Übungsblatt 2: Approximation von Funktionen

Aufgabe 1: Abschnittsweise quadratische Basisfunktionen

Es lassen sich auch Basisfunktionen definieren, die Abschnittsweise quadratisch sind. Hierzu wird jeweils noch ein Knoten in der Mitte jedes Elements angeordnet. Die Eigenschaft

$$\varphi_i(x_j) = \begin{cases} 1 & \text{falls} \quad i = j \\ 0 & \text{sonst} \end{cases}$$

soll weiterhin erfüllt sein.

Aufgaben:

1. Skizzieren Sie die Basisfunktionen für $N_e = 3$ Elemente gleicher Länge.

2. Wie verläuft die Funktion $u_h = \sum \varphi_i \hat{u}_i$, wenn der Vektor der Knotenwerte mit

$$\hat{\mathbf{u}} = (2, 1.5, 1, 2, 1, 1, 2)$$

   vorgegeben wird.

3. Was können Sie hier über die Differenzierbarkeit von u_h sagen?

Tipp: Schauen Sie sich nochmal die Lagrange-Polynome an. Diese Aufgabe erfordert keinerlei Rechnungen, Skizze reicht!

Aufgabe 2: Berechnung des Approximationsfehlers

Häufig möchte man wissen, wie gut die Approximation einer Funktion $u : [0, l] \to \mathbb{R}$ durch eine Näherung u_h ist. Wenn man die Funktion u kennt, dann besteht eine einfache Möglichkeit darin, den Approximationsfehler als größte Abweichung zwischen den beiden Funktionen zu definieren:

$$e = \max_{0 \leq x \leq l} |u(x) - u_h(x)|.$$

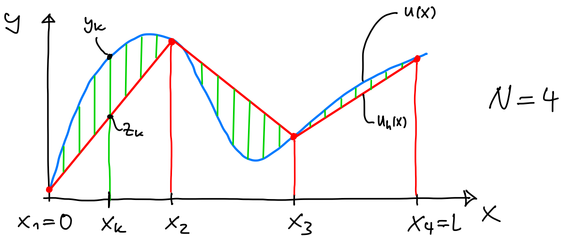
Diesen Fehler kann näherungsweise dadurch bestimmt werden, dass die maximale Differenz für eine sehr feine Unterteilung mit M Punkten berechnet wird.

Wir betrachten nun für eine gegebene Funktion $u : [0, l] \to \mathbb{R}$ eine spezielle Näherungslösung. Diese entsteht dadurch, dass wir das Intervall $[0, l]$ in $N - 1$ gleichlange Abschnitte unterteilen und dadurch den Knotenvektor $\mathbf{x}_h = (x_1, x_2, \ldots, x_N)$ erhalten. Die Funktion u

wird nun durch

$$u_h(x) = \sum_{i=1}^{N} \varphi_i(x) \cdot u(x_i)$$

angenähert, wobei φ_i unsere abschnittsweise linearen Basisfunktionen sind. Die Skizze erläutert das Vorgehen.



$$e = \max_{0 \leq x \leq L} |u(x) - u_h(x)| \approx \max_{1 \leq k \leq M} |y_k - z_k|$$

$$y_k = u(x_k), \quad z_k = u_h(x_k), \quad x_k = k \cdot \frac{L}{M-1}, \quad M = 100 \cdot (N-1)+1$$

Bearbeiten Sie hierzu folgende Punkte:

1. Wir betrachten die Funktion $u(x) = \sin(x)$ für den Bereich $0 \leq x \leq \pi$. Skizzieren Sie u und u_h für $N = 4$. Wie groß ist dann der Fehler e?

2. Programmieren Sie eine Funktion `compute_error` (in Matlab oder Python) mit der Sie den Fehler e für eine gegebenen Funktion u und eine abschnittsweise lineare Approximation u_h näherungsweise berechnen können. Verwenden Sie folgende Eingabeparameter:

`u`	-	Die Funktion u
`xh`	-	Der Vektor der Knotenpunkte $\mathbf{x}_h$
`uHat`	-	Die Koeffizienten $\hat{u}_i$

   Testen Sie die Funktion für das Beispiel aus Punkt 1.

   Tipp: Verwenden Sie zur Berechnung der Werte z_k die Funktion `interp1` (Matlab) oder `np.interp` (Python). Laden Sie eine der beiden Datei `aufgabe_2.mlx` bzw. `aufgabe_2.ipynb` herunter um die Aufgabe darin zu lösen.

3. Berechnen Sie für das Beispiel aus Punkt 1 den Fehler e_k mit $N_k = 2^k$ für $k = 1, \ldots, 10$. Plotten Sie den Fehler über die Länge der Elemente $h_k = \pi/(N_k - 1)$ sowohl in einer normalen als auch in einer doppelt-logarithmischen Skala.

4. Zusatzaufgabe: Wie lautet die Funktionsgleichung zum Graphen von $f(x) = C \cdot x^n$ in einer doppelt-logarithmischen Skala?