REPORT

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

Lab 1
Date: 19.10.2024
Topic: Task Windowing
Variant7

Mateusz Łysoń
Informatyka II stopień,
niestacjonarne,
1 semestr,
Gr.1b

1. Problem statement:

The task is to generate three sine signals with given frequencies for a specific sampling frequency and amplitude, within a specified number of samples. The objective is to analyze the signals using Discrete Fourier Transform (DFT) and Discrete-Time Fourier Transform (DTFT) with different window functions (rectangular, Hann and flat-top). The DFT spectra will be normalized, and the DTFT spectra will be normalized to their main lobe maximum. The goal is to assess the performance of different windows and understand the difference between the signals at .

2. Input data:

Input as stated in 7. variant:

- F1 = 400 Hz
- F2 = 400.25 Hz
- F3 = 399.75 Hz
- Fs = 600 Hz
- N = 3000
- A = 3

2. Commands used (or GUI):

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.signal.windows import hann, flattop
from numpy.fft import fft, fftshift

# Parameters
f1 = 400     # Frequency 1 in Hz
f2 = 400.25  # Frequency 2 in Hz
f3 = 399.75  # Frequency 3 in Hz
fs = 600     # Sampling frequency in Hz
N = 3000     # Number of samples
A = 3        # Amplitude

# Time vector
```

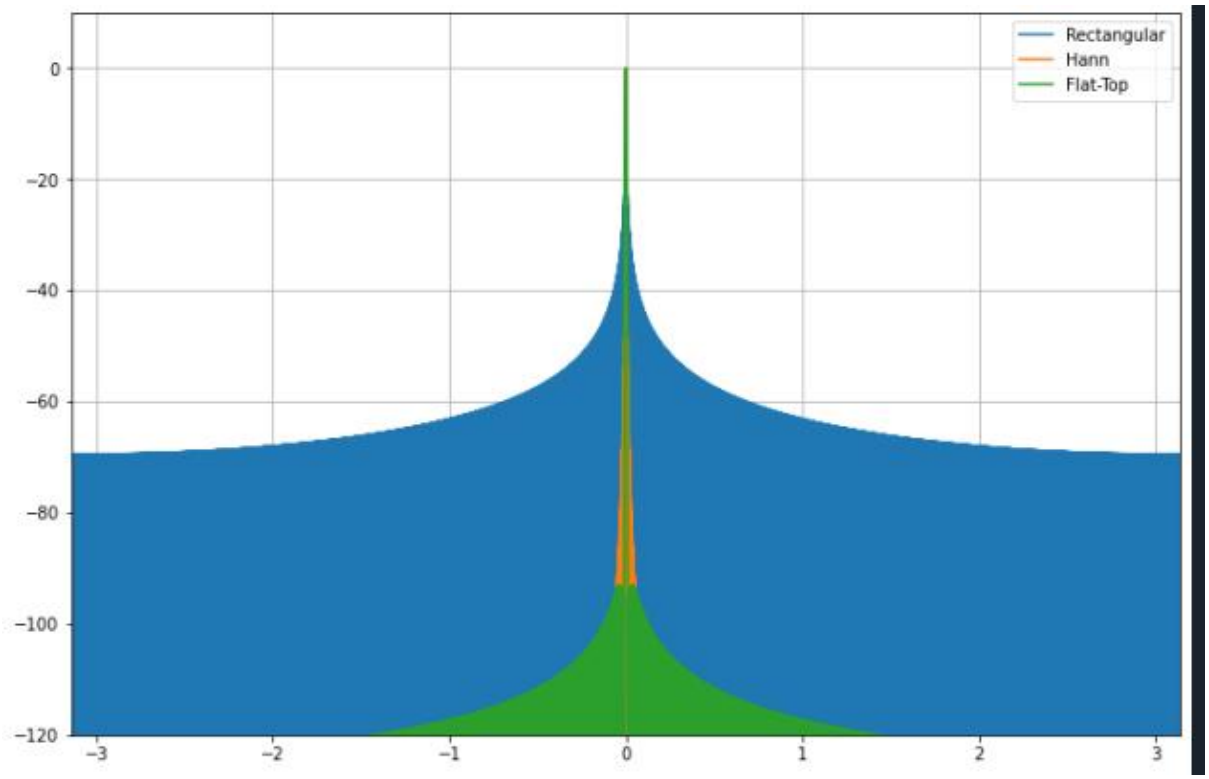
```
k = np.arange(N)

# Generate sine signals
x1 = A * np.sin(2 * np.pi * f1 * k / fs)
x2 = A * np.sin(2 * np.pi * f2 * k / fs)
x3 = A * np.sin(2 * np.pi * f3 * k / fs)

# Generate window functions
wrect = np.ones(N)                  # Rectangular window
whann = hann(N, sym=False)          # Hann window
wflattop = flattop(N, sym=False)    # Flat-top window

# Apply windows to signals
X1wrect = fft(x1 * wrect)
X2wrect = fft(x2 * wrect)
X3wrect = fft(x3 * wrect)

X1whann = fft(x1 * whann)
X2whann = fft(x2 * whann)
X3whann = fft(x3 * whann)

X1wflattop = fft(x1 * wflattop)
X2wflattop = fft(x2 * wflattop)
X3wflattop = fft(x3 * wflattop)

# Define DFT normalization function
def fft2db(X):
    N = X.size
    Xtmp = 2 / N * X  # Normalize for sine amplitudes
    Xtmp[0] /= 2      # bin for f=0 Hz exists only once
    if N % 2 == 0:
        Xtmp[N//2] /= 2  # fs/2 bin exists only once for even N
    return 20 * np.log10(np.abs(Xtmp))

# Frequency vector for DFT
df = fs / N
f = np.arange(N) * df
```

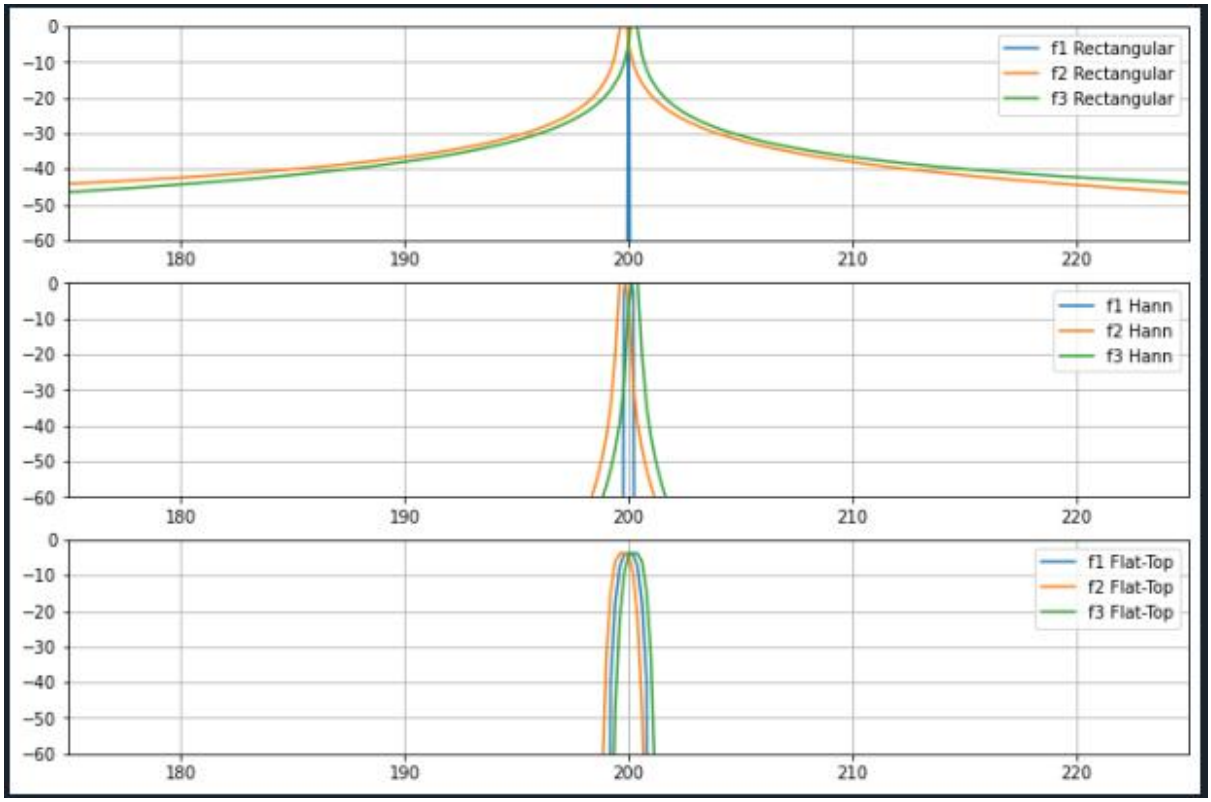
```
# Plot normalized DFT spectra (175 Hz to 225 Hz)
plt.figure(figsize=(12, 8))
plt.subplot(3, 1, 1)
plt.plot(f, fft2db(X1wrect), label='f1 Rectangular')
plt.plot(f, fft2db(X2wrect), label='f2 Rectangular')
plt.plot(f, fft2db(X3wrect), label='f3 Rectangular')
plt.xlim(175, 225)
plt.ylim(-60, 0)
plt.grid(True)
plt.legend()

plt.subplot(3, 1, 2)
plt.plot(f, fft2db(X1whann), label='f1 Hann')
plt.plot(f, fft2db(X2whann), label='f2 Hann')
plt.plot(f, fft2db(X3whann), label='f3 Hann')
plt.xlim(175, 225)
plt.ylim(-60, 0)
plt.grid(True)
plt.legend()

plt.subplot(3, 1, 3)
plt.plot(f, fft2db(X1wflattop), label='f1 Flat-Top')
plt.plot(f, fft2db(X2wflattop), label='f2 Flat-Top')
plt.plot(f, fft2db(X3wflattop), label='f3 Flat-Top')
plt.xlim(175, 225)
plt.ylim(-60, 0)
plt.grid(True)
plt.legend()
plt.show()

# DTFT-like spectra using zero-padding
def winDTFTdB(w):
    N = w.size
    Nz = 100 * N  # Zero-padding length
    W = np.zeros(Nz)
    W[0:N] = w
```

```
    W = np.abs(fftshift(fft(W)))  # FFT and shift
    W /= np.max(W)  # Normalize to mainlobe maximum
    W = 20 * np.log10(W)  # Convert to dB
    Omega = 2 * np.pi / Nz * np.arange(Nz) - np.pi  # Digital frequencies
    return Omega, W

# Plot window DTFT spectra normalized to mainlobe maximum
Omega, Wrect = winDTFTdB(wrect)
Omega, Whann = winDTFTdB(whann)
Omega, Wflattop = winDTFTdB(wflattop)

plt.figure(figsize=(12, 8))
plt.plot(Omega, Wrect, label='Rectangular')
plt.plot(Omega, Whann, label='Hann')
plt.plot(Omega, Wflattop, label='Flat-Top')
plt.xlim(-np.pi, np.pi)
plt.ylim(-120, 10)
plt.grid(True)
plt.legend()
plt.show()
```

Link to remote repostorium: https://github.com/sensorbtf/Digital-signal-processing

3. Outcomes:


Rectangular
Hann
Flat-Top
0
−20
−40
−60
−80
−100
−120
−3
−2
−1
0
1
2
3


f1 Rectangular
f2 Rectangular
f3 Rectangular
f1 Hann
f2 Hann
f3 Hann
f1 Flat-Top
f2 Flat-Top
f3 Flat-Top
0
−10
−20
−30
−40
−50
−60
180
190
200
210
220

5. Interpretation of the Results:

DFT Spectra:

1. The rectangular window shows more side lobes and spectral leakage, making it the worst choice in terms of frequency resolution.
2. The Hann window shows fewer side lobes and better attenuation but still allows some leakage.
3. The flat-top window has the best suppression of side lobes, resulting in cleaner spectra but with a wider main lobe.

DTFT Spectra:

1. The flat-top window has the best main lobe suppression, but at the cost of a wider lobe.
2. The Hann window performs better than the rectangular window in both side lobe suppression and main lobe width.

Why do the results for F1 and F2 differ?

1. The signals at f1 = 300 Hz and f2 = 300.25 Hz differ because their frequency spacing is very small (0.25 Hz). This small frequency difference is better resolved by windows that reduce spectral leakage, such as the Hann and flat-top windows, compared to the rectangular window, which spreads the energy across a broader range of frequencies due to leakage.