Computergrundlagen 2025

Blatt 8: Python

- Abgabetermin für die Lösungen: **14.12.2025, 20 Uhr/ für Montagsgruppe: 12.12.2025, 12 Uhr**
- Bei Fragen wendet euch bitte an eure/n Tutor/in:
  - Mo 11:30: Stephan Haag: `st170833@stud.uni-stuttgart.de`
  - Di 09:45: Julian Hoßbach: `julian.hossbach@icp.uni-stuttgart.de`
  - Mi 14:00: Julian Peters: `julian.peters@icp.uni-stuttgart.de`
  - Do 09:45: Rebecca Stephan: `rebecca.stephan@icp.uni-stuttgart.de`
  - Fr 09:45: Jonas Höpker: `st182335@stud.uni-stuttgart.de`
- Die Übungsaufgaben sollen in der Regel in **Zweiergruppen** bearbeitet werden. Nur in **begründeten Ausnahmefällen** sind Dreiergruppen möglich.
- Die Abgabe der Übungsblätter erfolgt über Ilias.
- Mit Abgabe der Lösungen erklärt Ihr, dass Ihr die Lösung euren Mitstudierenden im Rahmen der Übungsbesprechung vorstellen könnt. Um dies zu überprüfen, muss mindestens zweimal von jedem Teilnehmenden vorgetragen werden. Wenn Ihr das nicht könnt, werden euch die Punkte für die entsprechenden Aufgaben wieder abgezogen.
- **Befehle, die nicht in der Vorlesung besprochen wurden, müssen gegebenenfalls recherchiert werden.**
- **Alle erstellen Skripte (`.py` and `.ipynb`) sowie ein mit markown oder Latex erstellter Report (`.pdf`) sind Teil der Abgabe**

  Tipp: Für Abgaben in Python kann der Editor https://jupyter.org/ hilfreich sein. Hier könnt ihr euren Code *interactive* ausführen sowie mit Markdown Zellen ausführlich kommentieren. Die `*.ipynb*` Dateien können ebenfalls als Lösungen abgegeben werden und sind leicht zu validieren.

Numpy-arrays (3 Punkte)

Betrachte folgenden Code-Ausschnitt:

```
import numpy as np
np.random.seed(42)
data = np.random.rand(100, 2)
```

Der Code erzeugt einen `numpy` Array `data` mit 100 Zeilen und 2 Spalten. Gib an wie du folgende Werte von `data` ausgeben kannst. Dabei starten wir beim Zählen mit 0:

- Die Gesamtanzahl der Elemente (`size`)
- Das Element in der nullten Zeile und ersten Spalte
- Die gesamte letzte Zeile
- Die gesamte zehnte Spalte
- Einen Sub-Array, der die folgenden Bereiche abdeckt:
  - Nur Zeilen mit Index 50-59
  - Nur die nullte Spalte
- Den Mittelwert (`mean`) der Zeilen mit Indizes 50-99 in der nullten Spalte.

Ein Skript, das alle diese Werte ausgibt soll Teil der Abgabe sein.

Plotten einer gedämpften harmonischen Schwingung mit `matplotlib` und `argparse` (5 Punkte)

Die Auslenkung $x(t)$ einer gedämpften harmonischen Schwingung in Abhängigkeit von der Zeit t wird durch folgende Gleichung beschrieben:

$$x(t) = A \cdot e^{-\gamma t} \cdot \cos(\omega t)$$

Dabei bezeichnen die Symbole:

- A: Anfangsamplitude
- γ: Dämpfungskonstante
- ω: Kreisfrequenz

Teil 1: Implementierung und Darstellung

Nutze das bereitgestellte Python-Skript `damped_oscillator.py`, um die Bewegung einer gedämpften harmonischen Schwingung zu simulieren und grafisch darzustellen.

1. **Zeitskala:** Erstelle mithilfe von `numpy.linspace` ein Zeit-Array t, das den Bereich von 0 bis 10 Sekunden mit 500 äquidistanten Punkten abdeckt.
2. **Berechnung:** Berechne den Auslenkungs-Array $x(t)$ unter Verwendung der Funktion `damped_oscillation` und den folgenden Standardparametern:
   - $A = 1$ (Anfangsamplitude)
   - $\gamma = 0.2$ (Dämpfungskonstante)
   - $\omega = 2\pi$ (Kreisfrequenz)
3. **Visualisierung:** Erstelle mithilfe von `matplotlib.pyplot` einen Plot der Funktion $x(t)$ über der Zeit t. Füge dem Plot einen Titel, geeignete Achsenbeschriftungen (z.B. "Zeit t", "Auslenkung x(t)") und eine Legende hinzu.
4. **Abgabe:** Speicher den generierten Plot im PDF-Format ab und füge sowohl die Plot-Datei als auch das Skript `damped_oscillator.py` der Abgabe bei.

Teil 2: Erweiterung mit `argparse` (Kommandozeilen-Argumente)

Erstelle eine Kopie des Skriptes aus Teil 1 und nenne diese `damped_oscillator_argparse.py`. Erweitere dieses neue Skript, sodass die zentralen Parameter über Kommandozeilen-Argumente festgelegt werden können.

1. **Modul:** Importiere das Modul `argparse`.
2. **Argumente:** Definiere die folgenden Kommandozeilen-Argumente:
   - `--amplitude` oder `-A`: Setzt die Anfangsamplitude A. Standardwert: 1.0.
   - `--damping` oder `-gamma`: Setzt die Dämpfungskonstante γ. Standardwert: 0.2.
   - `--frequency` oder `-omega`: Setzt die Kreisfrequenz ω. Standardwert: 2π (numerisch ca. 6.283).
3. **Nutzung:** Passe die Berechnung und die Plot-Erzeugung im Skript so an, dass die übergebenen oder die Standard-Werte verwendet werden.

Teste die Funktionalität des Skriptes, indem du es mit verschiedenen Argumenten ausführst, z.B.:

```
python3 damped_oscillator_argparse.py -A 2.5 -gamma 0.1 -omega 10.0
```

Das angepasste Skript `damped_oscillator_argparse.py` soll abgegeben werden.

Array-Slicing (2 Punkte)

Genau wie bei Listen kann man bei Numpy-Arrays einzelne Teilbereiche auswählen, aber eben auch mehrdimensional. Erstelle ein NumPy-Array namens `data` mit 10×10 Nullen und verwendet Array-Zugriffe und -Slicing, um Werte zu setzen. Der erste Index ist die x-Koordinate und der zweite die y-Koordinate.

Am Ende soll

```
plt.imshow(data.T, origin='lower')
plt.colorbar()
plt.show()
```

folgendes Bild erzeugen: