REPORT

Zajęcia: Analog and digital electronic circuits

Teacher: prof. dr hab. Vasyl Martsenyuk

Lab 2

28 October 2023

Topic: "Windowing"

Variant 1

Tomasz Steblik

Informatyka II stopień,

niestacjonarne,

1 semestr,

Gr. 1B

1. Problem statement:

Generate three sine signals of given f1, f2, and f3 and amplitude |x[k]|max
for the sampling frequency fs in the range of $0 \le k < N$.
Plot: 1 1. the "normalized" level of the DFT spectra. 2. the window DTFT spectra normalized to their mainlobe maximum. The intervals for f , Ω, and amplitudes should be chosen by yourself for the best interpretation purposes.
Interpret the results of the figures obtained regarding the best and worst case for the different windows. Why do the results for the signals with frequencies f1 and f2 differ?

2. Input data:

No	f_1	f_2	f_3	$\lvert x[k]\rvert_{\max}$	f_s	N
1	300	300.25	299.75	2	400	2000
2	400	400.25	399.75	2	600	3000
3	500	500.25	499.75	2	800	1800
4	600	600.25	599.75	2	500	2000
5	300	300.25	299.75	2	400	2000
6	600	600.25	599.75	3	800	2000
7	400	400.25	399.75	3	600	3000
8	500	500.25	499.75	3	800	1800
9	600	600.25	599.75	3	500	2000
10	300	300.25	299.75	3	400	2000
11	200	200.25	199.75	4	400	2000
12	400	400.25	399.75	4	600	3000
13	500	500.25	499.75	4	800	1800
14	600	600.25	599.75	4	500	2000
15	500	500.25	499.75	4	800	2000

Table 1: Variants

3. Commands used (or GUI):

Link to remote repozytorium: https://github.com/TomaszSteblik/Aadec_2

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
from numpy.fft import fft , ifft , fftshift
from scipy.signal.windows import hann , flattop
```

In [2]:
```
f1 = 300 # Hz
f2 = 300.25 # Hz
f3 = 299.75 # Hz
fs = 400 # Hz
N = 2000
k = np.arange(N)
x1 = np.sin(2*np.pi*f1/fs*k)
x2 = np.sin(2*np.pi*f2/fs*k)
x3 = np.sin(2*np.pi*f3/fs*k)
```

In [3]:
```
wrect = np.ones(N)
whann = hann(N, sym=False)
wflattop = flattop(N, sym=False)
plt.plot(wrect, 'C0o-', ms=3, label='rect')
plt.plot(whann, 'C1o-', ms=3, label='hann')
plt.plot(wflattop, 'C2o-', ms=3, label='flattop')
plt.xlabel(r'$k$')
plt.ylabel(r'window$w[k]$')
plt.xlim(0, N)
plt.legend()
plt.grid(True)
```

In [4]:
```
X1wrect = fft(x1)
X2wrect = fft(x2)
X3wrect = fft(x3)
X1whann = fft(x1 * whann)
X2whann = fft(x2 * whann)
X3whann = fft(x3 * whann)
X1wflattop = fft(x1 * wflattop)
X2wflattop = fft(x2 * wflattop)
X3wflattop = fft(x3 * wflattop)
```

In [5]:
```
# this handling is working for N even and odd:
def fft2db(X):
    N = X.size
    Xtmp = 2/N * X  # independent of N, norm for sine amplitudes
    Xtmp[0] *= 1/2  # bin for f=0 Hz is existing only once,
    # so can cancel *2 from above
    if N % 2 == 0:  # fs/2 is included as a bin
        # fs/2 bin is existing only once, so can cancel *2 from above
        Xtmp[N//2] = Xtmp[N//2] / 2
    return 20 * np.log10(np.abs(Xtmp))  # in dB
```

In [6]:
```
# set up of frequency vector this way is independent of N even/odd:
df = fs / N
f = np.arange(N) * df
```

In [7]:
```
plt.figure(figsize=(16/1.5, 10/1.5))
plt.subplot(3, 1, 1)
plt.plot(f, fft2db(X1wrect), 'C0o-', ms=3, label='best case rect')
plt.plot(f, fft2db(X2wrect), 'C2o-', ms=3, label='central case rect')
plt.plot(f, fft2db(X3wrect), 'C3o-', ms=3, label='worst case rect')
plt.xlim(75, 125)
plt.ylim(-60, 0)
plt.xticks(np.arange(75, 125, 5))
plt.yticks(np.arange(-60, 10, 10))
plt.legend()
plt.ylabel('A / dB')
plt.grid(True)
plt.subplot(3, 1, 2)
plt.plot(f, fft2db(X1whann), 'C0o-', ms=3, label='best case hann')
plt.plot(f, fft2db(X2whann), 'C2o-', ms=3, label='central case hann')
plt.plot(f, fft2db(X3whann), 'C3o-', ms=3, label='worst case hann')
plt.xlim(75, 125)
plt.ylim(-60, 0)
plt.xticks(np.arange(75, 125, 5))
plt.yticks(np.arange(-60, 10, 10))
plt.legend()
plt.ylabel('A / dB')
plt.grid(True)
plt.subplot(3, 1, 3)
plt.plot(f, fft2db(X1wflattop), 'C0o-', ms=3, label='best case flattop')
plt.plot(f, fft2db(X2wflattop), 'C2o-', ms=3, label='central case flattop')
plt.plot(f, fft2db(X3wflattop), 'C3o-', ms=3, label='worst case flattop')
plt.xlim(75, 125)
plt.ylim(-60, 0)
plt.xticks(np.arange(75, 125, 5))
plt.yticks(np.arange(-60, 10, 10))
plt.legend()
plt.xlabel('f / Hz')
plt.ylabel('A / dB')
plt.grid(True)
plt.show()
```

In [8]:
```
def winDTFTdB(w):
    N = w.size  # get window length
    Nz = 100 * N  # zero-padding length
    W = np.zeros(Nz)  # allocate RAM
    W[0:N] = w  # insert window
    W = np.abs(fftshift(fft(W)))  # fft, fftshift and magnitude
    W /= np.max(W)  # normalize to maximum, i.e., the mainlobe maximum here
    W = 20 * np.log10(W)  # get level in dB
    # get appropriate digital frequencies
    Omega = 2 * np.pi / Nz * np.arange(Nz) - np.pi  # also shifted
    return Omega, W
```

In [9]:
```
plt.plot([-np.pi, +np.pi], [-3.01, -3.01], 'gray')  # mainlobe bandwidth
plt.plot([-np.pi, +np.pi], [-13.3, -13.3], 'gray')  # rect max sidelobe
plt.plot([-np.pi, +np.pi], [-31.5, -31.5], 'gray')  # hann max sidelobe
plt.plot([-np.pi, +np.pi], [-93.6, -93.6], 'gray')  # flattop max sidelobe
Omega, W = winDTFTdB(wrect)
plt.plot(Omega, W, label='rect')
Omega, W = winDTFTdB(whann)
plt.plot(Omega, W, label='hann')
Omega, W = winDTFTdB(wflattop)
plt.plot(Omega, W, label='flattop')
plt.xlim(-np.pi, np.pi)
plt.ylim(-120, 10)
plt.xlim(-np.pi/100, np.pi/100)  # zoom into mainlobe
plt.xlabel(r'$\Omega$')
plt.ylabel(r'$|W(\Omega)|$ / dB')
plt.legend()
plt.grid(True)
plt.show()
```

4. Outcomes:

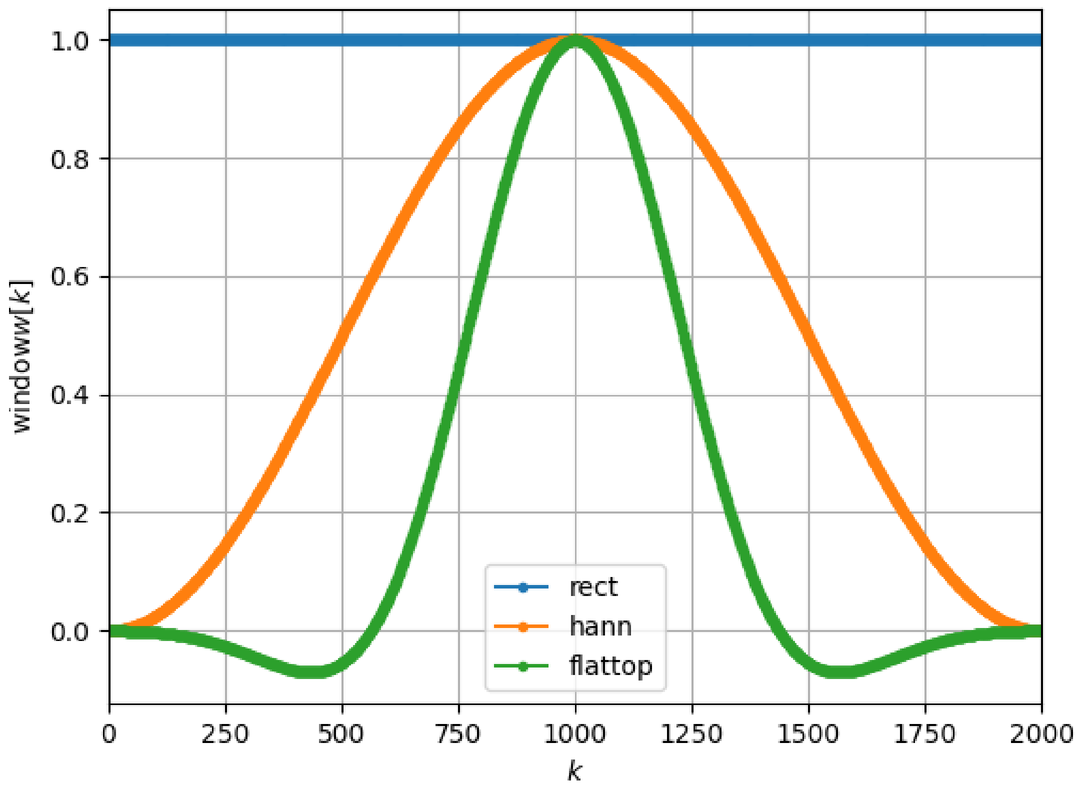

Figure 1. Obtained window signals over k

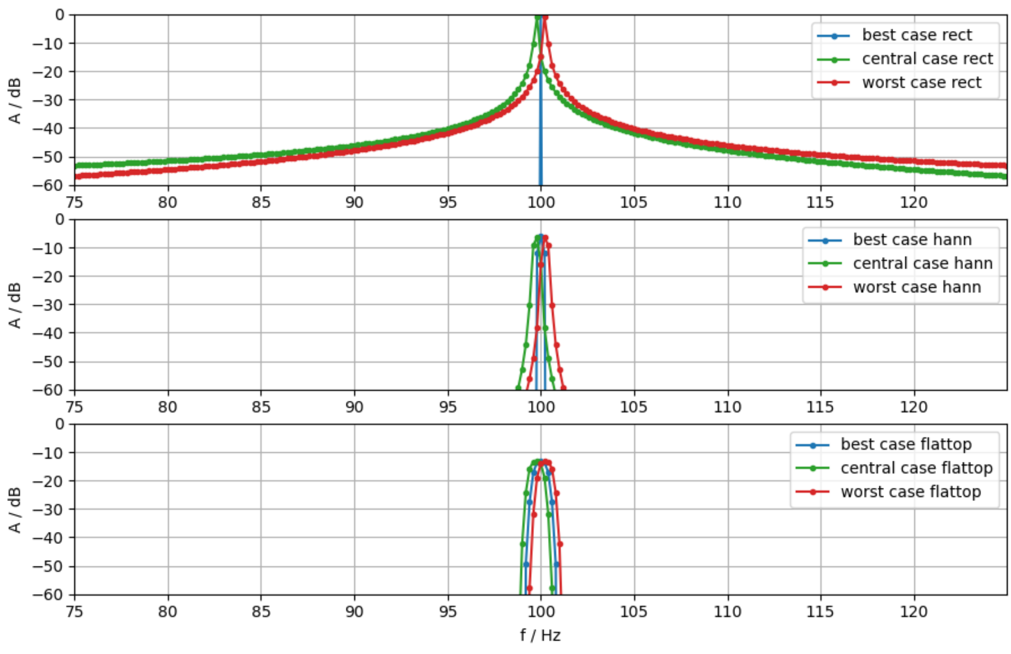

Figure 2. DFT spectra using FFT algorithm

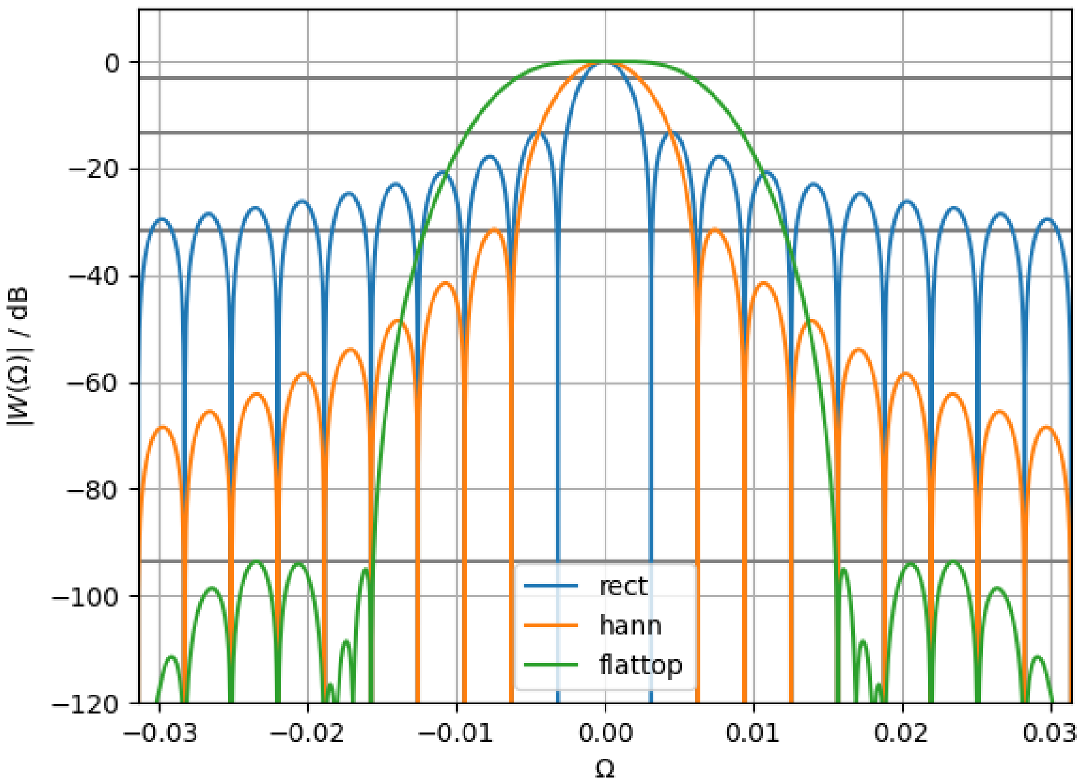

Figure 3. Window DTFT spectra normalized to their mainlobe maximum

5. Conclusions:

The differences in the results for the signals with frequencies f1 and f2 can be attributed to the spectral leakage and windowing effect in the Discrete Fourier Transform (DFT). Spectral leakage occurs when the signal's frequency does not exactly match one of the basis frequencies of the DFT. This can cause the signal's energy to "leak" into adjacent frequency bins, distorting the spectrum. The signals with frequencies f1 and f2 are likely experiencing different amounts of spectral leakage due to their frequencies relative to the DFT basis frequencies. This, combined with the effects of the window functions, can lead to differences in their DFT spectra.