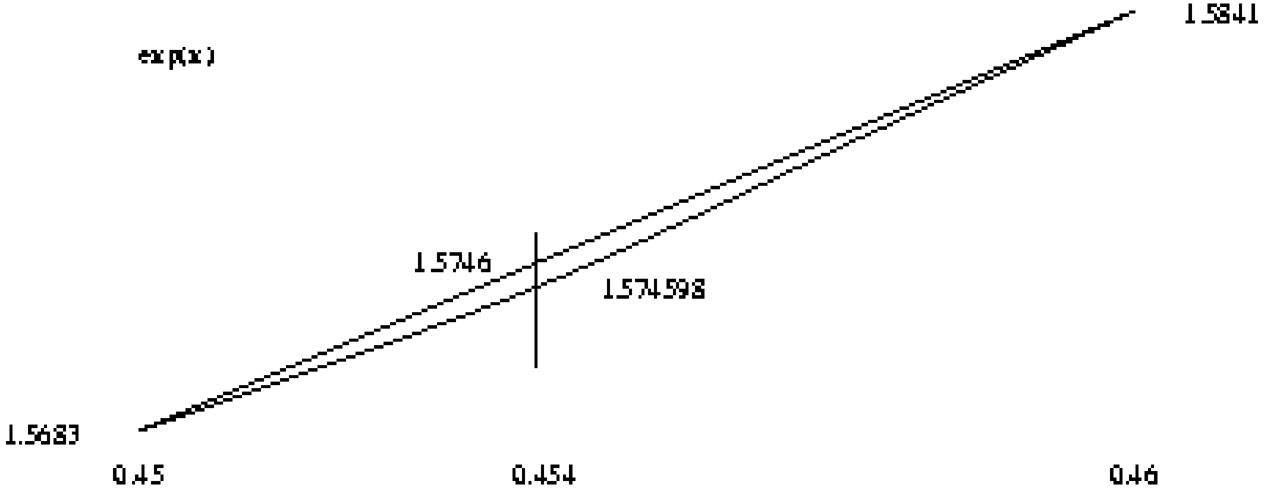
IV. Interpolation und Quadratur

4.1. Interpolation

4.1.1. Beispiel: Interpolation mit Tafelwerken für exp, sin, cos, log
Gesucht: exp(0.454) ; Tabelliert: exp(0.45) und exp(0.46)
Lineare Interpolation.



x:	0.42	0.43	0.44	0.45	0.46
exp(x):	...	...	...	1.5683	1.5841

4.1.2. Allgemeine Problemstellung:

Gegeben: Punktepaare (x_j, y_j), $j=0,1,\ldots,n$, paarweise verschieden
und linear unabhängige Funktionen $g_k(x)$, $k=0,1,\ldots,n$

Gesucht: Koeffizienten c_k , $k=0,1,\ldots,n$ mit

$$G(x_j) = \sum_{k=0}^{n} c_k g_k(x_j) \overset{!}{=} y_j \quad \text{für } j=0,1,\ldots,n$$

$$\begin{pmatrix} g_0(x_0) & \cdots & g_n(x_0) \\ \vdots & & \vdots \\ g_0(x_n) & \cdots & g_n(x_n) \end{pmatrix} \cdot \begin{pmatrix} c_0 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} y_0 \\ \vdots \\ y_n \end{pmatrix}$$

$(n+1)\times(n+1)$ lineares Gleichungssystem

Interpolation führt auf quadratisches Gleichungssystem

Man unterscheide Interpolation und Approximation!

Beispiel zu Approximation:
Ausgleichsgerade führt auf überbestimmtes Gleichungssystem (Normalgleichung, QR)

4.1.3. Spezialfall Polynom-Interpolation:

Ansatzfunktionen $g_k(x)$ sind Polynome x^k

Gesucht: Koeffizienten c_k , k=0,1,...,n mit

$$p(x_j) = \sum_{k=0}^{n} c_k x_j^k \overset{!}{=} y_j \quad \text{für } j=0,1,\ldots,n$$

$$\begin{pmatrix} 1 & x_0 & \cdots & x_0^n \\ \vdots & \vdots & & \vdots \\ 1 & x_n & \cdots & x_n^n \end{pmatrix} \cdot \begin{pmatrix} c_0 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} y_0 \\ \vdots \\ y_n \end{pmatrix}$$

n+1 Gleichungen für n+1 Unbekannte: Teuer! $O(n^3)$

4.1.4. Lösung mit Lagrange-Polynomen

Definiere geschickt Basis-Polynome:

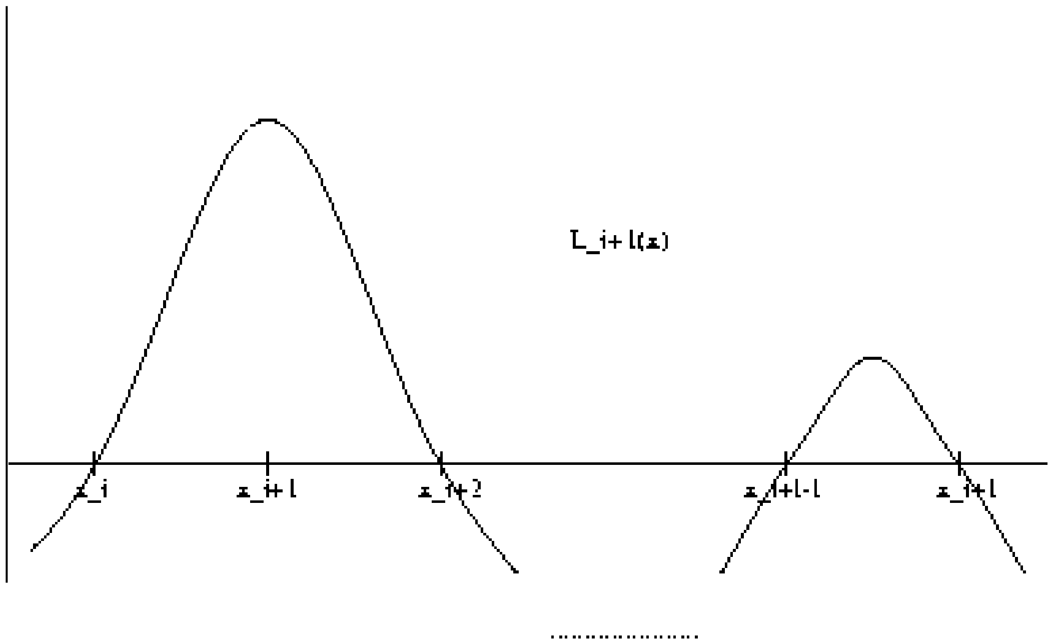
$$L_j(x) := \prod_{i=0, i\neq j}^{n} \frac{x - x_i}{x_j - x_i} = \frac{(x-x_0)\ldots(x-x_{j-1})(x-x_{j+1})\ldots(x-x_n)}{(x_j-x_0)\ldots(x_j-x_{j-1})(x_j-x_{j+1})\ldots(x_j-x_n)}$$

n+1 Polynome vom Grad n, besser als 1, x, x^2, .., x^n

Eigenschaften der Lagrange-Polynome: Grad n mit

$$L_j(x_i) = \begin{cases} 1 & falls \quad i = j \\ 0 & falls \quad i \neq j \end{cases}$$

Gesucht $p(x) = \sum_{j=0}^{n} c_j L_j(x)$, das die Interpolationsbedingungen erfüllt



Aus diesen Eigenschaften ergibt sich zur Lösung des Interpolationsproblems ein lineares Gleichungssystem für die Koeffizienten:

$$\begin{pmatrix} L_0(x_0) & \cdots & L_n(x_0) \\ \vdots & & \vdots \\ L_0(x_n) & \cdots & L_n(x_n) \end{pmatrix} \cdot \begin{pmatrix} c_0 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} 1 & & 0 \\ & \ddots & \\ 0 & & 1 \end{pmatrix} \cdot \begin{pmatrix} c_0 \\ \vdots \\ c_n \end{pmatrix} = \begin{pmatrix} y_0 \\ \vdots \\ y_n \end{pmatrix}$$

Die Lösung ist $c_j = y_j$; daher ist das Interpolationspolynom:

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

denn es ist $p(x_i) = \sum_{j=0}^{n} y_j L_j(x_i) = y_i L_i(x_i) = y_i$.

Damit ist die Existenz eines interpolierenden Polynoms gezeigt!

Eindeutigkeit?

Hauptsatz der Algebra:

Jedes Polynom p(x) vom Grad n kann als Produkt von n linearen Faktoren (den ev. komplexen Nullstellen z_k) geschrieben werden in der Form

$$p(x) = \alpha (x - z_1)(x - z_2)\ldots(x - z_n)$$

Annahme: Es gibt zwei Polynome p und q vom Grad <= n, die beide die Interpolationsbedingungen erfüllen.
Definiere neues Polynom $h(x) := p(x) - q(x)$. Dann gilt

$$h(x) = \alpha(x - z_1)(x - z_2)\ldots(x - z_n)$$

und $h(x_j) := p(x_j) - q(x_j) = 0$ für j=0,1,...,n

Daher hat das Polynom h(x) den Grad n und n+1 Nullstellen. Aus dem Hauptsatz der Algebra folgt daher, dass

$\alpha = 0$ sein muss, und daher ist $h(x) \equiv 0$, oder

$$p(x) \equiv q(x).$$

Also es existiert <u>genau ein</u> Interpolationspolynom!
Lagrange zur Lösung der Interpolation nicht geeignet, da numerisch problematisch und etwas teuerer als Neville: $O(n^2)$.

4.1.5. Berechnung des Interpolationspolynoms

Löse nicht das lineare Gleichungssystem, da dies zu teuer ist! Außerdem wird oft nur der Wert des Polynoms an einer einzigen Stelle gesucht!

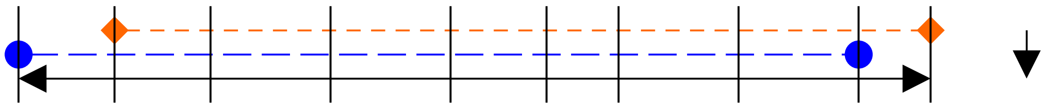
Idee: Berechne induktiv interpolierende Polynome für immer mehr Stützstellen.

Setze dazu $p_{i,\cdots,i+l}(x)$ als das interpolierende Polynom vom Grade l, das genau an den Stellen $x_i, x_{i+1}, \cdots, x_{i+l}$ die Interpolations-Bedingungen erfüllt.

Zur Bestimmung von $p_{i,\cdots,i+l}(x)$ verwende die interpolierenden Polynome vom Grade l-1 $p_{i+1,\cdots,i+l}(x)$ und $p_{i,\cdots,i+l-1}(x)$, die zu den Stützstellen $x_{i+1}, \cdots, x_{i+l}$, bzw. $x_i, \cdots, x_{i+l-1}$, gehören.

Wesentliche Formel :

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



Beweis: Nachprüfen der Interpolationsbedingung.

$$p_{i,\cdots,i+l}(x_i) = \frac{0 - (x_i - x_{i+l})\, y_i}{x_{i+l} - x_i} = y_i$$

$$p_{i,\cdots,i+l}(x_{i+l}) = \frac{(x_{i+l} - x_i)\, y_{i+l} - 0}{x_{i+l} - x_i} = y_{i+l}$$

und für alle anderen j mit i < j < i+l :

$$p_{i,\cdots,i+l}(x_j) = \frac{(x_j - x_i)y_j - (x_j - x_{i+l})y_j}{x_{i+l} - x_i} = y_j$$

Wegen der Eindeutigkeit des interpolierenden Polynoms ist jedes der so definierten Polynome genau die eindeutige Lösung des jeweiligen Interpolationsproblems!

Daher ist $p_{i,\cdots,i+l}(x)$ die gesuchte Lösung!

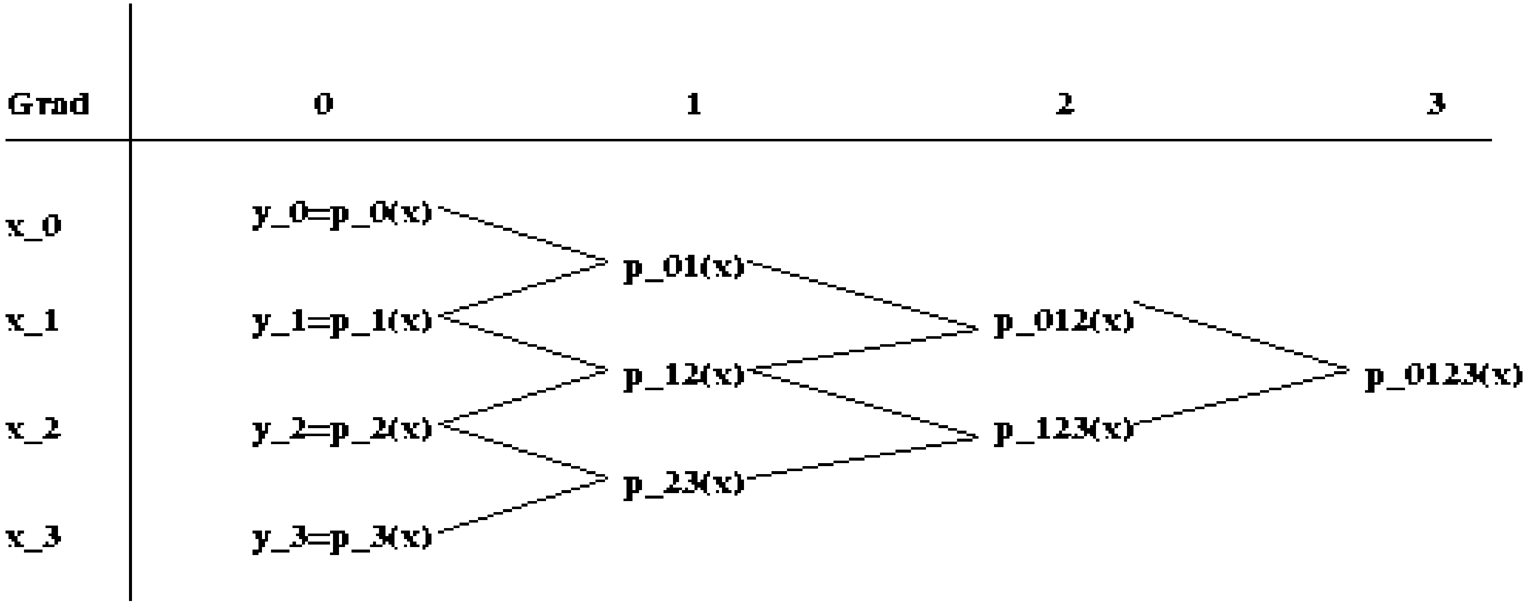
Anwendung der Formel (*) zur punktweisen Auswertung des Interpolationspolynoms an einer Stelle $\bar{x}$:

Eingabe: Stützwerte (x_j , y_j), j=0,...,n und Stelle $\bar{x}$; $p(\bar{x}) = ?$

Ausgabe: $p_{i,\cdots,i+l}(\bar{x})$.

Tableau-artige Berechnung der Interpolationspolynome aufsteigenden Grades, aber nur an der Stelle $\bar{x}$.

Neville-Tableau:



Erste Spalte sind konstante interpolierende Polynome, also genau die jeweils vorgegebenen Werte y_i .
Zweite Spalte sind interpolierende lineare Polynome zu jeweils zwei benachbarten Stützstellen.
Letzte Spalte enthält das interpolierende Polynom zu allen vorgegebenen Stützstellen.

Neue Stützstelle x_4 mit Wert y_4 kann in das Tableau eingefügt werden und führt zu einer neuen ‚Zeile' und einer neuen Endspalte p_01234(x).

Auswertung des Tableaus an einer festen Stelle x:

Beispiel: $x_0 = 0, y_0 = 1,$

$$x_1 = 1, y_1 = 3, \qquad x_2 = 3, y_2 = 2$$

Auswertung des interpolierenden Polynoms an der Stelle x=2 mit Lagrange, bzw. Neville-Tableau:

Lagrange: $L_0(x) = \frac{(x-1)(x-3)}{(0-1)(0-3)}$, $L_1(x) = \frac{(x-0)(x-3)}{(1-0)(1-3)}$ und

$$L_2(x) = \frac{(x-0)(x-1)}{(3-0)(3-1)} \qquad \rightarrow$$

$$p_{012}(2) = 1 \cdot L_0(2) + 3 \cdot L_1(2) + 2 \cdot L_2(2) = -\frac{1}{3} + 3 + 2 \cdot \frac{1}{3} = \frac{10}{3}$$

Neville-Tableau:

$$
\begin{array}{l|lll}
x\_0=0 & y\_0=p\_0(2)=1 & & \\
 & & 5 & \\
x\_1=1 & y\_1=p\_1(2)=3 & & 10/3 \\
 & & 5/2 & \\
x\_2=3 & y\_2=p\_2(2)=2 & &
\end{array}
$$

Da nach (*)

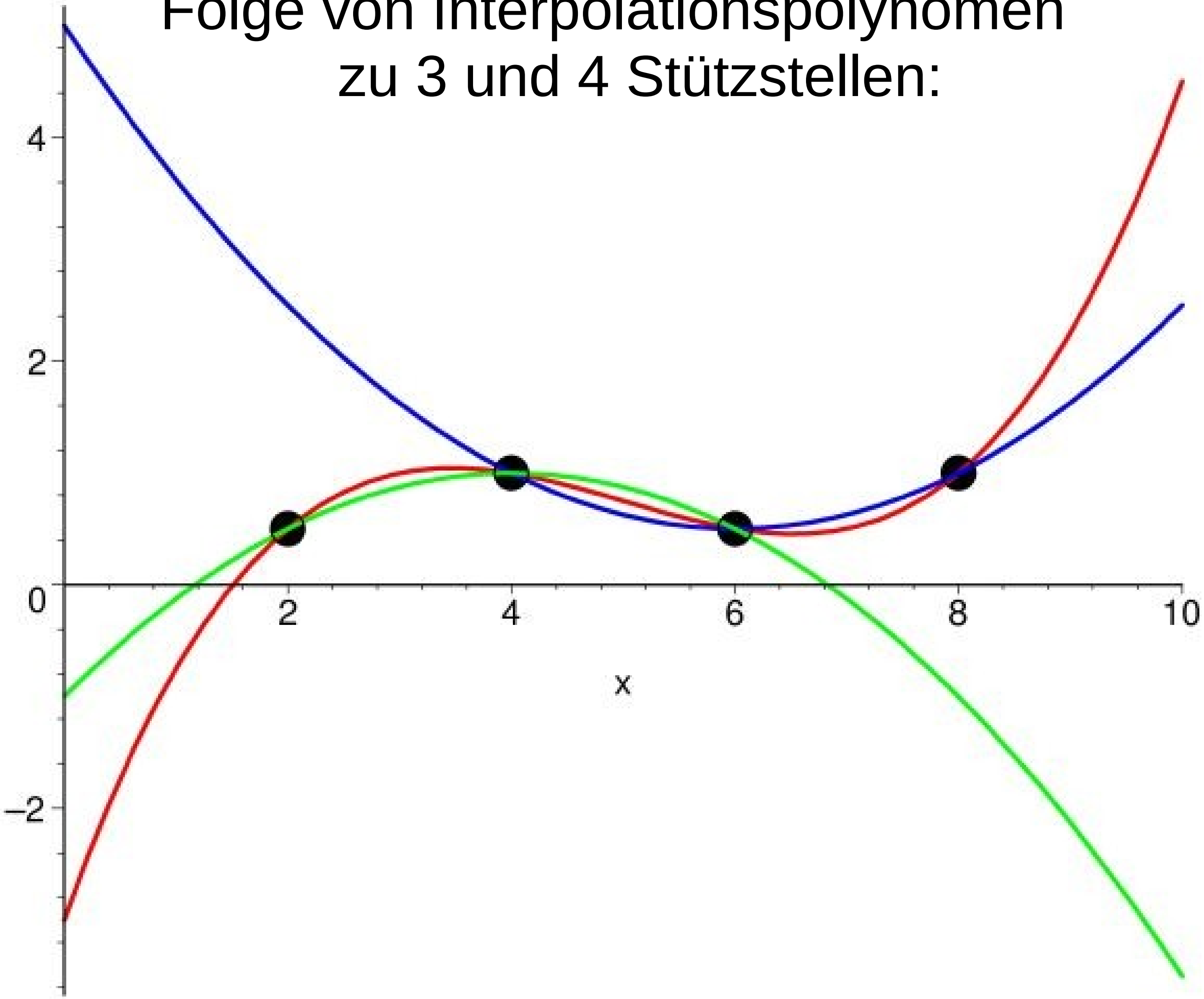
$$p_{01}(2)=\frac{(2-0)\cdot 3-(2-1)\cdot 1}{1-0}=5$$

$$p_{12}(2)=\frac{(2-1)\cdot 2-(2-3)\cdot 3}{3-1}=\frac{5}{2}$$

und daher auch

$$p_{012}(2)=\frac{(2-0)\cdot {}^{5}\!/_{2}-(2-3)\cdot 5}{3-0}=\frac{10}{3}\ .$$

Folge von Interpolationspolynomen zu 3 und 4 Stützstellen:



4.1.6. Fehler bei der Polynominterpolation

Satz: Gegeben Stützstellen $x_0 < \ldots < x_n$ und genügend oft diff'bare Funktion f(x). p(x) sei das interpolierende Polynom vom Grad n mit $p(x_i)=f(x_i)$, i=0,...,n
An einer beliebigen Stelle $\bar{x}$ gilt dann für die Abweichung zwischen p und f

$$f(\bar{x}) - p(\bar{x}) = \frac{f^{(n+1)}(\chi)}{(n+1)!} \cdot (\bar{x} - x_0) \cdots (\bar{x} - x_n)$$

Dabei ist $f^{(n+1)}(\chi)$ die (n+1)-te Ableitung von f an einer Zwischenstelle χ aus dem Intervall

$$I := \left[\min\{x_0, x_n, \bar{x}\}, \max\{x_0, x_n, \bar{x}\}\right]$$

Frage: Wie gut wird f durch p dargestellt?

Beweis: Definiere Hilfsfunktion

$$g(x) = f(x) - p(x) - K \cdot (x - x_0) \cdots (x - x_n)$$

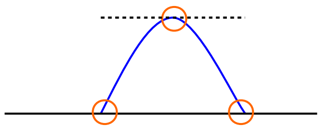
mit p(x) = das interpolierende Polynom, und K Konstante.
Nullstellen von g: x_0 , ..., x_n
Durch die Festlegung von K

$$K := \frac{f(\bar{x}) - p(\bar{x})}{(\bar{x} - x_0) \cdots (\bar{x} - x_n)}$$

wird auch $g(\bar{x}) = 0$

Also hat die Funktion g(x) n+2 Nullstellen!

Mit Mittelwertsatz besagt der Satz von Rolle, dass zwischen zwei Nullstellen einer stetig diff'baren Funktion f stets mindestens eine Nullstelle der Ableitung f ' liegen muss
(relatives Extremum mit waagrechter Tangente)



Also hat die erste Ableitung g' mindestens noch n+1 Nullstellen im Intervall I.

Die zweite Ableitung g'' noch n Nullstellen

...

die n+1-te Ableitung noch eine Nullstelle in I

$$0 = g^{(n+1)}(\chi) =$$

$$= f^{(n+1)}(\chi) - p^{(n+1)}(\chi) - K \cdot \frac{d^{n+1}}{dx^{n+1}}\big((x - x_0)\cdots(x - x_n)\big)\Big|_{\chi} =$$

$$= f^{(n+1)}(\chi) - K \cdot (n+1)!$$

Daher folgt: $$K = \frac{f^{(n+1)}(\chi)}{(n+1)!}$$

Denn die (n+1)-te Ableitung von $(x-x_0)\ldots(x-x_n)$ ist gleich der (n+1)-ten Ableitung von x^{n+1} allein.

Die (n+1)-te Ableitung ist auf dem Intervall I beschränkt, wenn f (n+1)-mal stetig diff'bar ist. Dann existiert M mit

$$|K| \le \frac{M}{(n+1)!} \text{ , z.B. mit } M = \max_{x \in I} \left| f^{(n+1)}(x) \right| < \infty$$

Insgesamt ergibt sich daher

$$\left| f(\bar{x}) - p(\bar{x}) \right| = \left| g(\bar{x}) + K(\bar{x} - x_0) \cdots (\bar{x} - x_n) \right| =$$

$$= \left| \frac{f^{(n+1)}(\chi)}{(n+1)!} (\bar{x} - x_0) \cdots (\bar{x} - x_n) \right| \le$$

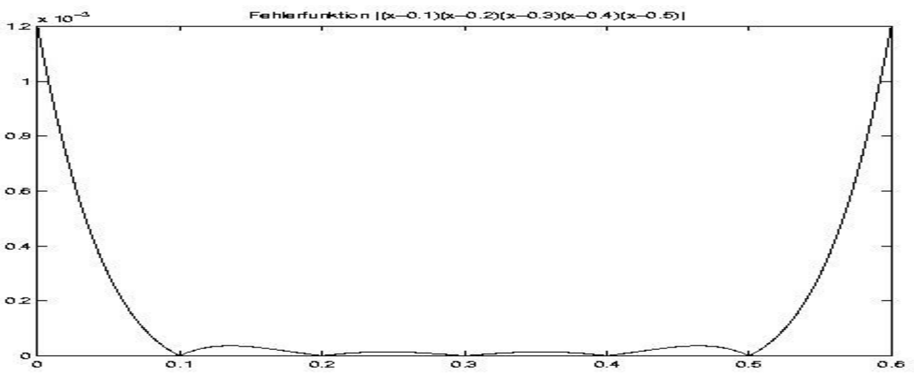
$$\le \frac{M}{(n+1)!} \left| (\bar{x} - x_0) \cdots (\bar{x} - x_n) \right|$$

Frage: Wie gut stellt p(x) die gegebene Funktion f(x) dar?
Entscheidend dafür sind

- die Größe der n+1-ten Ableitung von f auf dem Intervall I
- die Größe der Funktion $w(x) := (x - x_0)\cdots(x - x_n)$ an der Stelle $\bar{x}$

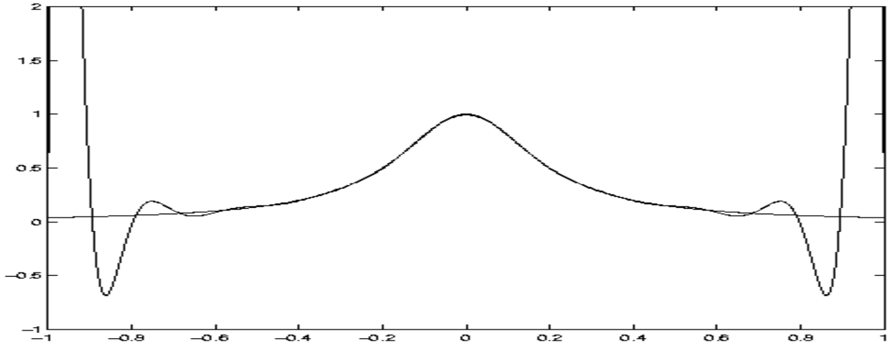
abhängig von Wahl der Stützstellen x_i

Verlauf der Funktion
w(x)=|x-0.1|*|x-0.2|*
|x-0.3||x-0.4|*|x-0.5|



Folgerung: Näherung am Rand schlechter!
Der Abstand zwischen Funktion und interpolierendem Polynom ist klein in der Mitte der Stützstelle.
Am Rand und außerhalb kann der Fehler schnell groß werden.

Beispiel: Runge-Funktion $$R(x) = \frac{1}{1+25x^2}, \qquad x \in [-1,1]$$



R(x) und p(x) vom Grad 20, äquidistante Stützstellen

Bessere Wahl der Stützstellen, um den Fehler gleichmäßig auf dem ganzen Intervall klein zu halten:

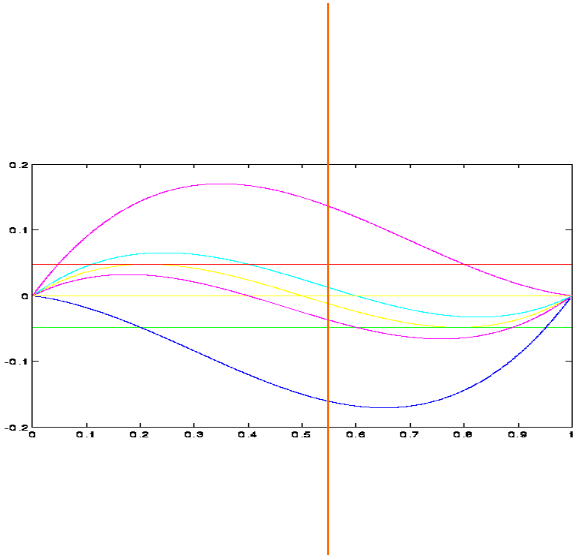
Wähle die Stützstellen so, dass $w(x)=x^{n+1}+\ldots = (x-x_0)\ldots$ als Polynom vom Grad (n+1) auf dem Intervall I möglichst **gleichmäßig klein** wird.

Spezialfall $n=2$, $x_0=0$ und $x_2=1$;
Bestimme x_1 so, dass w(x) möglichst klein:

$$\min_{x_1} : \max_x |w(x)| = \max_x |(x-0)(x-x_1)(x-1)| =$$

$$= \max_x |x(x-1)(x-x_1)| \quad in \quad [0,1]$$

Lösung: $x_1=0.5$, gelbe Kurve

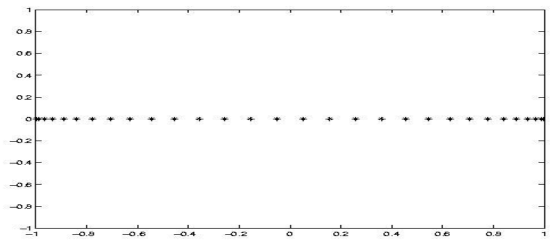


w(x) für verschiedene x_1 .
Optimal, wenn alle Werte in schmalem Band liegen.

Allgemeine Lösung im Intervall [-1,1]: Tchebycheff-Polynom

Nullstellen:

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



Also verteile Stützstellen besser so, dass am Rand mehr Punkte sind, um den ev. großen Fehler dort auszugleichen.

Allgemein gilt:

> Polynominterpolation mit Polynomen <u>hohen</u> Grades neigt zu Oszillationen (siehe Runge-Funktion) und wird kaum verwendet.

An Stelle von äquidistanten Stützstellen

$$x_j = a + \frac{j}{n}(b-a), \quad j = 0,1,\dots,n$$

verwende man besser eine Verteilung, bei der am Rande mehr Stützstellen sind, z.B. Tchebycheff-knoten (s.o.)

Allgemeines Stützstellenproblem im Intervall [-1,1]:

$$\min_{a_0,\ldots,a_n}\left\{\max_{-1\le x\le 1}\left|x^{n+1}+a_n x^n+\ldots+a_1 x+a_0\right|\right\}=$$
$$\min_{x_0,\ldots,x_n}\left\{\max_{-1\le x\le 1}\left|(x-x_0)\cdots(x-x_n)\right|\right\}$$

4.1.7 Newtonform des Interpolationspolynoms

Manchmal ist man auch an einer expliziten Darstellung des Interpolationspolynoms selbst interessiert.
Allerdings eignet sich dazu nicht die übliche Form

$$p(x) = a_n x^n + \cdots + a_1 x + a_0 \,.$$

Effiziente Auswertung dieser Form mittels Hornerschema:

$$(\cdots((a_n x + a_{n-1})x + a_{n-2})x + \cdots + a_1)x + a_0$$

durch das Programm:

```
y=a(n);
FOR j=n-1,…,0:
      y=y*x+a(j);
ENDFOR
```

An Stelle der Standardform verwendet man p(x) in einer Darstellung, in der die Stützstellen eingehen:

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

Die Koeffizienten $f[x_0,\ldots,x_j]$ erhält man wieder aus der wesentlichen Formel (*) aus 4.1.5:

$$\boxed{\begin{aligned} p_{i,\cdots,i+k}(x) &= \frac{(x-x_i)\,p_{i+1,\cdots,i+k}(x) - (x-x_{i+k})\,p_{i,\cdots,i+k-1}(x)}{x_{i+k}-x_i} = \\ & f[x_i] + f[x_i x_{i+1}](x-x_i) + \ldots + f[x_i,\ldots,x_{i+k}](x-x_i)\cdots(x-x_{i+k-1}) \end{aligned}}$$

Sie lassen sich wieder aus einem Tableau (Tableau der ‚Dividierten Differenzen') der Reihe nach berechnen mittels

$$\boxed{f[x_i,\ldots,x_{i+k}] = \frac{f[x_{i+1},\ldots,x_{i+k}] - f[x_i,\ldots,x_{i+k-1}]}{x_{i+k} - x_i}}$$

Dann kann durch Horner-artiges Schema p(x) an der Stelle x ausgewertet werden. *MATLAB-BEISPIEL*

4.1.8. <u>Weitere Interpolationsansätze</u>

(a) Hermite-Interpolation:

Um Oszillationen zu vermeiden will man auch die Ableitungen an den Stützstellen kontrollieren, im einfachsten Fall genau die ersten Ableitungen:

Vorgegeben sind (x_j , y_j , y'_j) , $j=0,...,n$

Gesucht ein Polynom vom Grad 2n+1 mit

$$p(x_j) = y_j \quad \text{und} \quad p'(x_j) = y'_j \;, \; j=0,1,...,n$$

(1) Lösung durch Aufstellen des dadurch gegebenen linearen $2n+2 \times 2n+2$ Gleichungssystems für die gesuchten Koeffizienten des Polynoms:

$$\begin{pmatrix} 1 & x_0 & \cdots & x_0^{2n+1} \\ 0 & 1 & \cdots & (2n+1)x_0^{2n} \\ 1 & x_1 & \cdots & x_1^{2n+1} \\ \vdots & \vdots & & \vdots \end{pmatrix} \cdot \begin{pmatrix} c_0 \\ c_1 \\ \vdots \\ c_{2n+1} \end{pmatrix} = \begin{pmatrix} y_0 \\ y_0' \\ y_1 \\ \vdots \end{pmatrix}$$

(2) Lösung durch ‚Lagrange'-Polynome, die genau an einer Stützstelle z.B. Wert 0 und Ableitung 1 haben, sonst an allen anderen Stützstellen Wert und Ableitung 0.

Setze dazu $$L_j(x) := (x-x_0)^2 \cdots (x-x_{j-1})^2 (x-x_{j+1})^2 \cdots (x-x_n)^2$$

1. Aufgabe:
Finde Polynom vom Grad 2n+1, das an der Stelle x_j den Wert 1 und Ableitung 0 hat, und an allen anderen Stützstellen Wert und Ableitung 0:

Ansatz: $$G_j(x) = L_j(x) \cdot (\alpha + \beta x)$$

α und β sind daher so zu wählen, dass

$$L_j(x_j)(\alpha + \beta x_j) = 1 \quad \text{gilt} \quad \text{und} \quad L_j'(x_j)(\alpha + \beta x_j) + L_j(x_j)\beta = 0$$

Dies liefert zwei Bedingungen für α und β $\rightarrow$ $G_j(x)$

2. Aufgabe:
Finde Polynom $H_j(x)$, das an der Stelle x_j den Wert 0 und Ableitung 1 hat, an allen anderen Stützstellen Wert und Ableitung 0.

Dann sind α und β so zu wählen, dass

$$L_j(x_j)(\alpha+\beta x_j)=0 \quad \text{gilt und} \quad L_j'(x_j)(\alpha+\beta x_j)+L_j(x_j)\beta=1$$

Lösung liefert α und β $\rightarrow$ $H_j(x)$

Damit erhält man als Lösung der Hermite-Interpolation:

$$p(x)=\sum_{j=0}^{n}\left(y_j G_j(x)+y'_j H_j(x)\right)$$

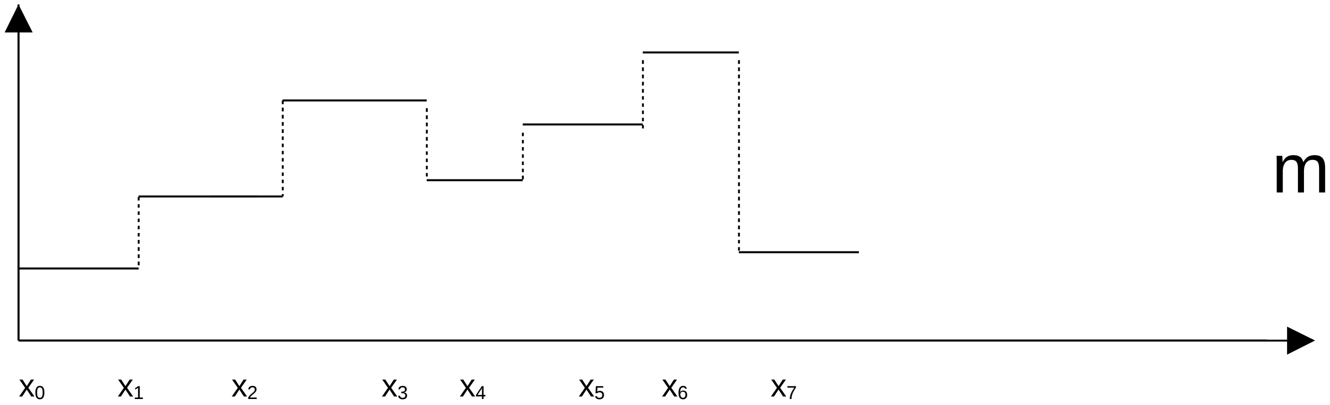
(b) Spline-Interpolation:

Versuche, die Oszillationen zu verhindern, indem man die interpolierende Funktion <u>stückweise</u> aus Polynomen niedrigen Grades zusammenbaut.

Definition einer Splinefunktion S vom Grade k zu Stützstellen $x_0 ,..., x_n$:

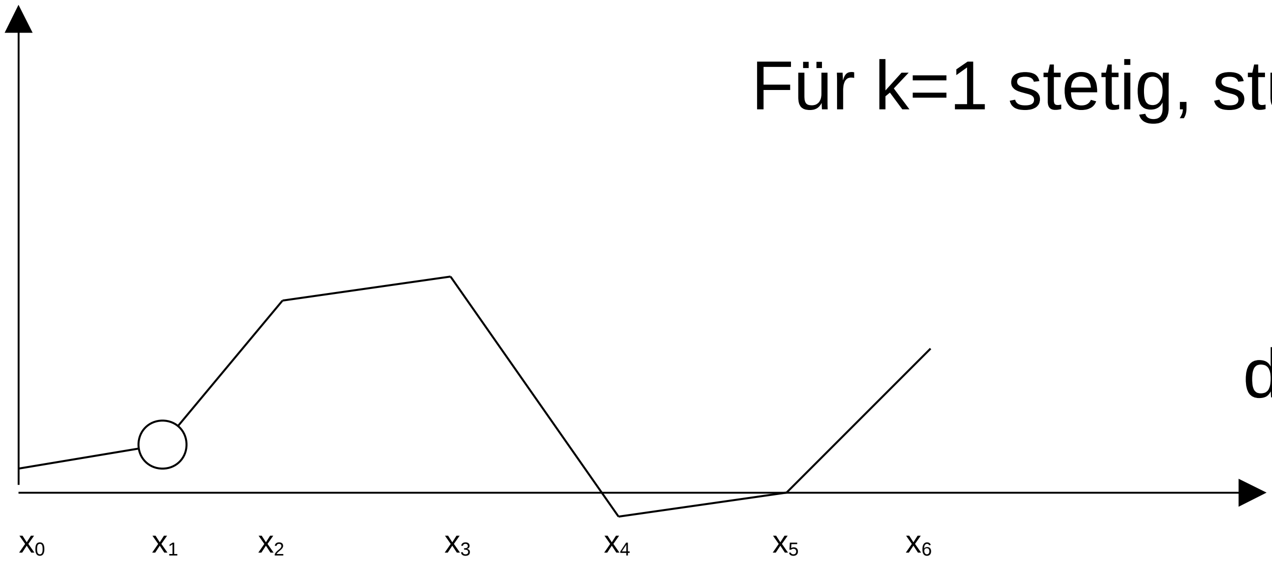
S(x) sei <u>insgesamt</u> (k-1)-mal stetig diff'bar und auf den <u>einzelnen Teil-Intervallen</u> $[x_i , x_{i+1}]$ *jeweils ein Polynom k-ten Grades.*

Für k=0 stückweise konstante Funktionen (Treppenfunktionen):



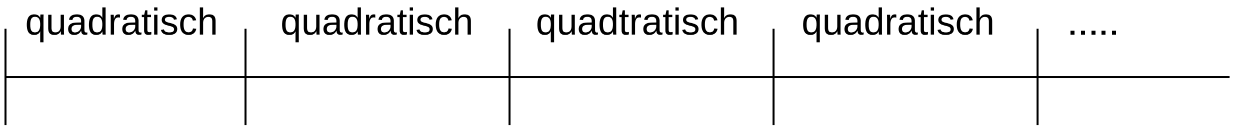
mit Sprungstellen

Für k=1 stetig, stückweise linear:



○: Unstetigkeitsstelle der ersten Ableitung

Für k=2 stückweise quadratische Polynome, stetig diff'bar verbunden.

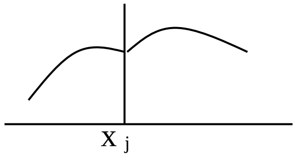


Für k=3 (kubische Splines) stückweise Polynome dritten Grades; an den Nahtstellen sind die Funktion samt erster und zweiter Ableitung stetig. Diese Splines werden am häufigsten verwendet.

Das Interpolationsproblem mit Splinefunktionen lautet dann: Bestimme Spline S(x) zu k so, dass $S(x_j)=y_j$, $j=0,\ldots,n$

Dies führt auf lineares Gleichungssystem für die Parameter, die S(x) bestimmen, abhängig von k.

Neben den n+1 Interpolationsbedingungen gehen noch die Bedingungen an den Nahtstellen ein (stetig, stetig diff'bar,...), z.B. in der Form



$$p_{links}(x_j) = p_{rechts}(x_j)$$

$$p'_{links}(x_j) = p'_{rechts}(x_j)\text{, usw.}$$

p sollen dabei Polynome k-ten Grades sein und S, S',... stetig bis einschließlich (k-1)-te Ableitung.

Bestimmt werden die Koeffizienten der Teilpolynome vom Grad k, so dass die Interpolationsbedingungen erfüllt sind und die Übergänge entsprechend stetig sind.

Anzahl Unbekannte: n Polynome vom Grad k: n(k+1) Koeff.

Anzahl Bedingungen: n+1 Interpol.-Bedingungen $S(x_j)=y_j$

und (n-1)k stetige Übergänge $p_j^{(i)}(x_j) = p_{j+1}^{(i)}(x_j)$

für i=0,…,k-1 an den inneren Stellen j=1,…,n-1

.

Insgesamt also nk+n-k+1=n(k+1)+(1-k) Bedingungen.

Füge noch k-1 neue Bedingungen hinzu, um ein quadratisches lineares Gleichungssystem mit genauso vielen Gleichungen wie Unbekannten zu erhalten,

z.B. - Bedingungen über Ableitungen an den Endpunkten,
oder - Periodizität, o.ä.
also $S''(a)=S''(b)=0$, oder $S^j(a)=S^j(b)$.

Dann ergibt sich dann eine n(k+1) x n(k+1) - Matrix.

Geschickter Ansatz:

Bestimme wieder geeignete ***Basis*** von Splinefunktionen und löse das Interpolationsproblem in dieser Basis →

B-Spline-Basis (analog zu Lagrange-Polynomen):

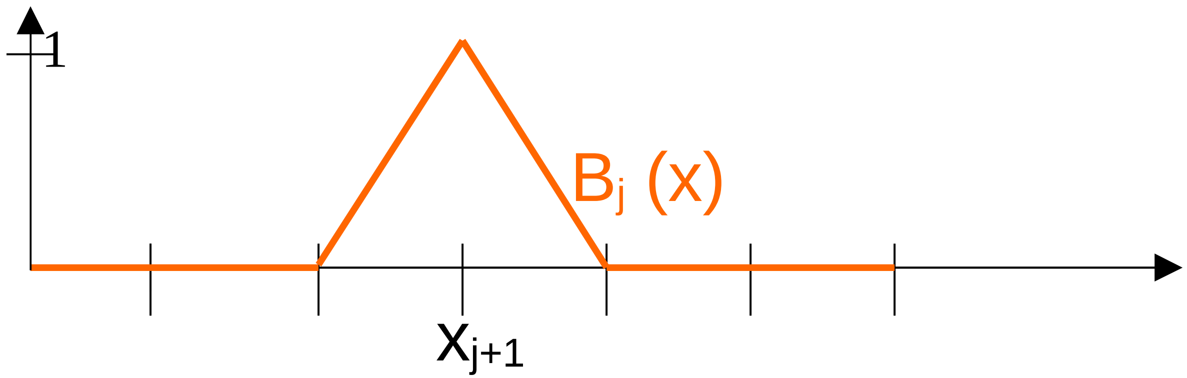
Elementare Splinefunktion vom Grade k,

- die höchstens an einer Stützstelle den Wert 1 hat,
- und an ‚fast' allen anderen 0.

Der Fall k=1:

Gesucht: stückweise lineare, stetige Funktion, die genau bei x_{j+1} den Wert 1 annimmt, und an anderen Stützstellen 0.

Hut-Funktion

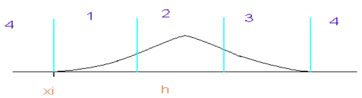


$$B_j(x) := \begin{cases} \dfrac{x - x_j}{x_{j+1} - x_j} & \text{für} \quad x \in [x_j, x_{j+1}] \\[2ex] \dfrac{x - x_{j+2}}{x_{j+1} - x_{j+2}} & \text{für} \quad x \in [x_{j+1}, x_{j+2}] \\[2ex] 0 & \text{sonst} \end{cases}$$

Lösung der Interpolations-Aufgabe:

$$\boxed{S(x) = \sum_{j=0}^{n} y_j B_{j-1}(x)}$$

B-Spline für k=2 (quadratisch):



$$B_i(x) := \frac{1}{2h^2} \begin{cases} (x - x_i)^2 & \textit{für} \quad x \in]x_i, x_{i+1}[& (1) \\ h^2 + 2h(x - x_{i+1}) - 2(x - x_{i+1})^2 & \textit{für} \quad x \in [x_{i+1}, x_{i+2}] & (2) \\ (x_{i+3} - x)^2 & \textit{für} \quad x \in]x_{i+2}, x_{i+3}[& (3) \\ 0 & \textit{sonst} & (4) \end{cases}$$

Stetig diff'bar, stückweise quadratisch.