**REPORT**

Zajęcia: Analog and digital electronic circuits

Teacher: prof. dr hab. Vasyl Martsenyuk

**Lab 7 and 8**

17.02.2025

**Topic:**

7. Sampling and Reconstruction of Signals: Analysis of Aliasing Effects and Proper Signal Reconstruction.

8. Coding and Decoding Digital Signals

**Variant 15**

Tobiasz Wojnar  
 Informatyka II stopień,   
niestacjonarne,   
1 semestr,   
Gr. B

# 1. Problem statement:

The aim of the first task is to reconstruct of a sine wave.

The aim of the second task is to compare signal distortion and compression ratio for specified thresholds in DCT compression for the given signal

# 2. Input data:

Task one data

* f = 8 Hz – signal frequency
* fs = 16 Hz, sampled frequency

Task two data

* signal = [3,6,9,12, 15, 18]
* thresholds = [4,8,12]

# 3. Commands used (or GUI):

## a) source code

**Sampling**

**import** numpy **as** np

**import** matplotlib.pyplot **as** plt

**from** scipy.signal **import** resample

f\_signal = 8 *# Frequency of the signal (Hz)*

f\_sample = 16 *# High sampling frequency (Hz)*

t = np.linspace(0, 1, 1000, endpoint=**False**) *# Time vector*

signal = np.sin(2 \* np.pi \* f\_signal \* t) *# Original signal*

*# Sampling the signal*

t\_sample = np.arange(0, 1, 1 / f\_sample)

samples = np.sin(2 \* np.pi \* f\_signal \* t\_sample)

*# Reconstructing the signal using high sampling rate*

num\_samples = 1000

reconstructed\_signal = resample(samples, num\_samples)

*# Plotting the reconstruction*

plt.figure(figsize=(10, 6))

plt.plot(t, signal, label='Original Signal')

plt.plot(t, reconstructed\_signal, label='Reconstructed Signal', linestyle='--')

plt.title('Signal Reconstruction')

plt.xlabel('Time (s)')

plt.ylabel('Amplitude')

plt.legend()

plt.grid()

plt.show()

plt.figure(figsize=(10, 6))

plt.plot(t, signal, label='Original Signal 8Hz')

plt.stem(t\_sample, samples, linefmt='g-', markerfmt='go', basefmt=' ', label='Sampling frequency 16Hz')

plt.title('Aliasing Demonstration')

plt.xlabel('Time (s)')

plt.ylabel('Amplitude')

plt.legend()

plt.grid()

plt.show()

**Trade off analysis**

import numpy as np

import matplotlib.pyplot as plt

from scipy.fftpack import dct, idct

signal = np.array([3, 6, 9, 12, 15, 18])

N = len(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)

results["distortions"].append(mse)

return results

*# Perform Analysis for a Range of Thresholds*

thresholds = np.linspace(4,12,3) *# Threshold values*

results = analyze\_tradeoff(signal, thresholds)

print("Threshold | Compression Ratio | Mean Squared Error (Distortion)")

print("----------|-------------------|--------------------------------")

**for** i **in** range(len(thresholds)):

print(f"{thresholds[i]:10}|{results['compression\_ratios'][i]:19.2f}|{results['distortions'][i]:32.2f}")

*# 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()

## b) Link to remote repositorium

<https://github.com/TobiaszWojnar/DSP>

# 4. Outcomes:

## Signal reconstruction

A graph of a signal reconstruction

AI-generated content may be incorrect.

A graph with green and blue lines

AI-generated content may be incorrect.

## Trade off analysis

|  |  |  |
| --- | --- | --- |
| Threshold | Compression Ratio | Mean Squared Error (Distortion) |
| 4 | 3 | 0.26 |
| 8 | 3 | 0.26 |
| 12 | 3 | 0.26 |

A graph with a dotted line

AI-generated content may be incorrect.

# 5. Conclusions

The application of DCT did not successfully transformed the original time-domain signal into a frequency-domain representation. Since sampling frequency was exactly twice as signal frequency. Due to this reason we always sampled signal when the amplitude was equal to zero.