

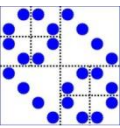
Die $(n+1)$ -te Ableitung ist auf dem Intervall I beschränkt, wenn f $(n+1)$ -mal stetig diff'bar ist. Dann existiert M mit

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

Insgesamt ergibt sich daher

$$\begin{aligned} |f(\bar{x}) - p(\bar{x})| &= |g(\bar{x}) + K(\bar{x} - x_0) \cdots (\bar{x} - x_n)| = \\ &= \left| \frac{f^{(n+1)}(\chi)}{(n+1)!} (\bar{x} - x_0) \cdots (\bar{x} - x_n) \right| \leq \\ &\leq \frac{M}{(n+1)!} |(\bar{x} - x_0) \cdots (\bar{x} - x_n)| \end{aligned}$$

□



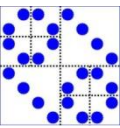
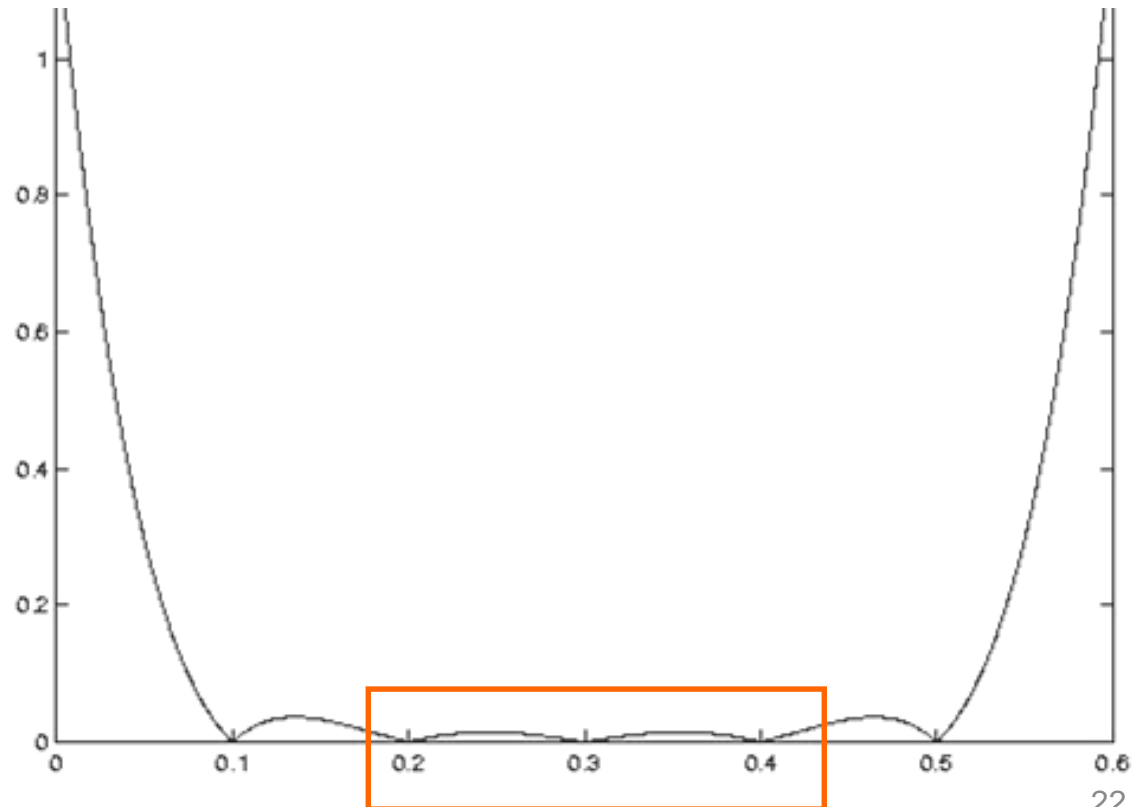
Frage: Wie gut stellt $p(x)$ die gegebene Funktion $f(x)$ dar?

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

$$w(x) = |x - 0.1| * |x - 0.2| * |x - 0.3| * |x - 0.4|$$

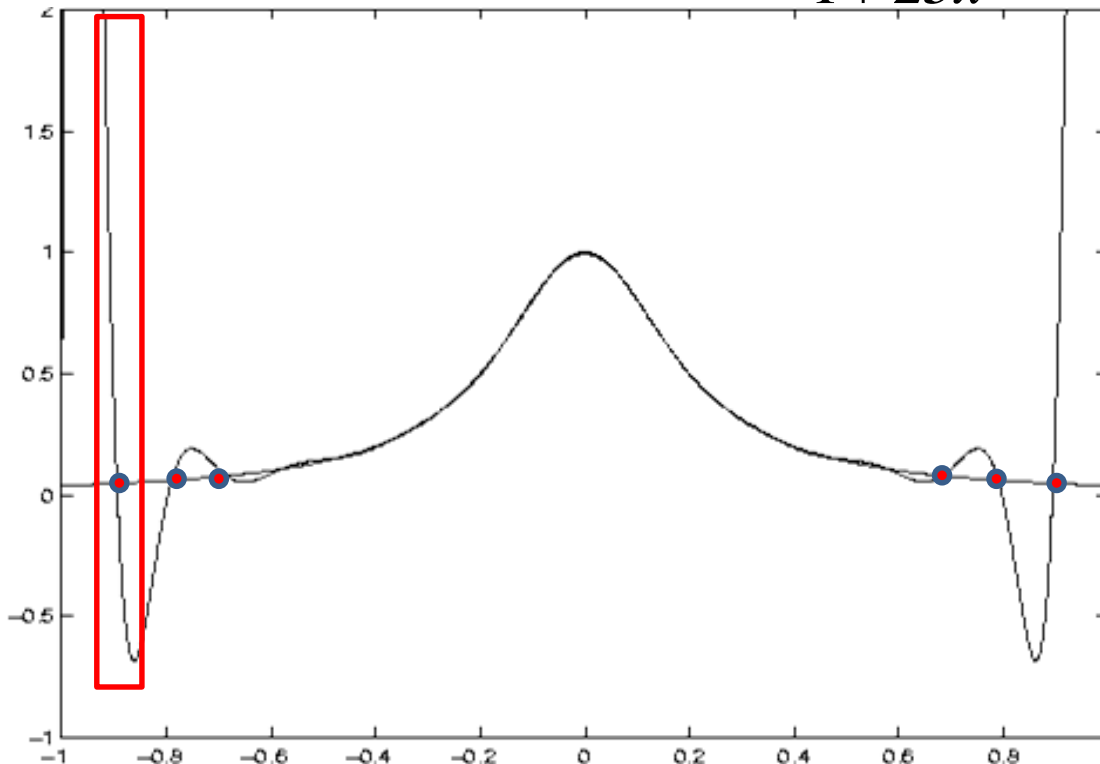


Folgerung: Näherung am Rand schlechter!

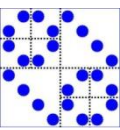
Der Abstand zwischen Funktion und interpolierendem Polynom ist klein in der Mitte der Stützstelle.

Am Rand und außerhalb kann der Fehler schnell groß werden.

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



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



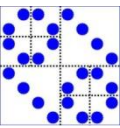
Bessere Wahl der Stützstellen, um den Fehler gleichmäßig auf dem ganzen Intervall klein zu halten:

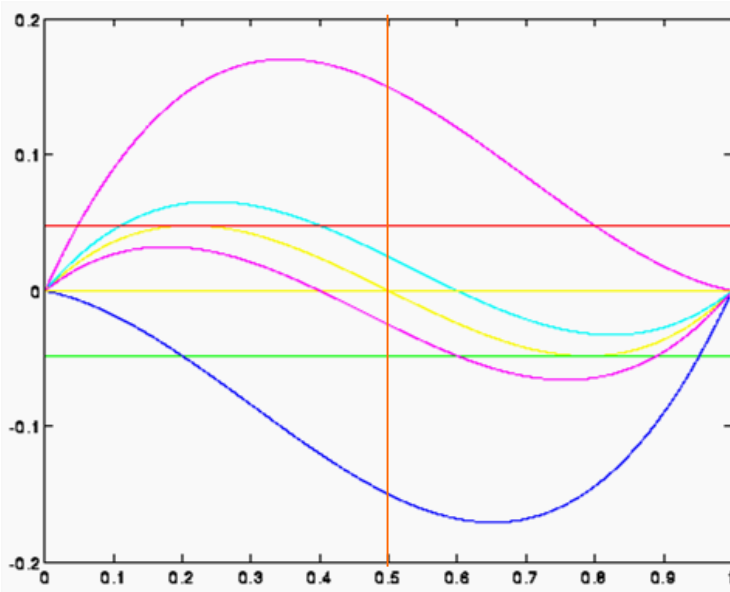
Wähle die Stützstellen so, dass $w(x)=x^{n+1}+\dots = (x-x_0)\dots$ als Polynom vom Grad $(n+1)$ auf dem Intervall I möglichst **gleichmäßig klein** wird.

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

$$\begin{aligned} \min_{x_1} : \max_x |w(x)| &= \max_x |(x-0)(x-x_1)(x-1)| = \\ &= \max_x |x(x-1)(x-x_1)| \quad \text{in } [0,1] \end{aligned}$$

Lösung: $x_1=0.5$, gelbe Kurve





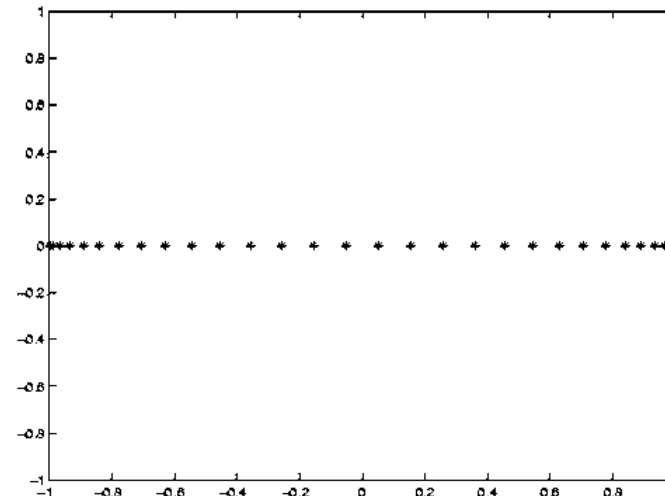
$w(x)$ für verschiedene x_1 .
Optimal, wenn alle Werte
in schmalem Band liegen.

Beispiel: stuetz.m

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

Nullstellen: \rightarrow optim. Stützst.:

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



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

Allgemein gilt:

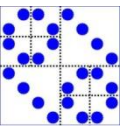
Polynominterpolation mit Polynomen hohen Grades neigt zu Oszillationen (siehe Runge-Funktion) und wird kaum verwendet.

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.).

Ev. adaptive Stützstellenwahl je nach Verhalten von f .



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

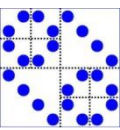
$$p(x) = a_n x^n + \cdots + a_1 x + a_0$$

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:

$$\begin{aligned}
 p(x) &= f[x_0] + f[x_0, x_1](x - x_0) + \\
 &+ f[x_0, x_1, x_2](x - x_0)(x - x_1) + \dots + \\
 &+ f[x_0, \dots, x_n](x - x_0) \cdots (x - x_{n-1}) = \\
 &= P_{0,1,\dots,n}(x)
 \end{aligned}$$

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

$$p_{i,\dots,i+k}(x) = \frac{(x - x_i)p_{i+1,\dots,i+k}(x) - (x - x_{i+k})p_{i,\dots,i+k-1}(x)}{x_{i+k} - x_i} = \\
 f[x_i] + f[x_i, x_{i+1}](x - x_i) + \dots + f[x_i, \dots, x_{i+k}](x - x_i) \cdots (x - x_{i+k-1})$$

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

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

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

4.1.8. Weitere Interpolationsansätze

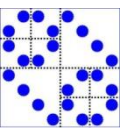
(a) Hermite-Interpolation:

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) , $j=0, \dots, n$

Gesucht ein Polynom vom Grad $2n+1$ mit

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



- (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 $\beta \rightarrow G_j(x)$

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

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

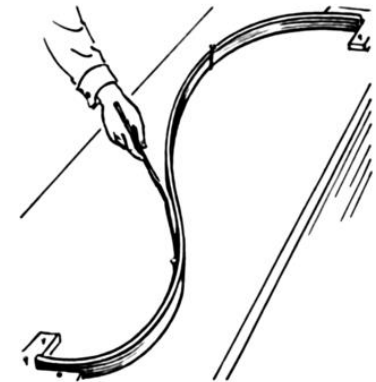
Lösung liefert α und $\beta \rightarrow H_j(x)$

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: **Minimiere Krümmung**

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

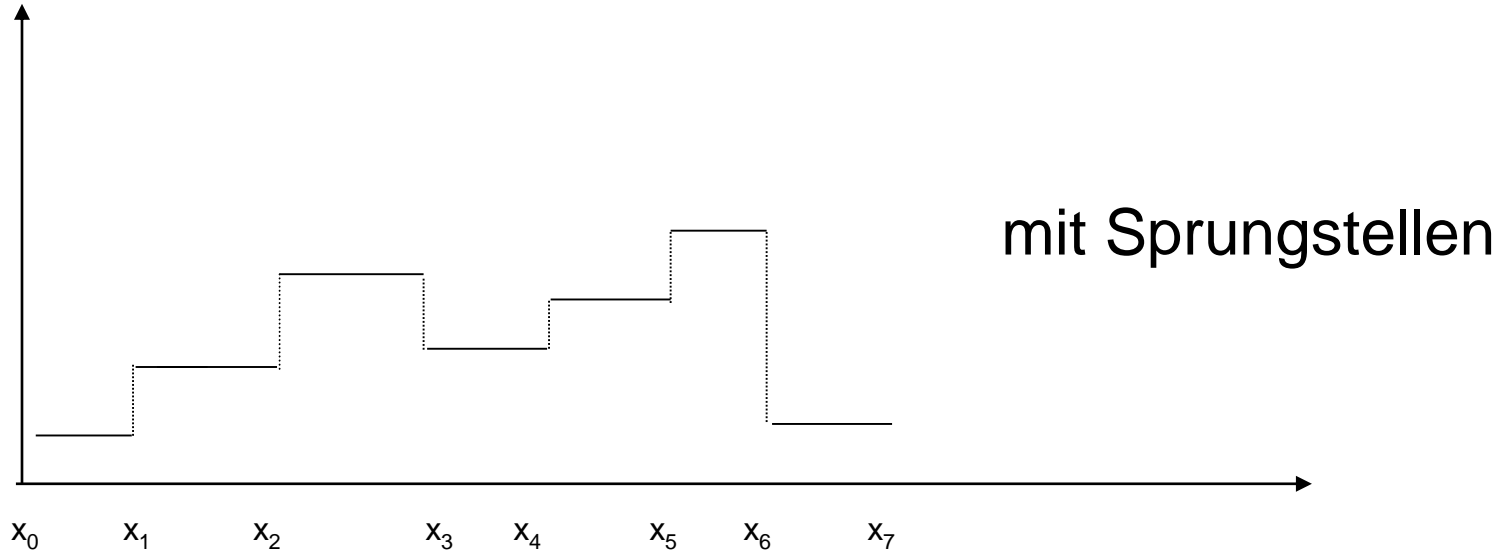


Kein Interesse an geschlossener Darstellung! Paradigmenwechsel!

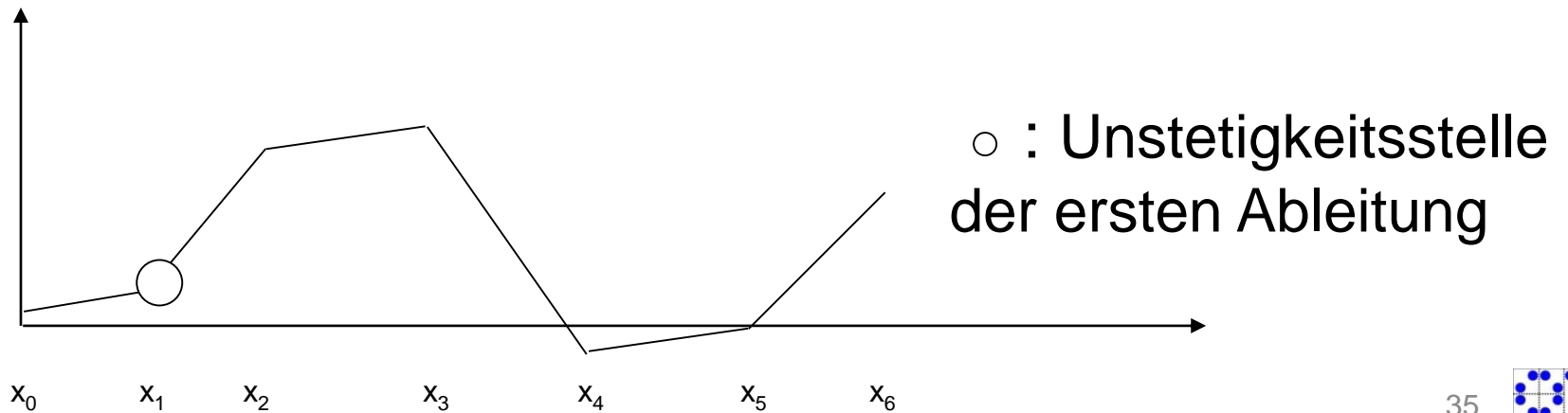
Definition einer Splinefunktion S vom Grade k zu Stützstellen x_0, \dots, x_n :

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

Für $k=0$ stückweise konstante Funktionen (Treppenfunktionen):

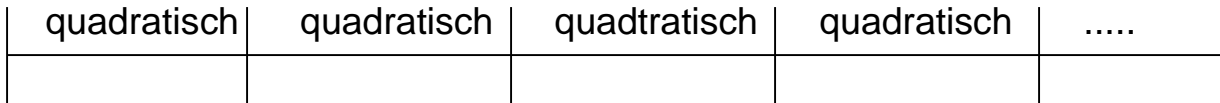


Für $k=1$ stetig, stückweise linear:





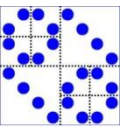
Für $k=2$ stückweise quadratische Polynome, stetig diff'bar verbunden.



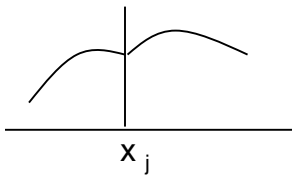
Für $k=3$ (kubische Splines) stückweise Polynome dritten Grades; an den Nahtstellen sind die Funktion samt erster und zweiter Ableitung stetig. Diese Splines werden am häufigsten verwendet.

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

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



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

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

p sollen dabei Polynome k -ten Grades sein und S, S', \dots 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 $i=0, \dots, k-1$ an den inneren Stellen $j=1, \dots, n-1$

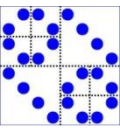
Insgesamt also $n+1+(n-1)k = n(k+1)+(1-k)$ Bedingungen.

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 $n(k+1) \times n(k+1)$ - 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

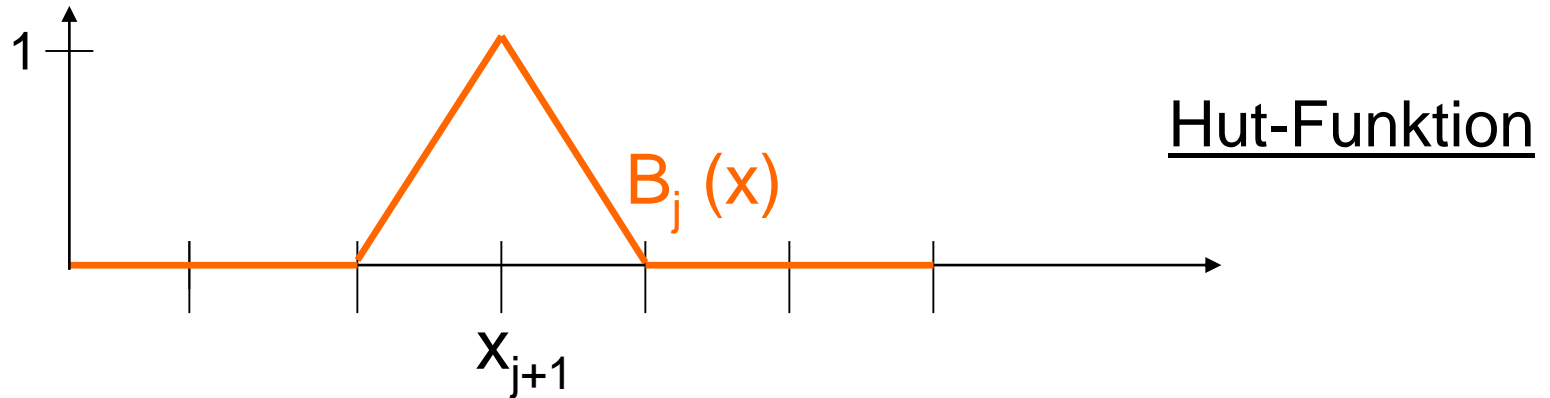
- höchstens an einer Stützstelle den Wert 1 hat,
- und an ‚fast‘ allen anderen 0.



Der Fall $k=1$:



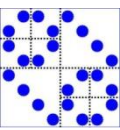
Gesucht: stückweise lineare, stetige Funktion, die genau bei x_{j+1} den Wert 1 annimmt, und an anderen Stützstellen 0.



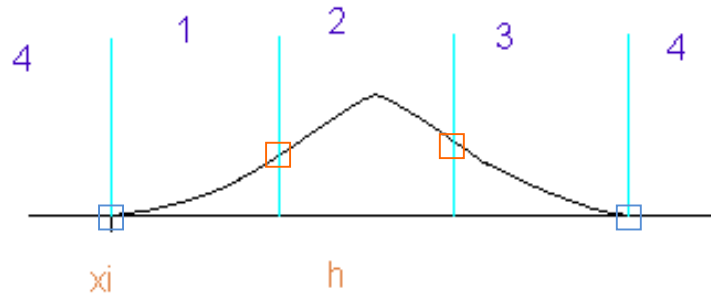
$$B_j(x) := \begin{cases} \frac{x - x_j}{x_{j+1} - x_j} & \text{für } x \in [x_j, x_{j+1}] \\ \frac{x - x_{j+2}}{x_{j+1} - x_{j+2}} & \text{für } x \in [x_{j+1}, x_{j+2}] \\ 0 & \text{sonst} \end{cases}$$



Lösung der Interpolations-Aufgabe: $S(x) = \sum_{i=0}^n y_i B_{i-1}(x)$ ₄₀



B-Spline für $k=2$ (quadratisch):



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

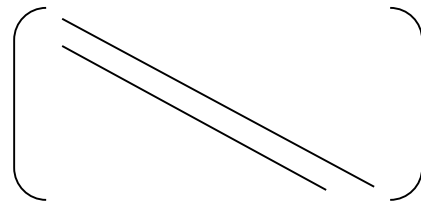
Stetig diff'bar, stückweise quadratisch.

Interpolationsproblem:

Ansatz:
$$p(x) = \sum_{i=-1}^{n-1} a_i B_i(x), \quad a_i = ?$$

$$y_j = p(x_j) = \sum_{i=-1}^{n-1} a_i B_i(x_j) = a_{j-2} B_{j-2}(x_j) + a_{j-1} B_{j-1}(x_j) \quad \text{für } j=0,1,\dots,n.$$

Ergibt bidiagonales lineares Gleichungssystem zur Berechnung der Koeffizienten a_j :



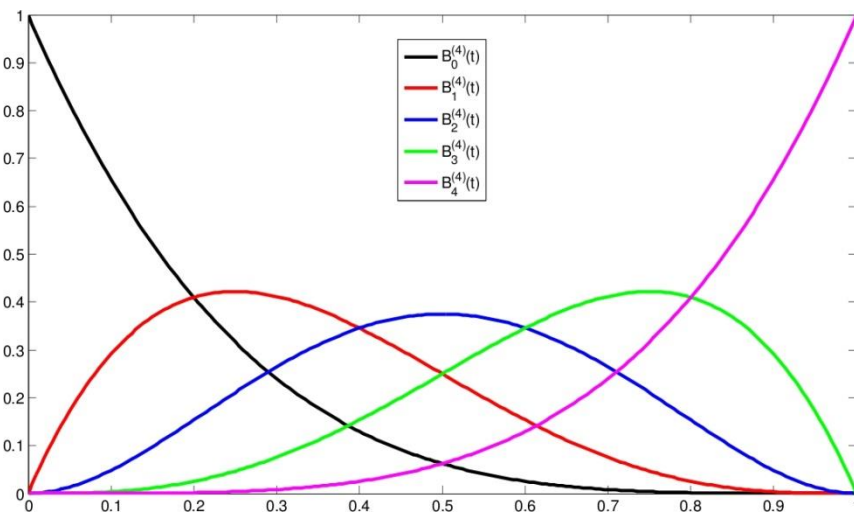
Trivial zu lösen! (Hier $B_{-2}(x) \equiv 0$, und x_{-1} dazu)

(c) Bezier-Kurven: (keine Interpolation)

Gesucht sind Kurven, die nicht exakt durch die Stützpunkte laufen, aber sich durch die Stützpunkte gut kontrollieren lassen.
(lokale Änderungen sollen nur lokal wirken)

Ausgangspunkt:

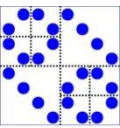
Bernsteinpolynome in $[0, 1]$



$$B_i^{(n)}(t) := \binom{n}{i} \cdot (1-t)^{n-i} t^i \quad \text{für } i = 0, 1, \dots, n$$

Für $n=4$ sind dies also 5 Polynome.

Es gilt: $\sum_{i=0}^n B_i^{(n)}(t) = \sum_{i=0}^n \binom{n}{i} (1-t)^{n-i} t^i = 1$; Darstellung der 1



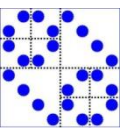
Jedes $B_i^{(4)}(t)$ hat einen Teilbereich, indem es die anderen Polynome dominiert! Alle zusammen ergeben immer 1.

Gegeben: Stützpunkte $\vec{b}_i \in R^k$, $i=0, \dots, n$, als Vektoren im R^k .

Bezierkurve zu diesen Punkten ist definiert als Vektorsumme:

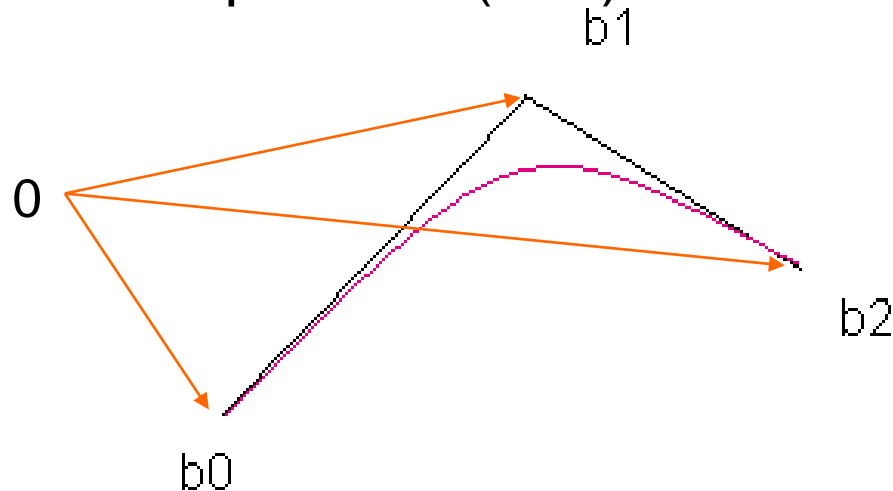
$$X(t) := \sum_{i=0}^n \vec{b}_i B_i^{(n)}(t) \quad \text{für } t \in [0,1]$$

Beginnend mit $t=0$ hat der Vektor $X(t)$ zunächst den Wert b_0 , und fährt dann für wachsendes t der Reihe nach die anderen Kontrollpunkte b_i näherungsweise ab, bis er für $t=1$ bei b_n anlangt.



Dabei überwiegt in einem Teilbereich von $[0,1]$ jeweils eines der Bernsteinpolynome $B_i(x)$ die anderen, und sorgt dafür, dass die Kurve zu dem entsprechenden Kontrollpunkt b_i gezogen wird.

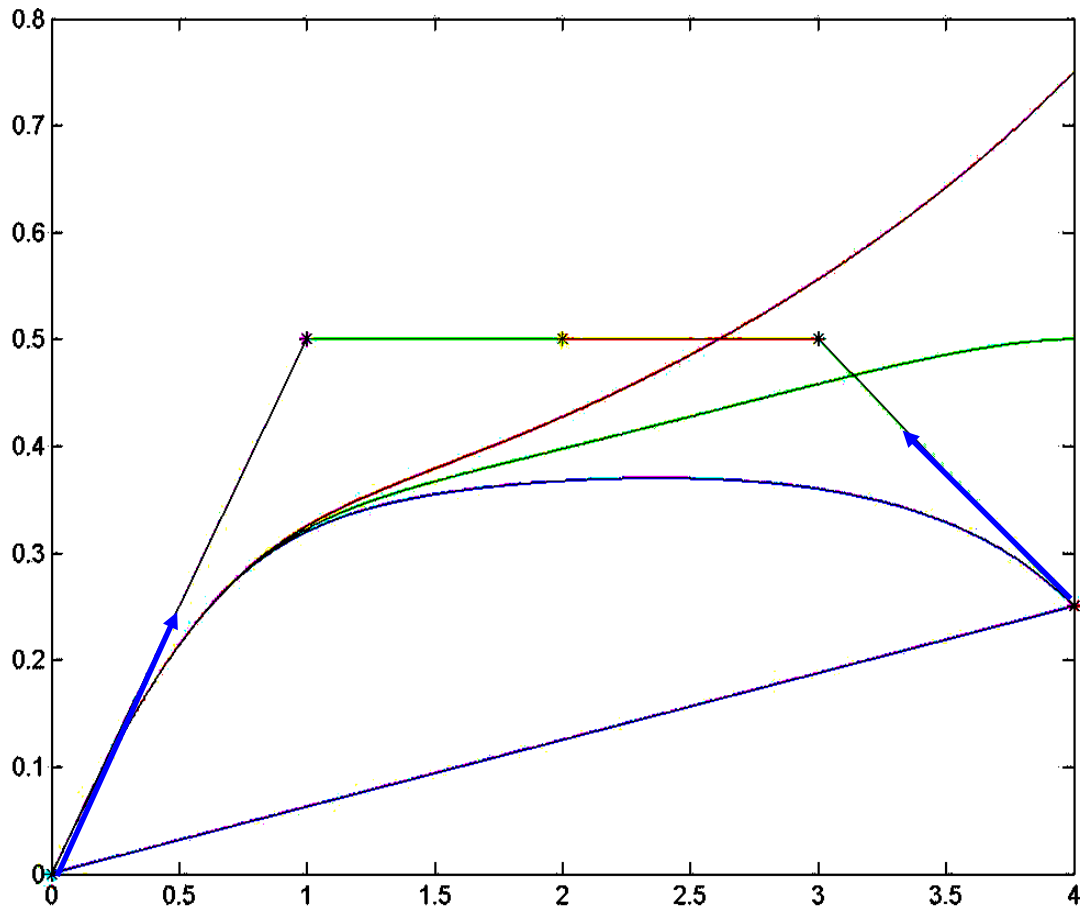
Verlauf bei drei Kontrollpunkten ($n=2$):



Die sich ergebende Bezier-Kurve ist eine gewichtete Summe von Vektoren!

Ergibt keine interpolierende Kurve, aber gut steuerbar →
Computergraphik (Word, CorelDraw, ...)

Beispiel Bezierkurve (n=3):



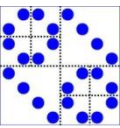
Hier sind drei verschiedene Bezierkurven dargestellt, die sich nur im vierten Kontrollpunkt unterscheiden.

- Die Kurve verläuft stets innerhalb der konvexen Hülle der Kontrollpunkte (als Polygon eingezeichnet für den Fall, dass der rechte Kontrollpunkt den kleinsten Wert hat.).
- Am Anfangs-, bzw. Endpunkt zeigt die Tangente in Richtung des nächsten Nachbarpunktes (blauer Pfeil).
- Bei Veränderung des rechten Kontrollpunktes bleibt die linke Kurve fast unverändert; d.h. lokale Änderungen wirken sich kaum auf den Rest der Kurve aus!

Bezier-Applets: MATLAB bezier.m

<http://www.theparticle.com/applets/nyu/BezierApplet/>

<http://sunsite.ubc.ca/LivingMathematics/V001N01/UBCExamples/Bezier/bezier.html>



Verbesserung:

Ersetze Bernsteinpolynome durch B-Splines (Nurbs)

Non uniform rational B-Splines , Gewichte w_i

$$R_{i,k}(x) = \frac{N_{i,k}(x)w_i}{\sum_{j=0}^n N_{j,k}(x)w_j}, \quad i = 0, \dots, n, \quad C(x) = \sum_{i=0}^n R_{i,k}(x) \cdot \vec{P}_i$$

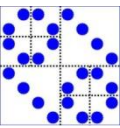
Vorteile:

- bessere Approximationseigenschaften
- einstellbare Differenzierbarkeit (k-mal stetig diff'bar)
- bessere Lokalitätseigenschaften
(lokale Änderungen wirken wirklich nur lokal)

B-Splines Applets:

<http://www.cs.technion.ac.il/~cs234325/Applets/applets/bspline/html/index.html>

<http://userpage.fu-berlin.de/~vratiska/Bildverarbeitung/Bspline/Bspline.html>



4.1.9. Eines der Hauptprobleme der Numerik: Finde geeignete Ansatzfunktionen (Basis)

Klassisch: Polynome $p(x)$ {Momome, Lagrange, Newton,
Bernstein, Tchebycheff, Orthogonal P., ...}
rationale Funktionen $p(x)/q(x)$,
Taylor-, Potenz-, Laurent-Reihe , z.B. $\sum a_k x^k$
Fourierreihen, $\cos(kx)$, $\sin(kx)$, z.B. $\sum a_k e^{ikx}$

Analytische Darstellung

Heutzutage erforderlich:

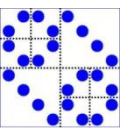
Algorithmische Darstellung

Anforderungskatalog:

- Variable Diff'barkeit, angepasst an das gestellte Problem
- Variable Approximationsgüte, kleiner Fehler
- Adaptivität, Auflösung angepasst an (lokale) Anforderungen
- Lokalität, d.h. lokale Änderungen wirken nur lokal
- Schnelle Algorithmen, dünne Gleichungen
- *Analysemöglichkeiten, vgl. Fouriertransformation Kap. 5*

Die ersten Punkte sind z.B. durch B-Splines erfüllt.

Es fehlt noch die Möglichkeit einer **„Frequenzanalyse“**, die wir bei der Fouriertransformation kennenlernen werden → Multiskalendarstellung, Wavelet, Multigrid



Vergleich der Interpolationsverfahren:

- Standard-Polynom-Interpolation:

Gegeben $(x_j, y_j), j=0,1,\dots,n$

Gesucht interpolierendes Polynom vom Grad n

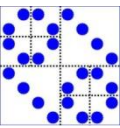
- Hermite-Interpolation:

Gegeben $(x_j, y_j, y'_j), j=0,1,\dots,n$

Gesucht interpolierendes Polynom vom Grad $2n+1$

- Spline-Funktionen:

stückweise Polynome vom Grad k , $(k-1)$ -mal stetig diff'bar, aneinandergeheftet, so dass sie die Interpolations-Bedingungen erfüllen.



- Nicht-interpolierende Kurven der Form

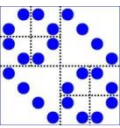
$\sum_{i=1,\dots,n} b_j \cdot B_j(t)$, $t \in I$, mit Kontrollpunkten b_j und Gewichten $B_j(t)$ (z.B. Bezier-, B-Spline-Gewichte).

- Stückweise Hermite-Interpolation, d.h.

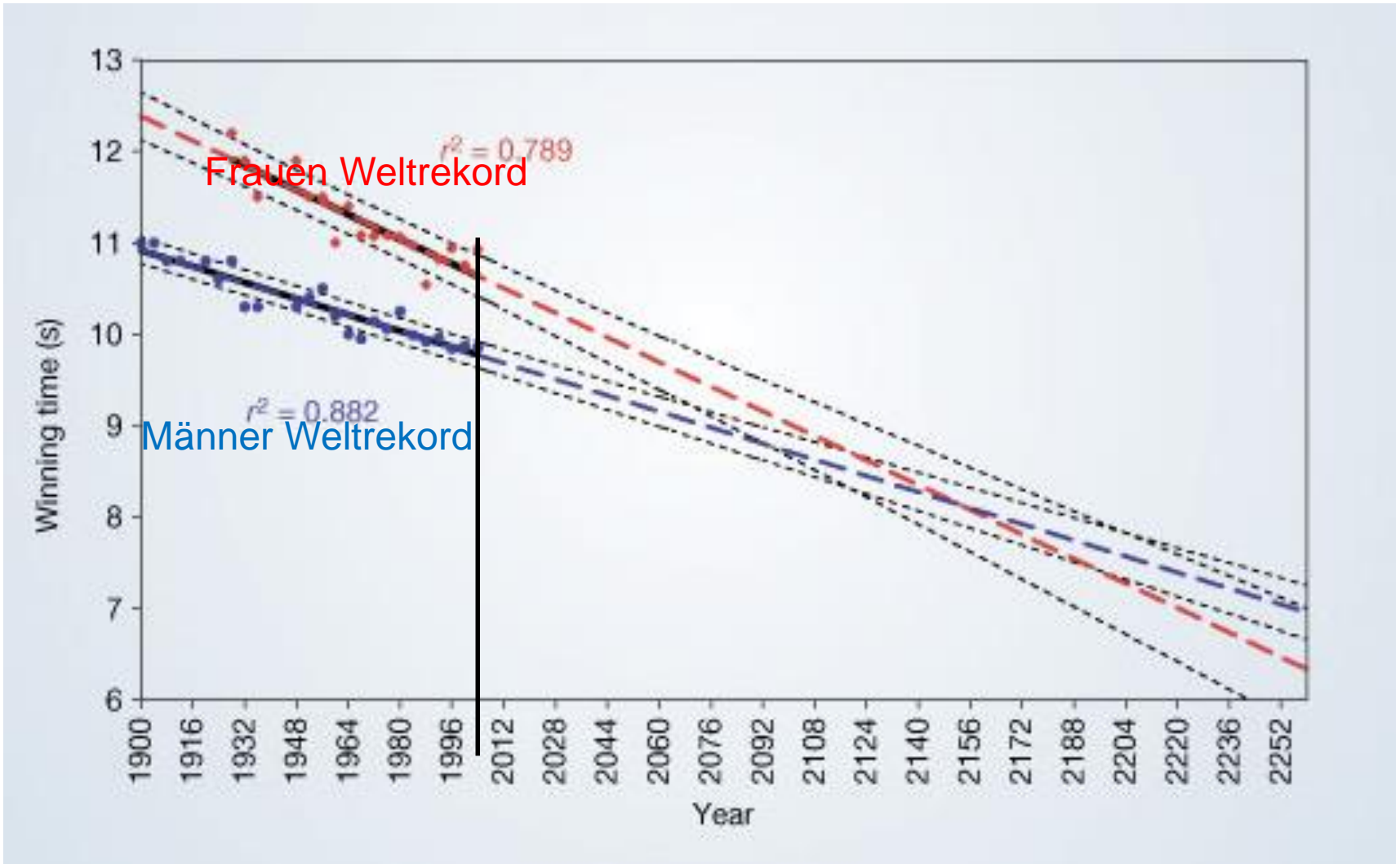
Gegeben (x_j, y_j, y'_j) , $j=0,1,\dots,n$

Gesucht stückweise kubisch, interpolierende Polynome

- Stückweise Bezierkurven, z.B. fasse für $n=3$ jeweils 4 Kontrollpunkte zusammen zur Bestimmung einer Teil-Bezierkurve. Hefte Teilkurven ‚glatt‘ aneinander.



Gefahren der Extrapolation:



Black Swan – Effekt und die Schwierigkeit von Vorhersagen.

Problem:

Gegeben Punkte (x_j, y_j) , Werte $z_j, j=0, \dots, N$

Gesucht Polynom in x und y mit $p(x_j, y_j) = z_j, j=0, \dots, N,$

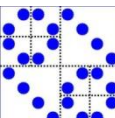
z.B.
$$p(x, y) = \sum_{r=0}^R \sum_{s=0}^S a_{r,s} x^r y^s \quad \text{mit} \quad (R+1)(S+1) = N+1$$

Ergibt $N+1 \times N+1$ lineares Gleichungssystem für die unbekanntenen Koeffizienten $a_{r,s}$.

Spezialfall ‚konstant‘: $R=S=0, N+1=1;$

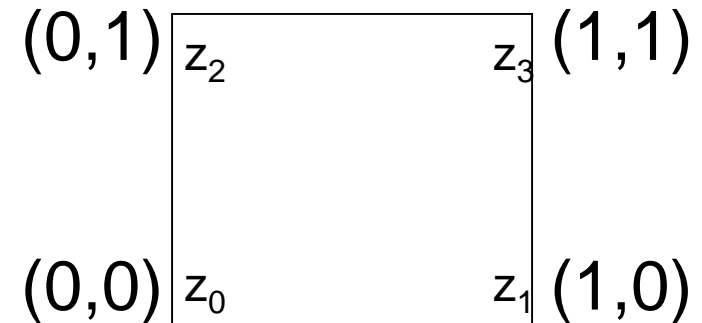
Eine Stützstelle (x_0, y_0) mit Wert z_0

Lösung: $p(x, y) \equiv z_0$



Spezialfall ‚bilinear‘: $R=S=1$, $N+1=4$:
 4 Stützstellen, z.B. $(0,0)$, $(1,0)$, $(0,1)$, $(1,1)$
 bilineares Polynom $p(x,y)=a+b \cdot x+c \cdot y+d \cdot xy$

Lösung der Interpolationsaufgabe :

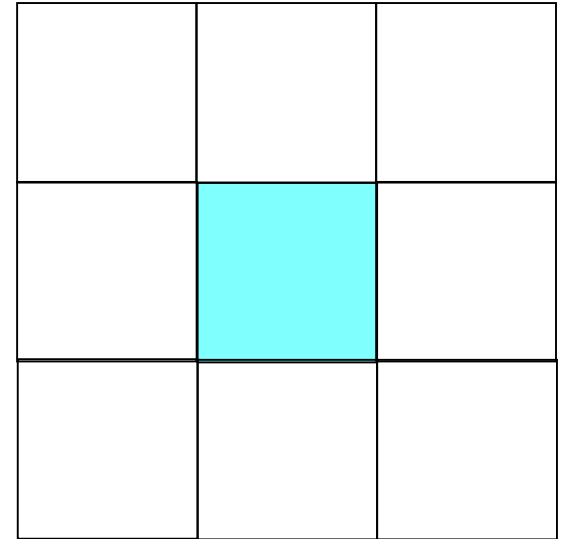


$$\begin{aligned}
 a &= z_0 \\
 a + b &= z_1 \\
 a + c &= z_2 \\
 a + b + c + d &= z_3
 \end{aligned}
 \Leftrightarrow
 \begin{pmatrix}
 1 & 0 & 0 & 0 \\
 1 & 1 & 0 & 0 \\
 1 & 0 & 1 & 0 \\
 1 & 1 & 1 & 1
 \end{pmatrix}
 \cdot
 \begin{pmatrix}
 a \\
 b \\
 c \\
 d
 \end{pmatrix}
 =
 \begin{pmatrix}
 z_0 \\
 z_1 \\
 z_2 \\
 z_3
 \end{pmatrix}$$

$$p(x, y) = z_0 + (z_1 - z_0)x + (z_2 - z_0)y + (z_3 + z_0 - z_1 - z_2)xy$$

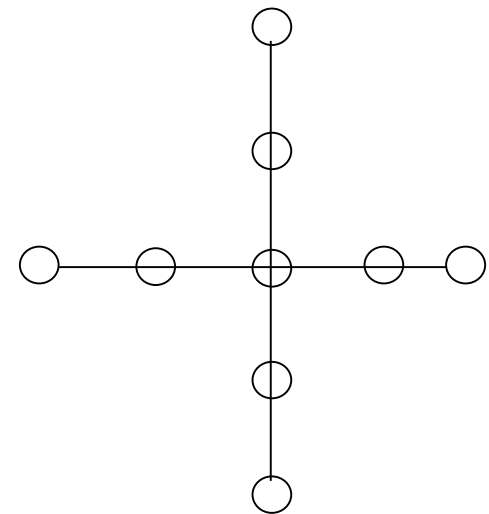
Spezialfall $R=S=3$, $N+1=16$,
16 Stützstellen:

bikubisch

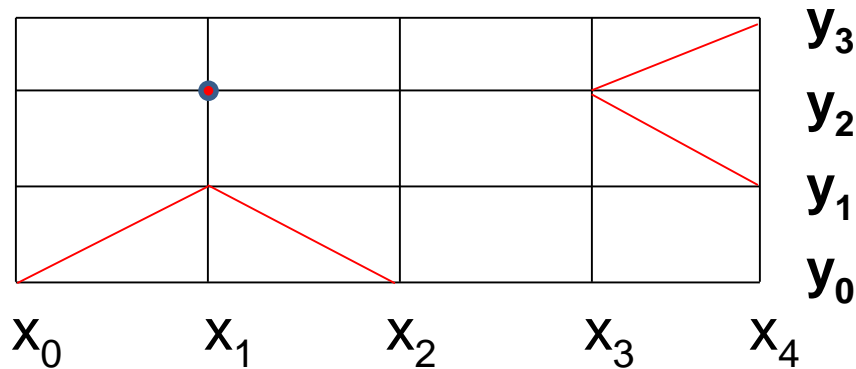


Spezialfall $R=S=2$, $N+1=9$,
9 Stützstellen:

biquadratisch



Spezielle Problemstellung: Funktion auf Rechteckgitter.
 (x_0, \dots, x_n) und (y_0, \dots, y_m) führen zu Stützstellen auf Gitter
 (x_j, y_k) , $j=0, \dots, n$ und $k=0, \dots, m$ mit Stützwerten $z_{j,k}$

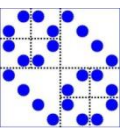


Lösung z.B. durch 1D-Lagrange-Polynome:

$$L_{j,k}(x, y) := L_{x_j}(x) \cdot L_{y_k}(y)$$

in der Form:

$$p(x, y) = \sum_{j=0}^n \sum_{k=0}^m z_{j,k} \cdot L_{j,k}(x, y)$$



Andere Problemstellung: Grad $r + s \leq n$ (Gesamtgrad $\leq n$)

$$p(x, y) = \sum_{r+s \leq n} a_{r,s} x^r y^s$$

z.B. $n=2$: $p(x, y) = a + bx + cy + dx^2 + exy + fy^2$

Hier ist der Gesamtgrad $r+s$ aller Terme $x^r y^s$ durch n beschränkt!

Bestimme Koeffizienten wieder aus linearem Gleichungssystem.

$n = 0$: $p(x, y) = a,$

$n = 1$: $p(x, y) = a + bx + cy$

$n = 2$: $p(x, y) = a + bx + cy + dx^2 + exy + fy^2$

$n = 3$: $p(x, y) = a + bx + cy + dx^2 + exy + fy^2 + gx^3 + hx^2y + ixy^2 + jy^3$

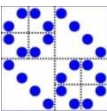
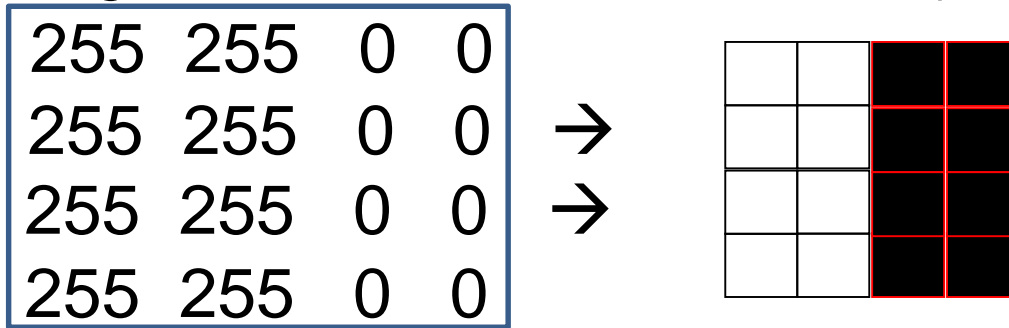
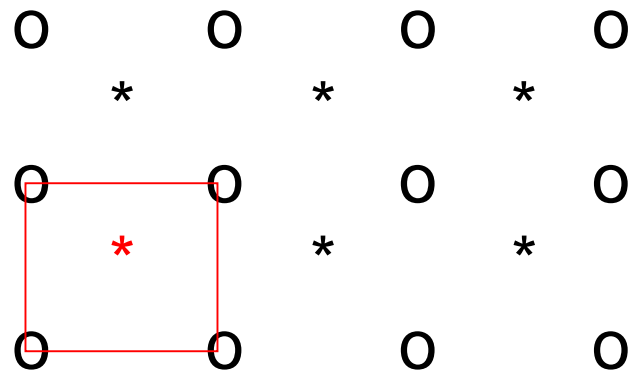


Bild als Matrix von Werten zwischen 0 und 255.

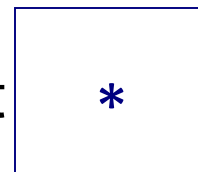
Jeder Wert gibt Grauwert an einer Stelle (einem Pixel) an.



(a) Umskalierung, Drehung, Verschiebung führt auf neues Gitter!
 Wie werden Pixelwerte des neuen Gitters bestimmt?



Bestimmung des neuen Wertes *
 z.B. durch bilineare Interpolation mit



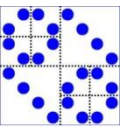
(b) Registrierung von Bilddaten:

Man verfügt über zwei verschiedene Aufnahmen desselben Objekts, z.B. zu verschiedenen Zeiten, von verschiedenen Aufnahmemedien, oder von verschiedenen Blickwinkeln aus aufgenommen. Z.B. Röntgen vs. Ultraschall.

Ziel ist es, diese Bilder (x, y) und (\tilde{x}, \tilde{y}) zur Deckung zu bringen. Z.B. eine neue Aufnahme mittels einer geeigneten Transformation (Verschiebung, Drehung) in das Referenzobjekt abzubilden.

Bestimme dazu a , b , und φ , so dass

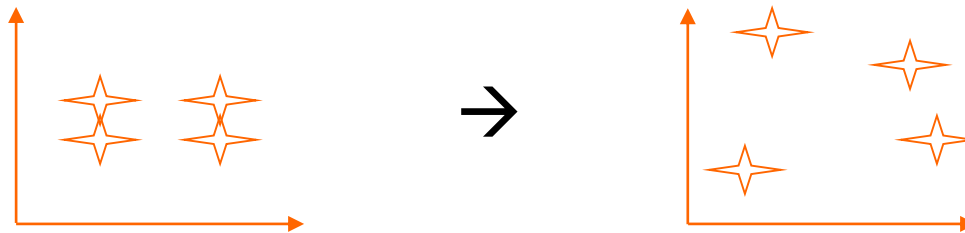
$$\min_{\varphi, a, b} \left\| \begin{pmatrix} \cos(\varphi) & -\sin(\varphi) \\ \sin(\varphi) & \cos(\varphi) \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} a \\ b \end{pmatrix} - \begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} \right\|$$



Wieder zwei verschiedene (aber deckungsgleiche) Aufnahmen desselben Objekts.

Nun ist aber eine der Aufnahmen verzerrt, z.B. im Falle von Röntgenaufnahmen durch Erdmagnetfeld, usw.

Wir nehmen an, dass wir Markierungspunkte in beiden Aufnahmen haben, die zur Übereinstimmung gebracht werden sollen, diesmal aber mit einer nichtlinearen Transformation.



Transformiere die Koordinaten der verzerrten Markierungspunkte möglichst gut auf die originalen Positionen,

z.B. durch zwei Polynome dritten Grades und Minimierung des Fehlers über alle Punkte:

(x_d, y_d) verzerrte Position von (x_p, y_p) :

Minimiere über alle Markierungspunkte den Abstand

$$\begin{pmatrix} x_p \\ y_p \end{pmatrix} - \begin{pmatrix} a_0 + a_1 x_d + a_2 y_d + a_3 x_d y_d + a_4 x_d^2 + a_5 y_d^2 + a_6 y_d x_d^2 + a_7 x_d y_d^2 + a_8 x_d^3 + a_9 y_d^3 \\ b_0 + b_1 x_d + b_2 y_d + b_3 x_d y_d + b_4 x_d^2 + b_5 y_d^2 + b_6 y_d x_d^2 + b_7 x_d y_d^2 + b_8 x_d^3 + b_9 y_d^3 \end{pmatrix}$$

Bikubischer Ansatz.

Betrachte z.B. $x_p = a_0 + a_1 x_{d,k} + \dots + a_9 y_{d,k}$ über alle Markierungspunkte:

ergibt Normalgleichungen zur Bestimmung der optimalen Koeffizienten a_0, \dots, a_9 .

Bestimme b_0, \dots, b_9 genauso.

10 Koeffizienten, also mindestens 10 Marker!

Integration \leftrightarrow *Flächenberechnung*

Problem:

$$I(f) := \int_a^b f(x) \, dx$$

In einigen Fällen kann das Integral mittels Stammfunktion berechnet werden:

Stammfunktion von x^k ist $x^{k+1}/(k+1)$;

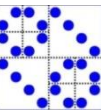
Daher
$$\int_a^b x^k \, dx = \left[\frac{x^{k+1}}{k+1} \right]_a^b = \frac{b^{k+1} - a^{k+1}}{k+1}$$

Stammfunktion von $\exp(x)$ ist $\exp(x)$, usw. (Formelsammlung)

In vielen Fällen existiert keine explizite Formel für die Stammfunktion!

Dann muss das Integral numerisch angenähert werden (vgl.

Riemann'sche Summen, Obersumme, Untersumme)

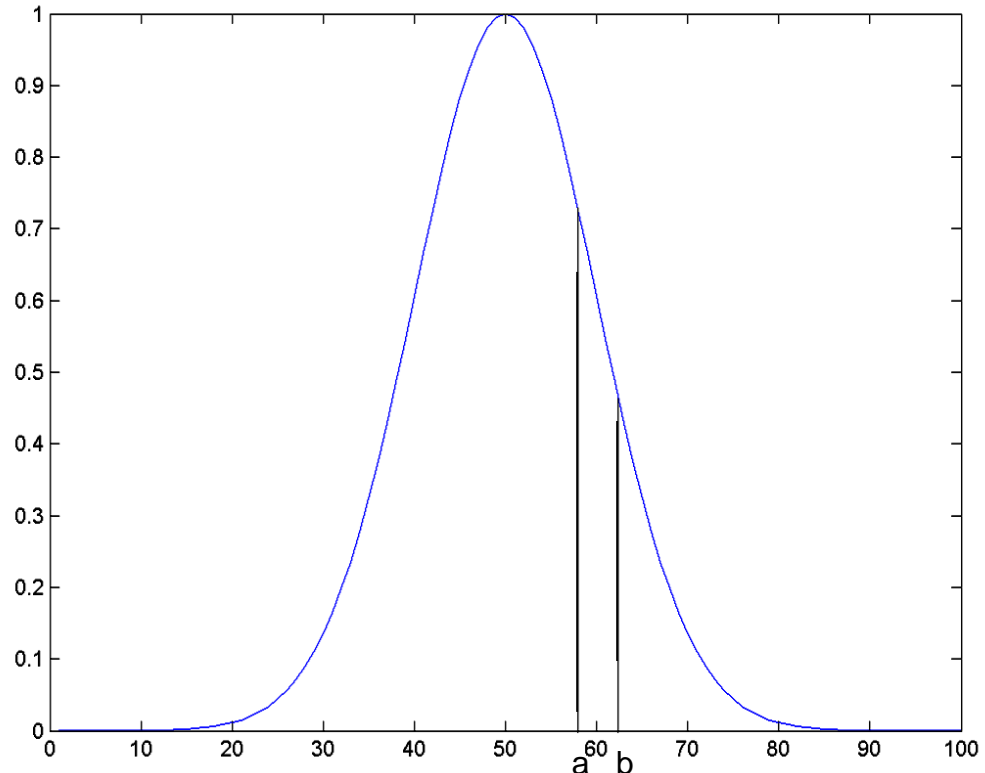


4.2.1. Beispiel aus der Wahrscheinlichkeitsrechnung:

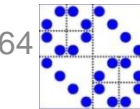
Gauss-Verteilung mit Mittelwert m und Standardabweichung σ ; gesucht ist die Wahrscheinlichkeit dafür, dass ein gemessener Wert zwischen a und b liegt:

$$\frac{1}{\sigma \sqrt{2\pi}} \cdot \int_a^b \exp\left(-\frac{(x-m)^2}{2\sigma^2}\right) dx$$

Entspricht der Fläche unter der Kurve:



$m=50$
 $\sigma=10$



4.2.2. Allgemeine Form numerischer Quadraturregeln zur näherungsweise Berechnung solcher Integrale:

$$I(f) = \int_a^b f(x) \, dx \approx \sum_{i=0}^n w_i f(x_i)$$

- Bestandteile:
- Stützstellen x_i mit Werten $f(x_i)$
 - Gewichte w_i

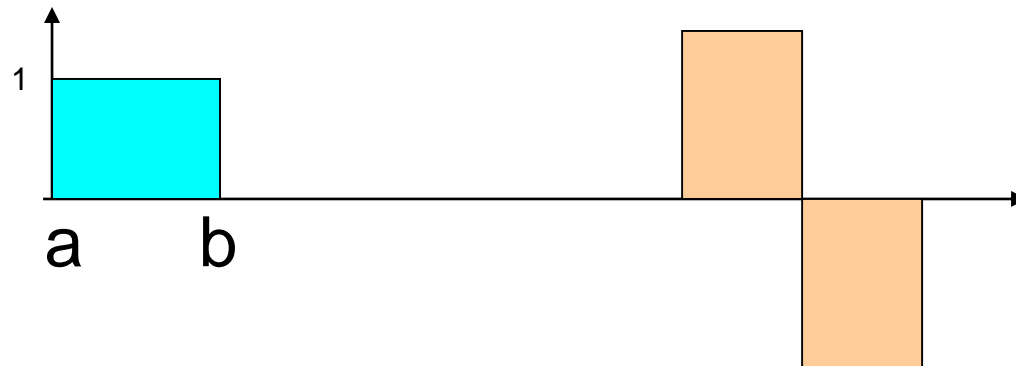
Frage: Wie sind die Stützstellen und Gewichte zu wählen, damit der Näherungswert möglichst gut ist.

Die Gewichte sollten dabei größer oder gleich 0 sein, um Rundungsfehler durch Auslöschung in der obigen Summe zu vermeiden!

Möglichst wenige Funktionsauswertungen für möglichst genaue Näherung!

4.2.3. Quadratur als Flächenberechnung

Das Integral zwischen a und b gibt den Flächeninhalt der Kurve an, die zwischen $f(x)$ und der x -Achse eingeschlossen ist.



Die blaue, linke Fläche entspricht $(b-a) \cdot 1$

Und die rosa, rechte Fläche entspricht dem Wert 0, da obere und untere Fläche sich weg heben.

Trapezregel (n=1):

$$I(f) = \int_a^b f(x)dx \approx \frac{f(a) + f(b)}{2} \cdot (b - a) =$$

$$= \frac{b-a}{2} f(a) + \frac{b-a}{2} f(b)$$

Stützstellen a,b; Gewichte $w_0 = w_1 = (b-a)/2$

