Signály a systémy

Projekt 2020/2021

Zaťovič Martin (xzatov00)

1. a 2. úloha

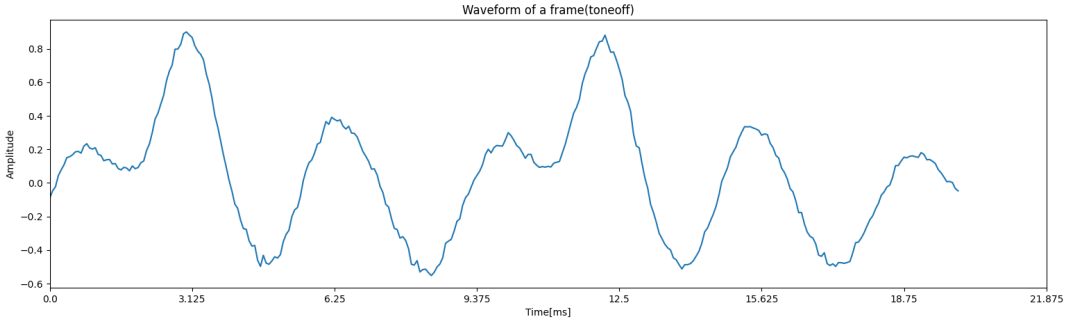
Názov	Dĺžka[s]	Dĺžka[vzorky]
maskoff_tone.wav	03.52	56255
maskon_tone.wav	6.66	106496
maskoff_sentence.wav	4.06	64958
maskon_sentence.wav	4.57	73116

3. úloha

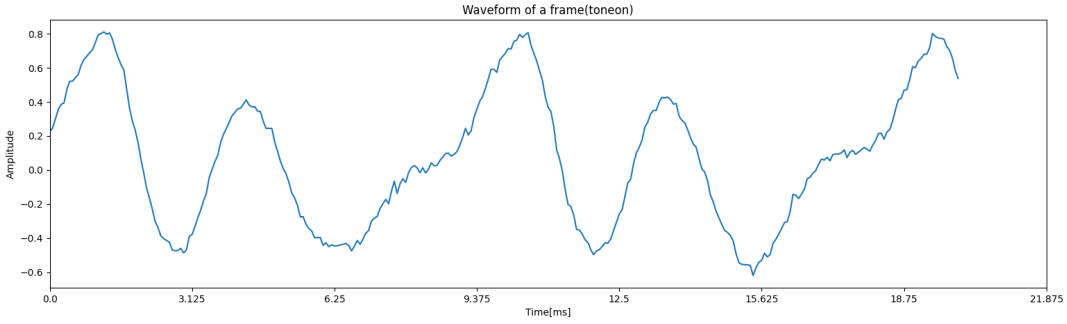
Vzorec pre výpočet veľkosti rámcu vo vzorkoch:

$$\begin{array}{ccccc} \text{veľkosť rámcu[vzorky]} & = & \text{veľkosť rámcu[s]} & * & \text{vzorkovacia frekvencia nahrávky} \\ N_f & = & t_f & * & F_s \\ N_f & = & 0.02 & * & 16000 \\ N_f & = & 320 & & \end{array}$$

Graf rámcu nahrávky tónu bez rúška:

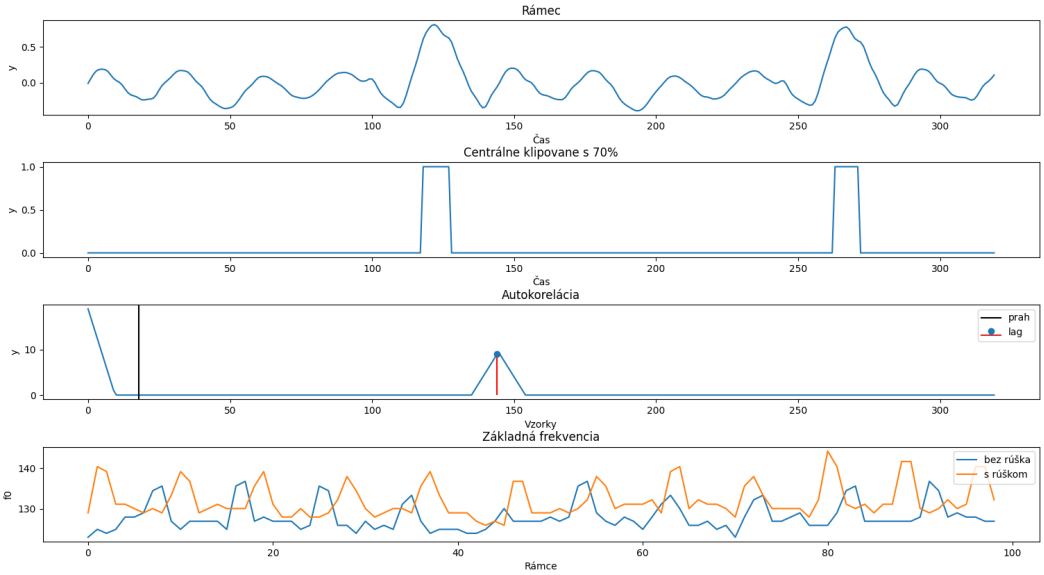


Graf rámcu nahrávky tónu s rúškom:



4. úloha

a)



b)

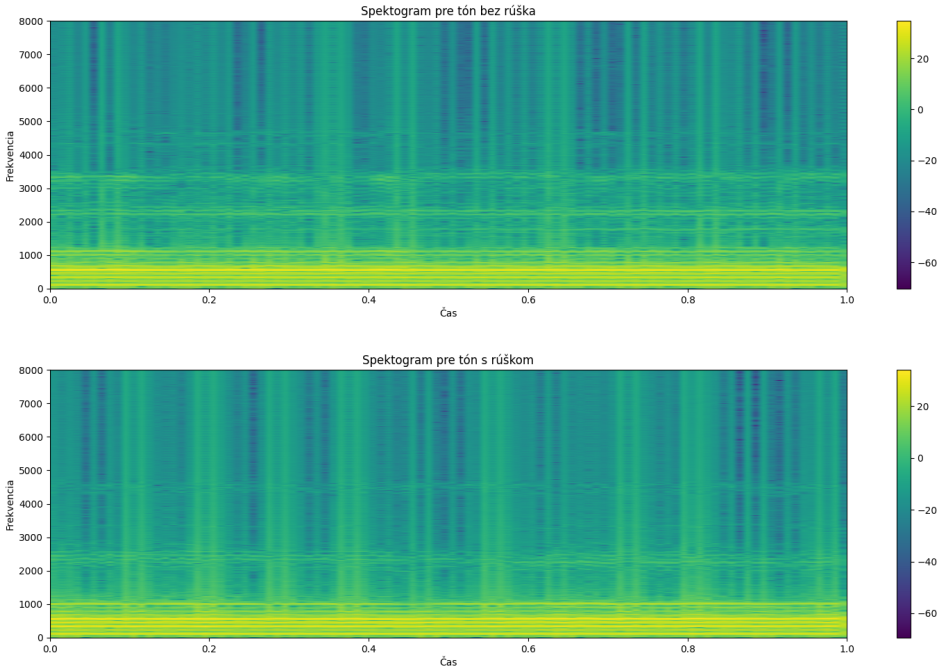
Stredná hodnota základnej frekvencie nahrávky bez rúška: 126.765
Stredná hodnota základnej frekvencie nahrávky s rúškom: 130.746
Rozptyl základnej frekvencie nahrávky bez rúška: 173.732
Rozptyl základnej frekvencie nahrávky s rúškom: 189.172

5. úloha

a)

```
1. def dft(array):
2.         fill = numpy.zeros(1024-len(array))
3.         array = numpy.append(array, fill)
4.         dft_arr = []
5.         arr_len = len(array)
6.         for i in range(arr_len):
7.                 coef = 0
8.                 for j in range(arr_len):
9.                         coef += array[j] * cmath.exp(−2j * cmath.pi * i * j * (1 / arr_len))
10.                dft_arr.append(coef)
11.        return dft_arr
```

b)



6. úloha

a)

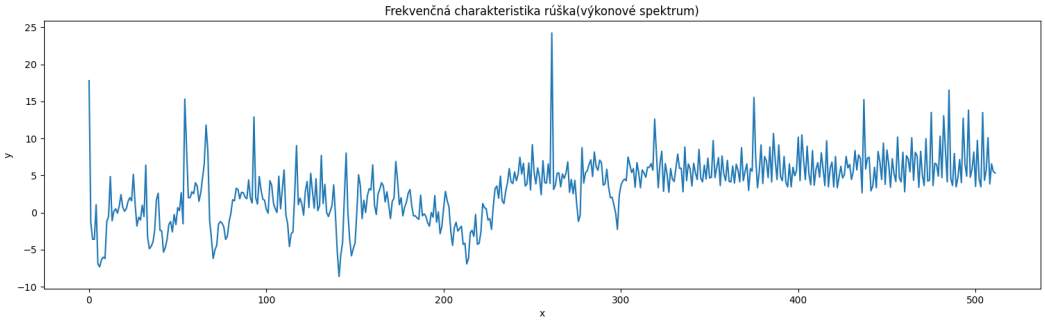
Postup:

Absolútne hodnoty koeficientov DFT z nahrávky s rúškom vydelíme absolútnou hodnotou koeficientu na zhodnej pozícií z DFT nahrávky bez rúška. Hodnoty týchto podielov zo všetkých rámcov na zhodných pozíciách sčítame a nakoniec vydelíme počtom rámcov.

Implementácia:

```
for i in range(0, len(maskoff_tone_frames)): # 0 až počet rámcov
    for j in range(0, 512):          # 0 až požadovaný počet vzorkov
        frekvencna_char_div[i][j] = abs(maskon_tone_dft[i][j]) // abs(maskoff_tone_dft[i][j])
for i in range(0, 512):          # 0 až počet vzorkov
    tmp = complex(0,0)
    for j in range(0, len(maskoff_tone_frames)):          # 0 až počet rámcov
        tmp += frekvencna_char_div[j][i]
    frekvencna_char_avg[i] = abs(tmp.real) / float(len(maskon_tone_frames))
    frekvencna_char_spectrum[i] = 10.0 * math.log((abs(frekvencna_char_avg[i]) ** 2), 10)
```

b)



c)

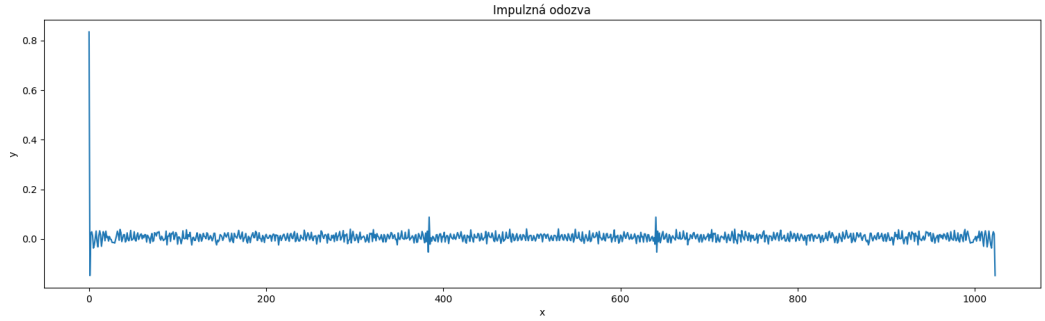
Hodnoty frekvenčnej charakteristiky rúška patria intervalu <0, 8> až na jedinú výmku, kedy dosiahne maximum o hodnote 16. Vo výkonovom spektre frekvenčnej charakteristiky rúška sa tento rozdiel medzi maximom a ostatnými hodnotami podstatne zmenší.

7. úloha

a)

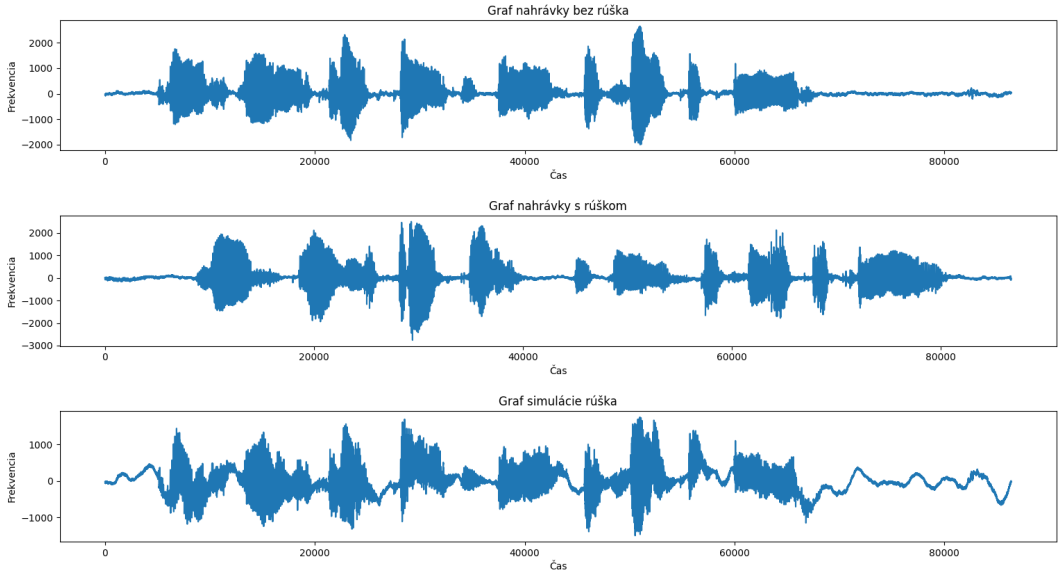
```
1. def idft(array):
2.         fill = numpy.zeros(1024-len(array))
3.         array = numpy.append(array, fill)
4.         idft_arr = []
5.         arr_len = len(array)
6.         for i in range(arr_len):
7.                 coef = 0
8.                 for j in range(arr_len):
9.                         coef += array[j] * cmath.exp(2j * cmath.pi * i * j * (1/arr_len))
10.                coef /= arr_len
11.                idft_arr.append(coef)
12.        return idft_arr
```

b)



8. úloha

a)



b)
Signál simulácie rúška sa od reálneho signálu nahranému s rúškom líši na viacerých miestach. Osobne najväčší rozdiel vidím vo zvlnení odsimulovaného signálu, avšak bádateľný rozdiel je aj v hodnotách frekvencií. Grafy sa anjviac podobajú na miestach s vyššími frekvenciami.

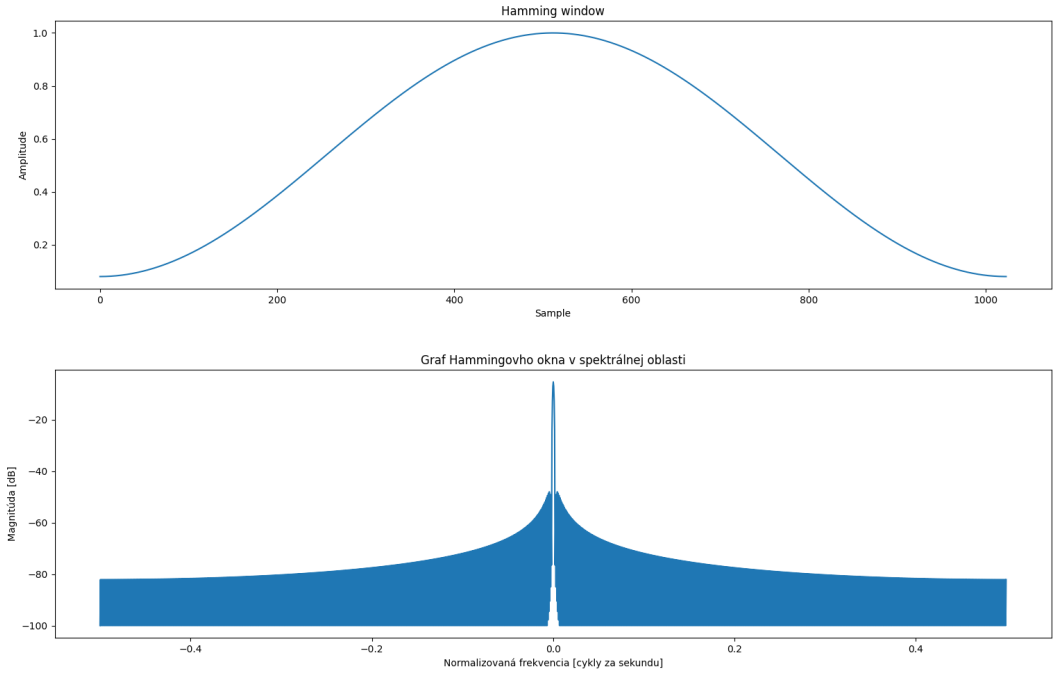
9. úloha

Moje riešenie v úrovni grafov na mňa pôsobilo správne až do momentu, kedy som prvý krát vykreslil simuláciu vety s rúškom. Zvlnenie grafu simulácie, obzvlášť viditeľné pri nízkych frekvenciách vstupu(nahrávky bez rúška) nepôsobí správne. Keď som si simuláciu vypočul, nepresvedčila ma o opaku mojej hypotézy. Simulácia znie skreslene, až pokrivene - akoby ju niekto nahral pokazeným, veľmi nekvalitným mikrofónom. Vstupné nahrávky som nahral viac krát, aby som sa presvedčil, že chyba nie je v nich, avšak vždy s podobným výsledkom. Skúšal som taktiež upraviť nápadne vysoké hodnoty frekvenčnej charakteristiky rúška, skontroloval som grafy jednotlivých krokov získavania filtra, avšak ani toto nepomohlo. Nie som si teda istý, kde moje riešenie zlyháva.

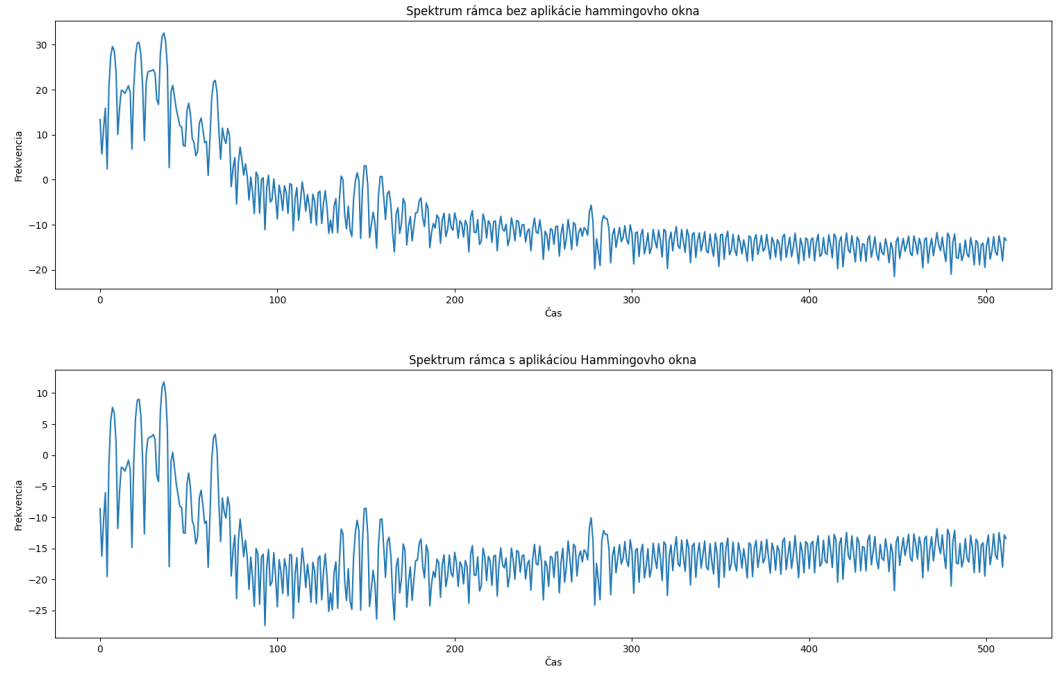
11. úloha

a)
Pre túto úlohu som si vybral **Hammingové okno**.

b)



c)



Toto okno je jedno z najpoužívanejších. Vďaka jeho tvaru(viz. obr. Hammingové okno) utlmuje začiatok a koniec rámcu a dá sa tak odfiltrovať šum vzniknutý "vystrihnutím"rámcu. Pri výslednej simulácií som si značného šumu všimol a tak som sa snažil odstrániť ho aplikáciou Hammingovho okna na všetky rámce pri výpočte ich spektier. Výsledné spektrum vyzerá podľa očakávaní - začiatok a koniec rámcu je tlmenejší, zatiaľčo jeho stred dosahuje hodnoty veľmi podobné tým pôvodným.