Blatt 8

Vincent Kümmerle und Elvis Gnaglo

11. Dezember 2025

1 Numpy-arrays

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

# Gibt die gesamte Anzahl aller Elemente aus
anzahl_elemente = data.size
print(f"1. Gesamtanzahl der Elemente: {anzahl_elemente}")

# Gibt das Element aus Zeile 0 und Spalte 1 aus
element_0_1 = data[0, 1]
print(f"2. Element (Zeile 0, Spalte 1): {element_0_1}")

# Gibt die Elemente aus der gesamten letzten Zeile aus
letzte_zeile = data[-1, :]
print(f"3. Die gesamte letzte Zeile: {letzte_zeile}")

# Gibt die Elemente aus der 10. Spalte aus. Fehler: es gibt nur 2
    Spalten.
# zehnte_spalte = data[:,9]
# print(f"4. Die Zehnte Spalte: {zehnte_spalte}")

# Definiert einen sub_array von Zeile 50 bis 59 und Spalte 0. Dabei
    ist zu beachten, dass der Start inklusiv ist, weswegen der Index
    50 angegeben ist und das Ende exklusiv ist, weswegen der Index 60
    angegeben werden muss, damit der Wert 59 mitinbegriffen ist.
sub_array = data[50:60, 0]
print(f"5. Sub-Array (Zeile 50-59, Spalte 0):\n{sub_array}")

# Berechnet den Mittelwert der Zeilen 50 bis 99 in der Spalte 0.
mean_val = np.mean(data[50:100, 0])
print(f"6. Mittelwert (Zeile 50-99, Spalte 0): {mean_val}")
```

2 Plotten einer gedämpften harmonischen Schwingung mit matplotlib und argparse

2.1 Teil 1: Implementierung und Darstellung

```python
import numpy as np
import matplotlib.pyplot as plt

# Funktion zur Berechnung der gedämpften harmonischen Schwingung
def damped_oscillation(t, A, gamma, omega):
    """
    Berechnet die Auslenkung einer gedämpften harmonischen Schwingung zur Zeit t.
    x(t) = A * e^(-gamma * t) * cos(omega * t)
    """
    return A * np.exp(-gamma * t) * np.cos(omega * t)

def main():
    A = 1.0
    gamma = 0.2
    omega = 2 * np.pi
    t = np.linspace(0, 10, 500)
    x = damped_oscillation(t, A, gamma, omega)

    plt.figure(figsize=(8,4))
    plt.plot(t, x, label="x(t)")
    plt.title("Gedämpfte harmonische Schwingung")
    plt.xlabel("Zeit t [s]")
    plt.ylabel("Auslenkung x(t)")
    plt.legend()
    plt.savefig("damped_oscillator.pdf")
    plt.show()
if __name__ == "__main__":
    main()
```

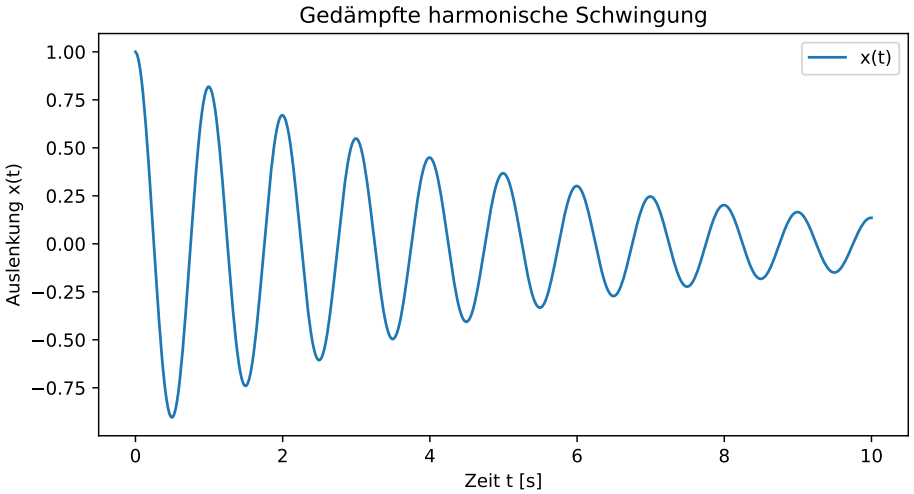

Abbildung 1: Plot der gedämpften harmonischen Schwingung.

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

```python
import numpy as np
import matplotlib.pyplot as plt
import argparse

# Funktion zur Berechnung der gedämpften harmonischen Schwingung
def damped_oscillation(t, A, gamma, omega):
    """
    Berechnet die Auslenkung einer gedämpften harmonischen Schwingung zur Zeit t.
    x(t) = A * e^(-gamma * t) * cos(omega * t)
    """
    return A * np.exp(-gamma * t) * np.cos(omega * t)

def main():
    parser = argparse.ArgumentParser(description="Gedämpfte harmonische Schwingung")

    parser.add_argument(
        "-A", "--amplitude", type=float,
        default=1.0,
        help="Anfangsamplitude A (Standard: 1.0)")

    parser.add_argument(
        "-gamma", "--damping", type=float,
        default=0.2,
        help="Dämpfungskonstante gamma (Standard: 0.2)")

```

```
    parser.add_argument(
        "-omega", "--frequency", type=float,
        default=2 * np.pi,
        help="Kreisfrequenz omega (Standard: 2 pi)")

    args = parser.parse_args()

    t = np.linspace(0, 10, 500)
    x = damped_oscillation(t, args.amplitude, args.damping, args.
       frequency)

    plt.figure(figsize=(8,4))
    plt.plot(t, x, label="x(t)")
    plt.title("Gedämpfte harmonische Schwingung")
    plt.xlabel("Zeit t [s]")
    plt.ylabel("Auslenkung x(t)")
    plt.legend()
    plt.savefig("damped_oscillator.pdf")
    plt.show()
if __name__ == "__main__":
    main()
```

Das Testen hat mit der Eingabe von

```
& "absoluter Pfad von Python" "Pfad von der .py" -A 2.5 -gamma 0.1 -
   omega 10
```

in die PowerShell Kommandozeile funktioniert

3 Array-Slicing

```
    import numpy as np
import matplotlib.pyplot as plt

# Erstellt ein Array mit dem Namen data mit 10x10 Nullen
data = np.zeros((10, 10))

# Der Mund:
# x-Bereich geht von 1 bis 6
# y-Position ist fest auf 3
data[1:7, 3] = 1

# Das linke Auge:
# x-Position ist fest auf 2
# y-Bereich geht von 5 bis 6
data[2, 5:7] = 1
```

```
16
17 # Das rechte Auge:
18 # x-Position ist fest auf 6
19 # y-Bereich geht von 5 bis 6
20 data[6, 5:7] = 1
21
22
23 plt.imshow(data.T, origin='lower')
24 plt.colorbar()
25 plt.show()
```

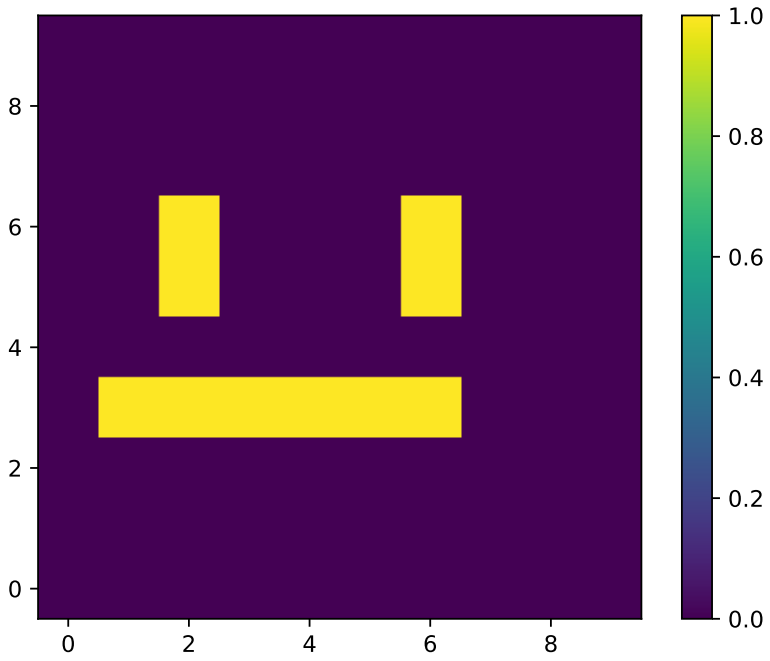

Abbildung 2: Erzeugtes Bild des Smileys.