

Blatt 8

Vincent Kümmerle und Elvis Gnaglo

11. Dezember 2025

1 Numpy-arrays

```
1 import numpy as np
2 np.random.seed(42)
3 data = np.random.rand(100, 2)
4
5 # Gibt die gesamte Anzahl aller Elemente aus
6 anzahl_elemente = data.size
7 print(f"1. Gesamtanzahl der Elemente: {anzahl_elemente}")
8
9 # Gibt das Element aus Zeile 0 und Spalte 1 aus
10 element_0_1 = data[0, 1]
11 print(f"2. Element (Zeile 0, Spalte 1): {element_0_1}")
12
13 # Gibt die Elemente aus der gesamten letzten Zeile aus
14 letzte_zeile = data[-1, :]
15 print(f"3. Die gesamte letzte Zeile: {letzte_zeile}")
16
17 # Gibt die Elemente aus der 10. Spalte aus. Fehler: es gibt nur 2
18 # Spalten.
19 # zehnte_spalte = data[:,9]
20 # print(f"4. Die Zehnte Spalte: {zehnte_spalte}")
21
22 # Definiert einen sub_array von Zeile 50 bis 59 und Spalte 0. Dabei
23 # ist zu beachten, dass der Start inklusiv ist, weswegen der Index
24 # 50 angegeben ist und das Ende exklusiv ist, weswegen der Index 60
25 # angegeben werden muss, damit der Wert 59 mitinbegriffen ist.
26 sub_array = data[50:60, 0]
27 print(f"5. Sub-Array (Zeile 50-59, Spalte 0):\n{sub_array}")
28
29 # Berechnet den Mittelwert der Zeilen 50 bis 99 in der Spalte 0.
30 mean_val = np.mean(data[50:100, 0])
31 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

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3
4 # Funktion zur Berechnung der gedämpften harmonischen Schwingung
5 def damped_oscillation(t, A, gamma, omega):
6     """
7         Berechnet die Auslenkung einer gedämpften harmonischen Schwingung
8         zur Zeit t.
9         x(t) = A * e^(-gamma * t) * cos(omega * t)
10    """
11
12 def main():
13     A = 1.0
14     gamma = 0.2
15     omega = 2 * np.pi
16     t = np.linspace(0, 10, 500)
17     x = damped_oscillation(t, A, gamma, omega)
18
19     plt.figure(figsize=(8,4))
20     plt.plot(t, x, label="x(t)")
21     plt.title("Gedämpfte harmonische Schwingung")
22     plt.xlabel("Zeit t [s]")
23     plt.ylabel("Auslenkung x(t)")
24     plt.legend()
25     plt.savefig("damped_oscillator.pdf")
26     plt.show()
27 if __name__ == "__main__":
28     main()
```

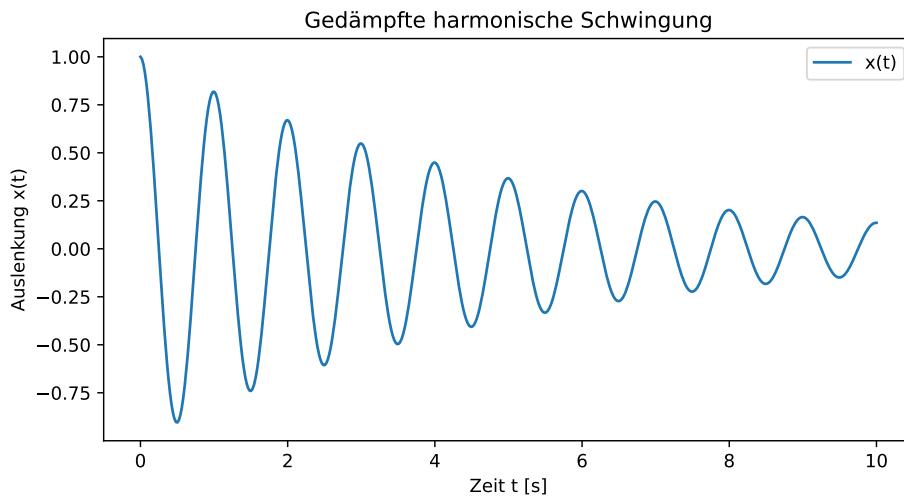


Abbildung 1: Plot der gedämpften harmonischen Schwingung.

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

```

1 import numpy as np
2 import matplotlib.pyplot as plt
3 import argparse
4
5 # Funktion zur Berechnung der gedämpften harmonischen Schwingung
6 def damped_oscillation(t, A, gamma, omega):
7     """
8         Berechnet die Auslenkung einer gedämpften harmonischen Schwingung
9             zur Zeit t.
10    x(t) = A * e^(-gamma * t) * cos(omega * t)
11    """
12
13 def main():
14     parser = argparse.ArgumentParser(description="Gedämpfte
15                                     harmonische Schwingung")
16
17     parser.add_argument(
18         "-A", "--amplitude", type=float,
19         default=1.0,
20         help="Anfangsamplitude A (Standard: 1.0)")
21
22     parser.add_argument(
23         "-gamma", "--damping", type=float,
24         default=0.2,
25         help="Dämpfungskonstante gamma (Standard: 0.2)")
```

```

26     parser.add_argument(
27         "-omega", "--frequency", type=float,
28         default=2 * np.pi,
29         help="Kreisfrequenz omega (Standard: 2 pi)")
30
31     args = parser.parse_args()
32
33     t = np.linspace(0, 10, 500)
34     x = damped_oscillation(t, args.amplitude, args.damping, args.
35                             frequency)
36
37     plt.figure(figsize=(8,4))
38     plt.plot(t, x, label="x(t)")
39     plt.title("Gedämpfte harmonische Schwingung")
40     plt.xlabel("Zeit t [s]")
41     plt.ylabel("Auslenkung x(t)")
42     plt.legend()
43     plt.savefig("damped_oscillator.pdf")
44     plt.show()
45 if __name__ == "__main__":
46     main()

```

Das Testen hat mit der Eingabe von

```

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

```

in die PowerShell Kommandozeile funktioniert

3 Array-Slicing

```

1 import numpy as np
2 import matplotlib.pyplot as plt
3
4 # Erstellt ein Array mit dem Namen data mit 10x10 Nullen
5 data = np.zeros((10, 10))
6
7 # Der Mund:
8 # x-Bereich geht von 1 bis 6
9 # y-Position ist fest auf 3
10 data[1:7, 3] = 1
11
12 # Das linke Auge:
13 # x-Position ist fest auf 2
14 # y-Bereich geht von 5 bis 6
15 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 plt.imshow(data.T, origin='lower')  
23 plt.colorbar()  
24 plt.show()
```

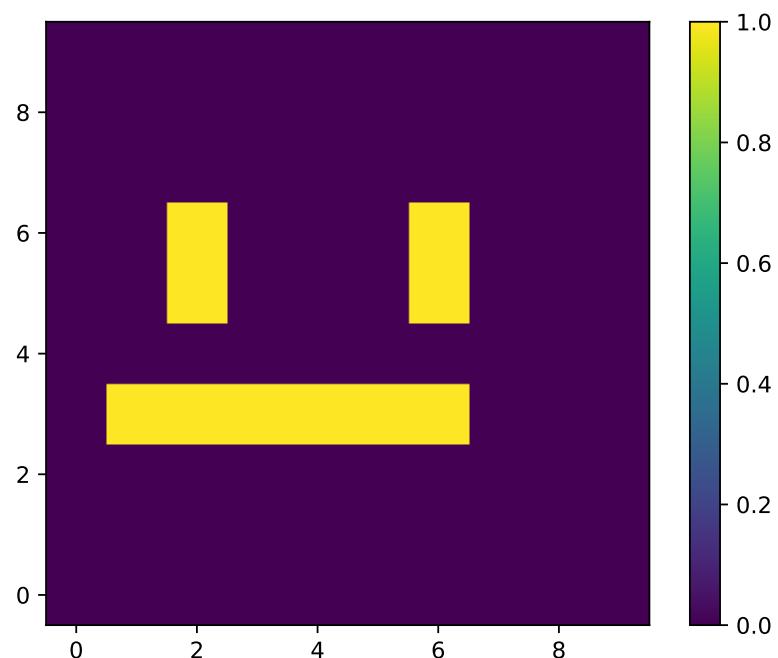


Abbildung 2: Erzeugtes Bild des Smileys.