R_n(x) \tag{22}$$

$$R_n(x) = (x - x_0)(x - x_1)\dots(x - x_n)\frac{f^{(n+1)}(\xi)}{(n+1)!}, \quad \xi \in [x_0, x_n] \tag{23}$$

Man sieht $R_n = 0$ für

- alle x_i
- für Polynome vom Grad $\leq n$

Achtung:
Bei hohen Graden $n \geq 5$ neigt die Polynominterpolation zu starken Oszill. zwischen den Stützstellen, insbesondere an den Rändern.

Fehler der Polynominterpolation:

$$R_n(x) = \prod_i (x - x_i) \frac{f^{(n+1)}(\xi)}{(n+1)!} \tag{24}$$

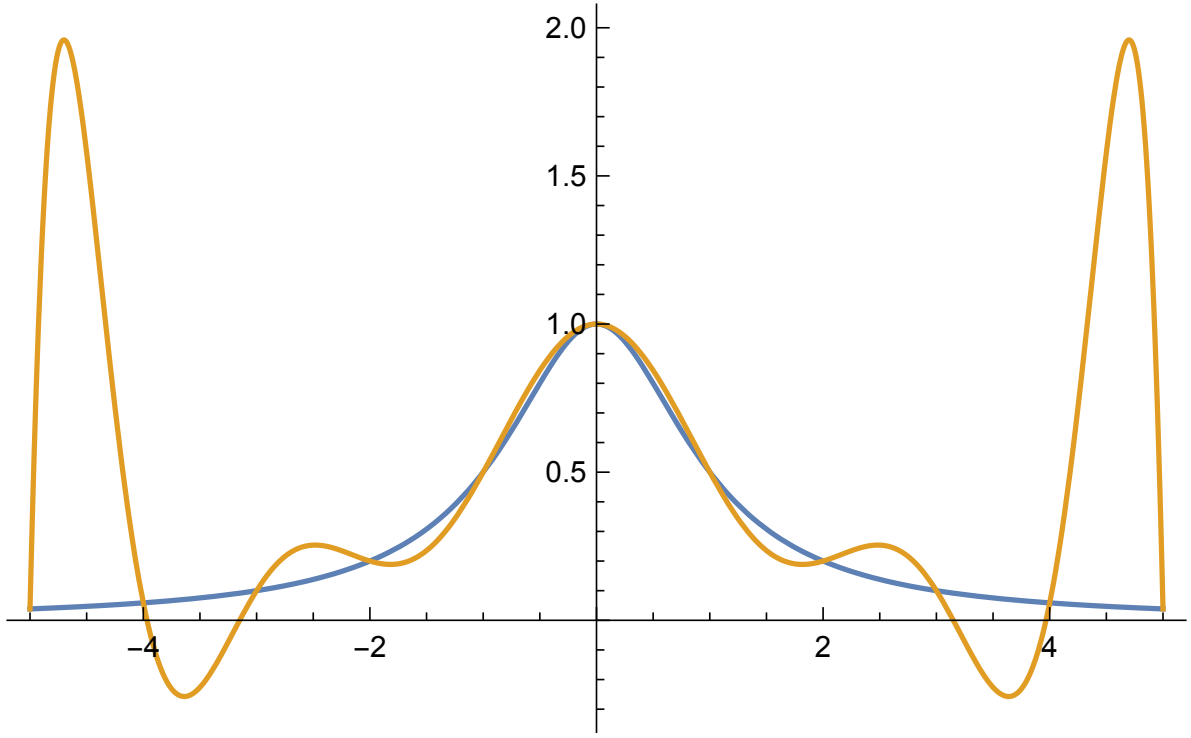
Seien die Stützstellen äquidistant $x_{i+1} - x_i = h$ dann

$$\left| \prod (x - x_i) \right| \leq \frac{h^{n+1} n!}{4} \tag{25}$$

und damit

$$\max_{x \in I} |R_n| \leq \frac{h^{n+1}}{4(n+1)} \max_{x \in I} |f^{(n+1)}(x)| \tag{26}$$

Kann divergieren!



Polynom vom Grad 10

Ausweg Splines

Wortherkunft:

Der Begriff stammt aus dem Schiffbau: eine lange dnne Latte (Straklatte, englisch spline), die an einzelnen Punkten durch Molche fixiert wird, biegt sich genau wie ein kubischer Spline mit natrlicher Randbedingung. Dabei wird die Spannungsenergie

minimal $E(S) = \int_a^b \frac{S''(x)^2}{(1+S'(x)^2)^{5/2}} dx$

Gegeben seien n Datenpunkte $x_i, y_i, i = 1, \ldots, n$ im wesentlichen fehlerfrei.

Methode:

stückweise stetige und glatte Polynome vom Grad $n = 3$, wobei das Ziel ist die Krümmung zu minimieren.

Stückweise stetige Polynome:
Anstelle eines Poly. mit hohem Grad $(n - 1)$ durch alle Punkte, legt man viele Polynome mit niedriegem Grad $n = 3$ durch jeweils zwei benachbarte Punkte, und stückelt die einzelnen Poly. glatt (stetig und diffbar) an.

Ansatz für das j-te Intervall

$$I_j = (x_j, x_{j+1}] \quad \text{mit} \quad h_j = x_{j+1} - x_j \ , \quad j = 1, \ldots, n+1 \tag{27}$$

Für $x \in I_j$

$$S_j(x) = a_j(x - x_j)^3 + b_j(x - x_j)^2 + c_j(x - x_j) + d_j \tag{28}$$

Mit der stückweise definierten Spline Funktion:

$$S(x) = \begin{cases} S_1(x), x \in I_1 \\ \vdots \\ S_{n-1}(x), x \in I_{n-1} \end{cases} \tag{29}$$

Führe Stetigkeitsbedingungen ein:

$$S_j(x_{j+1}) = S_{j+1}(x_{j+1}) \tag{30}$$
$$S'_j(x_{j+1}) = S'_{j+1}(x_{j+1}) \tag{31}$$
$$S''_j(x_{j+1}) = S''_{j+1}(x_{j+1}) \tag{32}$$
$$\tag{33}$$

S_j muss durch die Endpunkte gehen

$$S_j(x_j) = d_j = y_j \tag{34}$$
$$S_j(x_{j+1}) = a_j h_j^3 + b_j h_j^2 + c_j h_j + d_j = y_{j+1} \tag{35}$$

Die zweite Ableitung bekommt an den Endpunkten die Werte S_j

$$S''_j(x_j) = 2b_j = y''_j \tag{36}$$
$$S''_j(x_{j+1}) = 6a_j h_j + 2b_j = y''_{j+1} \tag{37}$$

Mit diesen Bedingunsggleichungen können die Polynome über die y_j und y_j'' geschrieben werden

$$b_j = \frac{y_j''}{2}, a_j = \frac{y_{j+1}'' - y_j''}{6h_j} \tag{38}$$

$$d_j = y_j, c_j = \frac{y_{j+1} - y_j}{h_j} - \frac{h_j}{6}\left(y_{j+1}'' + 2y_j''\right) \tag{39}$$

Einsetzen in den Ansatz für S_j ergibt

$$S_j(x) = \frac{y_{j+1}'' - y_j''}{6h_j}(x - x_j)^3 + \frac{y_j''}{2}(x - x_j)^2 + \tag{40}$$

$$+ \left(\frac{y_{j+1} - y_j}{h_j} - \frac{h_j}{6}(2y_j'' + y_{j+1}'')\right)(x - x_j) + y_j \tag{41}$$

Die Koeffizienten y_j'' legt man mit der Stet. der 1. Ableitung fest

$$S_j'(x_j) = S_{j-1}'(x_j) \tag{42}$$

$$\Rightarrow c_j = 3a_{j-1}h_{j-1}^2 + 2b_{j-1}h_{j-1} + c_{j-1} \tag{43}$$

$$\Rightarrow \frac{y_{j+1} - y_j}{h_j} - \frac{h_j}{6}(y_{j+1}'' + 2y_j'') = 3\frac{y_j'' - y_{j-1}''}{6h_{j-1}}h_{j-1}^2 \tag{44}$$

$$+ 2\frac{y_{j-1}''}{2}h_{j-1} + \frac{y_j - y_{j-1}}{h_{j-1}} - \frac{h_{j-1}}{6}(y_j'' + 2y_j'' \tag{45}$$

Hieraus folgt für $j = 1, \ldots, n-1$

$$h_{j-1}y_{j-1}'' + 2(h_j + h_{j-1})y_j'' + h_j y_{j+1}'' = 6\left(\frac{y_{j+1} - y_j}{h_j} - \frac{y_j - y_{j-1}}{h_{j-1}}\right) \tag{46}$$

Die Stetigkeitsbedingung gilt für alle Zwischenpunkte, d.h. $x_1, \ldots, x_{n-1}$. Es fehlt eine Bestimmungsgleichung für y_0'' und y_n''.
$\rightarrow$ An den beiden Randpunkten benötigt man zusätzliche Randbedingungen!

- Natürliche Randbedingun $y_0'' = y_n'' = 0 \Rightarrow$ Linearität an den Rändern.
- Vollständige Randbedingungen $s_0' = c_0$ und $s_n' = d$
- Periodische Randbedingung $y_0 = y_n, y_0' = y_n', y_0'' = y_n''$

$$h_{j-1}y''_{j-1} + 2(h_j + h_{j-1})y''_j + h_j y''_{j+1} = 6\left(\frac{y_{j+1} - y_j}{h_j} - \frac{y_j - y_{j-1}}{h_{j-1}}\right) \tag{47}$$

Für natürliche Splines Darstellung als Matrix

$$\begin{pmatrix} p_1 & h_1 & 0 & \dots & 0 & 0 \\ h_1 & p_2 & h_2 & \dots & 0 & 0 \\ 0 & h_2 & p_3 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & p_{n-2} & h_{n-2} \\ 0 & 0 & 0 & \dots & h_{n-2} & p_{n-1} \end{pmatrix} \begin{pmatrix} y''_1 \\ y''_2 \\ y''_3 \\ \vdots \\ y''_{n-2} \\ y''_{n-1} \end{pmatrix} = \begin{pmatrix} q_1 \\ q_2 \\ q_3 \\ \vdots \\ q_{n-2} \\ q_{n-1} \end{pmatrix} \tag{48}$$

mit

$$p_i = 2(h_{i-1} + h_i) \tag{49}$$

$$q_i = 6\left(\frac{y_{j+1} - y_j}{h_j} - \frac{y_j - y_{j-1}}{h_{j-1}}\right) \tag{50}$$

Lösung für Tridiagonalmatrix über Thomasalgorithmus

Idee: Wende Gauß-Verfahren an, hieraus ergibt sich eine Matrix mit Diagonal und oberem Nebenelement.
Ansatz:

$$y''_{j-1} = \alpha_j y''_j + \beta_j \tag{51}$$

Einsetzen in Dreipunktrekursion

$$h_{j-1}y''_{j-1} + 2(h_j + h_{j-1})y''_j + h_j y''_{j+1} = h_{j-1}\left(\alpha_j y''_j + \beta_j\right) \tag{52}$$

$$+ 2(h_j + h_{j-1})y''_j + h_j y''_{j+1} = q_j \tag{53}$$

Auflösen nach y''_j

$$y''_j = \frac{q_j - h_{j-1}\beta_j}{h_{j-1}\alpha_j + 2(h_j + h_{j+1})} - \frac{h_j}{h_{j-1}\alpha_j + 2(h_j + h_{j+1})} y''_{j+1} \tag{54}$$

$$= \beta_{j+1} + \alpha_{j+1} y''_{j+1} \tag{55}$$

Thomasalgorithmus:

1. Setze Startwerte $\alpha_0 = \beta_0 = 0$
2. Bestimme rekursiv $\alpha_j + 1$ und $\beta + j + 1$
3. Setze $y_n'' = 0$
4. Bestimme rekursiv $y_{j-1}'' = \alpha_j y_j'' + \beta_j$