ISS Projekt 2020/2021
Protokol

Datum: 02.01.2021

Autor: Roman Popelka(xpopel24)

Roman Popelka (xpopel24)

Tabulka tónů (Úkol č. 1)

- Na tabulce níže, jsou názvy nahrávek tónů a jejich délky v sekundách a vzorcích. Nahrávky jsou již přeformátované v předořezaném stavu na délce kolem 1s, přesně se ořezávají až v programu

Název nahrávky	Délka nahrávky [sekundy]	Délka nahrávky [vzorky]
maskoff_tone.wav	1.144187	18307
maskon_tone.wav	1.300312	20805

Tabulka vět (Úkol č.2)

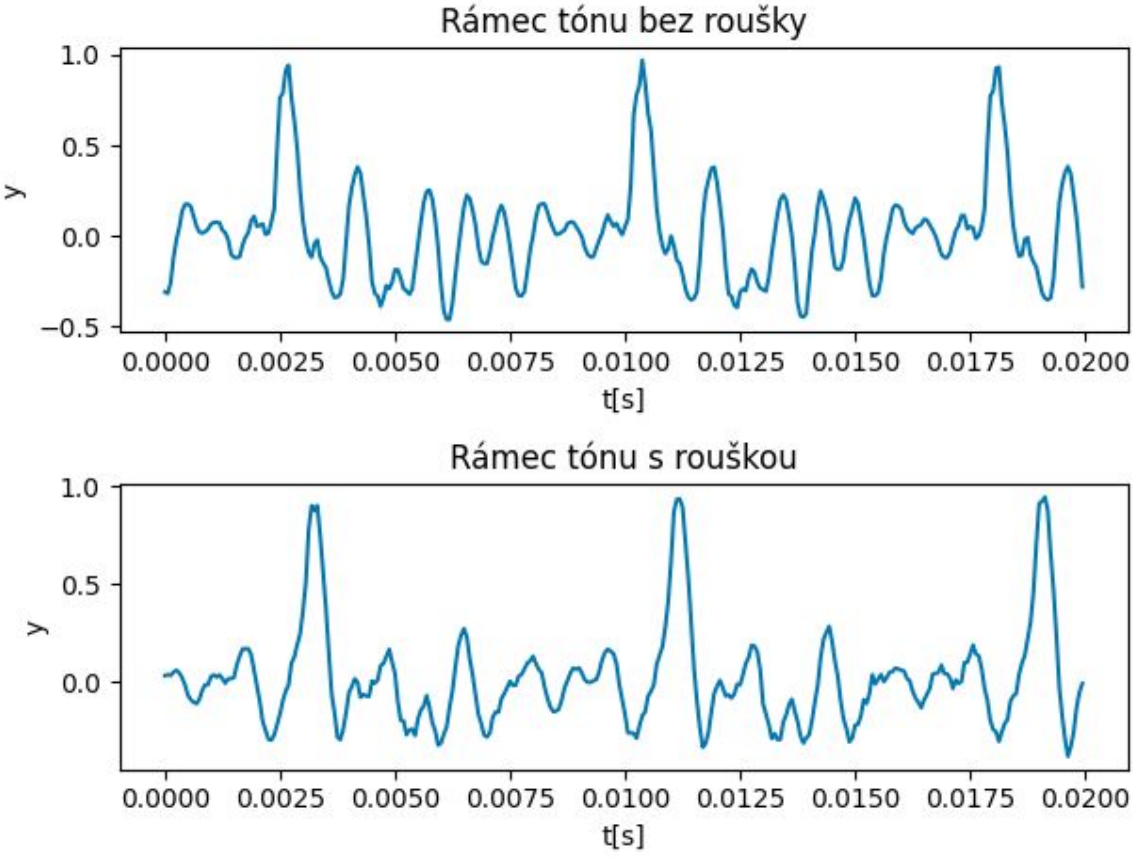
- Na tabulce níže, jsou názvy nahrávek vět a jejich délky v sekundách a vzorcích.

Název nahrávky	Délka nahrávky [sekundy]	Délka nahrávky [vzorky]
maskoff_sentence.wav	1.763875	28222
maskon_sentence.wav	2.170562	34729

Úkol č.3

- V této části jsem nejprve ořezal všechny nahrávky na přesně 16000 vzorků, tedy 1 s. Potom jsem je ustřednil pomocí knihovní funkce numpy mean, ustředněné data jsem pak normalizoval do rozsahu [-1,1] za použití knihovních funkcí numpy abs a max.
- Jako další jsem úseky rozdělil na rámce o délce 20 ms překrývající se o 10 ms. Rozhodl jsem se zvolit variantu s kratší nahrávkou a tedy 99 rámci.
- Jako vzorec pro výpočet velikost rámce ve vzorcích jsem použil:
  $velikost\ rámce = Fs * 0,02$,
  kde Fs zastupuje vzorkovací frekvenci a koeficient zastupuje část, kterou tvoří 20ms v sekundě. Při vzorkovací frekvenci 16000 Hz a délce rámce 20ms, plyne, že velikost rámce bude 320 vzorců

- Grafy rámců



Kontrola rozdílu výšky tónu (Úkol č.4)

Clipping

- Jako první metodu jsem aplikoval 70% clipping na všechny rámce tónů s rouškou a bez roušky. Jako maximální hodnotu při clippov

Autokorelace

- Další funkce, kterou jsem aplikoval na rámce tónů, byla autokorelace, která se používá pro porovnání signálu se sebou samotným. Extrahoval jsem z ní lagy ,které jsem pak použil pro zjištění základní frekvence tónů
- Jako práh jsem u autokorelace použil 500Hz, což je 32. vzorek
- Pro vlastní implementaci jsem použil vzorec pro vychýlený odhad autokorelačních koeficientů

Roman Popelka (xpopel24)

Základní frekvence tónů

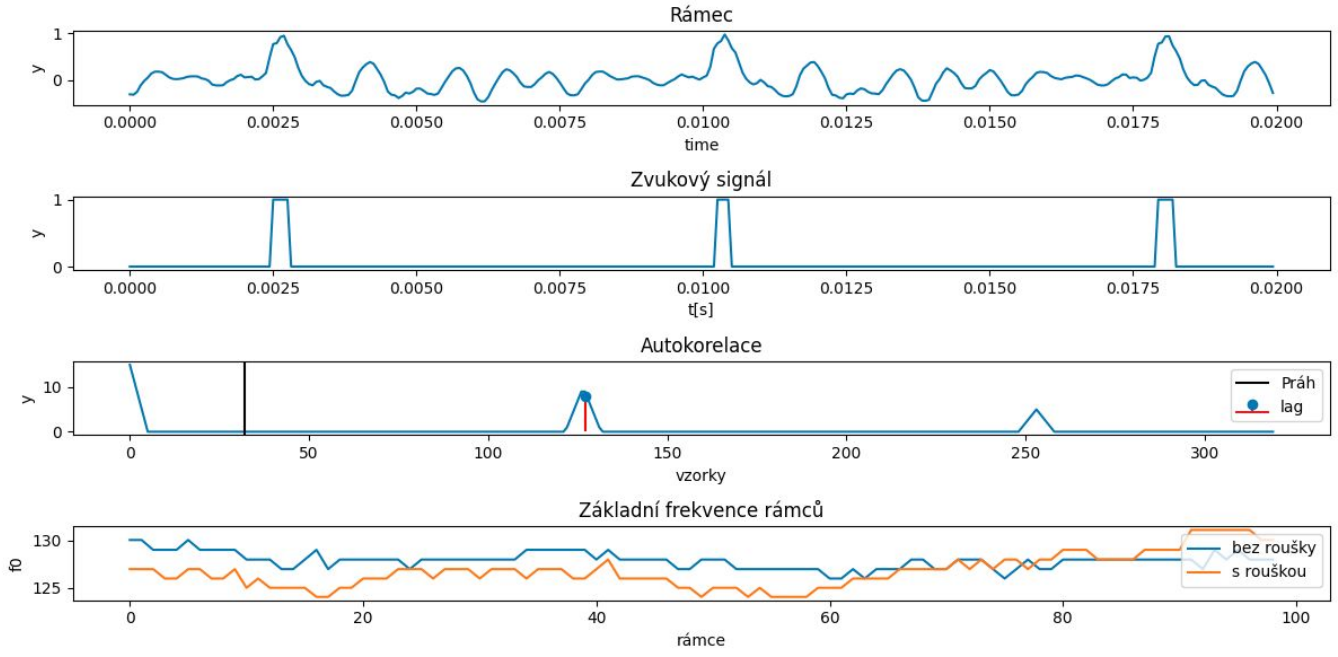
Po aplikaci autokorelace na rámce jsem z každého rámce tónů s rouškou a bez roušky pomocí knihovní funkce numpy - argmax zjistil lagy, potřebné pro zjištění základní frekvence.
Následně jsem vypočítal základní frekvenci pomocí vzorce $f0 = fs / \text{lag}$

Střední hodnota a rozptyl nahrávek

Název nahrávky	Střední hodnota	Rozptyl
maskoff_tone.wav	126.77521906942714	3.1750938203260257
maskon_tone.wav	127.90234820215142	0.7294824439322417

Odpověď na otázku ze zadání

Grafy



Roman Popelka (xpopel24)

DFT (Úkol č.5)

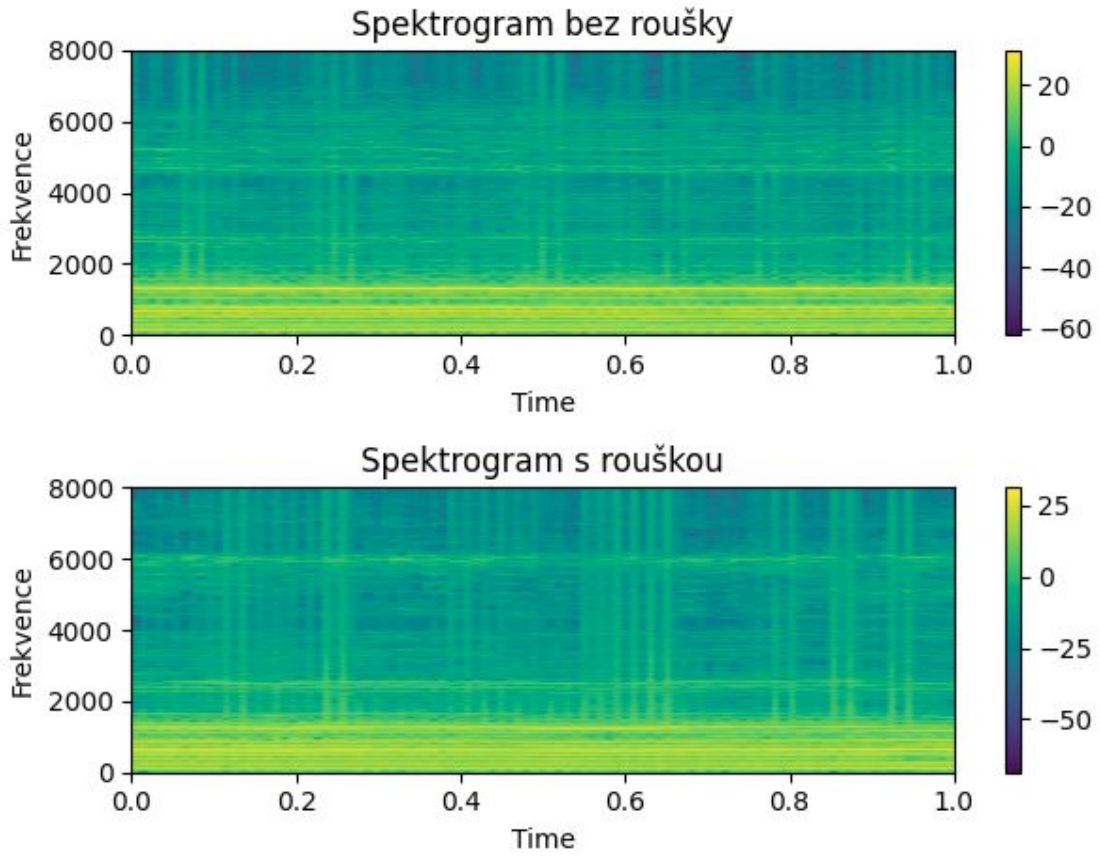
Postup

První krok, který jsem udělal, byla aplikace DFT s N = 1024 nad ustředněnými a normalizovanými rámci tónů s rouškou a bez roušky.
Poté jsem upravil výsledné hodnoty pomocí vzorce $P[k] = 10\,log10|X[k]|2$.
Nakonec jsem vykreslil spektrogramy.

Vlastní implementace DFT

```
def DFT(frame):
    dft = list()
    frame_in = list()
    frame_in.extend(frame) #Překopírování rámce
    frame_in.extend([0]*(1024-len(frame))) #Rozšíření rámce
    for k in range(len(frame_in)):
        sum_of = complex()
        for n in range(len(frame_in)):
            sum_of += frame_in[n]*cmath.exp(-2j*cmath.pi*n*k/1024)
        dft.append(sum_of)
    return dft
```

Spektrogramy



Frekvenční charakteristika roušky (Úkol č.6)

Postup

Nejprve jsem použil vzorec pro $H(ej\omega)$ roušky popsaný níže v tom jsem si pomohl knihovní funkce numpy - divide , pak jsem zprůměroval všechny rámce pomocí knihovní funkce numpy - mean (s parametrem axis = 0), po zprůměrování mi zbylo pole o 1024 prvcích s frekvenční charakteristikou. Nakonec jsem všechny hodnoty upravil tak, aby jsem mohl graf vykreslit jako výkonové spektrum.

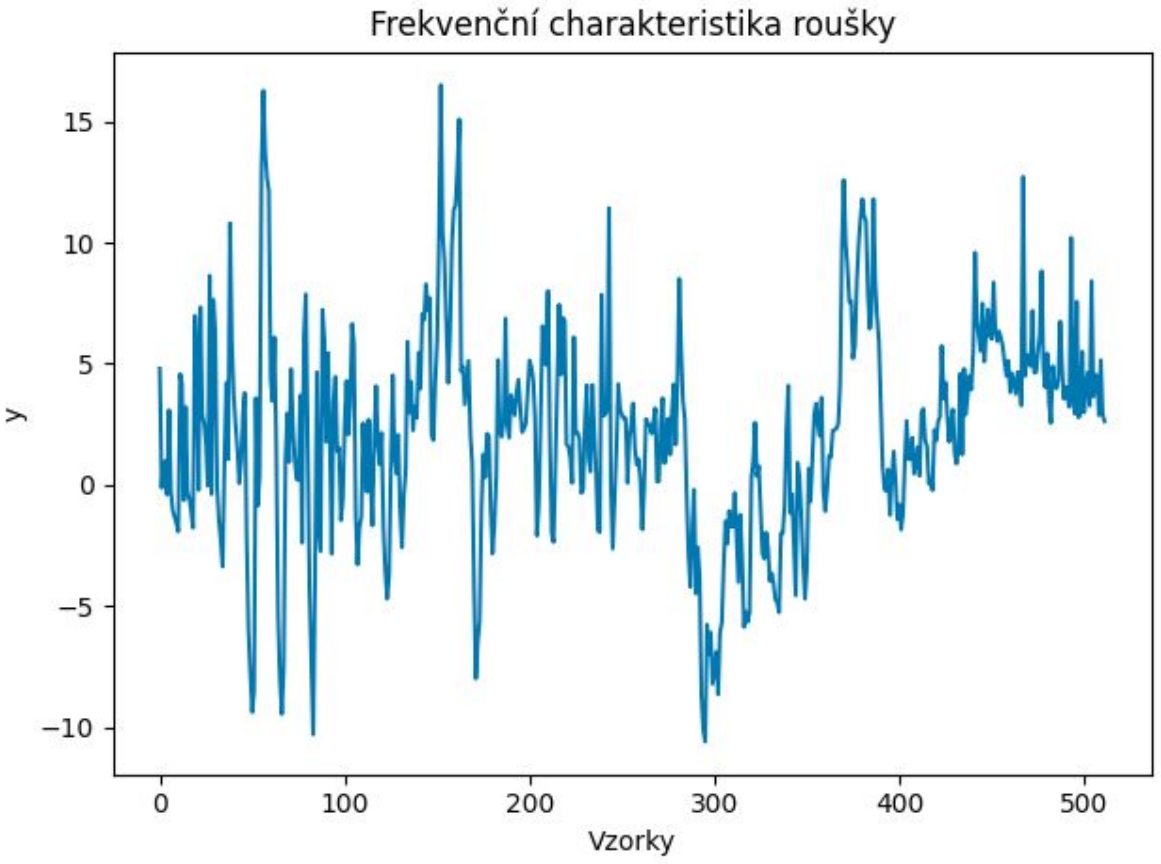
Vzorec pro výpočet H(ejω)

$$H(ej\omega) = DFT\ s\ rouškou\ /\ DFT\ bez\ roušky$$

Pro získání H(ejω) musíme vydělit všechny prvky rámců po aplikování DFT

Roman Popelka (xpopel24)

Frekvenční charakteristika



Komentář k filtru

filtr zeslabuje signál (ve většině případů)

Roman Popelka (xpopel24)

Impulsní odezva, IDFT (Úkol č.7)

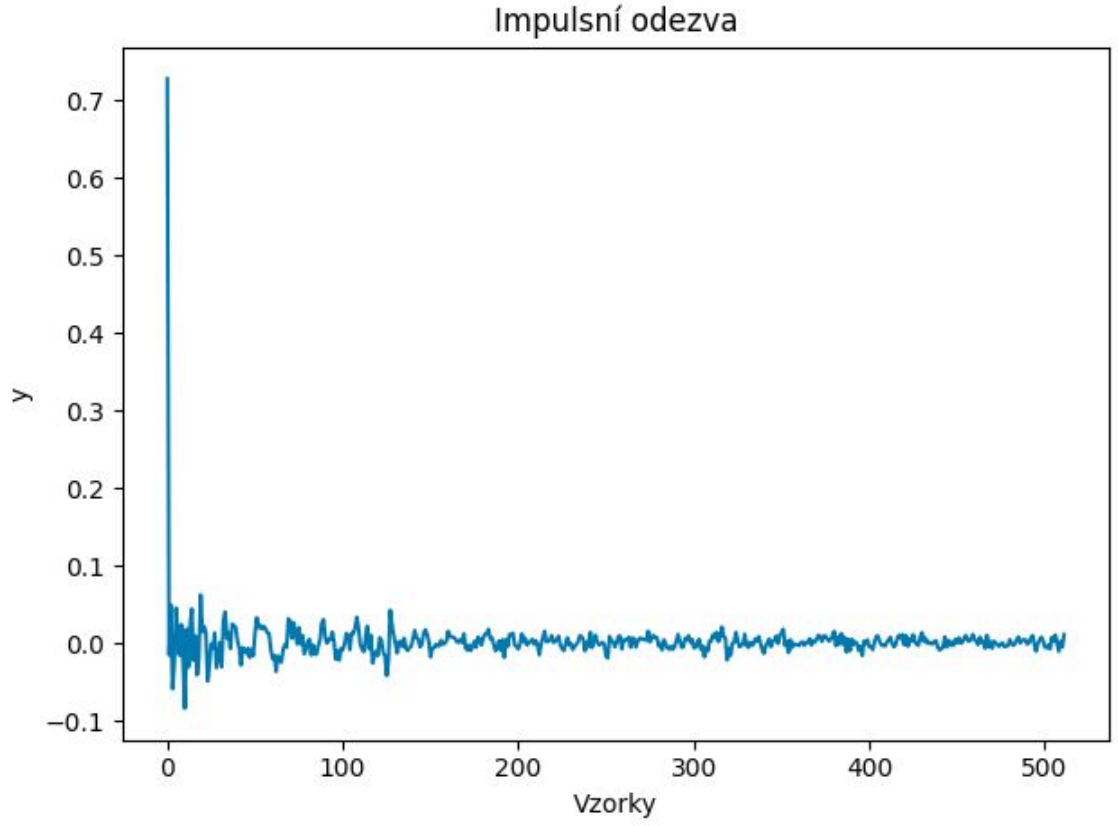
Postup

Na frekvenční charakteristiku jsem aplikoval IDTF s N = 1024, tím jsem získal impulsní odezvu

Vlastní implementace

```
def IDFT(frame):
    idft = list()
    frame_in = list()
    frame_in.extend(frame)
    frame_in.extend([0]*(1024-len(frame)))
    for n in range(len(frame)):
        sum_of = 0
        for k in range(len(frame)):
            sum_of += frame[k]*cmath.exp(2j*cmath.pi*n*k/1024)
        frame_in.append(sum_of/1024)
    return idft
```

Graf Impulsní odezvy



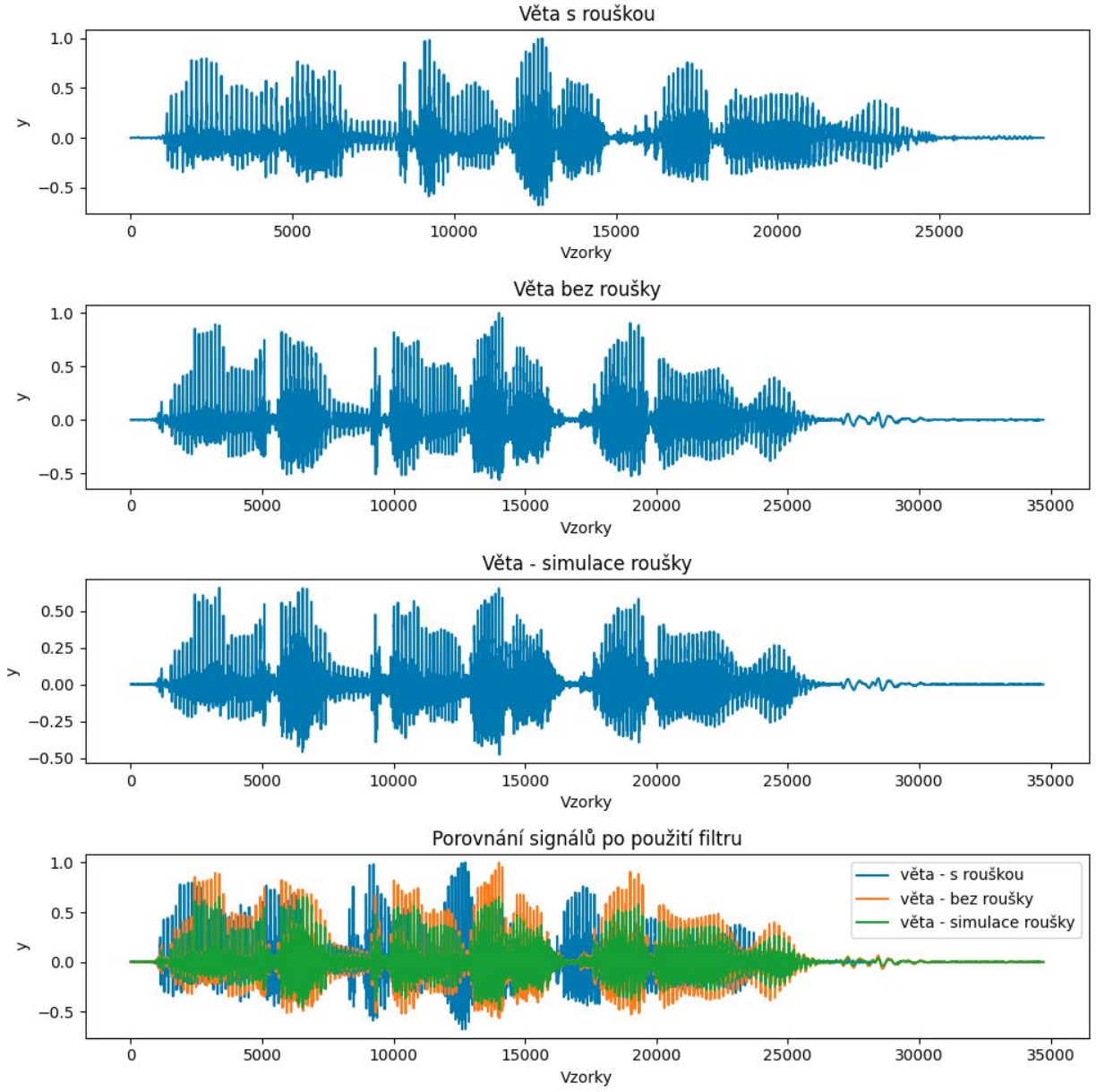
Roman Popelka (xpopel24)

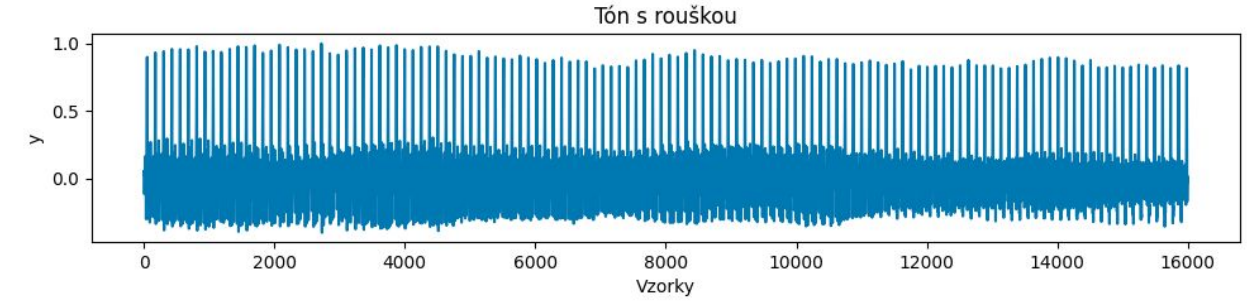
Filtrace nahrávek (Úkol č.8)

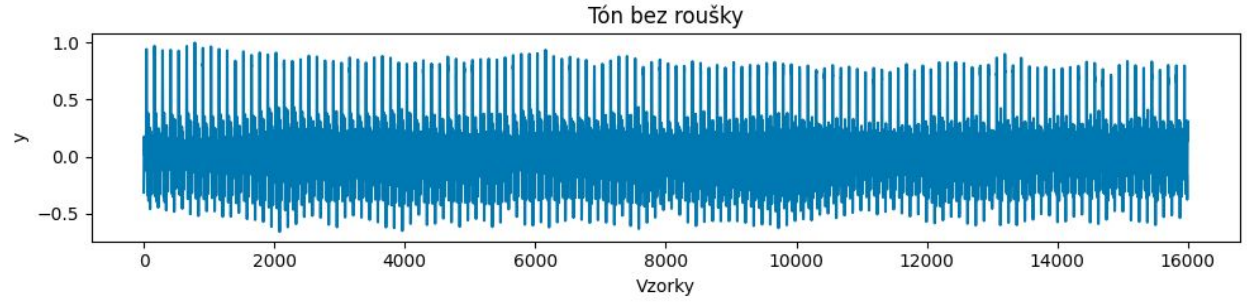
Postup

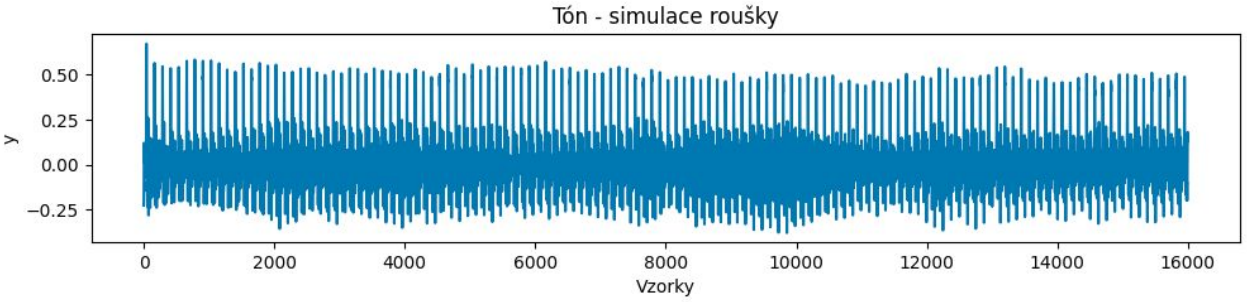
pro vytvoření filtru jsem použil funkci knihovny scipy - lfilter, kde jsem jako parametr filtru zadal impulzní odezvu. Poté jsem vy

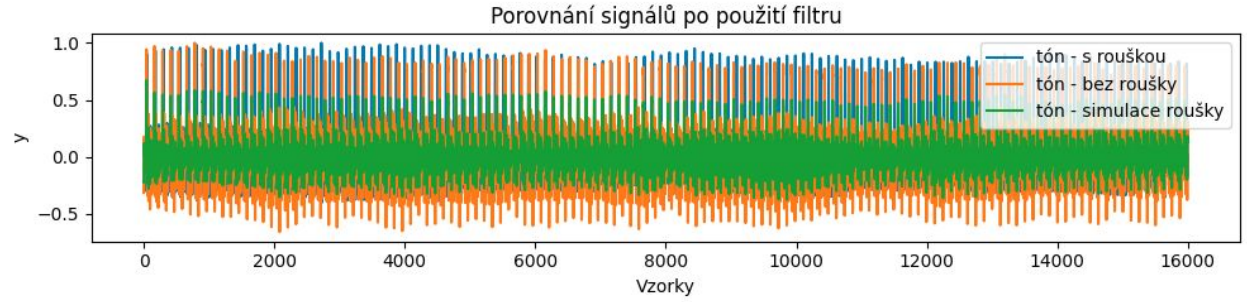
Graf




Tón s rouškou
y
Vzorky


Tón bez roušky
y
Vzorky


Tón - simulace roušky
y
Vzorky


Porovnání signálů po použití filtru
tón - s rouškou
tón - bez roušky
tón - simulace roušky
y
Vzorky

Odpovědi na otázky

- Jak se od sebe liší signály s rouškou a se simulovanou rouškou?
    - Simulovaná rouška má menší amplitudu a více “zubů” v signále
- Jsou signály podobné?
    - Dalo by se říct, že ano. Použití filtru vedlo k zeslabení signálů, lehkému vyhlazení výraznějších “zubů”, či ostrých hran signál simulované roušky, ale pořád uplně nedosahuje takové kvality jako signál bez roušky
- Kde se nejvíce podobají a kde se liší?
    - Nejvíce se podobají ve špičkách signálu (při vyšší amplitudě), kde filtr dokázal vyhladit “zuby” v signálu a částečně se mu podařilo signál zaoblit. Méně se podobají při vyšších amplitudách

Závěr (Úkol č.9)

Filtr snižuje vyšší amplitudy a přijde mi, že při nižších amplitudách zvyšuje frekvenci filtrovaného signálu. Zároveň se filtr snaží částečně vyhlazovat větší ostré hrany. Zvuk tedy zní o něco utlumeněji. Změna signálu sice není nějak výrazná, ale částečně poznat jde. Filtr by byl možná o něco lepší, kdyby byly základní frekvence tónů více podobné. Věty také nebyly úplně stejné, ale z mého výběru byly asi nejlepší. Výsledky hodnotím víceméně kladně.