Einführung in die Grundlagen der Numerik

Wintersemester 2014/2015
Prof. Dr. Marc Alexander Schweitzer
Sa Wu

universitätbonn

Übungsblatt 0.

Aufwärmblatt, keine Abgabe, Besprechung in den Übungen 09.10-10.10

Dieses Blatt dient der Wiederholung und der Vorbereitung für die weiteren Programmieraufgaben. Insbesondere soll auch ein Einstieg in Python/NumPy/matplotlib erleichtert werden. Es wird empfohlen, die ersten Progammieraufgaben in sowohl C/C++ als auch Python/Numpy zu bearbeiten.

Programmieraufgabe 1. (Python/NumPy/matplotlib Einführung)

a) Besorgen Sie sich eine Installation von Python (`http://www.python.org`, `http://ipython.org`), NumPy (`http://numpy.org`) und matplotlib (`http://matplotlib.org`). Die Pakete sind auch in Rechnern der CIP Pools in der Wegelerstraße 6 und Endenicher Allee 60 installiert. Sie sind auch in aktuellen Versionen als Teil des (sehr viel größeren) Gesamtpakets Sage (`http://sagemath.org`) erhältlich.

b) Machen Sie sich mit Python/NumPy/matplotlib vertraut, indem sie zu den Stützstellen $x_0 = -1, x_1 = 0, x_2 = 1$ die Lagrangepolynome

$$L_i = \prod_{j=0, j\neq i}^{2} \frac{x - x_j}{x_i - x_j}$$

for $i = 0, 1, 2$ plotten.

c) Überlegen Sie sich eine Abbildung $L_i \mapsto$ „Mitglied Ihrer Abgabegruppe“ und bauen Sie eine dazu passende Legende in eine Grafik mit Plots aller L_i mit ein.

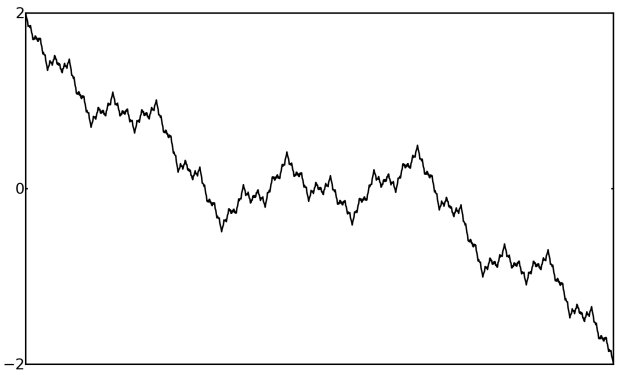
d) Erstellen Sie ein Bild von seperaten Plots der L_i, nebeneinander und mit Bildunterschriften versehen.

keine Abgabe

Programmieraufgabe 2. (Quadratur einer wenig glatten Funktion)
Wir betrachten für $M \in \mathbb{N}$ die Funktion

$$f_M : [0,1] \to \mathbb{R} \qquad\qquad x \mapsto \sum_{k=0}^{M} \frac{1}{2^k} \cos(3^k \pi x)$$

die für $M = 7$ abgebildet ist.



Für alle M sind die f_M beliebig oft differenzierbar, beim Grenzübergang $f_* = \lim_{M\to\infty} f_M$ kommt aber eine Variante der Weierstraß-Funktion heraus. Diese ist zwar stetig aber nirgends differenzierbar.
Wir schauen uns nun mal an, ob und wie das beim numerischen Integrieren eine Rolle spielt.

a) Berechnen Sie

$$\int_0^1 f_M(x)dx.$$

b) Implementieren Sie für allgemeine $f : [a,b] \to \mathbb{R}$ die Quadratur per Trapezsumme $T_{N,a,b}(f)$ und Simpson-Summe $S_{N,a,b}(f)$ in N Teilintervallen von $[a,b]$.

   Berechnen Sie damit für $f(x) = \cos(x)$ die Näherungen $T_{10,0,\frac{\pi}{2}}(f)$ und $S_{10,0,\frac{\pi}{2}}(f)$ an $\int_0^{\frac{\pi}{2}} \cos(x)dx$.

c) Wir teilen uns nun das Integral über $[0,1]$ von f_M in jeweils ein Integral über $[0,0.4]$ und über $[0.4,1]$ auf.

   Plotten Sie für $M = 2,4,6,8,20$ den Fehler von $T_{N,0,0.4}(f_M)+T_{N,0.4,1}(f_M)$ zur exakten Lösung aus a) gegen $N = 2^0, 2^1, \ldots, 2^{17}$ in einem geeigneten Koordinatensystem (beide Achsen logarithmisch, am besten zur Basis 10, `loglog`) und kommentieren Sie das Verhalten im Hinblick auf die Fehlerabschätzungen für Quadraturen.

keine Abgabe

Programmieraufgabe 3. (Iterationsverfahren nochmal mit NumPy)

Um den Umgang mit Matrizen zu schulen, nehmen wir uns Progammieraufgabe 5 von Blatt 12 der Algorithmischen Mathematik 2 noch einmal mit Python/NumPy an. Insbesondere, sollten die `NumPy` Klasse `array` und deren Funktionalität kennen gelernt werden.

a) Schreiben Sie je eine Routine, die ein lineares Gleichungssystem $Ax = b$ iterativ nach der Methode von Jacobi und Gauß-Seidel.

b) Testen Sie die Löser an den 16×16 Matrix

$$A = \begin{pmatrix} T & -I & & \\ -I & T & -I & \\ & -I & T & -I \\ & & -I & -T \end{pmatrix}, \qquad \text{wobei} \qquad T = \begin{pmatrix} 8 & -1 & 0 & 0 \\ -1 & 8 & -1 & 0 \\ 0 & -1 & 8 & -1 \\ 0 & 0 & -1 & 8 \end{pmatrix}$$

und I die 4×4 Identität ist. Wählen Sie als rechte Seite $b = (b_1 b_2 b_2 b_1)^T \in \mathbb{R}^{16}$ mit $b_1^T = (6, 5, 5, 6)$, $b_2^T = (5, 4, 4, 5) \in \mathbb{R}^4$. Als Startiterierte verwenden Sie jeweils $x^0 = 0$. Die Iteration soll abgebrochen werden, falls für das Residuum

$$\left\| Ax^k - b \right\|_2 \leq \left\| Ax^0 - b \right\|_2$$

gilt. Als Ausgaben sollen die Anzahl der Iterationen und der danach erreichte absolute Fehler $\|Ax^k - b\|$ dienen.

c) Erstellen Sie einen Plot des Konvergenzverhaltens der Verfahren mit logarithmisch skalierter Ordinate (`semilogy`) und Legende.

keine Abgabe