

Blatt 9

Vincent Kümmerle und Elvis Gnaglo

16. Dezember 2025

1 Fitten von Funktionen

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3 from scipy.optimize import curve_fit
4
5 # Daten mit delimiter , laden
6 try:
7     data = np.loadtxt('C:/Studium/5. Semester/AC II lab/Protokolle/
8         CGL/Blatt 9/data.txt', delimiter=',')
9     x_data = data[:, 0]
10    y_data = data[:, 1]
11 except Exception as e:
12     print(f" Fehler: {e}")
13 # Fallback simulation, damit der Code hier läuft
14 x_data = np.linspace(0, 3, 300)
15 y_data = 1.0 * np.exp(-0.8 * x_data) * np.sin(10 * x_data) + np.
16     random.normal(0, 0.1, 300)
17
18 # Fitfunktion definieren
19 def damped_sine(x, amplitude, decay, omega, phase, offset):
20     """
21         amplitude: Start-Höhe der Welle
22         decay: Wie schnell die Welle abklingt (Dämpfung)
23         omega: Kreisfrequenz (bestimmt den Abstand der Wellenberge)
24         phase: Verschiebung nach links/rechts
25         offset: Verschiebung nach oben/unten
26     """
27     return amplitude * np.exp(-decay * x) * np.sin(omega * x + phase)
28     + offset
```

```

29 p0_guess = [1.0, 0.8, 10.0, 0.0, 0.0]
30
31 try:
32     params, covariance = curve_fit(damped_sine, x_data, y_data, p0=
33         p0_guess)
34
35     # Parameter ausgeben
36     labels = ["Amplitude", "Decay (Dämpfung)", "Omega (Frequenz)", "Phase", "Offset"]
37     print("Gefundene Parameter:")
38     for label, val in zip(labels, params):
39         print(f" {label}: {val:.4f}")
40
41     # Plotten
42     plt.figure(figsize=(10, 6))
43
44     # Messdaten
45     plt.scatter(x_data, y_data, label='Messdaten', color='black',
46                 alpha=0.5, s=15)
47
48     # Fit-Kurve
49     x_fit = np.linspace(min(x_data), max(x_data), 1000)
50     y_fit = damped_sine(x_fit, *params)
51
52     plt.plot(x_fit, y_fit, 'r-', linewidth=2, label='Fit: Gedämpfte Schwingung')
53
54     # Plot anzeigen und speichern
55     plt.xlabel('x')
56     plt.ylabel('y')
57     plt.savefig('fit_plot.pdf')
58     plt.show()
59
60 except RuntimeError:
61     print("Der Fit hat nicht konvergiert. Versuche, die p0_guess Werte anzupassen.")

```

2 SymPy

3 Primzahlen

```

1 N = int(input("N eingeben: "))
2 gestrichen = [False] * (N + 1) # False = noch Primzahl

```

```

3
4 # Jede zusammengesetzte Zahl j=a*b hat mindestens einen Faktor
5   kleiner gleich sqrt(j)
6 for i in range(2, int(N**0.5) + 1): # Hat eine Zahl keinen Teiler
7   kleiner gleich sqrt(), dann ist sie prim
8   if not gestrichen[i]: # i ist Primzahl
9     print(i, end=", ")
10    for j in range(i*i, N + 1, i): # Vielfache streichen
11      gestrichen[j] = True
12
13 # Restliche Primzahlen ausgeben:
14 for i in range(int(N**0.5) + 1, N + 1):
15   if not gestrichen[i]:
16     print(i, end=", ")

```