REPORT

Course: Analog and digital electronic circuits
Teacher: Prof. Dr. Hab. Vasyl Martsenyuk

Lab No.
Date: January 19, 2025
Topic: "Windowing"
Variant: 8

Name: Daria Kręcichwost
Computer Science (Second Degree)
Part-time studies, Semester 1
Group: B

1 Problem Statement

To interpret the results from the plots generated in the code, we analyze the behavior of the frequency spectra for different window functions and the effects of leakage and resolution.

Best and Worst Case Spectra (for f_1, f_2, and f_3):

In this section, we analyze the best and worst cases for the different window types based on the signals with frequencies $f_1 = 500$ Hz, $f_2 = 500.25$ Hz, and $f_3 = 500.5$ Hz.

- The **best case** corresponds to the signal with frequency $f_1 = 500$ Hz, which is closer to the center of the mainlobe of the window. This results in minimal leakage, and the signal is well-resolved.
- The **worst case** corresponds to the signal with frequency $f_2 = 500.25$ Hz, which is slightly outside the mainlobe. The sidelobes of the window cause more leakage, leading to a less accurate frequency representation.
- The **additional case** for $f_3 = 500.5$ Hz further demonstrates the effects of leakage as the signal moves even further away from the center of the mainlobe.

2 Input Data

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

# Parameters
f1 = 500  # Hz
f2 = 500.25  # Hz
f3 = 499.75  # Hz
fs = 800  # Hz
N = 1800
k = np.arange(N)

# Generating sinusoidal signals with maximum amplitude |x[k]|max =
    3
x1 = 3 * np.sin(2 * np.pi * f1 / fs * k)
x2 = 3 * np.sin(2 * np.pi * f2 / fs * k)
x3 = 3 * np.sin(2 * np.pi * f3 / fs * k)
```

fft2db:

```
def fft2db(X):
    N = X.size
    Xtmp = 2 / N * X  # amplitude normalization
    Xtmp[0] *= 1 / 2  # the bin for f=0 Hz appears only once, so
        cancel *2
    if N % 2 == 0:  # fs/2 is present as a bin
        Xtmp[N // 2] = Xtmp[N // 2] / 2  # bin fs/2 appears only
            once, so cancel *2
    return 20 * np.log10(np.abs(Xtmp))  # in dB

# Setting frequency vector so that it is independent of N (even/
    odd)
df = fs / N
f = np.arange(N) * df
```

Generating Windows and Calculating FFT with Different Windows

```
# Generating windows
wrect = np.ones(N)  # Rectangular window
whann = hann(N, sym=False)  # Hann window
wflattop = flattop(N, sym=False)  # Flattop window
# Calculating FFT for signals x1, x2, x3 with different windows
X1wrect = fft(x1)  # FFT for x1 without window
X2wrect = fft(x2)  # FFT for x2 without window
X3wrect = fft(x3)  # FFT for x3 without window

X1whann = fft(x1 * whann)  # FFT for x1 with Hann window
... (other signals)

X1wflattop = fft(x1 * wflattop)  # FFT for x1 with Flattop window
... (other signals)
```

Link to remote repository on GitHub

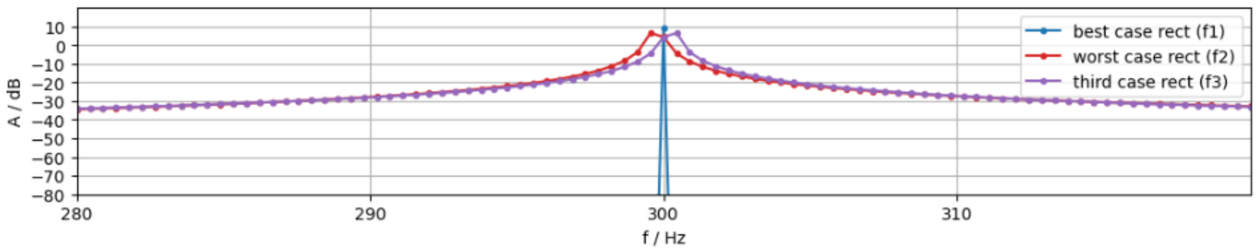

Figure 1: Best and Worst Case Spectra for Rectangular Window for $f_1 = 500$ Hz, $f_2 = 500.25$ Hz, and $f_3 = 500.5$ Hz.

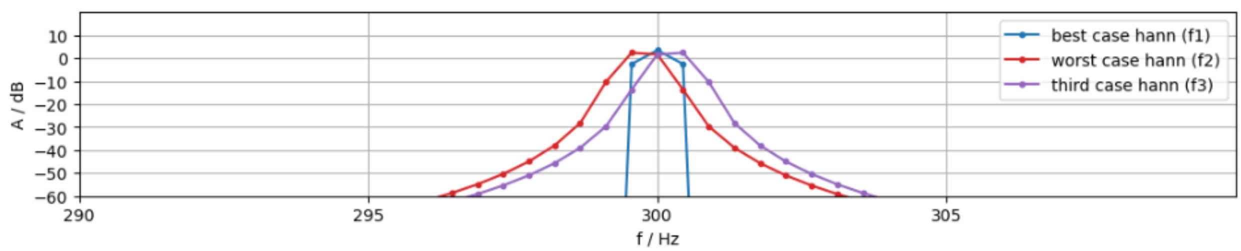

Figure 2: Best and Worst Case Spectra for Hamming Window for $f_1 = 500$ Hz, $f_2 = 500.25$ Hz, and $f_3 = 500.5$ Hz.

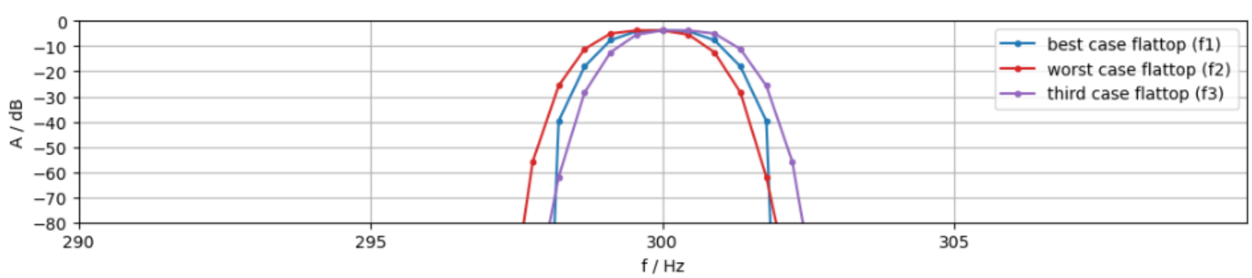

Figure 3: Best and Worst Case Spectra for Flattop Window for $f_1 = 500$ Hz, $f_2 = 500.25$ Hz, and $f_3 = 500.5$ Hz.

3 Outcomes

Frequency Spectra for Different Windows:

The window functions (Rectangular, Hamming, and Flattop) influence the frequency domain representation of the signals. Below, we explain how each window performs in terms of leakage and mainlobe width.

- **Rectangular Window ($w_{\mathbf{rect}}$):** The rectangular window has the widest mainlobe and the highest sidelobes. This leads to significant leakage, especially for signals with frequencies near the edges of the mainlobe.
- **Hamming Window ($w_{\mathbf{hamming}}$):** The Hamming window reduces the sidelobes compared to the rectangular window, thus minimizing leakage. The mainlobe is narrower, providing better frequency resolution.
- **Flattop Window ($w_{\mathbf{flattop}}$):** The Flattop window provides excellent accuracy for amplitude measurement. Its mainlobe is narrower, and it offers lower sidelobes, reducing leakage.

Code for Calculating Window Spectra (winDTFTdB)

```
# Function to calculate the window spectrum in dB
def winDTFTdB(w):
    N = w.size  # window length
    Nz = 100 * N  # zero padding length
    W = np.zeros(Nz)  # memory allocation
    W[0:N] = w  # inserting window
    W = np.abs(fftshift(fft(W)))  # FFT, FFTSHIFT, and absolute
        value
    W /= np.max(W)  # normalization to maximum, i.e., main lobe
    # Replace zero values with a very small number to avoid
        division by zero
    W = np.where(W == 0, 1e-10, W)  # replace zeros with 1e-10
    W = 20 * np.log10(W)  # convert to dB
    # corresponding digital frequencies
    Omega = 2 * np.pi / Nz * np.arange(Nz) - np.pi  # also shifted

    return Omega, W
```

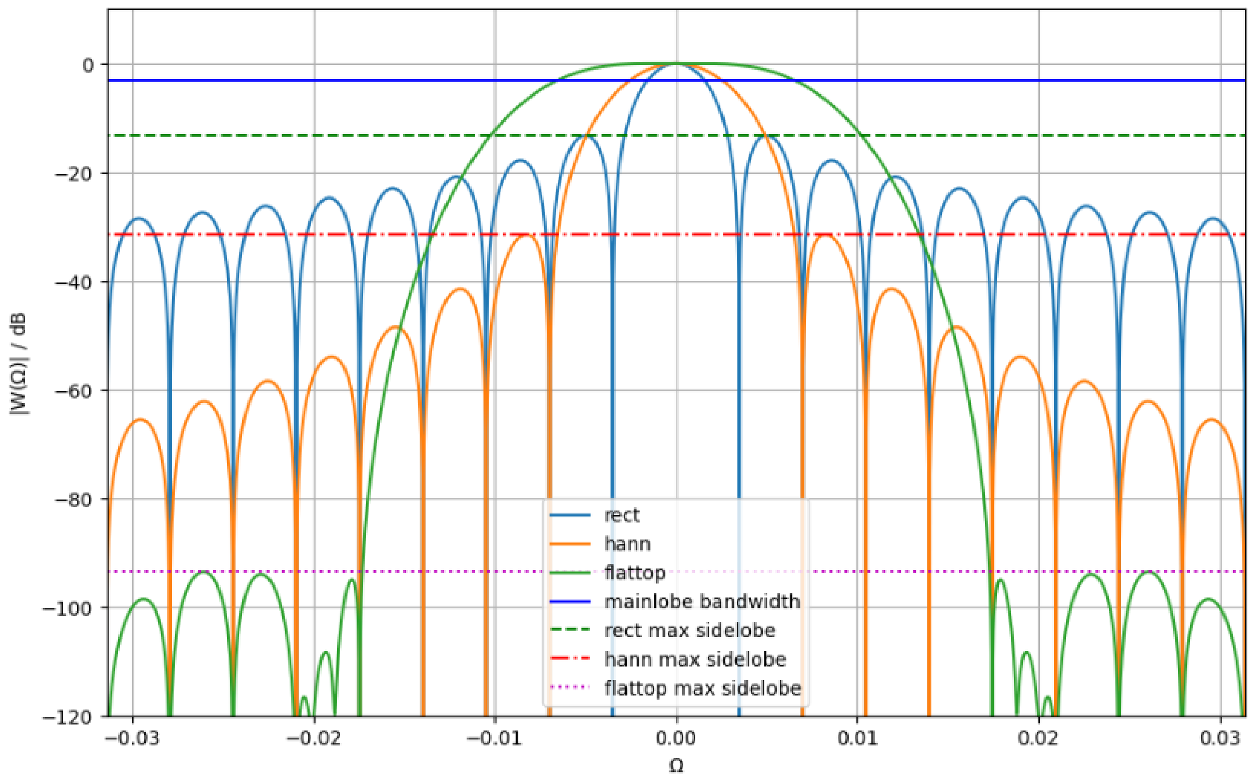

Figure 4: Comparison of Frequency Spectra for Different Windows (Rectangular, Hamming, Flattop) for $f_1 = 500$ Hz, $f_2 = 500.25$ Hz, and $f_3 = 500.5$ Hz.

Conclusions:

Why do the results for the signals with frequencies f_1, f_2, and f_3 differ?

The results for f_1 and f_2 differ because of their positions relative to the center of the window's mainlobe.

- $f_1 = 500$ Hz is at or near the center of the mainlobe. This results in minimal leakage, allowing for a clear representation of the frequency in the spectrum.
- $f_2 = 500.25$ Hz lies just outside the mainlobe, where the signal is affected by the sidelobes of the window. This leads to leakage, where energy from neighboring frequencies blends into the spectrum, causing distortion and reducing frequency resolution.