Protocol for the Signals and Systems course project
Author: Sabína Gulčíková, xgulci00

Base tasks - Introduction

The following pages contain the outcomes of functions implemented according to the assigned tasks. The final output (simulated masks) is not perfect. I would advise you not to listen to the recordings, because for some reason they are incredibly amplified, which means you get scared any time the output record starts playing. By the end of this project, I was so terrified of playing them, that I kept on programming functions blindly. If you have already tried it, I apologise.

Structure:

`/audio` - contains all recordings, both input and output
`/src/plot` - contains all plotted graphs
`/src/extract` - contains seconds extracted from both mask-on and mask-off tones

1. Tone recordings

Tone		
Name	Length [samples]	Length [seconds]
maskoff_tone.wav	95915	05.99
maskon_tone.wav	102059	06.38

2. Sentence recordings

Sentence		
Name	Length [samples]	Length [seconds]
maskoff_sentence.wav	53931	03.37
maskon_sentence.wav	50517	03.16

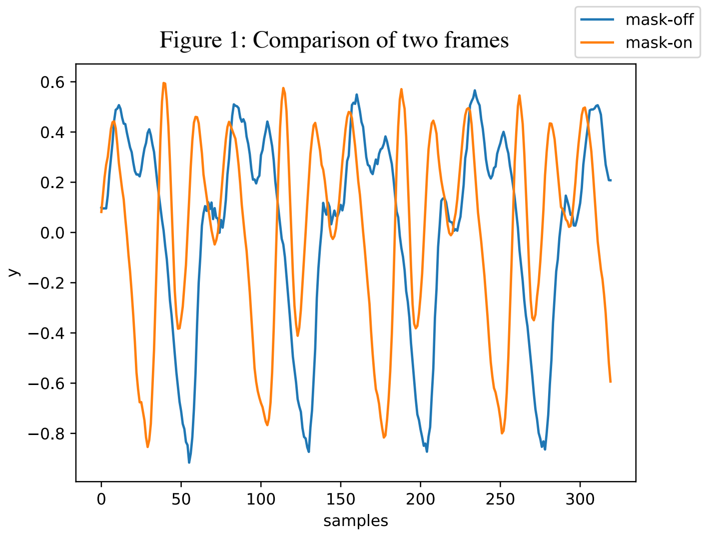
3. Extraction of frames

Extracted seconds were chosen manually from both recordings. The size of one frame in samples can be calculated by formula:

$$size = frame\_ms * samples\_per\_ms,$$

where *frame_ms* is the length of one frame in milliseconds, and *samples_per_ms* is number of samples per one millisecond ($F_s \times 10^{-3}$)

Figure 1: Comparison of two frames

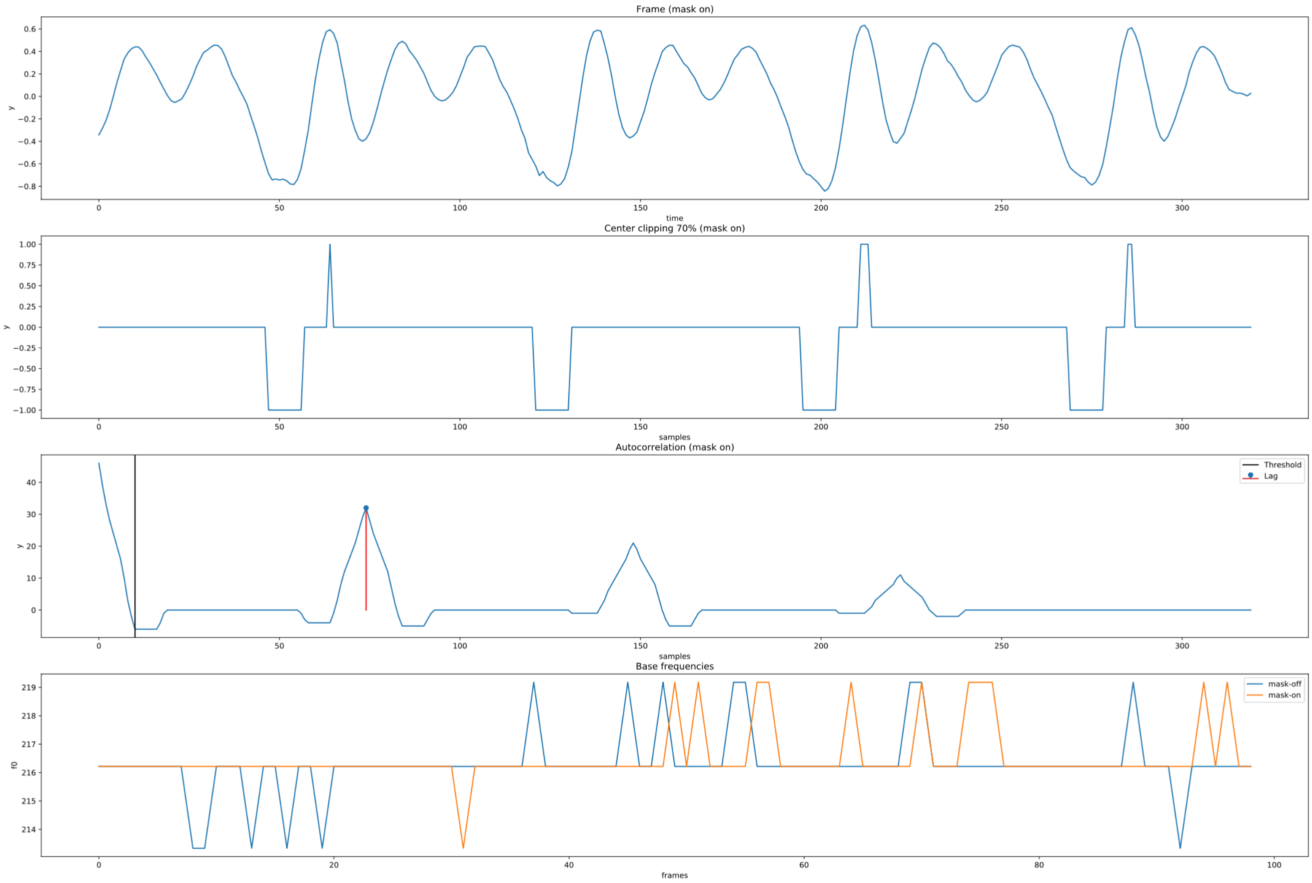


4. Similarity check

The table below contains calculated values of mean and variance. The graph shows center clipping and auto-correlation performed on one frame of the mask-on tone, as well as the comparison of base frequencies of both mask-on and mask-off tones.

Name	Mean	Variance
Mask-off f_0	216.281	1.21
Mask-on f_0	216.516	0.97

Figure 2: Center clipping, autocorrelation and base frequencies



Answer to the question:
The frequency is highly affected because the frequency of particular frame is calculated as f0 = lag / Fs. Any significant change in the numerator of the fraction results in the significant change of the result. Perhaps we could use some “error-smoothing” coefficient, by which the fraction would be multiplied, in order to prevent outstanding changes.

5. Discrete Fourier transform

Figure 3: Implementation of DFT

```
import numpy as np
def my_dft(x_n, zero_pad):
    #zero padding
    curr_len = len(x_n)
    to_pad = zero_pad - curr_len
    np.pad(x_n, [(0, 0), (0, to_pad)], mode='constant')

    N = len(x_n)
    x_k = []
    const = 0
    for k in range(N):
        for n in range(N):
            exponent = -2j*np.pi*k*n*(1/N)
            const += x_n[n] * np.exp(exponent)
        x_k.append(const)
        const = 0

    return x_k
```

Figure 4: mask-off spectogram

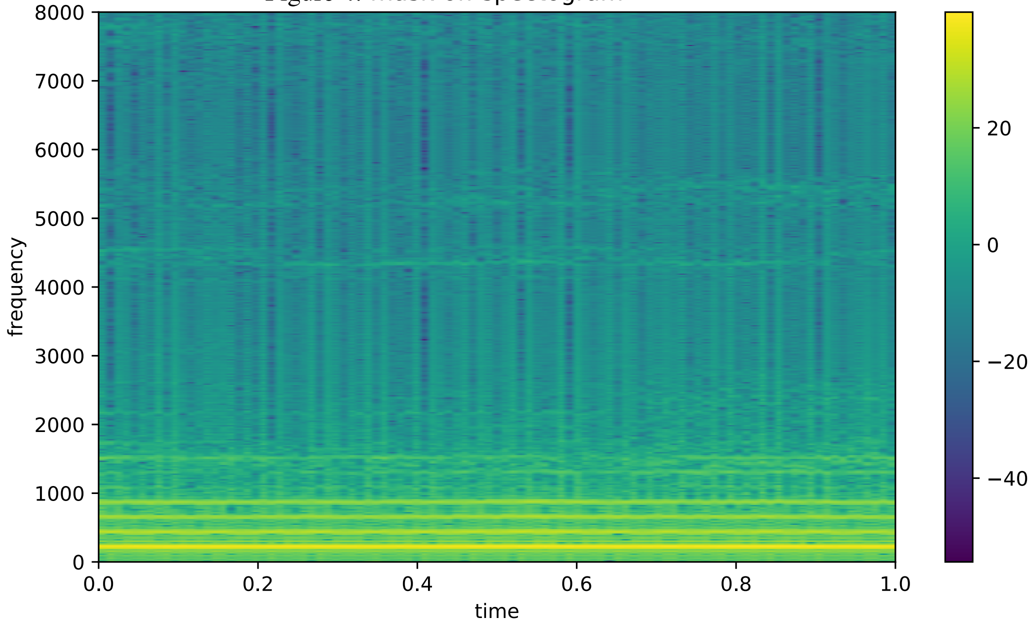
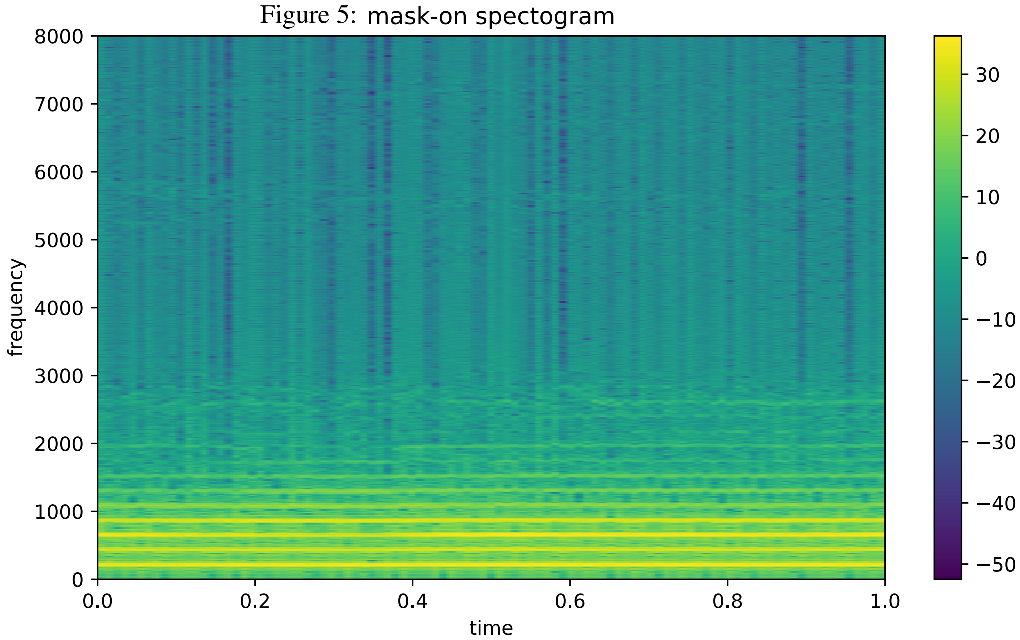


Figure 5: mask-on spectogram

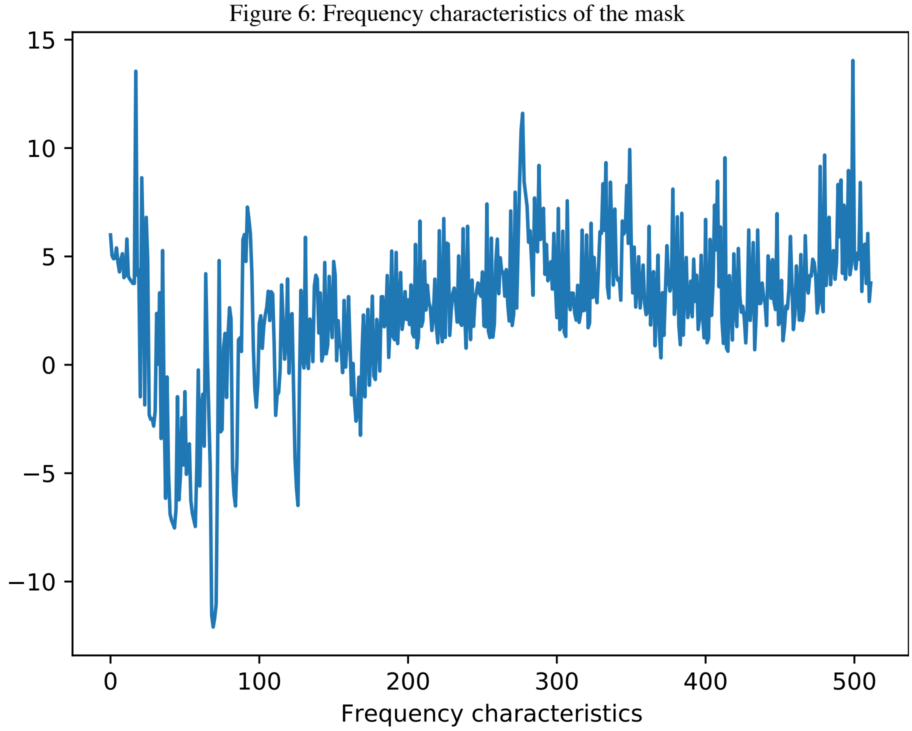


6. Frequency characteristics

Formula for calculating $H(e^{j\omega})$ is:

$$H(e^{j\omega}) = \frac{B(e^{j\omega})}{A(e^{j\omega})} = \frac{b[0] + b[1]e^{-j\omega} + \ldots + b[M]e^{-j\omega M}}{a[0] + a[1]e^{-j\omega} + \ldots + a[N]e^{-j\omega N}}$$

Figure 6: Frequency characteristics of the mask



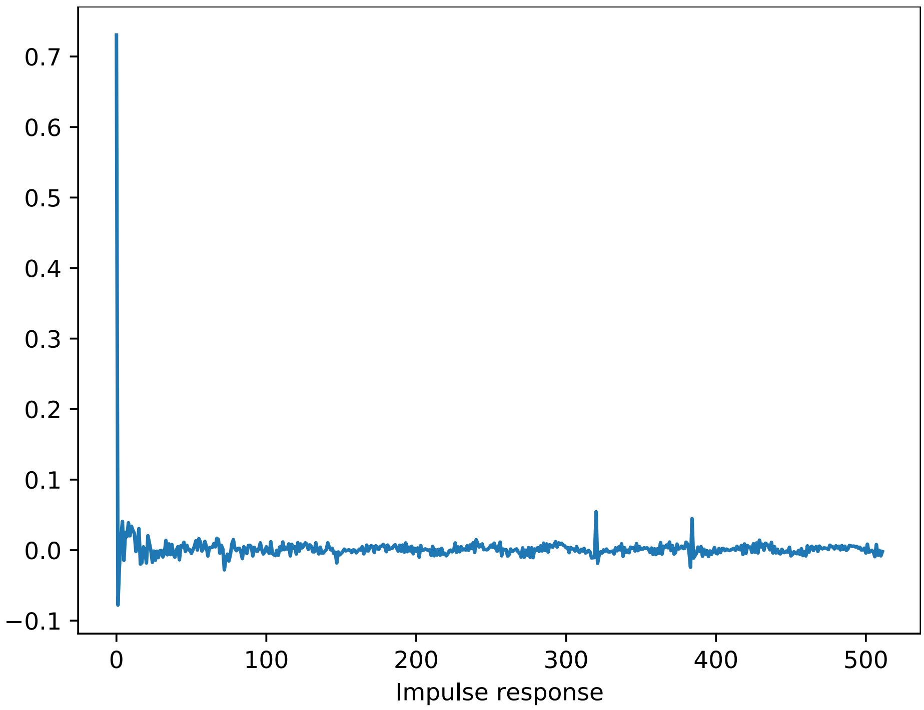
Comment: According to the resulting spectre, it is not very easy to predict the outcome after the application od this filter. Because of the effect that masks have on speech, it is expected for the final recording to be reduced, and a bit muffled. The mask should smooth out higher frequencies (might resemble low-pass filter). Since I have used three masks for recording, this filter should be a more “harsh” and higher frequencies should be eliminated completely.

7. Impulse response

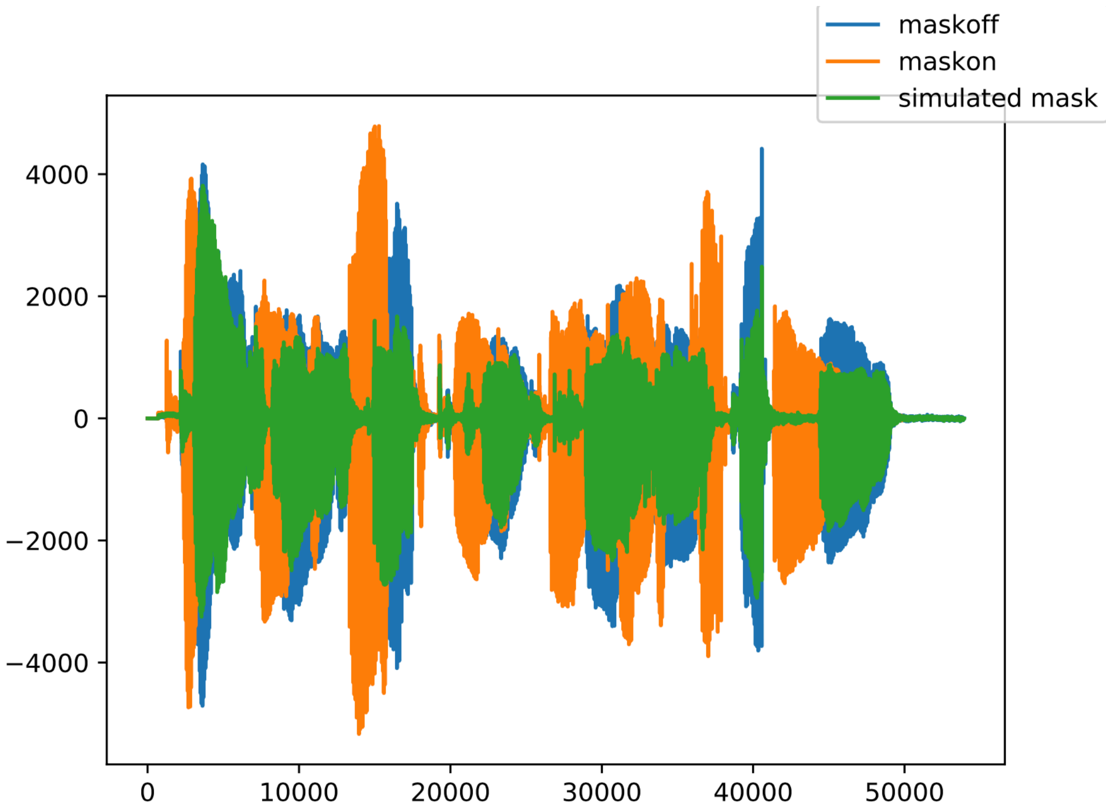
Figure 7: Implementation of IDFT

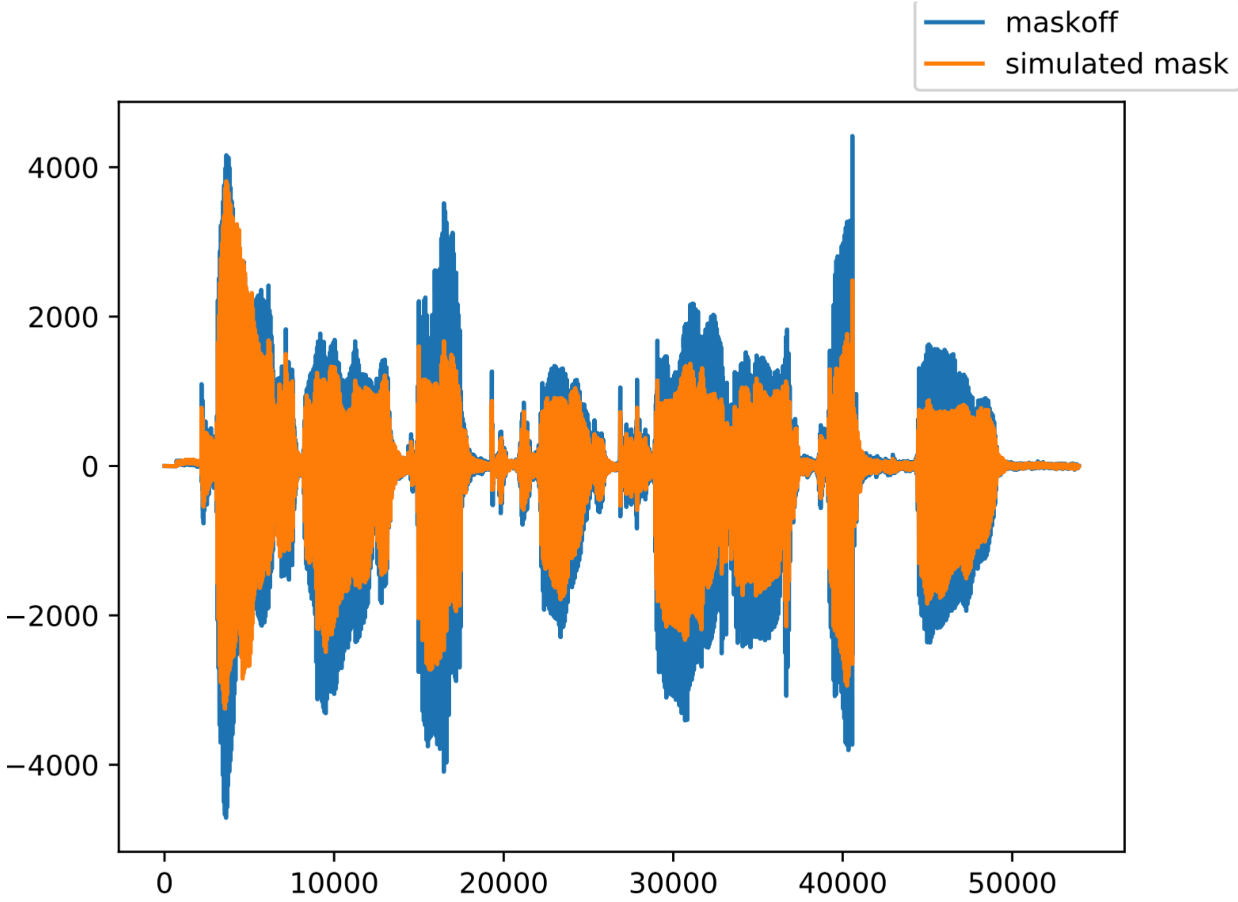
```
import numpy as np
def my_idft(x_k, zero_pad):
    # zero padding
    curr_len = len(x_k)
    to_pad = zero_pad - curr_len
    x_k = np.pad(x_k, [(0, 0), (0, to_pad)], mode='constant')

    N = len(x_k)
    x_n = []
    const = 0
    for n in range(N):
        for k in range(N):
            exponent = 2j*np.pi*k*n*(1/N)
            const += x_k[k]*np.exp(exponent)
        const /= N
        x_n.append(const)
        const = 0

    return x_n
```

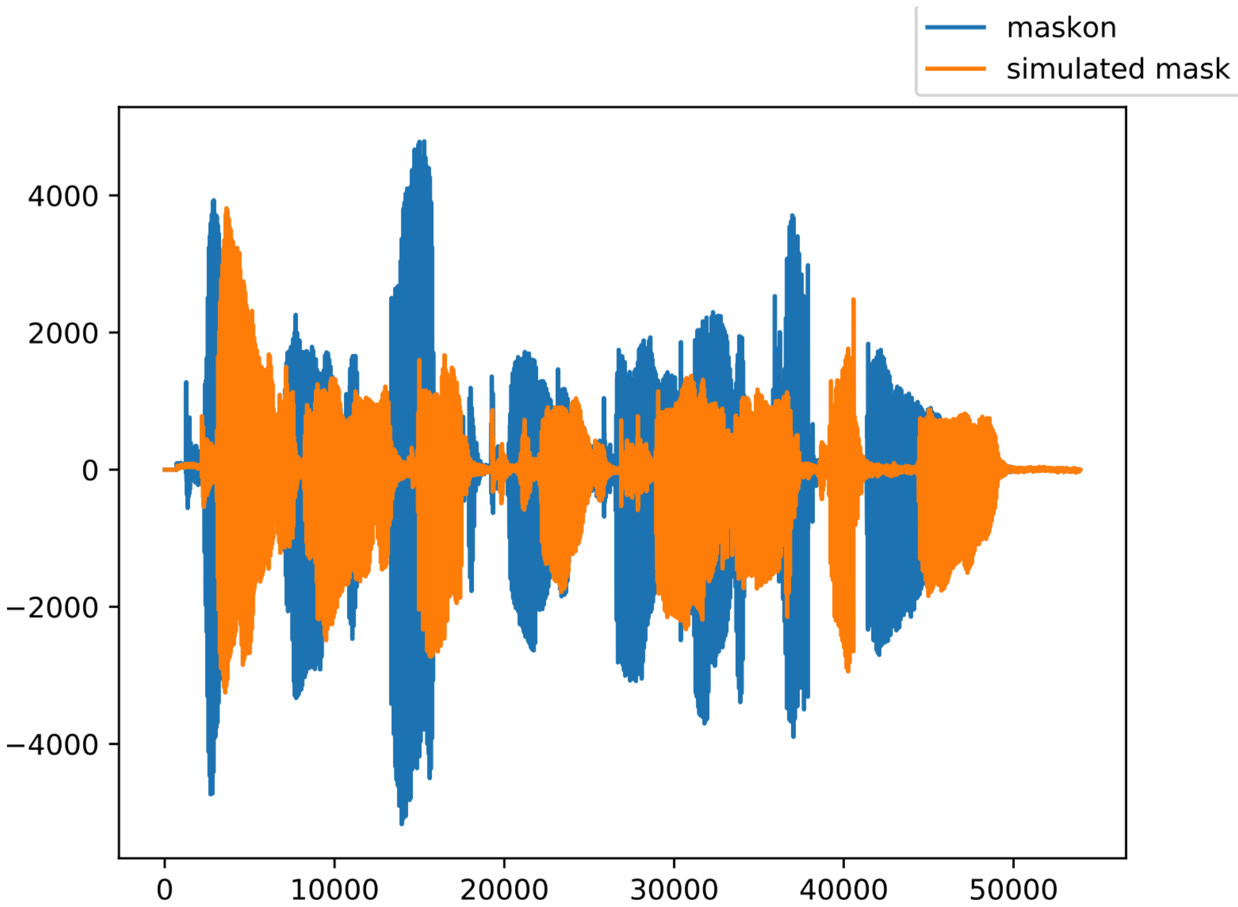
Figure 8: Impulse response



8. Mask simulation




maskoff
simulated mask
4000
2000
0
−2000
−4000
0
10000
20000
30000
40000
50000


maskon
simulated mask
4000
2000
0
−2000
−4000
0
10000
20000
30000
40000
50000

According to the graph showing us the sound recorded without mask, and its comparison to the simulated mask, we see that the higher frequencies were eliminated (nearly) successfully. Our expectations were pretty correct, and the outcome of the application of the simulated mask resembles more muffled and reduced version of the original recording.

In comparison to the real mask, we see that the simulated mask is way too harsh in some fragments, and eliminates even frequencies that were not modified by the real mask. Another difference is in the phase shift, whose removal is described in the last part of this protocol.

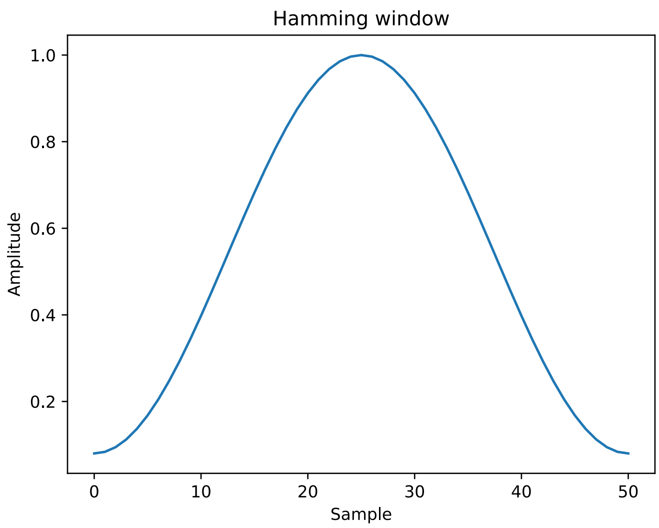
9. Conclusion

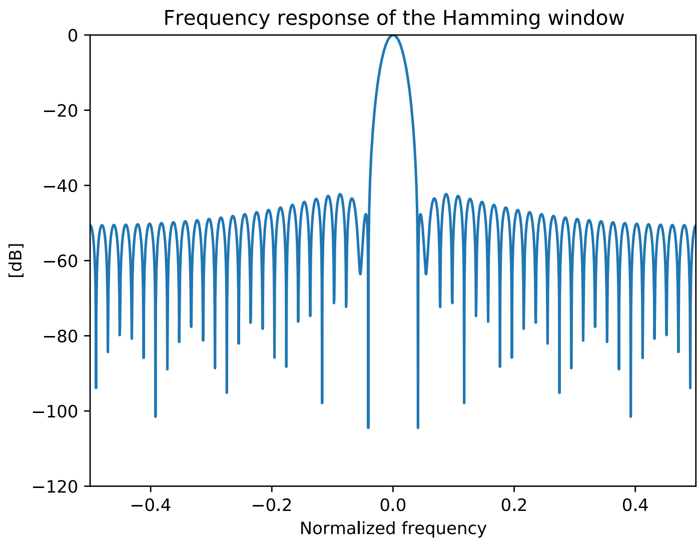
To conclude, working on this project was an amazing experience. It has given me a chance to find out what spectral analysis and working with signals looks like from the practical point of view. I have to admit that this project and topic seem very attractive, and even though my knowledge is very basic right now, and the filters do not work as well as they should, I will for sure try and improve my skills in the future. Thanks to this project I have discovered new interest of mine - signals.

Additional tasks:

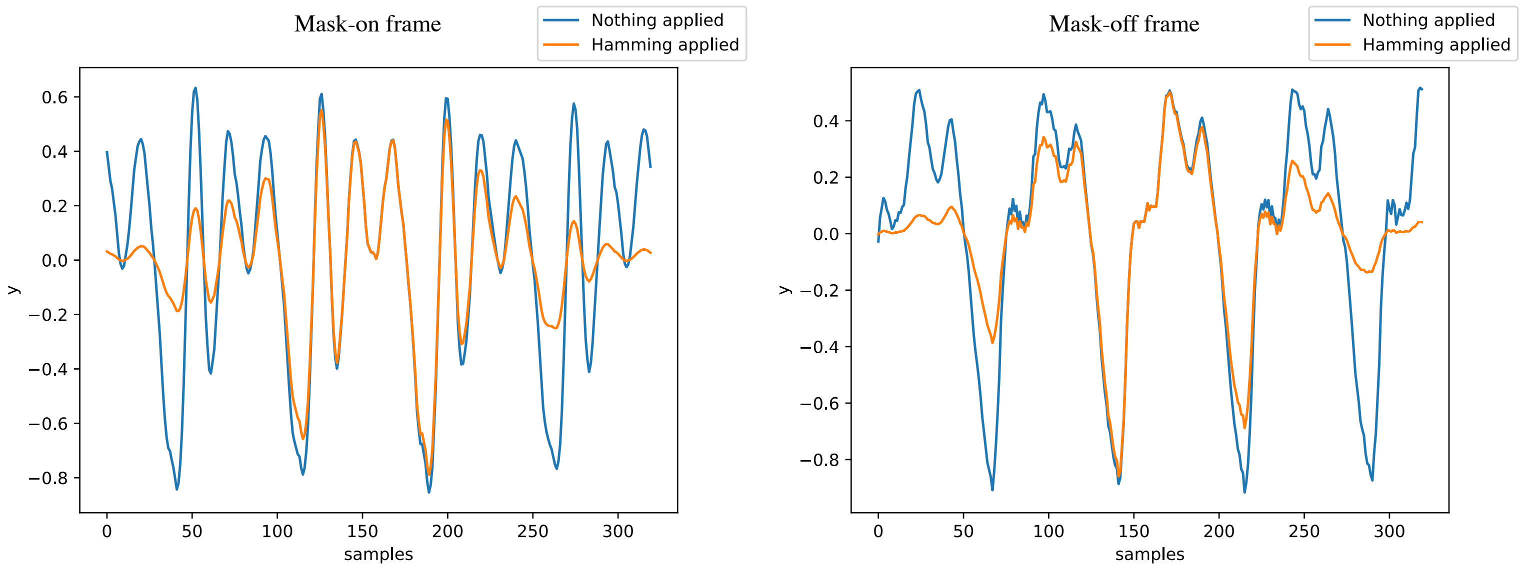
11. Window function

I have chosen the Hamming window for this task. You can find its characteristics in the graphs below.



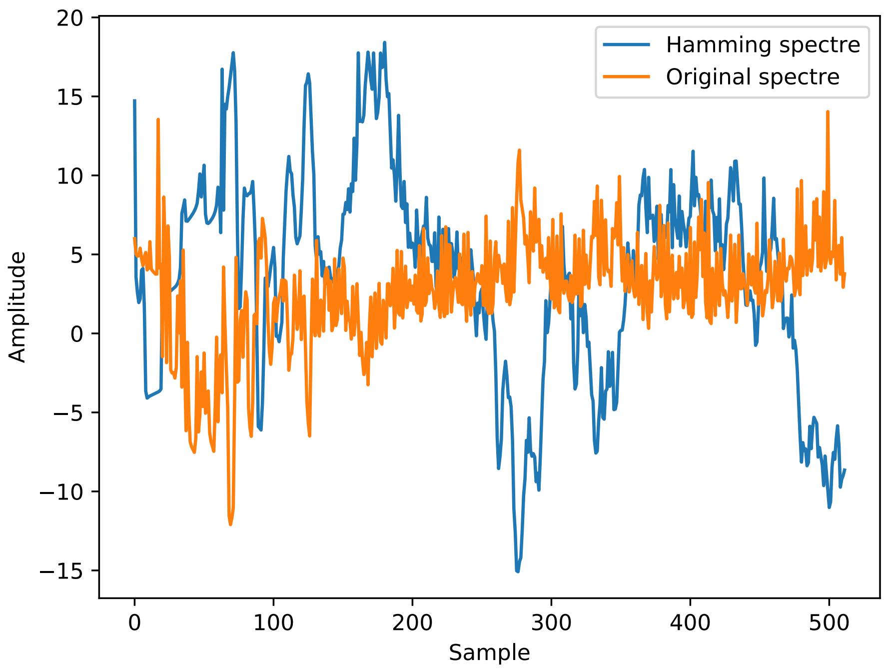


Comparison of frame before and after application of the Hamming window:

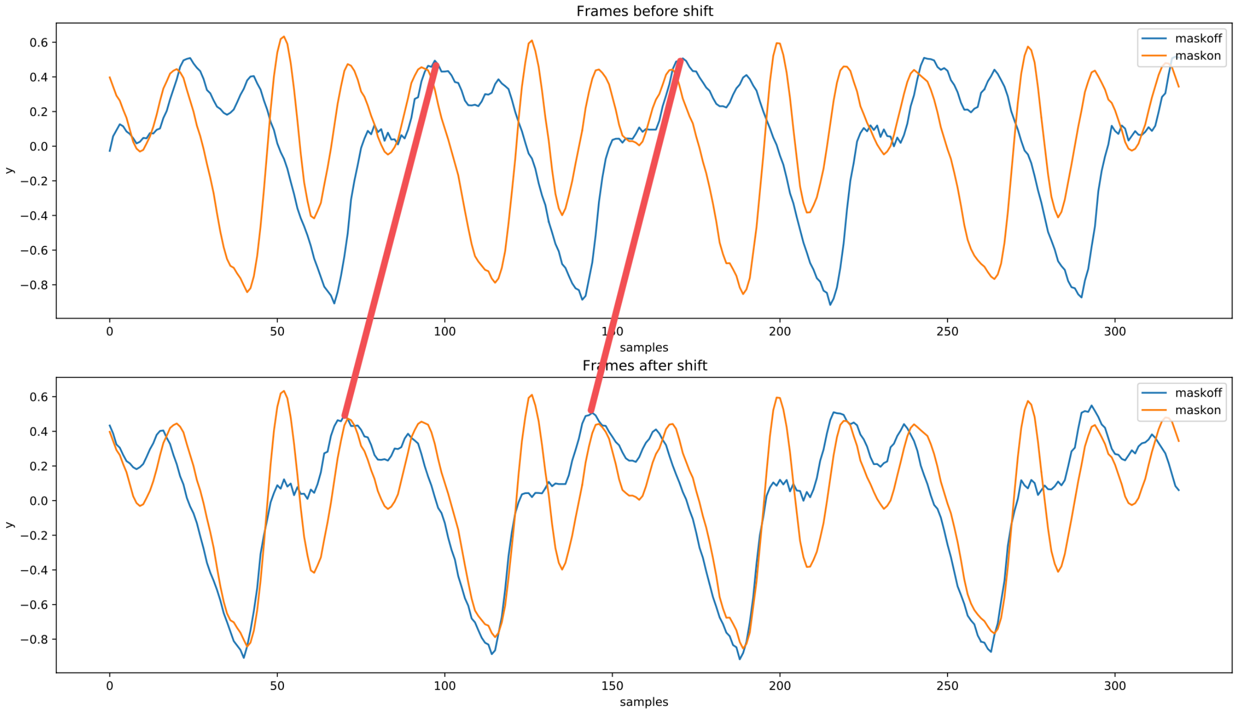


Window functions are usually used in spectral analysis, or when it comes to creation of impulse response filter. Application of a window function allows us to restrict any signal to a specific domain. Anything that does not belong to this domain is annulled. This phenomenon allows us to work with chosen part of the signal, for example with a single period.

In the following graph, we can see a comparison of frequency characteristics with and without application of the Hamming window:



15. Phase shift



The shift subtracted from the beginning of one recording, added to the shift subtracted from the end of another recording is very similar to the lag. In my opinion, this has something to do with the fact that phase shift was calculated as the maximum of the coefficients of correlation. In the auto-correlation which was done in the beginning of this project, we have chosen the value of lag similarly, as the index of maximal value. There might even be a direct relation between them: *lag = maximum(correlation coefficients) / 2*