Prof. Dr. Karsten Urban
Dipl.-Math. oec. Klaus Stolle
Institut für Numerische Mathematik
Universität Ulm

WiSe 2017/2018
Blatt 02
26.10.2017

Angewandte Numerik 2

Besprechung in den Tutorien in der Woche vom 06.11.2017 bis 10.11.2017

Für dieses Übungsblatt gibt es 27 Theorie- und 16 Matlab-Punkte, sowie 4 Theorie- und 8 Matlab-Zusatzpunkte. Punkte, die mit einem * gekennzeichnet sind, sind Zusatzpunkte.
Die 60-Prozent-Grenzen liegen aktuell (inklusive Blatt 02) bei 25,8 Theoriepunkten und 19,2 Matlabpunkten.

Aufgabe 5 *(Bernstein-Polynome)* (1T+2T+2T+2T* Punkte)

Bestimmen Sie

a) das konstante Bernstein-Polynom (also das Bernstein-Polynom vom Grad 0)

sowie mit Hilfe der Rekursionsformel

b) die linearen Bernstein-Polynome (also die Bernstein-Polynome vom Grad 1),

b) die quadratischen Bernstein-Polynome und

b) die kubischen Bernstein-Polynome.

Aufgabe 6 *(Programmieraufgabe: Darstellung der Bernstein-Polynome)* (8M* Punkte)

Schreiben Sie ein Matlabskript `bernsteinPolynome`, mit dem Sie die Bernstein-Polynome vom Grad 0 bis zum Grad n plotten können. Wählen Sie eine effiziente Implementierung. Testen Sie Ihr Matlabskript mit $n = 8$.

Aufgabe 7 *(Programmieraufgabe: Bézier-Kurven im $\mathbb{R}^2$)* (5M+5M+5M+1M+4T+2T Punkte)

a) Schreiben Sie eine Matlabfunktion `C = bezier(P,t)`, die die Bézier-Kurve zu den Kontrollpunkten `P` an den Stellen `t` auswertet. `P` ist dabei der Vektor $(\boldsymbol{P}_0, \dots, \boldsymbol{P}_n)$ der $n+1$ Kontrollpunkte $\boldsymbol{P}_j \in \mathbb{R}^2$, also eine Matrix $\in \mathbb{R}^{2\times(n+1)}$. `t` ist ein Vektor $(t_1, \dots, t_m)$. Ihre Funktion soll die Werte $\boldsymbol{C}(t_j) \in \mathbb{R}^2$, $j = 1, \dots, m$ der Bézier-Kurve in der Matrix `C` $= (\boldsymbol{C}(t_1), \dots, \boldsymbol{C}(t_m)) \in \mathbb{R}^{2\times m}$ zurück geben.

b) Schreiben Sie ein Matlabskript `bezierKurve`, bei dessen Aufruf der Benutzer aufgefordert wird, die Kontrollpunkte $\boldsymbol{P}_0, \dots, \boldsymbol{P}_n$ zu markieren. Bei der Eingabe soll Ihr Skript das zugehörige Kontrollpolygon zeichnen. Nach Eingabeende soll die Bézier-Kurve zu den eingegebenen Kontrollpunkten geplottet werden.

   Hinweis: Sie können sich beim Einlesen der Kontrollpunkte am Matlabskript `drawSpline` von Aufgabe 4 orientieren.

c) Schreiben Sie eine Matlabfunktion `C0 = deCasteljauVisual(P,t0)`, die den Punkt `C0` der Bézier-Kurve zu den Kontrollpunkten `P` (wie in Aufgabenteil a)) an der Stelle `t0` $\in [0,1]$ mit dem De-Casteljau-Algorithmus berechnet. Ihre Funktion soll den De-Casteljau-Algorithmus verdeutlichen und dazu die Polygonzüge durch die Punkte $\boldsymbol{P}_{k,j}(t_0)$ mit $k = 1,\dots,n$ und $j = 0,\dots,n-k$ zeichnen sowie die Ecken $\boldsymbol{P}_{k,j}(t_0)$ durch kleine Kreise markieren.

d) Ergänzen Sie Ihr Matlabskript `bezierKurve`, so dass Sie Ihre Matlabfunktion `deCasteljauVisual` testen können.

e) Erläutern Sie, wie man mit Hilfe des De-Casteljau-Algorithmus eine Bézier-Kurve berechnen kann. Wählen Sie für Ihre Erläuterungen ein einfaches Beispiel mit 4 Kontrollpunkten $P_j \in \mathbb{R}^2$, $(j = 0, ..., 3)$.

f) Haben Sie Ihre Matlabfunktion `bezier` aus Aufgabenteil a) effizient implementiert? Begründen Sie Ihre Antwort.

Aufgabe 8 *(Interpolation mit quadratischen B-Splines)* (5T+3T+3T+2T*+5T Punkte)

Gegeben seien die folgenden Messwerte

x_i	0	1	2	3
y_i	2	3	3	5

Interpolieren Sie diese durch einen quadratischen B-Spline:

a) Berechnen Sie hierzu zunächst alle B-Spline-Basisfunktionen $N_j^p(t)$ vom Grad $p \in 0,1,2$.

Hinweis: Konstruieren Sie sich eine Knotenfolge $\mathcal{T} = \{t_0,\dots,t_7\}$, in der die Werte $x_0 = 0$ und $x_3 = 3$ jeweils dreifach vorkommen.

b) Berechnen Sie aus den Interpolationsbedingungen $s(x_i) = y_i$ ein (unterbestimmtes) lineares Gleichungssystem zur Bestimmung der Koeffizienten c_j des quadratischen B-Splines $s(x) = \sum_{j=0}^4 c_j N_j^2(x)$.

c) Bestimmen Sie den frei wählbaren Parameter aus einer zusätzlichen Bedingung, beispielsweise aus der Randbedingung $s'(0) = 0$, und geben Sie die Koeffizienten c_j des daraus resultierenden quadratischen B-Splines $s(x) = \sum_{j=0}^4 c_j N_j^2(x)$ an.

d) Geben Sie den interpolierenden quadratischen B-Spline s explizit an.

e) Werten Sie den B-Spline s an der Stelle $x = \frac{1}{2}$ effizient von Hand aus.

Hinweise: Verwenden Sie die Knotenfolge $\mathcal{T} = \{t_0,\dots,t_7\}$ aus Aufgabenteil a). Zur effizienten Auswertung brauchen Sie die B-Spline-Basisfunktionen $N_j^2(t)$ nicht zu kennen.

Hinweise:

Die Programmieraufgaben sind in Matlab zu erstellen. Der Source Code muss strukturiert und dokumentiert sein. Senden Sie **spätestens 24 Stunden vor Ihrem Tutorium** alle Matlab-Files und alle Ergebnisse in einer E-mail mit dem Betreff **Loesung-Blatt02** an **angewandte.numerik@uni-ulm.de**.