Univerza v Ljubljani

University of Ljubljana
Faculty of Computer and Information Science

SEMINAR 3
Elimination of narrowband interference

Predmet: Digitalno procesiranje signalov
Avtor: Timotej Tim Rus, 63180256

Ljubljana, januar 2021

Kazalo vsebine

1. Introduction

Predstavitev osnovnih delov seminarske naloge.

Človeški govor

Človeški glas je sestavljen iz zvoka, ki ga človek uporablja z vokalnim traktom, vključno s pogovorom, petjem, smehom, jokom, kričanjem ali vpitjem. Frekvenca človeškega glasu je posebni del človeške produkcije zvoka, v katerem so vokalne gube (glasilke) glavni zvočni vir. (Drugi mehanizmi za ustvarjanje zvoka, proizvedeni iz istega splošnega področja telesa, vključujejo proizvodnjo zvočnih soglasnikov, klike, piskanje in šepetanje.)

Na splošno lahko mehanizem za ustvarjanje človeškega glasu razdelimo na tri dele; pljuča, vokalne gube v grlu (glasovno polje) in artikulatorji. Pljuča, "črpalka", morajo ustvarjati ustrezen pretok zraka in zračni tlak za vibriranje vokalnih gub. Vokalne gube (glasilke) nato zavibrirajo, da s pomočjo zračnega toka iz pljuč ustvarijo slišne impulze, ki tvorijo izvor grla v grlu. Mišice grla prilagodijo dolžino in napetost vokalnih gub, da se 'fino prilagodijo' višini in tonu.

Artikulatorji (deli glasovnega trakta nad grlom, ki jih sestavljajo jezik, nebo, lice, ustnice itd.) Artikulirajo in **filtrirajo** zvok, ki izvira iz grla, in do neke mere lahko vplivajo na zračni tok grla, da ga okrepijo ali oslabijo, vir zvoka.

Filtriranje

Tehnike digitalnega filtriranja se najpogosteje uporabljajo za signale v časovni domeni, v aplikacijah za filtriranje v realnem času. Odvisno od sistemskih parametrov lahko digitalni filter deluje hitreje kot z uporabo algoritma FFT, kjer posredni FFT pretvori signal časovne domene v frekvenčno domeno. Nato se frekvenčni signal pomnoži s funkcijo filtra in na koncu se frekvenčni signal prek IFFT pretvori nazaj v časovno domeno.

Dve pogosti vrsti pristopov digitalnega filtra sta Finite Impulse Responsev (FIR) in Infinite Impulse Response (IIR). Postopek filtriranja je dejansko konvolucija signala časovne domene s filtrirno funkcijo.

Opis problema

Iz zvoka, katerega smo dobili na učilnici (zvok klarineta, pomešan s človeškim glasom), je potrebno ugotoviti, katere frekvence pripadajo klarinetu. Nato je potrebno ustvariti filter, kateri bo filtriral zvok (narediti je potrebno bandstop filter) od klarineta in ohranil kar se da čist človeški glas.

Cilji naloge

Cilj naloge je, da naredim izris frekvenc iz podanega zvoka in ugotovim, katere pripadajo klarinetu. Nato te filtriram s pomočjo bandstop filtra in vse skupaj predvajam. Na koncu naloge je cilj, da odstranim zvok clarineta iz celotnega posnetka.

2. Methods and theory

Razlika med IIR in FIR filtroma

Digitalni filter je matematični algoritem, ki deluje na digitalnem nabour podatkov (npr. podatki senzorjev), da pridobi informacije, ki nas zanimajo in odstrani vse neželene informacije. Aplikacije te vrste tehnologije vključujejo odstranjevanje napak s podatkov senzorja ali celo čiščenje hrupa na izmerjenem signalu za lažjo analizo podatkov.

Poznamo dve kategoriji digitalnih filtrov:

- Infite impulse response (IIR)
- Finite impulse response (FIR)

FIR filtri

FIR (Finite impulse response) filtri so na splošno izbrani za aplikacije, kjer je linearna faza pomembna in je na voljo dostojna količina pomnilnika in računske zmogljivosti. Široko jih uporabljajo v aplikacijah za izboljšanje audio in biomedicinskih signalov. Njihova all-zero struktura zagotavlja, da nikoli ne postanejo nestabilni za kateri koli tip vhodnega signala, kar jim daje izrazito prednost pred IIR.

So dosti enostavnejši za implementacijo, imajo pa večjo računsko oz. Časovno komplksnost, kot IIR filtri.

$$y(n) = \sum_{k=0}^{q} b_k x(n-k) - \sum_{k=1}^{p} a_k y(n-k)$$

IIR filtri – uporabljen v seminarski nalogi

IIR (Infite impulse response) filtri so na splošno izbrani za aplikacije, kjer linearna faza ni preveč pomembna in je pomnilnik omejen. Veliko jih uporabljajo v izenačevanju zvoka, obdelavi signalov biomedicinskih senzorjev, pametnih senzorjih IoT / IIoT in hitrih telekomunikacijskih / RF aplikacijah.

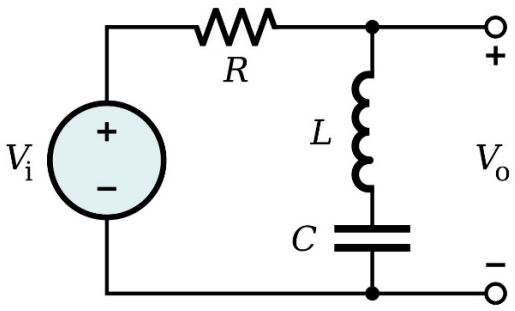
$$y(n) = \sum_{k=0}^{\infty} h(k) x(n-k)$$

IIR filter sem uporabil zato, ker ima manjšo računsko zahtevnost, ima majhno zakasnitev in je ekvivalnten analognem filtru.

- **Band Stop Filter** deluje tako, da izklopi frekvence, ki so v določenem območju, kar omogoča enostaven prehod samo na frekvence izven tega območja. Poznan tudi kot band-eliminacija, band-zavrnitev ali zareza filtri.Lahko predstavljen, kot kombinacija Low-pass in High-pass filtra, če je pasovna širina dovolj široka, da se filtra med seboj preveč ne stikata.

$$H(s) = \frac{s^2 + \omega_z^2}{s^2 + \frac{\omega_p}{Q}s + \omega_p^2}$$

Transfer Function



Shema enostavnega band-stop filtra

- **Low Pass Filter** omogoča, da frekvence, ki so pod neko določeno vrednostjo, prehajajo brez kakršnegakoli popačenja, frekvence nad to določeno vrednostjo pa omilijo.

- **High Pass Filter** enako, kot low pass tudi on omogoča, da signali prehajajo brez kakršnegakoli popačenja, v primeru, če imajo višjo frekvenco, kot določena meja, sicer se bo amplituda signala oslabila.

Primer:

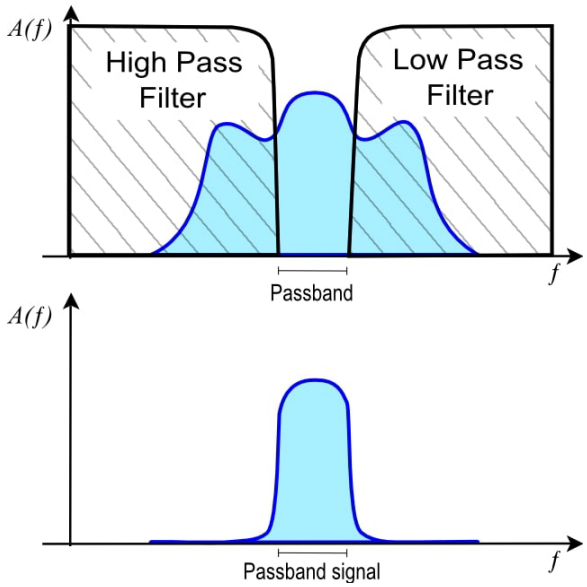
Filtriranje zvoka frekvence 400Hz

```
%uporabim stopband filter na zvoku
out = filter(filterF(420, 430, 450, 460, .5,  60, 1, Fs, 'stopband'),read);
```

Nižje vrednosti od 440 so Low Pass, višje pa High Pass

Passband

Je obseg frekvenc ali valovnih dolžin, ki lahko prehajajo skozi filter. Radijski sprejemnik na primer vsebuje bandpass filter za izbiro frekvence želenega radijskega signala med vsemi radijskimi valovi, ki jih zajema njegova antena. Passband sprejemnika je obseg frekvenc, ki jih lahko sprejme, ko je nastavljen na želeno frekvenco (kanal).

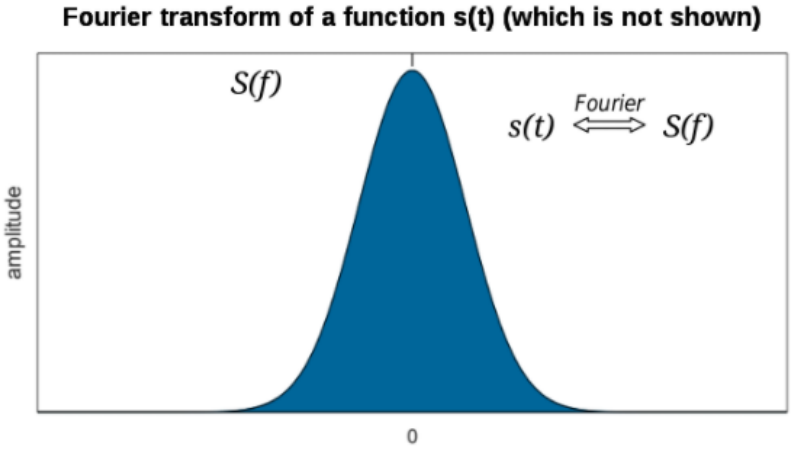


Primer delovanja

Discrete Fourier transform

V matematiki diskretna Fourierjeva transformacija (DFT) pretvori končno zaporedje enako razmaknjenih vzorcev funkcije v isto dolžinsko zaporedje, enako razmaknjenih vzorcev diskretne Fourierjeve transformacije (DTFT), katera je kompleksno vrednotena funkcija frekvence. Interval, v katerem se vzorči DTFT, je vzajemnost trajanja vhodnega zaporedja. Inverzni DFT je Fourierjeva serija, ki uporablja vzorce DTFT kot koeficiente kompleksnih sinusov na ustreznih frekvencah DTFT. Ima enake vzorčne vrednosti kot prvotno vhodno zaporedje.

DFT je najpomembnejša diskretna transformacija, ki se uporablja za izvajanje Fourierjeve analize v številnih praktičnih aplikacijah. Pri digitalni obdelavi signala je funkcija katera koli količina ali signal, ki se s časom spreminja, na primer tlak zvočnega vala, radijski signal ali odčitki dnevne temperature, vzorčeni v končnem časovnem intervalu.



V seminarski nalogi sem uporabil Fast fourier transform(FFT), ker deluje hitreje, kot DTFT, oba pa uporabljata DFT.

```
%fft za pretvorbo originalnega zvoka
firstTransform = fft(read);
```

Primer uporabe v seminarski nalogi

Opis delovanja kode

Task1:

S pomočjo funckije audioread preberem zvočno datoteko v spremenljivko »read«. Nato sem nad spremenljivko »read« naredil Fast Fourier Transform (FFT). S figure; sem ustvaril novo okno in plot-al signal absolutno vrednost spremenljivke zato, ker smo potrebovali pozitivne vrednosti y.

```
%task1
[read,fs] = audioread('1.danes_je_lep_dan_klarinet_22050.wav');
N = length(read);
f = (0:(N-1))/N*fs;
ftx = fft(read);
figure; plot(f,abs(ftx));
title('Amplitude Spectrum of the audio signal');
xlabel ('Frequency[Hz]');
ylabel('Amplitude');
```

Task2:

Signal »read«, katerega sem prebral, sem nato filtriral s pomočjo »stopband«. Filtru sem določil Low Pass (vrednosti so za 20Hz manjše od originalnih) in High Pass (vrednosti so za 20Hz višje od originalnih).

```
%task2
%uporabim stopband filter na zvoku
out = filter(filterF(420, 430, 450, 460, .5,  60, 1, Fs, 'stopband'),read);

out2 = filter(filterF(1300, 1310, 1330, 1340, .5,  60, 1, Fs, 'stopband'),out);

out3 = filter(filterF(2000, 2100, 2300, 2400, .5,  60, 1, Fs, 'stopband'),out2);

out4 = filter(filterF(3060, 3070, 3090, 3100, .5,  60, 1, Fs, 'stopband'),out3);

out5 = filter(filterF(3940, 3950, 3970, 3980, .5,  60, 1, Fs, 'stopband'),out4);

out6 = filter(filterF(4820, 4830, 4850, 4860, .5,  60, 1, Fs, 'stopband'),out5);
```

Funkcijo filter sem ustvavril malo nižje, na zgornji sliki jo pa program kliče.

```
function Hd = filterF(Fpass1, Fstop1, Fstop2, Fpass2, Apass1, Astop, Apass2, Fs, match)
    h  = fdesign.bandstop(Fpass1, Fstop1, Fstop2, Fpass2, Apass1, Astop, Apass2, Fs);
    Hd = design(h, 'butter', 'MatchExactly', match);
end
```

S funkcijo sound, sem filtrirano vrednost zaigral in jo nato z audiowrite zapisal v novo datoteko.

```
%zaigra filtriran zvok
sound(out6,Fs);
%zapiše novo filtrirano datoteko
audiowrite('izhod.wav',out6,Fs);
```

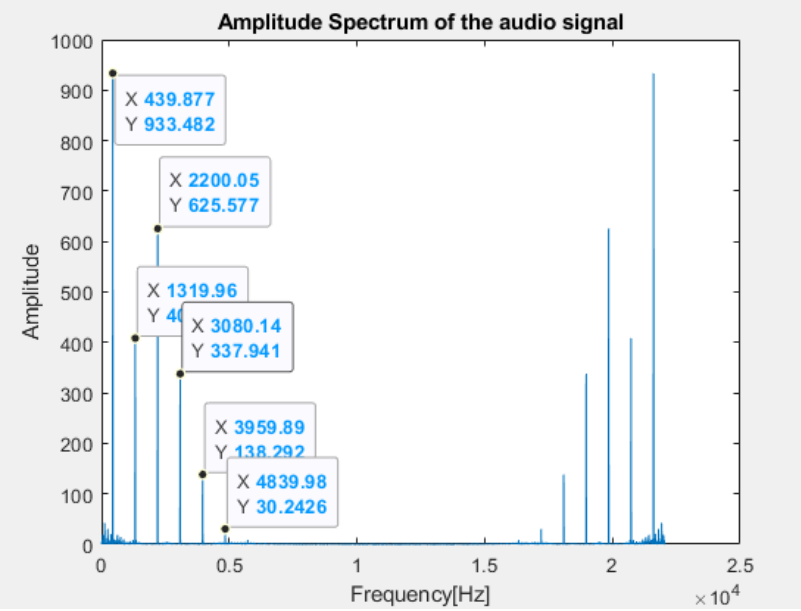
Nato sem ponovno pognal fft preko prebrane datoteke in ustvaril vektor frekvence za izris podatkov. Nato plot-am nefiltriran signal. (Moder signal na rezultatu)

```
%fft za pretvorbo originalnega zvoka
firstTransform = fft(read);
firstTransform = firstTransform(1:N/2+1);
%naredim vector frekvence za izris podatkov
f0 = 0:fs/N:fs/2;
figure; hold on;
plot(f0,abs(firstTransform));
```

Potem sem naredil enako, le da sem fft pognal preko filtriranega izhodnega zvočnega fila, katerega sem malo višje ustvaril na novo. Na enak način sem preračunal vektor in plot-al. (Oranžen signal na rezultatu)

```
[read1,fs1] = audioread('izhod.wav');
N1 = length(read1);
%fft za pretvorbo originalnega zvoka
secondTransform = fft(read1);
secondTransform = secondTransform(1:N1/2+1);
%naredim vector frekvence za izris podatkov
f1 = 0:fs1/N1:fs1/2;
plot(f1,abs(secondTransform));
title('Amplitude Spectrum after aplying Band Stop Filter');
xlabel ('Frequency[Hz]'); ylabel('Amplitude'); grid on;
hold off;
```

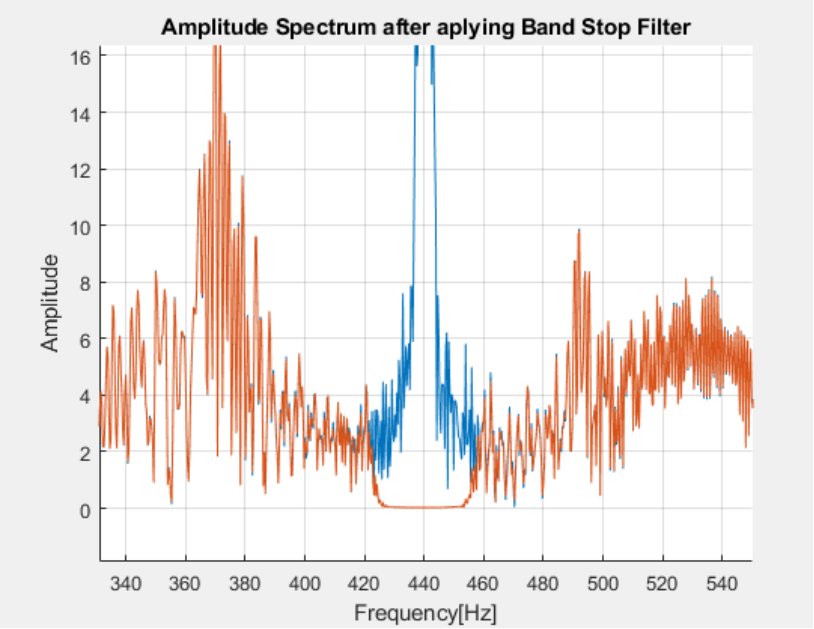
3. Results



Plot Task1: izpisan originalni vhodni audio file

Iz grafa je razvidno, katere frekvence pripadajo klarinetu in sicer:

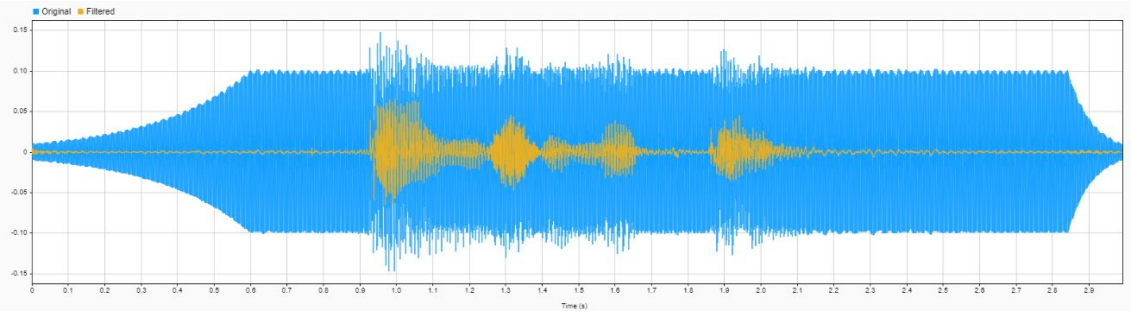
F1 = 440Hz, F2 = 1320Hz, F3 = 2200Hz, F4 = 3080Hz, F5 = 3960Hz, F6 = 4840Hz



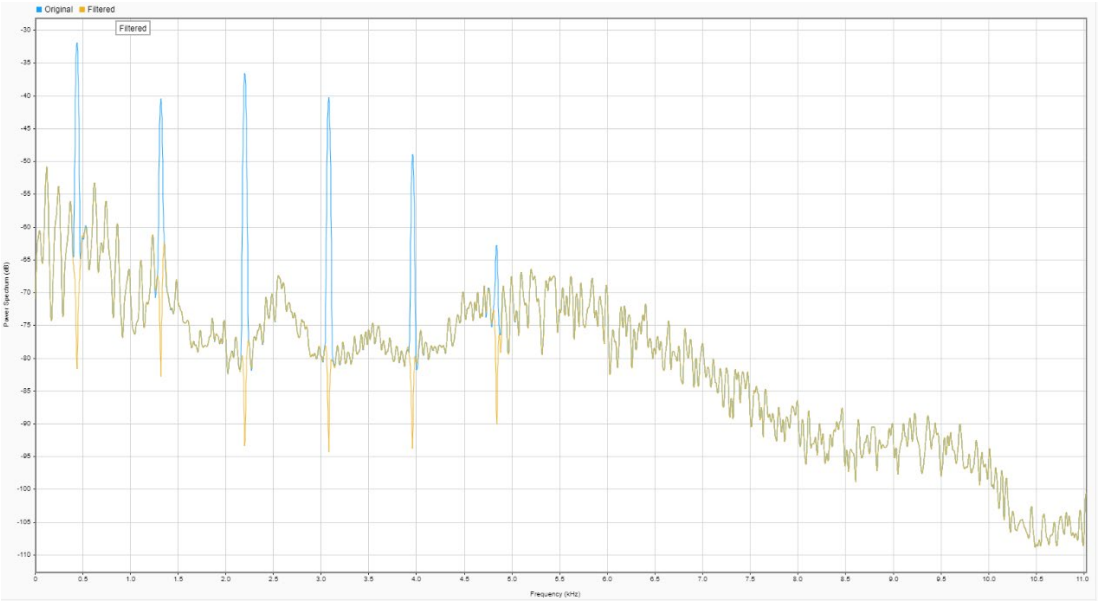
Plot Task2: filtriran in nefiltriran signal

Na sliki opazimo filtriran (oranžen) in nefiltriran (moder) signal.

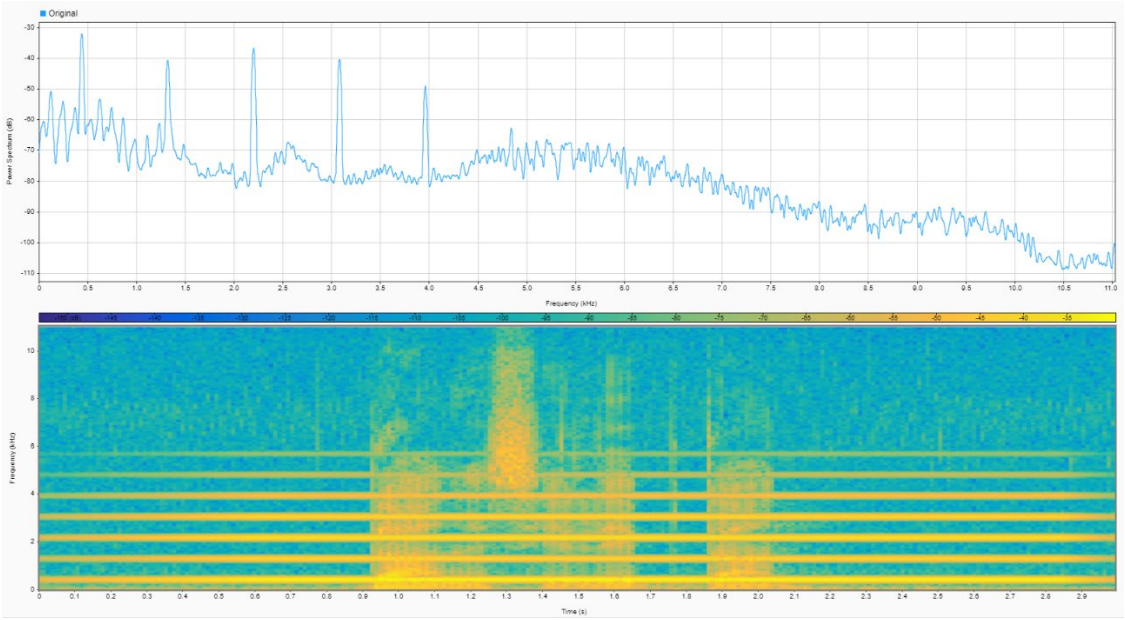
Za spodnje 4 grafe sem uporabljal program Signal Analyzer.



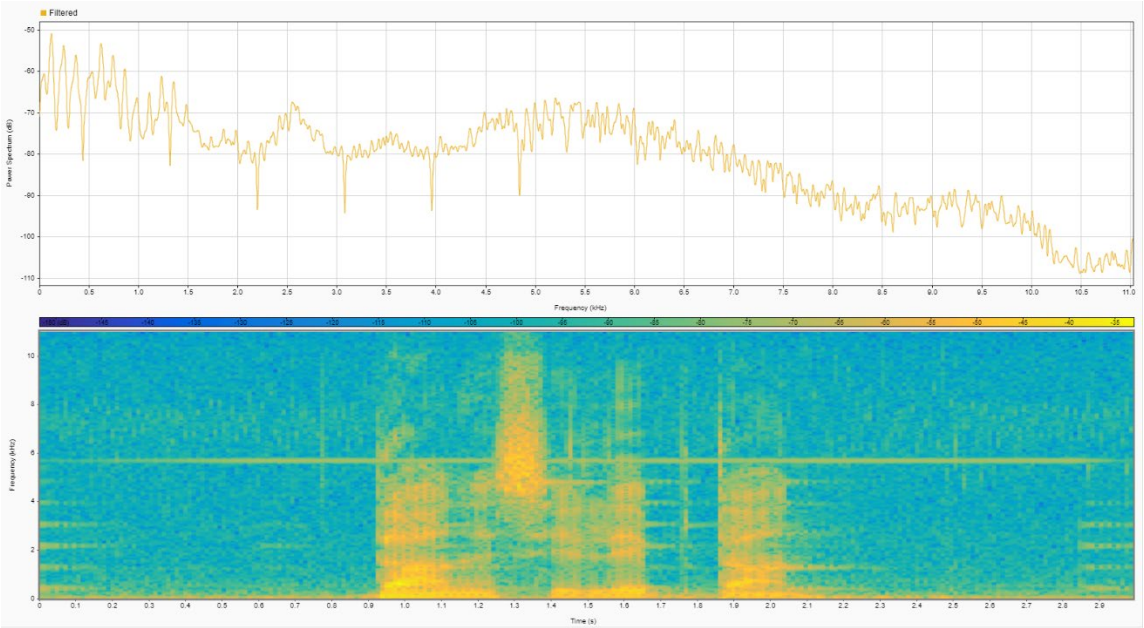
Audio signal in time domain



Power spectrum

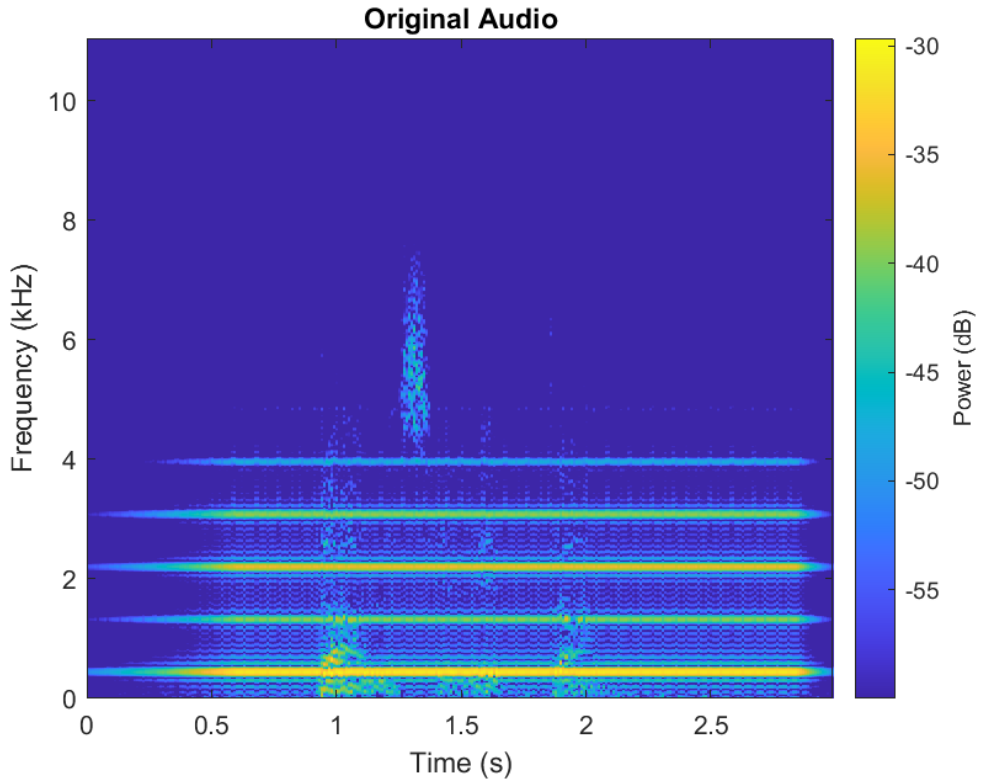


Signal originalne zvočne datoteke

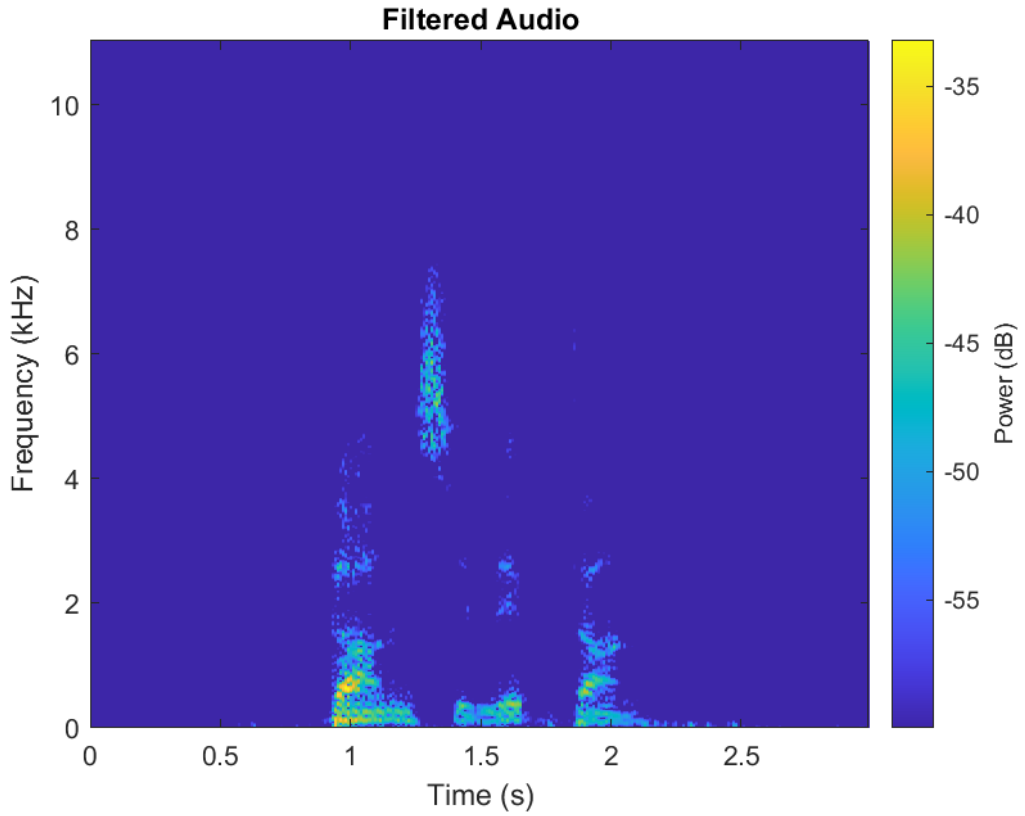


Signal filtrirane zvočne datoteke

Spodaj sta prikazana še 2 spectograma originalnega in filtriranega signala.




```
figure;
pspectrum(read,fs,'spectrogram','OverlapPercent',0,
'Leakage',1,'MinThreshold',-60,'TimeResolution', 10e-3);
title("Original Audio");
```




```
figure;
pspectrum(out6,fs,'spectrogram','OverlapPercent',0,
'Leakage',1,'MinThreshold',-60,'TimeResolution', 10e-3);
title("Filtered Audio");
```

4. Discussion

Pričakovane cilje sem na zaključku naloge izpolnil. Zvočni posnetek mi je uspelo filtrirati s pomočjo IIR filtra in odstraniti zvok klarineta.problematični so bili deli posnetka, kjer je imel klarinet »vrhove« oz. Velike frekvence, saj se je zvok zelo slišal.