Übung 1 *Kubischer Spline*

Für eine Zerlegung $X = (x_0, ..., x_n)$ des Intervalls $[a, b]$ mit $a = x_0 < x_1 < ... < x_n = b$ ist der Raum der kubischen Splines

$$S^3(X) = \{s \in \mathcal{C}^2([a,b]) : s|_{[x_i,x_{i+1}]} \in \mathbb{P}_3, 0 \leq i < n\}$$

gegeben durch abschnittsweise definierte Polynome dritten Grades, die sich über Interpolations- und Stetigkeitsbedingungen zu einer zweimal stetig differenzierbaren Funktion zusammenfügen:

$$s(x) = \begin{cases} p_i(x) & \text{für } x \in [x_{i-1}, x_i), \quad i \in \{1, ..., n\} \\ p_n(x_n) & \text{für } x = x_n \end{cases}$$

Die n kubischen Teilpolynome seien gegeben durch

$$p_i(x) = a_0^{(i)} + a_1^{(i)}(x - x_i) + a_2^{(i)}(x - x_i)^2 + a_3^{(i)}(x - x_i)^3 \qquad i = 1, ..., n \tag{11.1}$$

Weiterhin seien folgende Bedingungen vorgegebenen:

(a) s ist stetig und interpoliert die Werte $y_0, y_1, ..., y_n$ zu den Stützstellen aus X:

$$i = 1, ..., n : \begin{array}{rl} p_i(x_{i-1}) & = y_{i-1} \\ p_i(x_i) & = y_i \end{array}$$

(b) $s \in \mathcal{C}^2([a, b])$:

$$i = 1, ..., n-1 : \begin{array}{rl} p_i'(x_i) & = p_{i+1}'(x_i) \\ p_i''(x_i) & = p_{i+1}''(x_i) \end{array}$$

(c) Natürliche Randbedingungen an den beiden Enden:

$$p_1''(x_0) = 0 = p_n''(x_n)$$

Zeige:
Die gesuchten Koeffizienten der Teilpolynome lassen sich berechnen aus den folgenden Formeln, wenn man $a_2^{(0)} := 0$ setzt[1]:

$$h_i a_2^{(i-1)} + 2(h_i + h_{i+1})a_2^{(i)} + h_{i+1}a_2^{(i+1)} = 3\left(\frac{y_{i+1} - y_i}{h_{i+1}} - \frac{y_i - y_{i-1}}{h_i}\right) \text{ für } i = 1, ..., n-1, \tag{11.2}$$

wobei $a_2^{(n)} = 0$ und $h_i := x_i - x_{i-1}$,

$$a_0^{(i)} = y_i, \tag{11.3}$$

$$a_1^{(i)} = \frac{y_i - y_{i-1}}{h_i} + \frac{h_i}{3}\left(2a_2^{(i)} + a_2^{(i-1)}\right), \tag{11.4}$$

$$a_3^{(i)} = \frac{a_2^{(i)} - a_2^{(i-1)}}{3h_i}. \tag{11.5}$$

(6 Punkte)

[1] *Die Variable $a_2^{(0)}$ gehört gar nicht zu den Polynomkoeffizienten, sondern wird nur für die Darstellung der Gleichungen (11.2)-(11.5) benötigt.*

Übung 2 *Kubischer Spline - Praktische[2] Übung*

(a) Schreibe eine C++-Funktion `getCubicSpline(...)`, welche ausgehend von den Wertepaaren $(x_i, y_i), i = 0, ..., n$ die Koeffizienten $a_k^{(i)}, k = 0, ..., 3, i = 0, ..., n$ des natürlichen kubischen Splines wie oben beschrieben berechnet. Für die Lösung des linearen $(n-1) \times (n-1)$-Tridiagonalsystems (11.2) können die bereits vorhandenen Funktionen zur LR-Zerlegung verwendet werden.
`Input-Parameter:` x_i, y_i `Output-Parameter:` $a_k^{(i)}$

(b) Schreibe eine C++-Funktion `evaluateCubicSpline(...)`, die den oben berechneten kubischen Spline an einer beliebigen Stelle $x \in [x_0, x_n]$ auswertet.
`Input-Parameter:` $x, x_0, x_n, a_k^{(i)}$ `Output-Parameter:` $s(x)$

(c) Teste die beiden Funktionen an folgendem höchst realistischen Beispiel:

Auf der schwäbischen Alb wurden die Überreste von Versteinerungen gefunden, die zu einem Saurierskelett gehören. Um die Umrisse des Sauriers rekonstruieren zu können, wird das 14 mal 5 Meter große Fundgebiet mit einem Raster überzogen und jedes Fundstück Dinosaurierhaut in diesem Koordinatensystem lokalisiert. Nach einer logischen Reihung ergibt sich folgende Tabelle von Fundkoordinaten (x_i, y_i) für die von 0 bis 12 durchnummerierten Fundstücke:

Nr.	0	1	2	3	4	5	6	7	8	9	10	11	12
x_i	3.75	3.75	2.25	1.25	0.25	0.75	4.00	5.50	6.25	9.00	12.50	12.75	12.25
y_i	0.25	1.50	3.25	4.25	4.65	4.83	2.50	3.25	3.65	3.00	1.25	2.15	3.50

Zur Rekonstruktion des Sauriers gehe folgendermaßen vor: Die Gestalt des Sauriers wird durch eine Kurve

$$t \longmapsto \begin{pmatrix} \phi(t) \\ \psi(t) \end{pmatrix} \in \mathbb{R}^2, \quad t \in [0, T]$$

beschrieben, wobei die Funktionen ϕ und ψ durch je einen Spline genähert werden. Da das Durchlaufen der Kurve „mit konstanter Geschwindigkeit" geschehen soll, läßt man die Bogenlänge der Kurve einfließen, indem man als Stützstellen die Werte

$$t_0 := 0, t_i := t_{i-1} + \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2} \quad \text{für } i = 1, \ldots, 12 \tag{11.6}$$

benutzt. Berechne einen Spline s_ϕ zu den Stützstellen t_i und den Stützwerten x_i sowie einen weiteren Spline s_ψ zu den Stützstellen t_i und den Stützwerten y_i. Die genäherte Gestalt des Sauriers ergibt sich dann aus der Kurve

$$t \longmapsto \begin{pmatrix} s_\phi(t) \\ s_\psi(t) \end{pmatrix} \in \mathbb{R}^2, \quad t \in [t_0, t_{12}]$$

Werte diese Kurve an den Stellen $t = \xi_j := \frac{t_{12}}{100} \cdot j$ für $j = 0, 1, ..., 100$ aus, und fertige eine (grobe) Skizze des Sauriers an, z.B. mit gnuplot.

Um den Spline in `gnuplot` *zu visualisieren, schreibe die Ergebnisse in eine Datei* `result.dat`, *bei der in jeder Zeile ein Wertepaar steht:*

```
plot "result.dat" using 1:2 with lines
```

[2]*Immer dran denken: Vektorenindizes in C++ laufen von 0 bis n-1.*

trägt die Werte der zweiten Spalte gegen die der ersten Spalte auf und verbindet sie durch gerade Strecken. Erste Expertisen vermuten einen weiblichen Brontosaurus mittleren Alters.

(8 Punkte)

Übung 3 *Extrapolation zur Verbesserung der numerischen Differenziation*

Es gilt für die erste Ableitung $f'(x) = \lim_{h\to 0} a(h)$, wobei $a(h) = \frac{f(x+h)-f(x)}{h}$ ist. Nun ist aber $a(0)$ als Näherung für $f'(x)$ nicht direkt auswertbar und daher bietet sich hier die *Extrapolation zum Limes* für den Grenzfall $h \to 0$ an. In der Vorlesung wurde mit dieser Technik gezeigt, dass man mit einer absteigenden Folge von $n+1$ Stützstellen $h_0 > h_1 > ... > h_n > 0$ eine Näherung für $a(0)$ erzielen kann, deren Fehler von der Größenordnung $O(h^{n+1})$ ist, wenn $f \in \mathcal{C}^{n+2}(\mathbb{R})$ ist.

Zeige mittels Taylorreihen-Entwicklung, daß man mit der gleichen Anzahl an Stützstellen für die Approximation der 2.Ableitung durch den zentralen Differenzenquotienten

$$a(h) = \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}$$

die Fehlerordnung $O(h^{2n+2})$ erzielen kann, wenn $f \in \mathcal{C}^{2n+4}(\mathbb{R})$ ist. **(4 Punkte)**