REPORT

Zajęcia: Analog and digital electronic circuits
Teacher: prof. dr hab. Vasyl Martsenyuk

Lab 7 - 8

Date 21.12.2024

Topic: "7. Sampling and Reconstruction of Signals: Analysis of Aliasing Effects and Proper Signal Reconstruction. 8. Coding and Decoding Digital Signals"

Variant 10

Anna Więzik
Informatyka II stopień,
niestacjonarne,
1 semestr,
Gr.1b

1. **Problem statement:**

   Task Assignments for sampling and reconstruction

   Variant 10. Analyze a sawtooth wave with $f = 7\,\mathrm{Hz}$, sampled at $f_s = 10\,\mathrm{Hz}$.

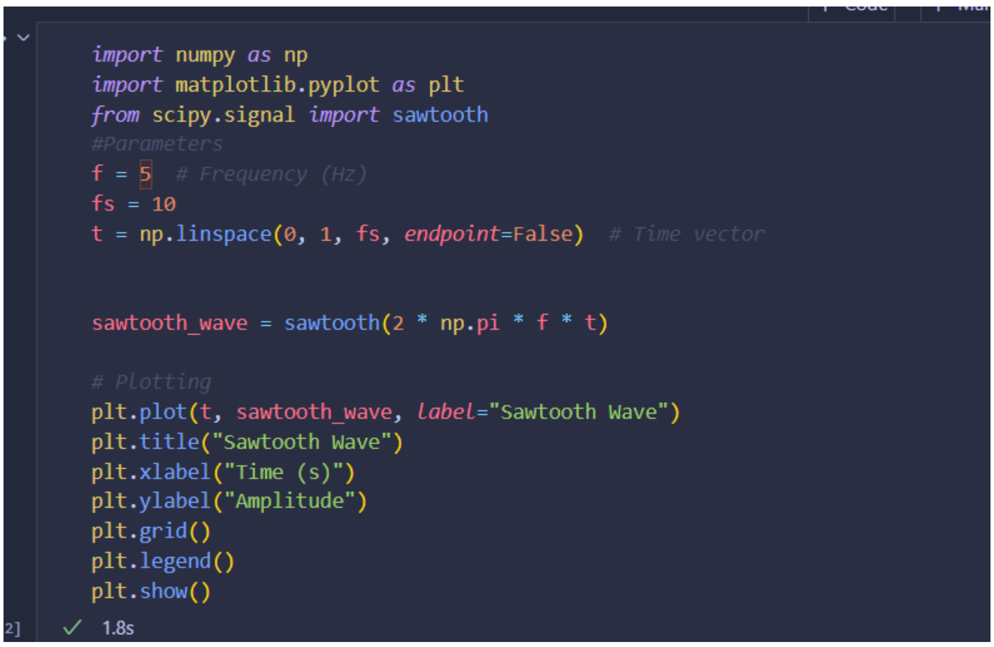
   Task Assignments on Coding/Decoding

   Variant 10. Solve Problem 4: Compare signal distortion and compression ratio for thresholds of 5, 10, and 15 in DCT compression for the signal $[8, 16, 24, 32, 40, 48]$.

   ### 2.4.3 Problem 4: Trade-off Analysis

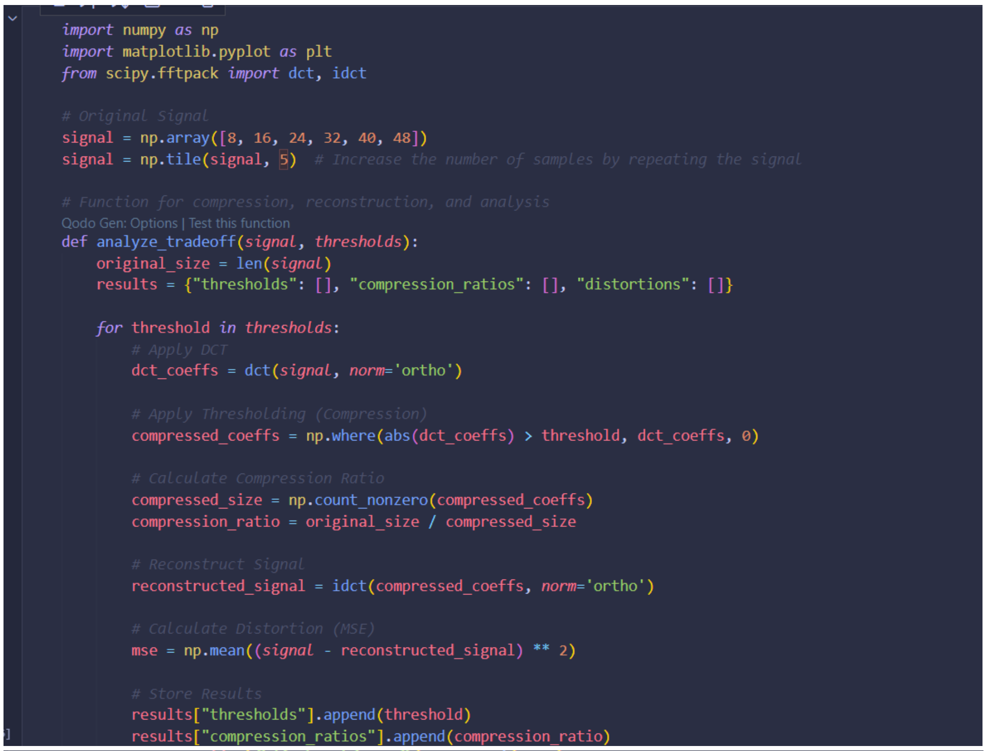
   Problem: Compare signal distortion and compression ratio for various thresholds in DCT compression.

2. **Input data:**
3. **Commands used (or GUI):**
   - source code

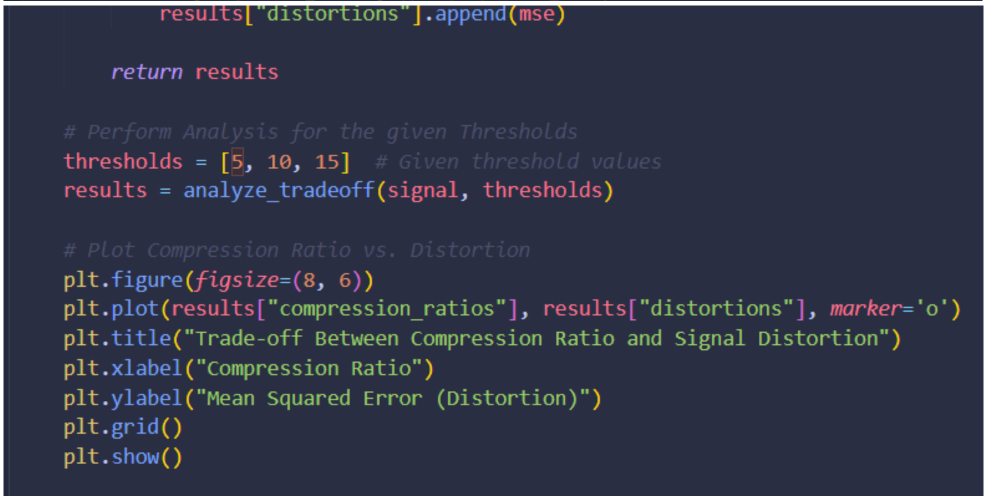


```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.signal import sawtooth
#Parameters
f = 5  # Frequency (Hz)
fs = 10
t = np.linspace(0, 1, fs, endpoint=False)  # Time vector


sawtooth_wave = sawtooth(2 * np.pi * f * t)

# Plotting
plt.plot(t, sawtooth_wave, label="Sawtooth Wave")
plt.title("Sawtooth Wave")
plt.xlabel("Time (s)")
plt.ylabel("Amplitude")
plt.grid()
plt.legend()
plt.show()
```

2] ✓ 1.8s

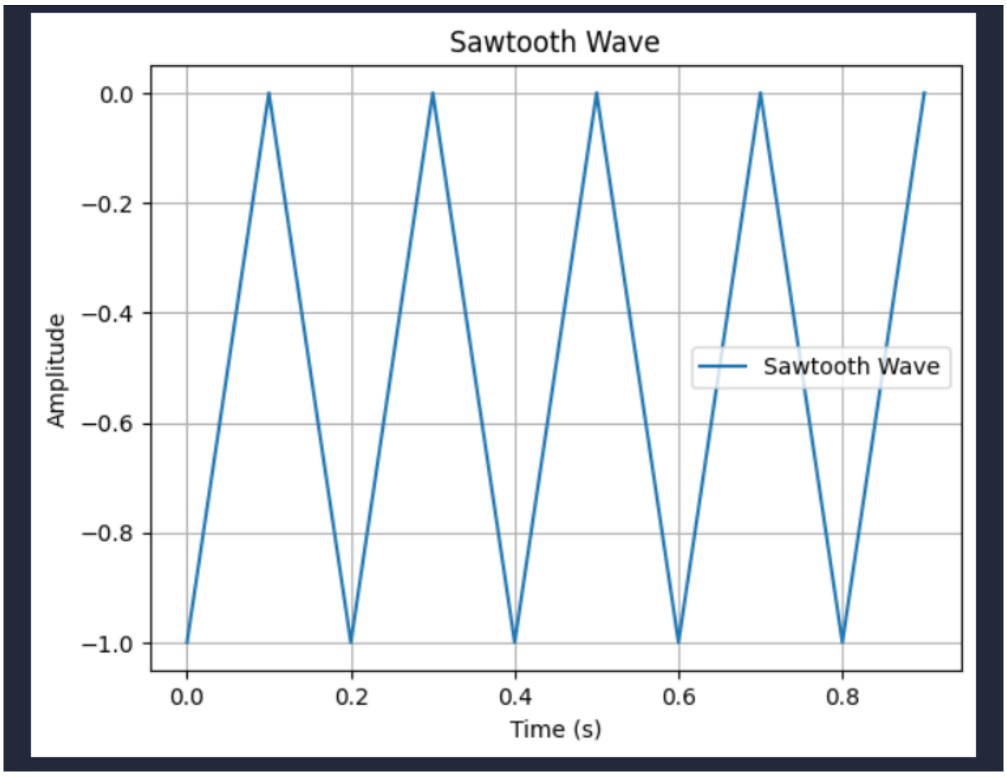


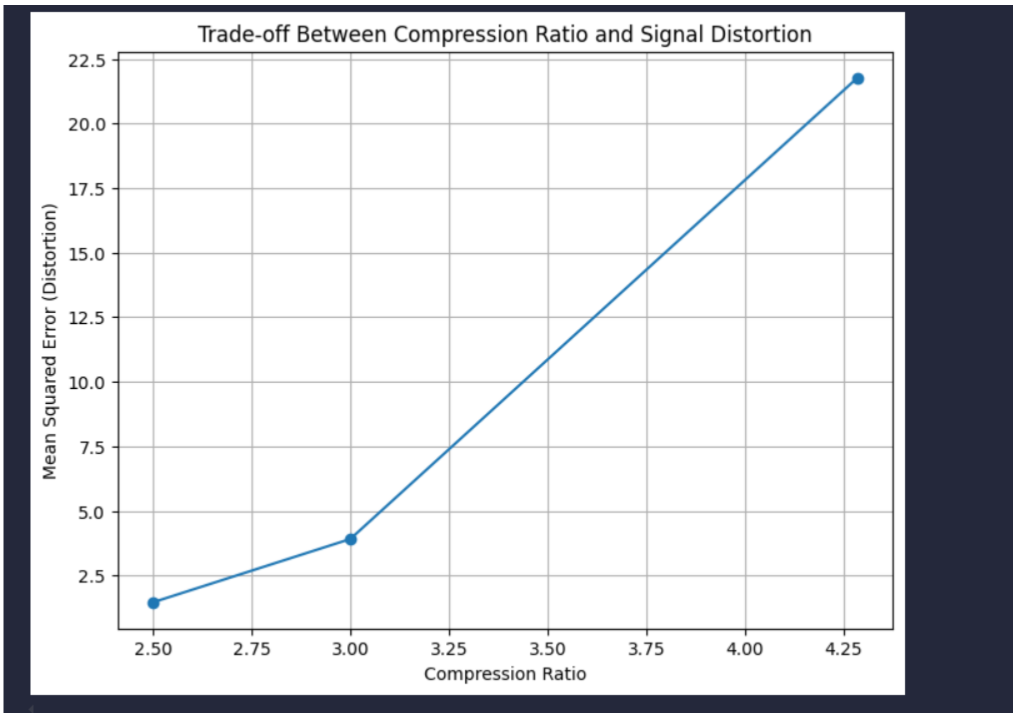
```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.fftpack import dct, idct

# Original Signal
signal = np.array([8, 16, 24, 32, 40, 48])
signal = np.tile(signal, 5)  # Increase the number of samples by repeating the signal

# Function for compression, reconstruction, and analysis
def analyze_tradeoff(signal, thresholds):
    original_size = len(signal)
    results = {"thresholds": [], "compression_ratios": [], "distortions": []}

    for threshold in thresholds:
        # Apply DCT
        dct_coeffs = dct(signal, norm='ortho')

        # Apply Thresholding (Compression)
        compressed_coeffs = np.where(abs(dct_coeffs) > threshold, dct_coeffs, 0)

        # Calculate Compression Ratio
        compressed_size = np.count_nonzero(compressed_coeffs)
        compression_ratio = original_size / compressed_size

        # Reconstruct Signal
        reconstructed_signal = idct(compressed_coeffs, norm='ortho')

        # Calculate Distortion (MSE)
        mse = np.mean((signal - reconstructed_signal) ** 2)

        # Store Results
        results["thresholds"].append(threshold)
        results["compression_ratios"].append(compression_ratio)
```



```python
        results["distortions"].append(mse)

    return results

# Perform Analysis for the given Thresholds
thresholds = [5, 10, 15]  # Given threshold values
results = analyze_tradeoff(signal, thresholds)

# Plot Compression Ratio vs. Distortion
plt.figure(figsize=(8, 6))
plt.plot(results["compression_ratios"], results["distortions"], marker='o')
plt.title("Trade-off Between Compression Ratio and Signal Distortion")
plt.xlabel("Compression Ratio")
plt.ylabel("Mean Squared Error (Distortion)")
plt.grid()
plt.show()
```

- screenshots





- Link to remote repozytorium https://github.com/AnaShiro/UM_2024

4. **Conclusions:**

   This laboratory delves into the essential concepts of sampling and reconstructing signals within the field of signal processing. It encompasses the Nyquist-Shannon sampling theorem, the phenomena of aliasing, and various techniques for signal reconstruction. The goal of this lab session is to acquaint students with the fundamentals of encoding and decoding digital signals. This includes practical applications of compression algorithms to enhance the representation and transmission of signals. The session specifically emphasizes the processes of signal coding, decoding, and reconstruction using Python.