

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ FAKULTA INFORMAČNÍCH TECHNOLOGIÍ



ISS projekt 2021/22

Obsah

1	Úvod	3
2	Riešenie	3
2.1	Základy	3
2.2	Predspracovanie rámca	3
2.3	DFT	4
2.4	Spektrogram	4
2.5	Určenie rušivých frekvencií	4
2.6	Generovanie signálu	4
2.7	Čistiaci filter	5
2.8	Nulové body a póly	6
2.9	Frekvenčná charakteristika	6
2.10	Filtrácia	7

1 Úvod

Projekt som riešil pomocou skriptovacieho jazyka *Python*. Na prácu so signálom som využíval knižnice *numpy*, *soundfile* a *scipy*. Na vykresľovanie grafov som použil knižnicu *matplotlib*.

2 Riešenie

2.1 Základy

Signál som načítal pomocou funkcie *read* z knižnice *soundfile*. Funkcia *read* načíta už normalizovaný signál.

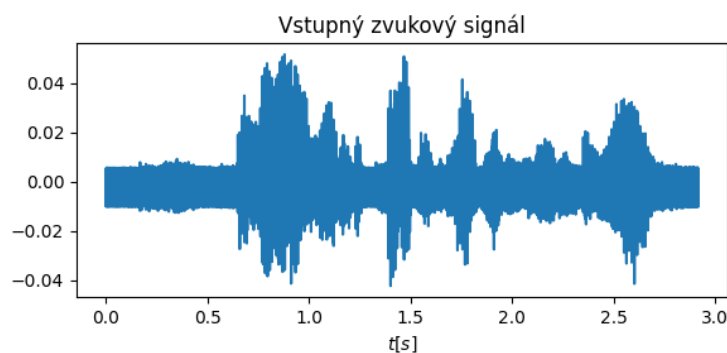
Vzorkovacia frekvencia signálu: 16 000 Hz

Dĺžka signálu vo vzorkách: 46695

Dĺžka signálu v sekundách: 2.9184375 s

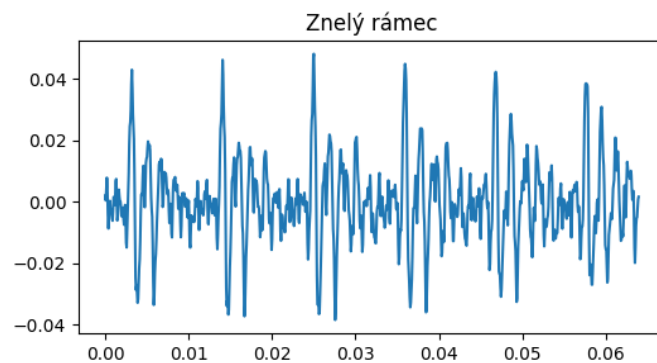
Minimálna hodnota: -0.04241943359375

Maximálna hodnota: 0.05169677734375



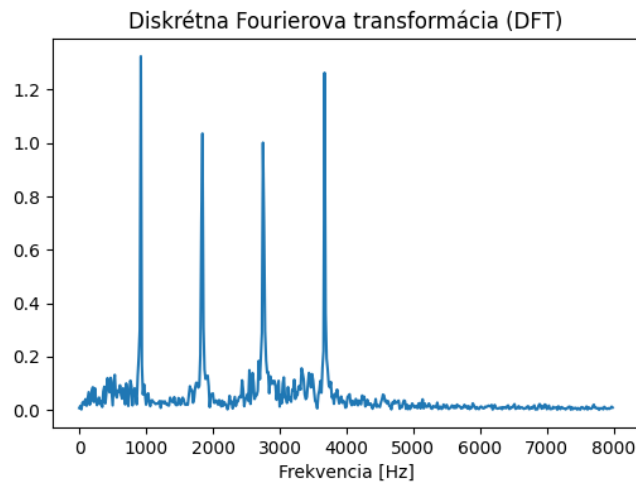
2.2 Predspracovanie rámca

Signál som si najskôr rozdelil do jednotlivých rámcov po 1024 vzorkoch, tzn. vytvoril som dvojrozmerné pole, ktoré obsahuje jednotlivé rámce. Následne som prechádzal jednotlivé rámce a hľadal rámec s periodickým charakterom.



2.3 DFT

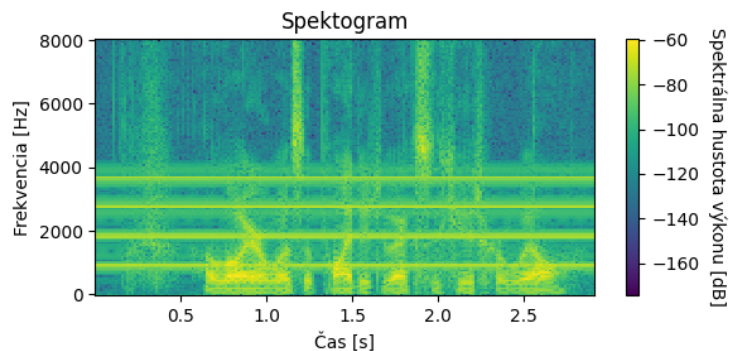
Na úvod som si vybral jeden z rámcov z predchádzajúcej úlohy, keďže máme pracovať s 1024 vzorkami. Následnej pomocou vlastnej funkcie *frame_dft* aplikujem DFT¹ na vybranom rámci. Funkcia mi vráti upravený rámec.



2.4 Spektrogram

Na vypočítanie a zobrazenie spektrogramu som využil funkciu *spectrogram* z knižnice *scipy.signal*. Argumentami funkcie je načítaný signál a vzorkovacia frekvencia signálu. Výstupom je pole vzorkovacích frekvencií, pole časových segmentov a spektrogram signálu.

```
f, t, sgr = spectrogram(sig, fs)
```



2.5 Určenie rušivých frekvencií

Rušivé frekvencie je možné vidieť aj na spektrograme aj na grafe DFT. Podľa hodnôt jednotlivých frekvencií môžeme vidieť, že sú cosínusovky harmonicky vzťahované.

$$f_1 = 920Hz \quad f_2 = 1840Hz \quad f_3 = 2760Hz \quad f_4 = 3680Hz$$

2.6 Generovanie signálu

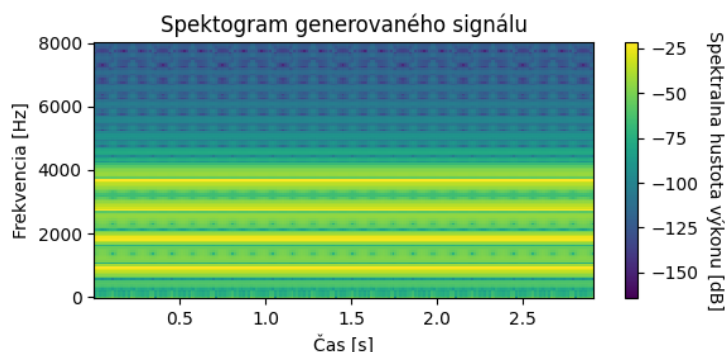
Za účelom generovaniu signálu z cosínusoviek som si vytvoril pole časových úsekov. Pole časových úsekov som použil pri vytváraní signálu z jednotlivých cosínusoviek.

```
cos1 = np.cos(2 * np.pi * 920 * casove_useky)
```

¹<https://pythonnumericalmethods.berkeley.edu/notebooks/chapter24.02-Discrete-Fourier-Transform.html>

Cosínusovky som následne spojil do jedného signálu. Výsledný signál som vygeneroval pomocou funkcie *wavfile.write*.

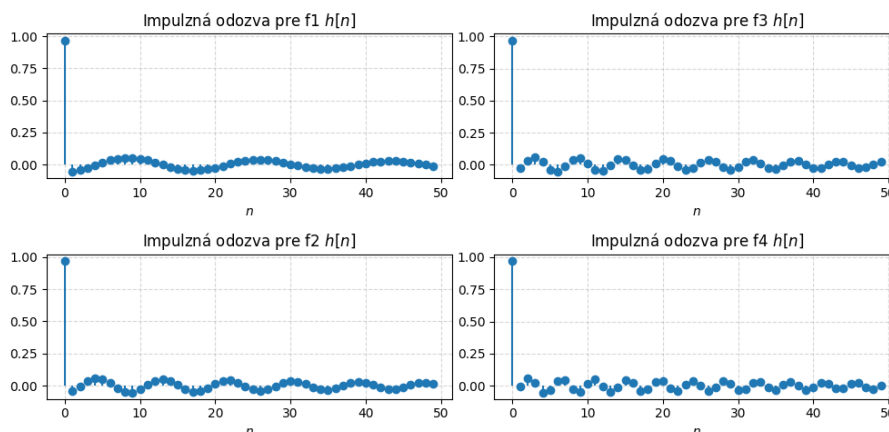
```
wavfile.write("audio/4cos.wav", fs, cos_merge.astype(np.float32))
```



2.7 Čistiaci filter

Čistiace filtre som vytvoril pomocou funkcie *scipy.signal.butter*. Zvolil som si zo zadania metódu č.3, návrh 4 pásmových zádrží.

```
b1, a1 = butter(4, [low, high], btype="bandstop")
```



Pre zobrazenie impulzných odoziev som zvolil 50 vzorkov. Keďže ide o IIR filtre je potrebné ich obmedziť na dĺžku vhodnú na zobrazenie.

Koeficienty filtrov:

```
b_1 = [ 0.96968306 -7.25717748 24.24619573 -47.17679161 58.43644899 -47.17679161
        24.24619573 -7.25717748 0.96968306]
a_1 = [ 1.          -7.42647266 24.62102491 -47.53815186 58.4323126 -46.81199214
        23.87458382 -7.09132154 0.94028524]
b_2 = [ 0.96968306 -5.81936384 16.97514878 -30.55735654 36.92422423 -30.55735654
        16.97514878 -5.81936384 0.96968306]
a_2 = [ 1.          -5.95511774 17.23747829 -30.79114816 36.92123644 -30.32080708
        16.71488794 -5.68636777 0.94028524]
b_3 = [ 0.96968306 -3.63020085 8.97512354 -14.07049444 16.7549157 -14.07049444
        8.97512354 -3.63020085 0.96968306]
a_3 = [ 1.          -3.71488604 9.11367444 -14.17791825 16.75319166 -13.96135026
        8.83737757 -3.54723603 0.94028524]
```

```

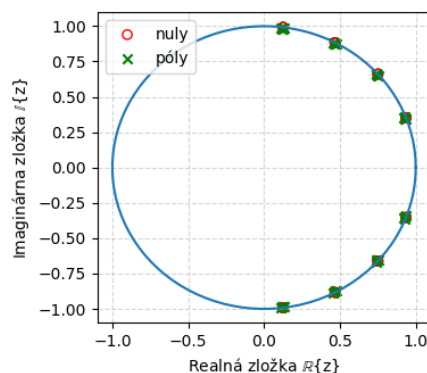
b_4 = [ 0.96968306 -0.97233559  4.24435552 -2.97811066  6.55317435 -2.97811066
        4.24435552 -0.97233559  0.96968306]
a_4 = [ 1.          -0.9950182  4.30971065 -3.00079861  6.55219762 -2.95496191
        4.179058   -0.95011377  0.94028524]

```

2.8 Nulové body a póly

Nulové body a póly som získal z filtrov pomocou funkcie `scipy.signal.butter` s výstupom `zpk`. Následne som nulové body a póly zo všetkých filtrov zobrazil na jednu jednotkovú kružnicu.

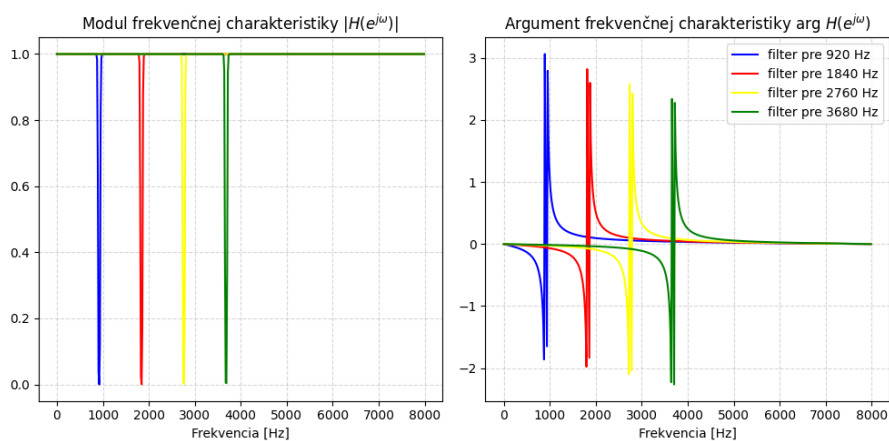
```
z1, p1, k1 = butter(4, [low, high], btype="bandstop", output="zpk")
```



2.9 Frekvenčná charakteristika

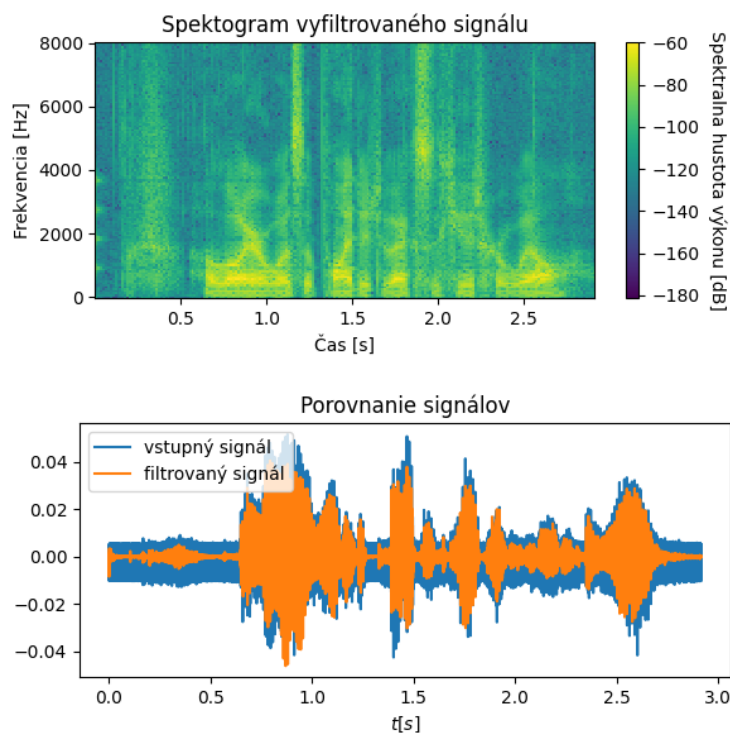
Frekvenčnú charakteristiku som realizoval pomocou funkcie `scipy.signal.freqz` postupne pre každý filter.

```
freq1, h1 = freqz(b1, a1)
```



2.10 Filtrácia

Signál som vyfiltroval pomocou funkcie `scipy.signal.lfilter` so všetkými 4 navrhnutými filtermi. Filtrovaný signál je vyčistený, avšak na začiatku signálu sa objavil artefakt. Je to spôsobené tým, že aj filter má nejakú impulznú odovzvu. Prvých pár vzorkov sa bude správať inak, tzn. na výstup sa prekopírujú vzorky z pôvodného signálu.



Podľa spektrogramu a vypočutia si výsledného zvuku usudzujem, že som rušivé frekvencie úspešne odstránil a splnil zadanie projektu.