Abteilung für
Angewandte Mathematik

Sommersemester 2020
Prof. Dr. Sören Bartels, Pascal Weyer

Praktikum zur Numerik Teil 2

Projektblatt Blockkurs – 16. und 18. September 2020

Abgabe: per E-Mail an den Tutor bis Freitag, den 25. September 2020, 20:00 Uhr

Für den Erhalt der Studienleistung müssen Sie 20 Punkte erreichen und ein Projekt im Wert von acht Punkten, d.h. eines der Projekte BK3, BK4 oder BK7, bei einem Praktikumstermin präsentieren.

Bitte erläutern Sie bei allen Aufgaben Ihre Beobachtungen in einigen Sätzen.

Projekt BK1 (4 Punkte)**.** Warm-up.
Wir betrachten die numerische Bestimmung der Eulerschen Zahl e, die sich durch die Grenzwerte

$$e = \lim_{n\to\infty} (1+1/n)^n, \quad e = \lim_{n\to\infty} \sum_{k=0}^{n} \frac{1}{k!}$$

charakterisieren lässt. Verwenden Sie ausschließlich arithmetische Grundoperationen und endliche Approximationen der obigen Grenzwerte mit $n = 10^j$, $j = 1, 2, \dots, 10$, um e zu approximieren. Bestimmen Sie jeweils die Approximationsfehler mit Hilfe der Annäherung $e \approx 2{,}718281828459045$ und stellen Sie diesen mit 15 Nachkommastellen in einer Tabelle dar.

Projekt BK2 (4 Punkte)**.** Polynom-Interpolation.
Schreiben Sie ein `Matlab`-Programm zur Bestimmung der Koeffizienten eines Interpolationspolynoms bezüglich der Newton-Basis für gegebene Stützstellen $x_0 < x_1 < \cdots < x_n$ und zugehörige -werte $y_0, \dots, y_n$.
Testen Sie Ihr Programm für die Funktionen $f(x) = \sin(\pi x)$, $g(x) = (1+25x^2)^{-1}$ und $h(x) = |x|$ im Intervall $[-1, 1]$ bei Verwendung von äquidistanten Stützstellen und Tschebyscheff-Knoten. Werten Sie die Interpolationspolynome an den Punkten $z_j = -1 + 2j/100$, $j = 0, 1, \dots, 100$ mit dem Horner-Schema aus und plotten Sie damit die Interpolationspolynome für $n = 1, 2, 4, 8$. Es bietet sich hier an, für jede Funktion eine `figure` zu erstellen, die vier Koordinatensysteme enthält – eines je Wert von n. Sie können dies durch den `subplot(4,1,i)`-Befehl erreichen, wobei `i` das aktuelle Koordinatensystem angibt.

Projekt BK3 (4+4 Punkte)**.** Spline-Interpolation.

(1) Der `Matlab`-Befehl `plot(X,Y)` stellt einen durch die Vektoren X und Y definierten Polygonzug grafisch dar. Ist $X = [x_0, x_1, \dots, x_n]^\top$ und $Y = [f(x_0), f(x_1), \dots, f(x_n)]^\top$, so wird eine stetige, stückweise lineare Interpolation der Funktion f dargestellt. Illustrieren Sie grafisch die stückweise lineare Approximation der Funktion $f(x) = x^{1/2}$ auf dem Intervall $[0, 1]$ mit den Gitterpunkten

(a) $x_i = i/n$, (b) $x_i = (i/n)^4$

für $i = 0, 1, \dots, n$ und $n = 2, 4, 8, 16$, indem Sie diese mit der Darstellung von f auf einem sehr feinen Gitter vergleichen.

(2) Schreiben Sie eine Routine zur Berechnung eines interpolierenden kubischen Splines mit periodischen Randbedingungen. Interpolieren Sie damit die folgende Funktion $f : [0, 1] \to \mathbb{R}^3$:

$$f(x) = \begin{bmatrix} (2 + \cos(2q\pi x)) \cos(2p\pi x) \\ (2 + \cos(2q\pi x)) \sin(2p\pi x) \\ \sin(2q\pi x) \end{bmatrix}$$

für $p = 1$ und eine natürliche Zahl $q > 2$ Ihrer Wahl und $2^s q + 1$ äquidistante Stützstellen für $s = 0, 1, 2, 3$. Wenden Sie Ihre Interpolations-Routine dafür auf alle Komponenten separat an und stellen Sie das Ergebnis graphisch mithilfe der Funktion `plot3` dar. Probieren Sie gerne auch andere natürliche Zahlen p und q aus. Sie erhalten dabei sogenannte »Torus-Knoten«.

Projekt BK4 (4+4 Punkte)**.** Schnelle Fourier-Transformation.

(1) Implementieren Sie die schnelle komplexe Fourier-Synthese als rekursive Funktion `my_ifft` und benutzen Sie diese sowie Bemerkung 13.2(i) in »Numerik 3x9«, um eine Funktion `my_fft` für die komplexe Fourier-Transformation zu definieren. Benutzen Sie folgende Funktionen:

$$f_1(x) = \sin(5x) + 0.5\mathrm{i}\cos(x) \quad f_2(x) = \begin{cases} 1, & x \in [\pi - 1/4, \pi + 1/4], \\ 0, & x \notin [\pi - 1/4, \pi + 1/4], \end{cases}$$

$$f_3(x) = \begin{cases} 1, & x \in [0, \pi), \\ -1, & x \in [\pi, 2\pi], \end{cases} \quad f_4(x) = \begin{cases} x, & x \in [0, \pi), \\ 2\pi - x, & x \in [\pi, 2\pi]. \end{cases}$$

Verwenden Sie Ihre Routine, um die Fourier-Transformation der Vektoren $y \in \mathbb{C}^n$ definiert durch $y_j = f_r(2\pi j/n)$, $j = 0, 1, \ldots, n-1$, $r = 1, 2, 3, 4$, mit $n = 2^s$, $s = 1, 2, \ldots, 5$, zu berechnen. Stellen Sie die zugehörigen komplexen trigonometrischen Polynome grafisch dar. Nutzen Sie bei der Erstellung Ihres Programms die `Matlab`-Realisierung komplexer Zahlen.

(2) In dieser Teilaufgabe wollen wir mithilfe der (schnellen) Fourier-Transformation drei Instrumente stimmen – Cello, Querflöte und Oboe. Dafür sollen die gespielten Frequenzen mithilfe der FFT berechnet und dann harmonisiert werden. Den Erfolg Ihrer Arbeit können Sie dabei am Ende selbst akustisch beurteilen und gleichzeitig sehen, warum diese Instrumente als verschieden klingend empfunden werden.
Laden Sie bitte die Datei
`aam.uni-freiburg.de/agba/lehre/ss20/num/materials/p11.zip`
herunter und entpacken Sie diese in Ihren `Matlab`-Arbeitsordner. Die Zugangsdaten sind identisch zu denen der übrigen Vorlesungsmaterialien. Bearbeiten Sie bitte die dort enthaltene `.m`-Datei entsprechend den im Quellcode vermerkten Arbeitsaufträgen (markiert mit `%TODO`).

Projekt BK5 (4 Punkte)**.** Numerische Integration.
Verwenden Sie die summierten Trapez- und Simpson-Regeln sowie eine summierte Gaußsche 3-Punkt-Quadraturformel, um die Integrale im Intervall $[0, 1]$ der Funktionen

$$f(x) = \sin(\pi x)e^x, \quad g(x) = x^{1/3}$$

mit Schrittweiten $h = 2^{-\ell}$, $\ell = 1, 2, \ldots, 10$, zu approximieren.
Berechnen Sie jeweils den Fehler e_h und bestimmen Sie eine experimentelle Konvergenzrate γ aus dem Ansatz $e_h \approx c_1 h^\gamma$ und der daraus folgenden Formel

$$\gamma \approx \frac{\log(e_h/e_H)}{\log(h/H)}$$

für zwei aufeinanderfolgende Schrittweiten $h, H > 0$. Vergleichen Sie die experimentellen Konvergenzraten mit den theoretischen Konvergenzraten der Verfahren und kommentieren Sie Ihre Ergebnisse. Stellen Sie die Paare (h, e_h) für die verschiedenen Quadraturformeln vergleichend als Polygonzüge grafisch in logarithmischer Achsenskalierung mit Hilfe des `Matlab`-Befehls `loglog` dar.

Projekt BK6 (4 Punkte)**.** Differenzenquotienten.
Aus der Taylor-Formel ergibt sich, dass die Quotienten

$$d_h^+ f(x) = \frac{f(x+h) - f(x)}{h}, \quad \hat{d}_h f(x) = \frac{f(x+h) - f(x-h)}{2h}$$

für eine gegebene Schrittweite $h > 0$ Approximationen von $f'(x)$ mit der Fehlerordnung $\mathcal{O}(h)$ beziehungsweise $\mathcal{O}(h^2)$ definieren. Überprüfen Sie diese Eigenschaft experimentell am Beispiel $f(x) = \tan(x)$ für $x = 1/2$ mit den Schrittweiten $h = 2^{-\ell}$, $\ell = 1, 2, \ldots, 30$.

Projekt BK7 (4+4 Punkte)**.** Nullstellensuche.

(1) Implementieren Sie das Sekanten-Verfahren und das Newton-Verfahren zur Nullstellensuche und testen Sie beide mit der Funktion

$$f(x) = \exp(x) + x^2 - 2,$$

den Start-Intervallen $[-2, 0]$ und $[0, 2]$ (Sekantenverfahren) bzw. den Startwerten $x_0 \in \{-1, 0, 1\}$ (Newton-Verfahren) sowie in beiden Fällen mit dem Abbruchkriterium. Beenden Sie das Newton-Verfahren bei Nichterreichen des Abbruchkriteriums mit 100 Iterationen.

(2) Für eine holomorphe Funktion $f : \mathbb{C} \to \mathbb{C}$ mit Nullstellen $z_1, z_2, \ldots, z_n \in \mathbb{C}$ kann die komplexe Ebene in Einzugsbereiche $E_j \subset \mathbb{C}$, die für $j = 1, 2, \ldots, n$, durch

$$E_j = \{z \in \mathbb{C} : \text{Newton-Verfahren mit Startwert } z \text{ konvergiert gegen } z_j\}$$

definiert sind, sowie die Restmenge $X = \mathbb{C} \setminus \cup_{j=1}^n E_j$, partitioniert werden. Betrachten Sie die Funktion $f(z) = z^3 - 1$ und verwenden Sie als Startwerte Gitterpunkte $z_\ell = x_\ell + \mathrm{i}y_\ell$ im Bereich $[-1, 1]^2 \subset \mathbb{C}^2$, die im Abstand $h = 1/200$ angeordnet sind.

Markieren Sie die Punkte unterschiedlich entsprechend der Zugehörigkeit zum Einzugsbereich einer Nullstelle und stellen Sie diese grafisch dar. Verwenden Sie dazu die `Matlab`-Befehle

```
[X,Y]=meshgrid(-1:h:1,-1:
scatter(X(:),Y(:),5,C(:))
```

mit einer geeignet definierten Matrix `C`.
Für ein anderes Polynom ergibt sich dabei nebenstehendes Bild. Die Nullstellen welcher Funktion wurden hier approximiert?