Lerntagebuch für Fourier-Analyse

Woche 1-2

Tag 1-10: Einführung und Vorbereitung

- Einführung in das Thema Fourier-Analyse.
- Überblick über die Aufgabenstellungen.
- Lesen und Verstehen der Aufgabenstellungen.
- Grundlagen der Fourier-Analyse und der Trapezregel auffrischen.
- Implementierung einer Methode zur Berechnung der k-ten Fourier-Komponente einer Funktion mit numpy und der Trapezregel (np.trapz).

Woche 3-4

Tag 11 - 14: Sonnenfleckendatenanalyse

- Einführung in die Sonnenfleckendaten (sunspots.txt).
- Einlesen und Vorverarbeitung der Daten.
- Plotten des Power-Spektrums.
- Analyse und Interpretation der Ergebnisse.
- Rücktransformation und Vergleich der Ergebnisse.
- Beschreibung und Erklärung der Ergebnisse.

Tag 15-17: Musikstückanalyse

- Einführung in die Analyse von Musikdateien.
- Einlesen und Vorverarbeitung des Musikstücks (music.wav).
- Plotten der Wellenform (waveplot) und des Amplitudenspektrums.
- Trennung des Soundsignals
- Rekonstruktion der gefilterten rekonstruierten Daten

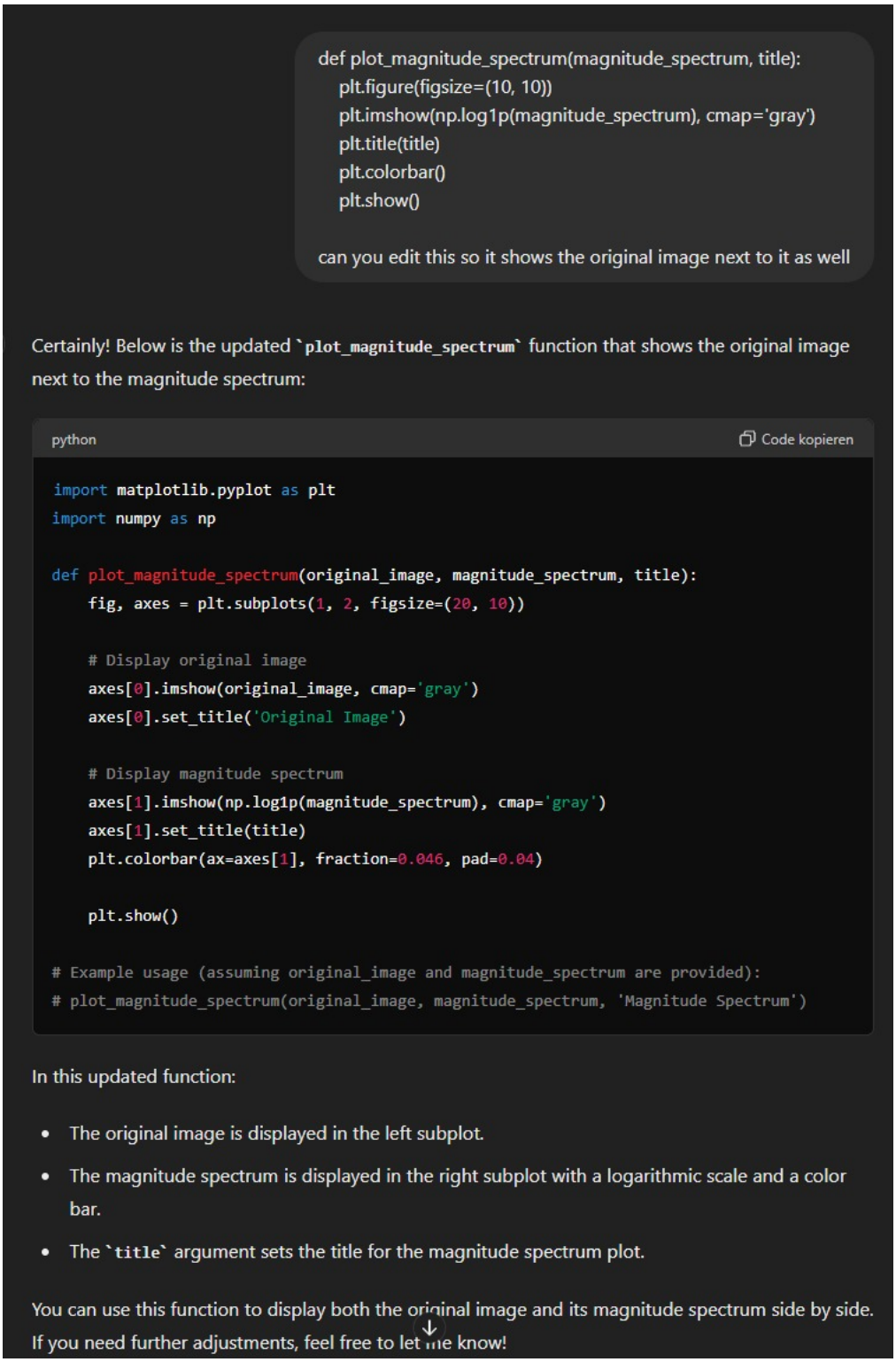
Tag 18-20: Bildanalyse

- Einführung in die Analyse von Bilddateien.
- Einlesen und Vorverarbeitung der Bilddatei (img.jpg).
- Erklärung der Natur der Verteilung der Fourier-Komponenten.
- Entfernen der Fourier-Komponenten mit absoluter Amplitude unter verschiedenen Schwellenwerten
- Durchführung der inversen Fourier-Transformation
- Analyse der Datenkompression.
-

Dieses Lerntagebuch hilft dabei, den Fortschritt systematisch zu dokumentieren und sicherzustellen, dass alle Aspekte der Aufgabenstellung gründlich bearbeitet werden.

Reflexion

Die Nutzung von KI für Programmierung und Code-Reviews war für uns äusserst wertvoll. Automatisierte Code-Analyse und Fehlererkennung haben unsere Effizienz und Effektivität gesteigert. KI-basierte Werkzeuge halfen uns, Fehler schnell und präzise zu identifizieren, Best Practices einzuhalten und die Codequalität zu verbessern. Die Integration von KI in unseren Entwicklungsprozess erhöhte unsere Effizienz und Zuverlässigkeit. Wir sind beeindruckt von den Fortschritten in der KI und freuen uns darauf, unsere Entwicklungspraktiken weiter zu verbessern und innovative Lösungen zu entwickeln.



```
def plot_magnitude_spectrum(magnitude_spectrum, title):
    plt.figure(figsize=(10, 10))
    plt.imshow(np.log1p(magnitude_spectrum), cmap='gray')
    plt.title(title)
    plt.colorbar()
    plt.show()
```

can you edit this so it shows the original image next to it as well

Certainly! Below is the updated `plot_magnitude_spectrum` function that shows the original image next to the magnitude spectrum:

python — Code kopieren

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

def plot_magnitude_spectrum(original_image, magnitude_spectrum, title):
    fig, axes = plt.subplots(1, 2, figsize=(20, 10))

    # Display original image
    axes[0].imshow(original_image, cmap='gray')
    axes[0].set_title('Original Image')

    # Display magnitude spectrum
    axes[1].imshow(np.log1p(magnitude_spectrum), cmap='gray')
    axes[1].set_title(title)
    plt.colorbar(ax=axes[1], fraction=0.046, pad=0.04)

    plt.show()

# Example usage (assuming original_image and magnitude_spectrum are provided):
# plot_magnitude_spectrum(original_image, magnitude_spectrum, 'Magnitude Spectrum')
```

In this updated function:

- The original image is displayed in the left subplot.
- The magnitude spectrum is displayed in the right subplot with a logarithmic scale and a color bar.
- The `title` argument sets the title for the magnitude spectrum plot.

You can use this function to display both the original image and its magnitude spectrum side by side. If you need further adjustments, feel free to let me know!

```
IndexError                                Traceback (most recent call last)
Cell In[31], line 8
      5 filtered_images =
FA_image.filter_fft_components(fft_img_shifted, magnitude_spectrum,
[max_threshold])
      7 # Plot filtered images
----> 8 FA_image.plot_filtered_images(original_image, filtered_images)
     10 print(f'Maximum compression threshold without exceeding
{max_error_percentage}% error: {max_threshold}%')

File
F:\DataScience\Semester2\Math\VTA\vta_minichallenge2\FA\Fourier_A
pproximation_image.py:50, in plot_filtered_images(original_image,
filtered_images)
     47 for i, (threshold, filtered_img) in enumerate(filtered_images):
     48     # Calculate and display FFT magnitude spectrum of the
filtered image
     49     _, filtered_magnitude_spectrum = perform_fft(filtered_img)
---> 50     axes[i, 0].imshow(np.log(1 + filtered_magnitude_spectrum),
cmap='gray')
     51     axes[i, 0].set_title(f'Filtered Fourier Spectrum, {threshold:.5f}%
filtered')
     53     # Display filtered image

IndexError: too many indices for array: array is 1-dimensional, but 2
were indexed
```

The `IndexError` occurs because the `axes` array is not in the shape you expected. This typically happens if you have a single set of axes instead of multiple subplots. We need to adjust the code to handle both cases when we have a single threshold or multiple thresholds.

Here is the corrected code with this consideration: