

Numerisches Programmieren, Übungen

5. Übungsblatt: Interpolation

1) Polynominterpolation mit Aitken-Neville

Zur Auswertung eines Interpolationspolynoms $p(x)$ zu den Punkten $P_i = (x_i, y_i)$, $i = 0, \dots, n$ an einer einzelnen Stelle x kann der Algorithmus von Aitken-Neville verwendet werden:

```

for i=0:n; p[i,0]:=y[i]; end
for k=1:n
    for i=0:n-k
        p[i,k] := p[i,k-1] + (x-x[i])/(x[i+k]-x[i])*(p[i+1,k-1] - p[i,k-1]);
    end
end
    
```

Die sukzessive Berechnung der $p[i, k]$ ist hier im Vergleich zur Vorlesungsfolie äquivalent umgeformt und kann mit einem Dreiecksschema veranschaulicht werden:

x_i	$i \setminus k$	0	1	2	...
x_0	0	$p[0,0] = y_0$	$\rightarrow p[0,1]$	$\rightarrow p[0,2]$	$\rightarrow \dots$
		\nearrow	\nearrow		
x_1	1	$p[1,0] = y_1$	$\rightarrow p[1,1]$	$\rightarrow \vdots$	
		\nearrow			
x_2	2	$p[2,0] = y_2$	$\rightarrow \vdots$		
\vdots	\vdots	\vdots			

Der Wert des Polynoms $p(x)$ an der Stelle x steht nach Abschluss des Algorithmus' in $p[0, n]$.

Berechnen Sie den Wert des kubischen Interpolationspolynoms $p(x)$ an der Stelle $x = 1$ für die Punkte $P_0 = (-2, -7)$, $P_1 = (-1, 0)$, $P_2 = (0, 1)$ und $P_3 = (2, 9)$ mit dem Aitken-Neville-Algorithmus! Stellen Sie dabei auch das Dreiecksschema auf.

Wann ist die Berechnung mit Aitken-Neville vorteilhaft und wann nicht?

2) Polynominterpolation nach Newton

Neben der Lagrangeschen Darstellungsweise bieten auch die Newtonschen dividierten Differenzen die Möglichkeit, ein Interpolationspolynom $p(x)$ analytisch zu beschreiben (d.h. seine Koeffizienten zu berechnen). Mit zu interpolierenden Punkten $P_i = (x_i, y_i)$ gelten die folgenden Formeln aus der Vorlesung:

$$[x_i]f = y_i \quad (1)$$

$$[x_i, x_{i+1}]f = \frac{y_{i+1} - y_i}{x_{i+1} - x_i} \quad (2)$$

$$[x_i, \dots, x_{i+k}]f = \frac{[x_{i+1}, \dots, x_{i+k}]f - [x_i, \dots, x_{i+k-1}]f}{x_{i+k} - x_i}. \quad (3)$$

Man ordnet die dividierten Differenzen in ein Neville-artiges Schema:

x_i	$i \setminus k$	0	1	2	...
x_0	0	$[x_0]f$	$\rightarrow [x_0, x_1]f$	$\rightarrow [x_0, x_1, x_2]f$	$\rightarrow \dots$
			\nearrow	\nearrow	
x_1	1	$[x_1]f$	$\rightarrow [x_1, x_2]f$	$\rightarrow \vdots$	
			\nearrow		
x_2	2	$[x_2]f$	$\rightarrow \vdots$		
\vdots	\vdots	\vdots			

Damit kann man in der ersten Zeile direkt die Koeffizienten $[x_0, \dots, x_{i+k}]f$ des Interpolationspolynoms $p(x)$ in der folgenden Gestalt ablesen:

$$p(x) = [x_0]f + [x_0, x_1]f \cdot (x - x_0) + \dots + [x_0, \dots, x_n]f \cdot \prod_{i=0}^{n-1} (x - x_i). \quad (4)$$

- i) Berechnen Sie die Newtonschen dividierten Differenzen für das Interpolationspolynom $p(x)$ durch die Punkte $P_0 = (-1, -3)$, $P_1 = (1, 1)$ und $P_2 = (3, -3)$ mit den Gleichungen (1) - (3). Stellen Sie dazu auch das Neville-artige Schema auf und berechnen Sie $p(x)$ mit Hilfe der Formel (4)!
- ii) Nun soll ein zusätzlicher Punkt $P_3 = (0, 0)$ zur Interpolation hinzugenommen werden. Berechnen Sie die noch fehlenden dividierten Differenzen, erweitern Sie das Neville-artige Schema aus Teilaufgabe i) und geben Sie die Koeffizienten des neuen Gesamtpolynoms $q(x)$ nach Formel (4) an!
- iii) Betrachten Sie nun eine Funktion $f \in C^{n+1}$, für die das in (ii) berechnete Polynom $p(x)$ ein zugehöriger Interpolant ist, d.h. $P_i = (x_i, f(x_i))$. Geben Sie eine Formel zur Abschätzung des Interpolationsfehlers an der Stelle $x = 2$ an.

3) Bezier-Kurven und Computergrafik

Bezier-Kurven gehören im engeren Sinne nicht zu den Interpolations-Methoden, weil nicht alle Kontrollpunkte interpoliert werden. Aufgrund ihrer Bedeutung bei der Modellierung und beim Design von realen Gegenständen (siehe Abb. 1) finden sie sich aber in den meisten CAD-Programmen und sollen hier in Grundzügen behandelt werden.

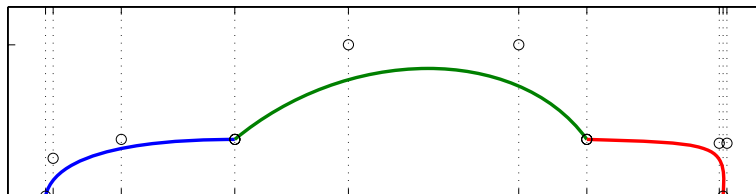


Abbildung 1: Einfache Fahrzeug-Silhouette, modelliert mit drei Bezier-Kurven zu je vier Kontrollpunkten.

In dieser Aufgabe werden lediglich zweidimensionale Bezier-Kurven mit vier Kontrollpunkten betrachtet. Man spricht nicht wie bisher von Stützpunkten, weil nicht mehr alle Kontrollpunkte von der resultierenden Kurve durchlaufen werden.

Eine Bezier-Kurve $s(t)$ ist definiert durch

$$s : [0; 1] \rightarrow \mathbb{R}^2, \quad s(t) = \begin{pmatrix} s_x(t) \\ s_y(t) \end{pmatrix}.$$

Die beiden Komponenten $s_x(t)$, $s_y(t)$ werden mit Hilfe der skalaren Bernstein-Polynome

$$B_i : [0; 1] \rightarrow \mathbb{R}, \quad B_i(t) = \binom{3}{i} t^i (1-t)^{3-i}, \quad i = 0, \dots, 3 \quad (5)$$

als kubische Polynome in t gewählt. Dabei bezeichnet $\binom{3}{i} = \frac{3!}{i!(3-i)!}$ den Binomialkoeffizient und keinen 2d-Vektor.

Zusammen mit den Kontrollpunkten $\mathbf{b}_i = (x_i, y_i)^T \in \mathbb{R}^2$ ($i = 0, \dots, 3$) erhält man somit eine Beschreibung der Gesamtkurve $s(t)$:

$$s(t) := \sum_{i=0}^3 \mathbf{b}_i \cdot B_i(t) \quad \text{mit} \quad t \in [0, 1]. \quad (6)$$

- i) Berechnen Sie die $B_i(t)$ ($i = 0, \dots, 3$) explizit mit Hilfe von Glg. (5).
- ii) Zeigen Sie, dass folgende Identität gilt:

$$\sum_{i=0}^3 B_i(t) = 1 \quad \forall t \in [0, 1].$$

Da auch $B_i(t) \geq 0$ gilt, bedeutet das, dass die Bezier-Kurve stets innerhalb der konvexen Hülle (hier ein Viereck) ihrer Kontrollpunkte verläuft.

iii) Zeigen Sie, dass die Bezier-Kurve $s(t)$ stets die Kontrollpunkte \mathbf{b}_0 bzw. \mathbf{b}_3 für $t = 0$ bzw. $t = 1$ interpoliert!

Zeigen Sie ferner, dass die Kontrollpunkte \mathbf{b}_1 bzw. \mathbf{b}_2 stets auf der Tangentengeraden τ an die Bezier-Kurve im Punkt \mathbf{b}_0 bzw. \mathbf{b}_3 liegen, indem Sie folgende Darstellung von τ verwenden:

$$\tau(\alpha)_{\mathbf{b}_0} = \mathbf{b}_0 + \alpha \cdot \left. \frac{ds(t)}{dt} \right|_{t=0}, \quad \alpha \in \mathbb{R}$$

$$\tau(\alpha)_{\mathbf{b}_3} = \mathbf{b}_3 + \alpha \cdot \left. \frac{ds(t)}{dt} \right|_{t=1}, \quad \alpha \in \mathbb{R}.$$

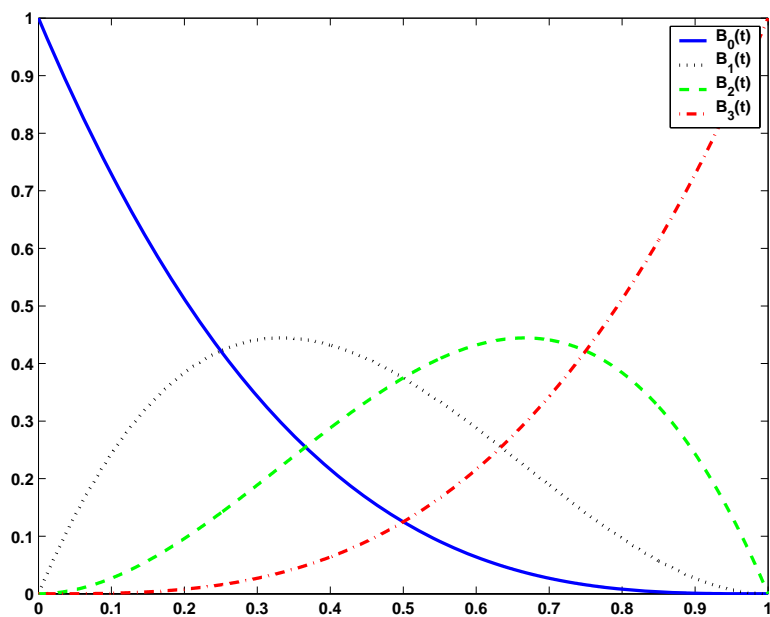


Abbildung 2: Visualisierung der Bernsteinpolynome $B_0(t), \dots, B_3(t)$, $t \in [0; 1]$.