MATHEMATISCHES INSTITUT
PROF. DR. CHRISTIANE HELZEL
ANDREAS TROLL

HEINRICH HEINE
UNIVERSITÄT DÜSSELDORF

9.11.2017

Computergestützte Mathematik zur linearen Algebra – 5. Übungsblatt

Vergessen Sie nicht Ihren Quelltext sinnvoll zu kommentieren!

Aufgabe 15: (*Zugriffe und Operationen*)

Befehle: `imag`, (`a`)`range`, `real`, `np.logical_or`, `np.logical_and`
Skriptname *array_zugriffe.py*
Erstellen Sie ein Array `d` mit folgenden beiden Eigenschaften. Geben Sie das Array nicht direkt ein, sondern verwenden Sie Methoden wie (`a`)`range`. Um die Teilarrays in den folgenden Teilaufgaben zu erhalten, tippen Sie die Indizes nicht direkt ein, sondern verwenden Sie Slices, logische Ausdrücke und ggf. (selbstgeschriebene) Funktionen!

- `np.real(d)=array([-4., -3., -2., -1., 0., 1., 2., 3., 4., 5.])`
- `np.imag(d)=array([ 49., 36., 25., 16., 9., 4., 1., 0., 1., 4.])`

Geben Sie Teilarrays mit folgenden Elementen von `d` aus:

(a) Das zweite bis neunte Element.

(b) Alle Elemente mit geradem Index.

(c) Alle Elemente mit geradem Imaginärteil.

(d) Alle Elemente deren Absolutbetrag des Realteils größer als 2 ist.

(e) Alle Elemente deren Realteil durch 2 teilbar ist oder deren Imaginärteil durch 3 teilbar ist.

(f) Alle Elemente deren Realteil größer als -4 ist und deren Imaginärteil kleiner/gleich 4 ist.

Aufgabe 16: (*Arrays konstruieren*)

Befehle: `concatenate`, `hstack`, `reshape`, `T`, `transpose`, `vstack`, `kron`
Skriptname: *arrays_konstruieren.py*

(a) Erstellen Sie ein `array` `a` mit den Zahlen von 1 bis 9 und eine 2×2 Einheitsmatrix `b`.

Wichtig: Lösen Sie die folgenden Teilaufgaben mit Ihren Arrays `a` und `b` (als Ganzes und nicht nur Teile eines Arrays), den oben genannten oder ähnlichen Befehlen und mathematischen Operationen. Jede Teilaufgabe soll ohne speichern von Zwischenergebnissen in nur einer Zeile gelöst werden!
Konstruieren Sie mit `a` und `b` folgende Matrizen:

(b) $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$

(c) $\begin{pmatrix} 1 & 0 & 0 & 1 \\ 2 & 0 & 0 & 2 \\ 3 & 0 & 0 & 3 \end{pmatrix}$

(d) $\begin{pmatrix} 1 & 4 & 7 & 2 & 5 & 8 & 3 & 6 & 9 \end{pmatrix}$

(e) $\begin{pmatrix} 1 & 0 & 2 & 0 & 3 & 0 \\ 0 & 1 & 0 & 2 & 0 & 3 \\ 4 & 0 & 5 & 0 & 6 & 0 \\ 0 & 4 & 0 & 5 & 0 & 6 \\ 7 & 0 & 8 & 0 & 9 & 0 \\ 0 & 7 & 0 & 8 & 0 & 9 \end{pmatrix}$

(f) $\begin{pmatrix} 1 & 2 & 3 & 0 & 0 & 0 \\ 4 & 5 & 6 & 0 & 0 & 0 \\ 7 & 8 & 9 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 & 3 \\ 0 & 0 & 0 & 4 & 5 & 6 \\ 0 & 0 & 0 & 7 & 8 & 9 \end{pmatrix}$

(f) einmal unter Verwendung von `a` und `b` und einmal nur mit `a`

Aufgabe 17: (*Integralapproximation*)

Befehle: `np.prod`, `np.sum`

Eine Approximation an das Integral einer Funktion f auf dem Intervall $[a, b]$ ist durch die sogenannte *linke Rechtecksregel* gegeben:

$$I^{(a,b)}(f) := \int_a^b f(x)\,\mathrm{d}x \approx h \sum_{k=1}^{N} f(x_{k-1}) =: I_N^{(a,b)}(f), \text{ mit } x_k = a + kh \text{ für } h = (b-a)/N. \qquad (1)$$

Schreiben Sie ein Skript *numerische_integration.py*:

(a) Schreiben Sie eine Funktion`IntApprox_schleife(f, a, b, N)`, welche mit Hilfe der linken Rechtecksregel (1) das Integral von f mittels einer Schleife näherungsweise berechnet (also $I_N^{(a,b)}(f)$) und als *float* zurückgibt. Die Eingabeparameter sind die reellwertige Funktion `f` als functionhandle (das `array`s punktweise auswerten kann), die Integrationsgrenzen `a` und `b` und die Anzahl der Auswertungen `N`. Setzen Sie `N`= 42, falls es nicht übergeben wird.

(b) Schreiben Sie eine Funktion `IntApprox_direkt(f, a, b, N)`, welche genau das gleiche wie `IntApprox_schleife` macht, jedoch keine Schleife verwendet (hilfreiche Funktionen stehen oben).

(c) Schreiben Sie eine Funktion
`(fehler_schleife, fehler_direkt)`=`IntApproxTest(f, a, b, Ns, ex)`:

- Die Eingaben `f`, `a` und `b` sind wie in (a), `Ns` sei eine Liste mit verschiedenen Werten für `N` und `ex` sei das exakte Integral (also $I^{(a,b)}(f)$).
- `IntApproxTest` soll jeweils mithilfe von `IntApprox_schleife` und `IntApprox_direkt` $I^{(a,b)}(f)$ mit jedem der Werte aus `Ns` approximieren (also $I_{Ns[k]}^{(a,b)}(f)$).
- Daraus sollen die Fehler $|I^{(a,b)}(f) - I_N^{(a,b)}(f)|$ beider Approximationen berechnet und als `array`s zurückgeben werden. Das `array fehler_schleife` soll also die Fehler enthalten, die bei Verwendung von `IntApprox_schleife` entstehen.
  `fehler_schleife[0]`=$|I^{(a,b)}(f) - I_{Ns[0]}^{(a,b)}(f)|$, `fehler_schleife[1]`=$|I^{(a,b)}(f) - I_{Ns[1]}^{(a,b)}(f)|$, usw. (`fehler_direkt` analog).

(d) Auf der Vorlesungsseite finden Sie das PYTHON-Modul *konTROLLeBlatt5.py*. Laden Sie die Datei in Ihr Arbeitsverzeichnis herunter und importieren Sie es. Testen Sie `IntApproxTest` mit $f(x) = \sin(x) + 1$ auf dem Intervall $[0, \pi]$ für`Ns = [ 1, 10, 100, 1000 ]` und führen Sie die Funktion `plottefehler` aus *konTROLLeBlatt5* mit Ihren Ergebnissen aus (gucken Sie ggf. in die Hilfe). Deuten Sie den Plot.

Besprechung in den Übungen vom 13.-17. November 2017.