ISS

Vengerová, Veronika, (xvenge01)
xvenge01@stud.fit.vutbr.cz

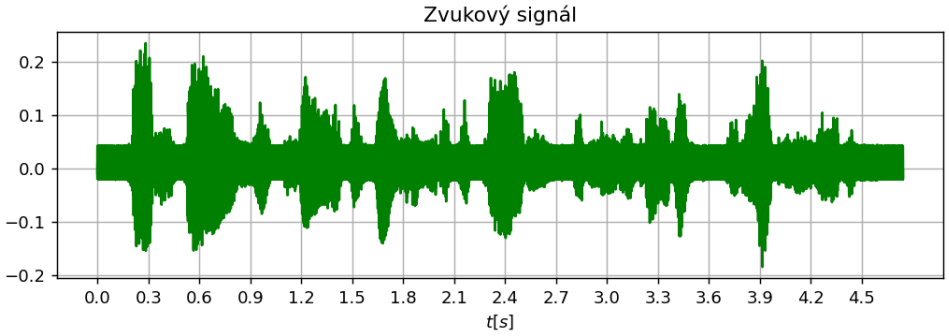
December 2021

1 Riešenie

Na nasledujúce úlohy sme častokrát čerpali z príkladov Katky Žmolíkovej[1]

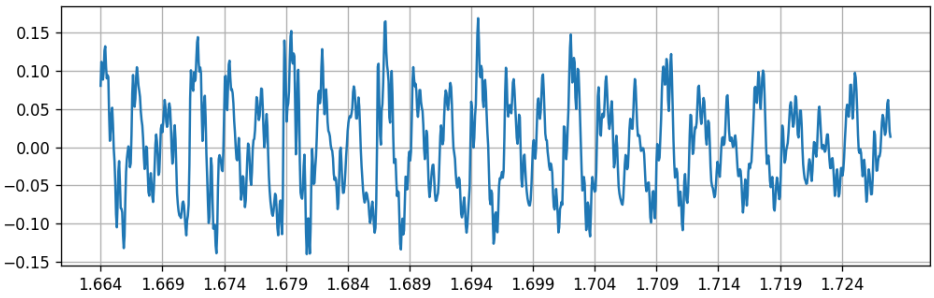
1.1 Úloha 1.

Signál sme načítali pomocou funkcie pythonu soundfile.read(), vrámci ktorej sme získali normované hodnoty v rozmedzí 1 až -1. Maximálna hodnota signálu je 0.234771728515625, najmenšia -0.184295654296875. Dĺžka signálu je 4.7424375[s] a 75879 vzorkov.



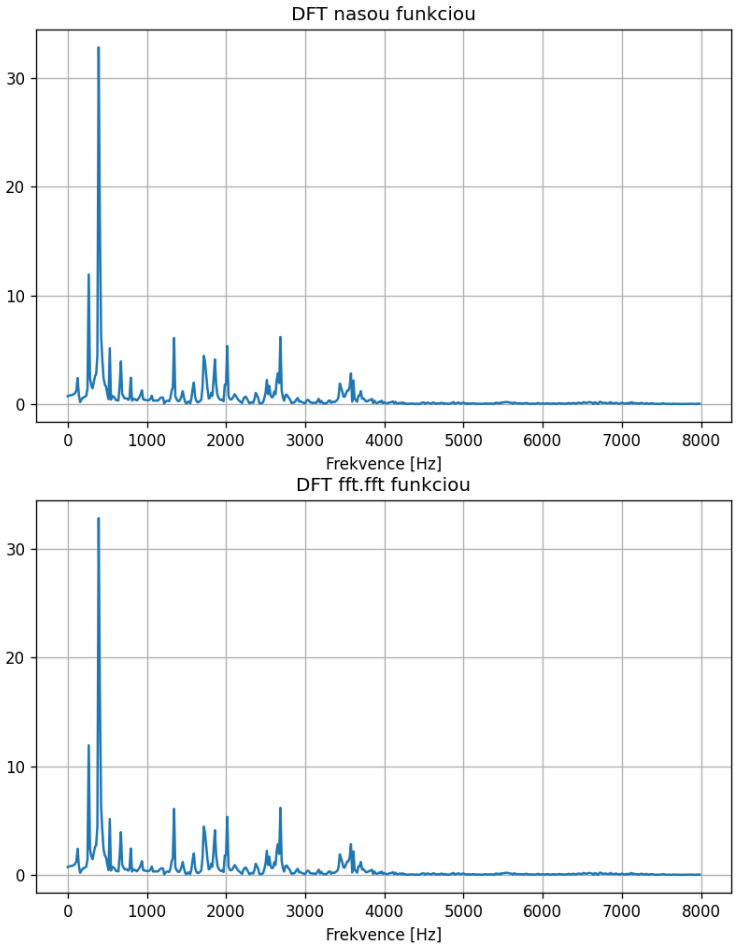
1.2 Úloha 2

Signál už je v rozmedzí -1,1, preto ho neupravujeme, ale iba delíme na úseky dlhé 1024 vzorkov s prekryvom 512 vzorkov. Získali sme tým 147 vzorkov (na konci signálu sme stratili 615 vzorkov, no pri dalšej práci nás to neovplyvní, keďže dané rámce využívame iba na určenie znelého rámcu s ktorým budeme ďalej pracovať). Postupne sme skúmali jednotlivé rámce uložené v matici a vybrali sme si ako pekný znelý rámec rámec s indexom 52 (od času 1,664[s]).



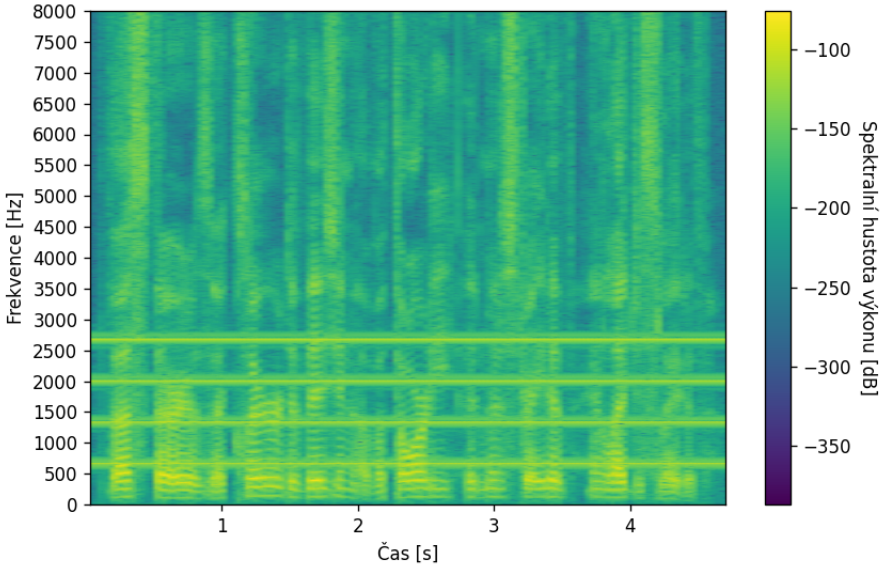
1.3 Úloha 3

Naimplementovali sme si funkciu *my_dft()*, s využitím vzorcu na výpočet DFT: $X(k) = \sum_{n=0}^{N-1} *x(n) * e^{-2*\pi*i*k*n/N}$.[2]
Ďalej sme našu funkciu spustili na v predošlej úlohe vybranom rámci.



1.4 Úloha 4

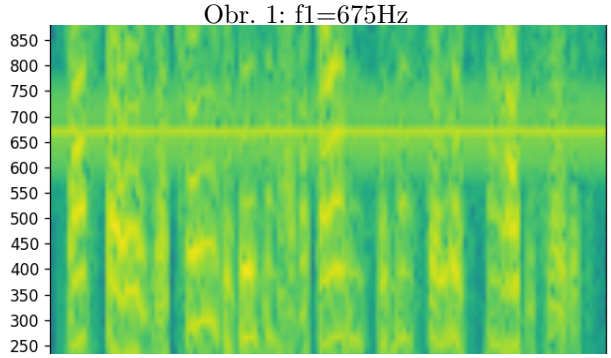
Na zhotovenie spektogramu sme využili funkciu spectrogram, s parametrami nperseg=N (počet vzorkov za segment, v našom prípade 1024), fs=fs (vzorkovacia frekvencia, v našom prípade 16kHz), noverlap=512 (počet vzorkov, ktoré sa prekrývajú). Na úpravu sme ďalej využili $P[k] = 10 * log_{10} \mid X[k] \mid^2$ a upravené hodnoty sme zobrazili do nižšie uvedeného spektogramu.



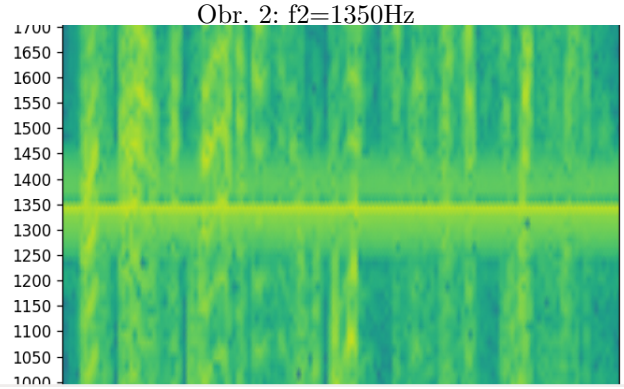
1.5 Úloha 5

Na vyššie uvedenom spektograme je vidieť 4 rušivé komponenty. Frekvencie sme určili pomocou spektogramu pri "priblížení"spektogramu. Na obrázkoch nižšie môžeme vidieť zobrazené priblížené úseky spektogramu na určenie frekvencií f_1, f_2, f_3, f_4 (na y osi je frekvencia v Hz). Frekvencie f_2, f_3, f_4 sú násobkami f_1, takže sú harmonicky vztažené.

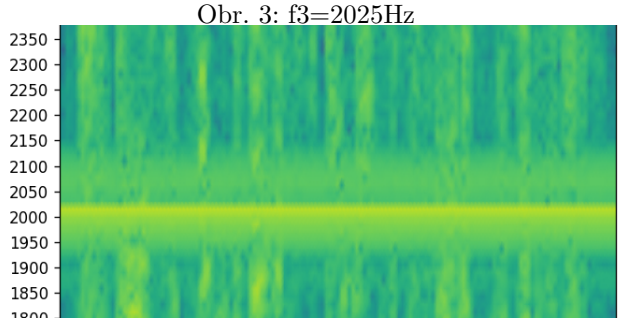
Obr. 1: f1=675Hz



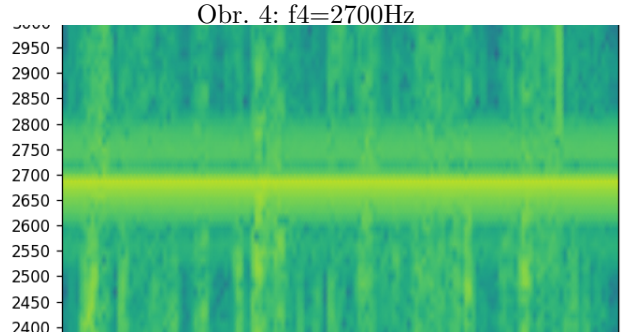
Obr. 2: f2=1350Hz



Obr. 3: f3=2025Hz

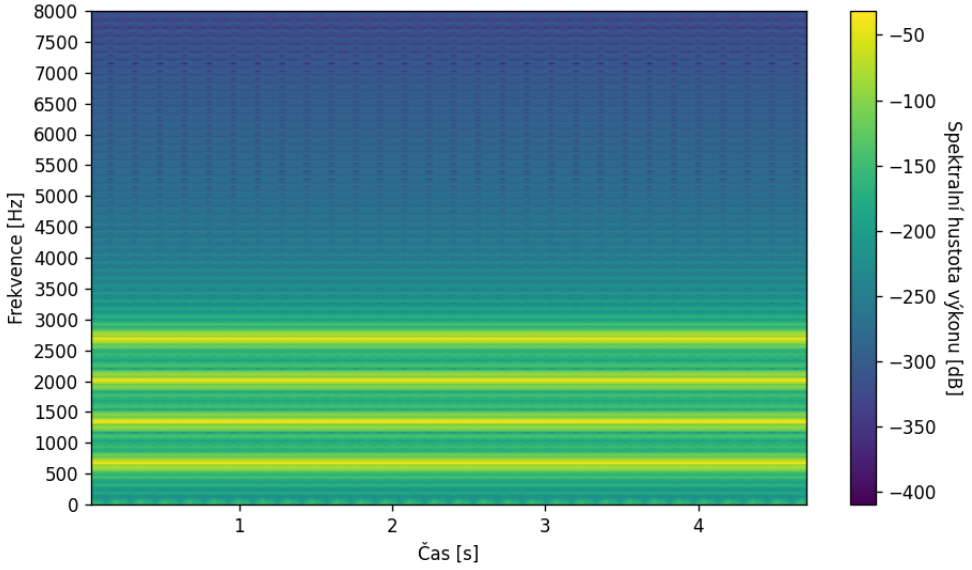


Obr. 4: f4=2700Hz



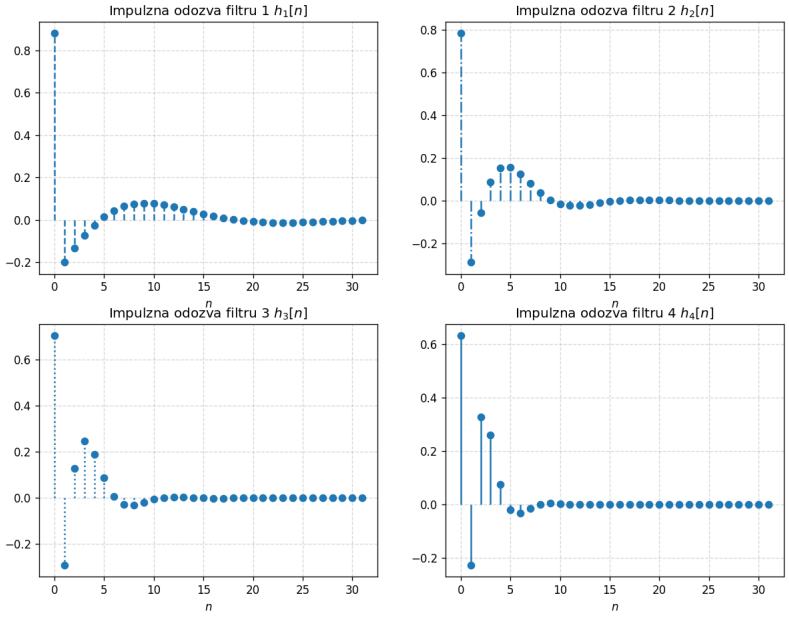
1.6 Úloha 6

Pomocou funkcie *cos()* sme si vygenerovali signál zmesi 4 cosinusoviek na frekvenciách určených v predchádzajúcej úlohe. (Vypočítali sme si postupne hodnoty cosinusu pre všetky vzorky signálu pre všetky 4 frekvencie a navzájom ich sčítali). Na kontrolu sme zobrazili spektogram získaného zmiešaného signálu a vypočuli nahrávku cos4.wav, v ktorom je daný signál uložený.



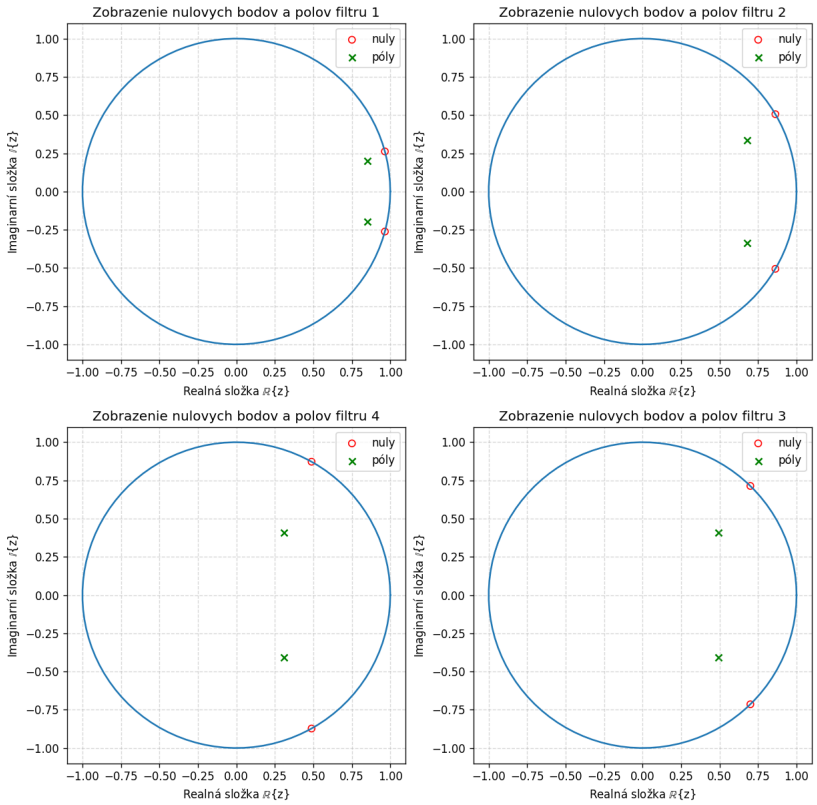
1.7 Úloha 7

Na vyfiltrovanie šumu, ktorý v pôvodnej nahrávke počuť sa vytvorili 4 notch filtre [3] pomocou scipy.signal.iirnotch(). Na zobrazenie impulznej odozvy sme si pripravili jednotkový impulz o dĺžke 32 vzorkov. Koeficienty filtrov sa vypisujú do jednotlivých súborov (f1_b.txt pre koeficienty numerátor filtru f1 a f1_a.txt pre denominátor polynómov filtru f1 (podobne pre všetky 4 filtre)).



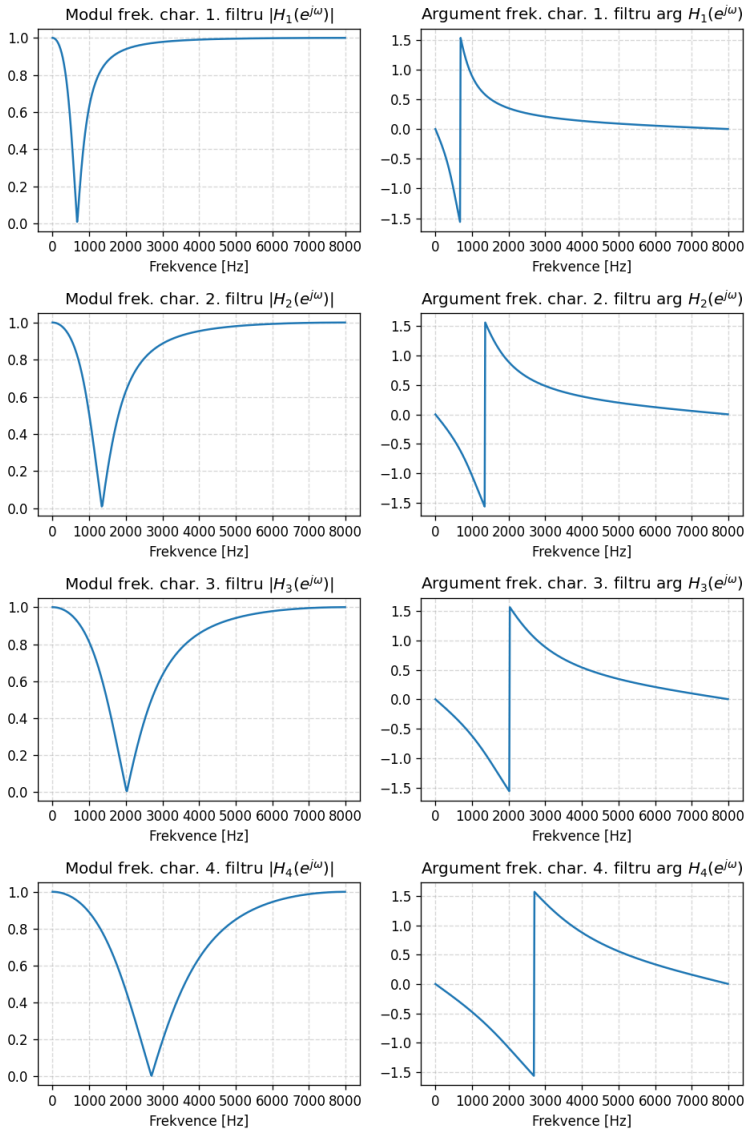
1.8 Úloha 8

Na získanie nulových bodov a pólov navrhnutých filtrov sme využili funkciu *tf2zpk()*a následne ich zobrazili do komplexnej roviny.



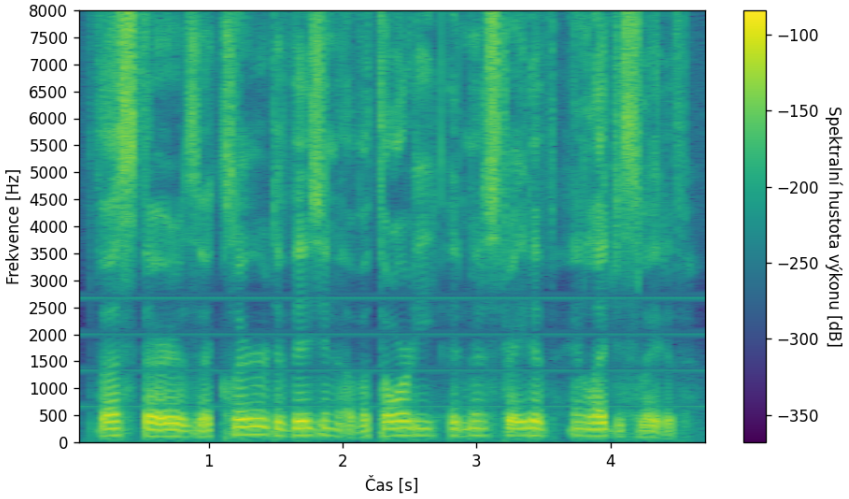
1.9 Úloha 9

Vypočítali sme si frekvenčnú charakteristiku využitím funkcie *freqz()*, ktorá vracia frekvencie na ktorých bolo počítané a frekvenčnú odozvu ako komplexné čísla, funkcie *abs()*, ktorá si poradí aj s komplexnými číslami a vráti nám modul a funkcie *angle()*, ktorá vracia uhol komplexného argumentu. Frekvenčnú charakteristiku zobrazíme a všímame si na grafoch, že frekvencie sú potlačované na vhodných frekvenciách (675, 1350, 2025, 2700 [Hz]).



1.10 Úloha 10

Na načítaný signál z nahrávky xvenge01.wav aplikujeme postupne všetky 4 filtre a výsledný signál vložíme do súboru clean_bandstop.wav. Na kontrolu, že došlo k vyčisteniu pôvodného signálu využijeme spektogram a vypočutie vyfiltrovaného signálu (vo vnútri jupyter notebooku použitím IPython.display.Audio, alebo vypočutím clean_bandstop.wav nahrávky).



Literatúra

[1] https://www.fit.vutbr.cz/~izmolikova/ISS/project/.

[2] https://jyhmiinlin.github.io/pynufft/misc/dft.html.

[3] https://stackoverflow.com/questions/54320638/how-to-create-a-bandstop-filter-in-python.