Fourieranalyse - Speicher

Bildquelle: Bild wurde von Copilot (DALL-E3) generiert:
„Nerd, der versucht eine Fourieranalyse zu programmieren und den Speicherbedarf zu analysieren. “

Jan Niclas Ruppenthal
1481198

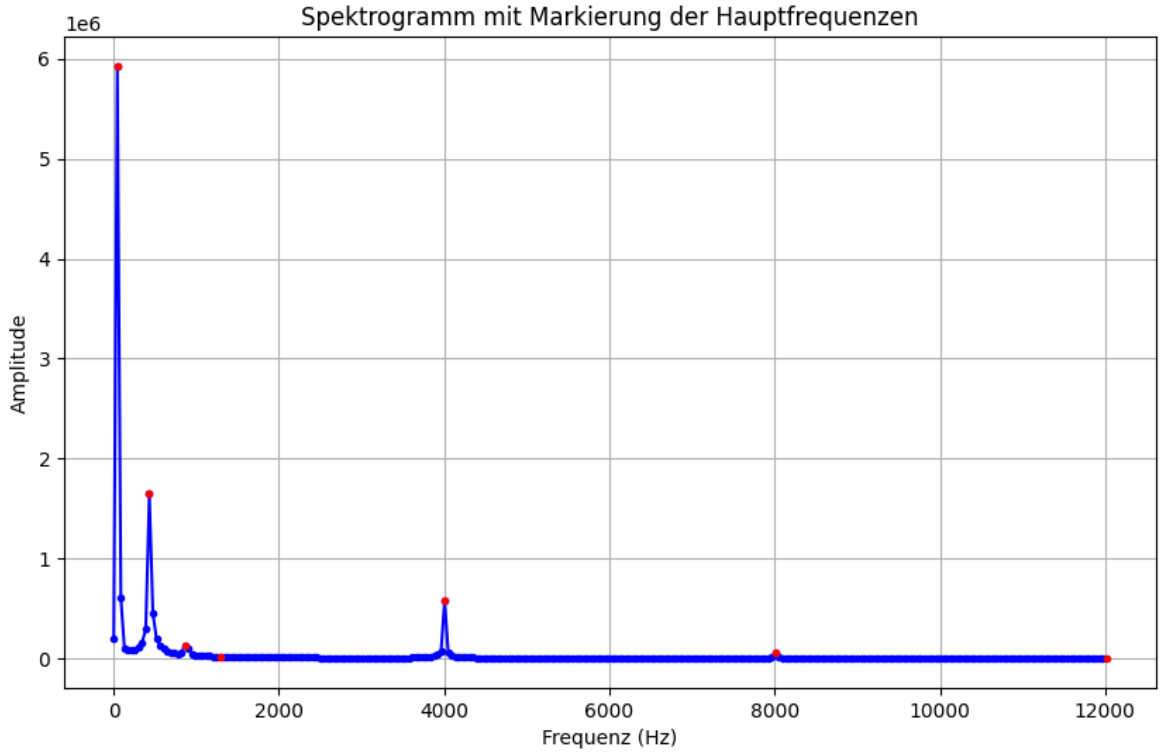
Python - Code

```
def analyze(data, block_size, fourier_function):
    num_samples = len(data)
    num_blocks = num_samples - block_size + 1
    aggregated_fft = np.zeros(block_size//2)
    for i in range(num_blocks):
        block = data[i:i+block_size]
        fft_result = fourier_function(block)
        aggregated_fft += np.abs(fft_result[:block_size//2])

    aggregated_fft /= num_blocks
    return aggregated_fft
```

- Lesen der übergebenen WAV-Datei
- Analyse über die Fouriertransformation
- Vier Varianten:
  - DFT
  - Rekursive FFT
  - Vektorisierte FFT
  - FFT (von numpy)
- Ergebnisse in eine Datei schreiben

Frequenzen (n = 1024)



	Hauptfrequenz	Amplitude
0	43.06640625	5926643.14962656
1	430.6640625	1645417.3411351012
2	861.328125	131049.86352009393
3	1291.9921875	22309.71339958239
4	4005.17578125	579259.092924025
5	8010.3515625	53893.82868761788
6	12015.52734375	5031.239973424879

Grundfrequenzen und Harmonische

1. **43.06640625 Hz**
2. **430.6640625 Hz**
    a. 861.328125 Hz (1. Harmonische)
    b. 1291.9921875 Hz (2. Harmonische)
3. **4005.17578125 Hz**
    a. 8010.3515625 Hz (1. Harmonische)
    b. 12015.52734375 Hz (2. Harmonische)

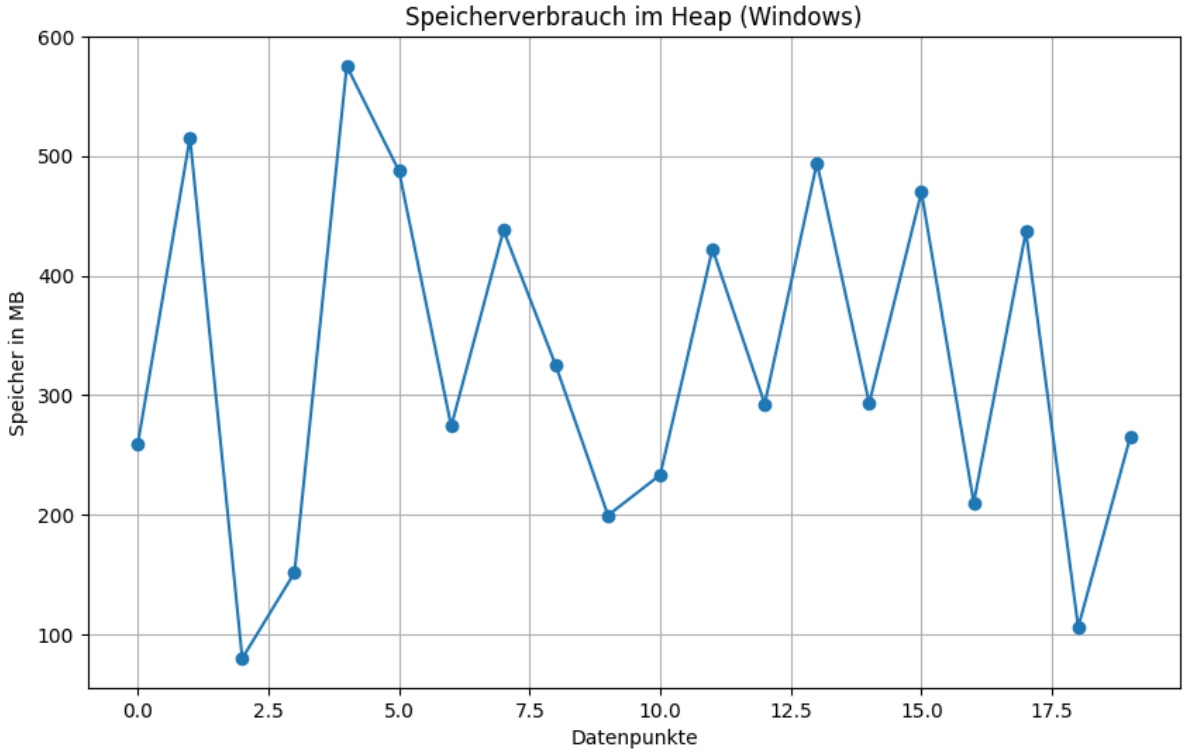
Speicherbedarf - Python

Verschiedene Varianten

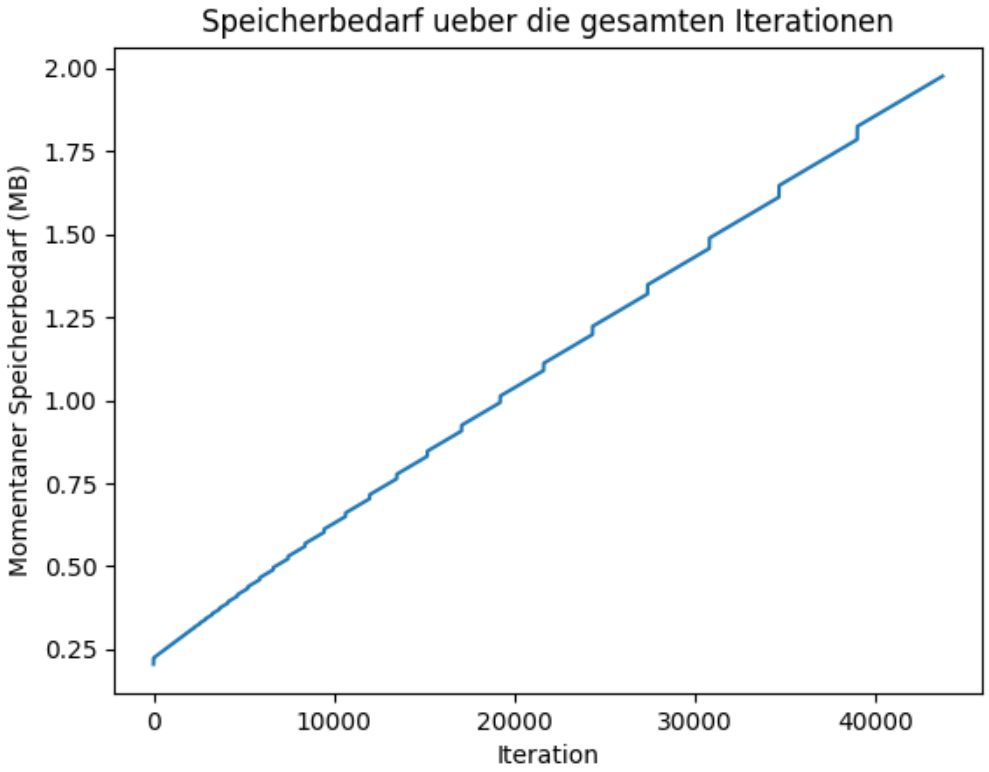
1. Windows:
    1. Python
    2. Java
2. Raspberry Pi Model B Revision 2
    1. Python
    2. Java
3. Linux Mint:
    1. Python
    2. Java

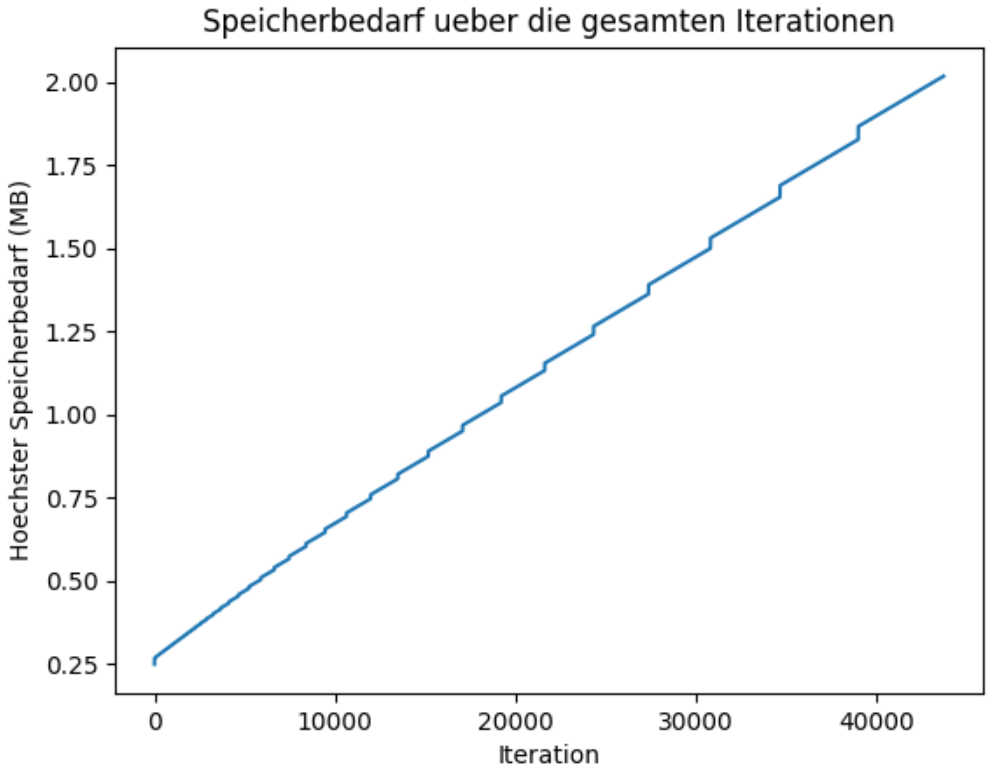
Bildquelle:
https://upload.wikimedia.org/wikipedia/commons/3/3b/Raspberry_Pi_Model_B_Rev._2.jpg

Speicher – Windows (Java)

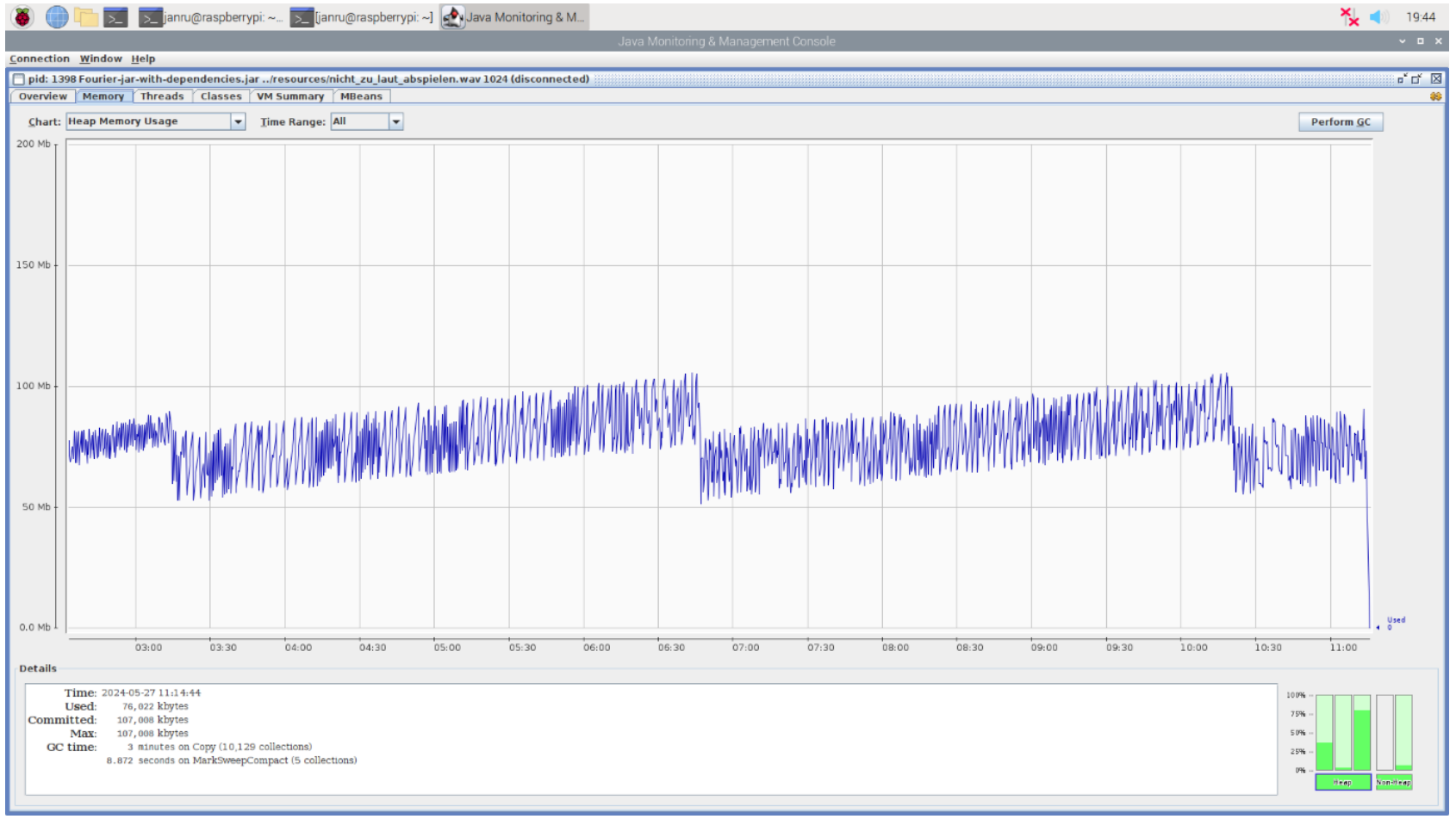


Speicher - Raspberry Pi (Python)





Speicher - Raspberry Pi (Java)



Speicher - Linux Mint (Python)

Speicher - Linux Mint (Java)