MATHEMATISCHES INSTITUT
PROF. DR. CHRISTIANE HELZEL
ANDREAS TROLL

HEINRICH HEINE
UNIVERSITÄT DÜSSELDORF

16.11.2017

Computergestützte Mathematik zur linearen Algebra – 6. Übungsblatt

WICHTIG: Kommentieren Sie Ihren Quelltext. Ihre Skripte müssen durch ausführen des „run"-Befehls (grünes Dreieck bzw. F5) lauffähig sein.

Aufgabe 18: (*Rekursive Funktionen*)

Befehle: `for`, `while`
Skriptname: *rekursive_funktion.py*
Die *Hermiteschen-Polynome* $H_n(x)$ sind wie folgt definiert:

$$\frac{1}{2}H_{n+1}(x) = xH_n(x) - nH_{n-1}(x) \text{ mit } n \in \mathbb{N}_0,\ H_0(x) = 1 \text{ und } H_{-1}(x) := 0$$

Schreiben Sie (ohne das Modul *sympy*!) Funktionen

(a) `hepoit(n,x)`, die das `n`-te Hermitesche-Polynom an den Stellen des `array`s `x` iterativ auswertet,

(b) `hepore(n,x)`, die das `n`-te Hermitesche-Polynom an den Stellen des `array`s `x` rekursiv auswertet.

Hinweis: Es reicht völlig, die Auswertungen an den Stellen iterativ/rekursiv zu berechnen, Sie müssen nicht iterativ/rekursiv die Funktionsvorschrift für $H_n(x)$ erstellen.

(c) Plotten Sie die ersten 5 Hermiteschen-Polynome $(H_0, \ldots, H_4)$ im Intervall $[-1.8, 1.7]$ mit passender Legende für die iterative und die rekursive Variante in zwei Subplots nebeneinander. Geben Sie jedem Subplot auch einen passenden Titel.

Aufgabe 19: (*Funktionen und Plots*)

Skriptname: *plotten1.py*
Hinweis: Die Funktionen in a) und b) sollen arrays verarbeiten können.

(a) Definieren Sie eine Funktion $f(x) = 0{,}7 + 5 \cdot 9^{-2}x(x + 2{,}5)(x - 3)$

(b) Definieren Sie eine Funktion $g(x) = 3\sin(49x) + x$

(c) Plotten Sie beide Funktionen jeweils in einen eigenen Plot im Intervall $[-\pi, \pi]$ (verwenden Sie dazu den `linspace` Befehl). Ein neues Plotfenster erhalten Sie mit `plt.figure()`. $f(x)$ soll in einer schwarzen, gestrichelten Linie und $g(x)$ in einer grünen, durchgezogenen Linie geplottet werden.

(d) Erstellen Sie für den Plot von $f(x)$ eine aussagekräftige Legende mit Schatten die sich in der oberen linken Ecke befindet. Geben Sie bei beiden Plots einen passenden Titel an und beschriften Sie die x- und y-Achsen.

Hinweis: Es kann sein, dass einer Ihrer Plots nicht so aussieht wie Sie es erwartet haben. Wieso ist das so und wie können Sie das Problem beheben?

Aufgabe 20: (*Stückweise definierte Funktion & Plot*)
Befehle: `logical_and`
Skriptname: *plotten2.py*

(a) Definieren Sie die stückweise definierte Funktion

$$f(x) = \begin{cases} 2 - 10\cos(3x) & x \leq 0, \\ -8 + 7x & x \in (0, 2), \\ 6 + 0{,}4(x-2)^4 & x \geq 2 \end{cases}$$

einmal als *lambda*-Funktion und einmal mit der „normalen“ Funktionsdefinition. Beide Varianten sollen `array`s auswerten können.

(b) Überprüfen Sie, ob Ihre beiden Varianten die gleichen Ergebnisse produzieren.

(c) Plotten Sie den Graphen von f mit einer roten Linie der Stärke 2 im Intervall $[-2, 4]$.

(d) Verwenden Sie mindestens 150 Punkte zum plotten und achten Sie darauf, dass die Funktion an den wichtigen Stellen 0 und 2 ausgewertet wird.

(e) Setzen Sie im Abstand von 0,5 weiße Dreiecke (Spitze oben) mit rotem Rand der Dicke 2 als Marker.

Aufgabe 21: (*Cramersche Regel*)

Befehle: `copy`, `det`, `for`, `norm`
Skriptname: *cramersche_regel.py*

(a) Mithilfe der *Cramerschen Regel* kann man lineare Gleichungssysteme lösen.
Schreiben Sie eine Funktion `x=solveCramer(A, b)`, die genau das tut.

Cramersche Regel:
Seien eine invertierbare Matrix $A \in \mathbb{R}^{n\times n}$ und ein Vektor $b \in \mathbb{R}^n$ gegeben. Dann löst x, mit $x_k = \frac{\det A_k}{\det A}$, das LGS $Ax = b$. Hierbei ist $\det A$ die Determinante von A. A_k ist die Matrix, bei der die k-te Spalte von A durch b ersetzt wurde.

Hinweise: Für die Berechnung der Determinanten können Sie `numpy.linalg.det` verwenden. Beachten Sie, dass die Eingabematrix A durch Ihre Funktion NICHT verändert werden soll

(b) Testen Sie Ihre Funktion anhand einer Zufallsmatrix Z der Dimension 10×10 und eines Zufallsvektors v passender Länge. Berechnen Sie hierfür die euklidische Norm des Residuums $Zx - v$. Vergleichen Sie Ihr Ergebnis mit der *numpy*-Funktion `numpy.linalg.solve`

Besprechung in den Übungen vom 20.-24. November 2017.