Mini Design Project
EECS 3451
Alain Ballen
216341703
December 6, 2022
Virtual

Table of Contents

INTRODUCTION **3**

EQUIPMENT AND SOFTWARE USED **3**

RESULTS AND DISCUSSION **3**

Question 1 3

Question 2 4

CONCLUSION **9**

APPENDIX **9**

Introduction

The objective of this project is to design two filters (time domain and frequency domain) to remove abnormal noise in the given wave file.

Equipment and Software Used

- Matlab
- Macbook Pro

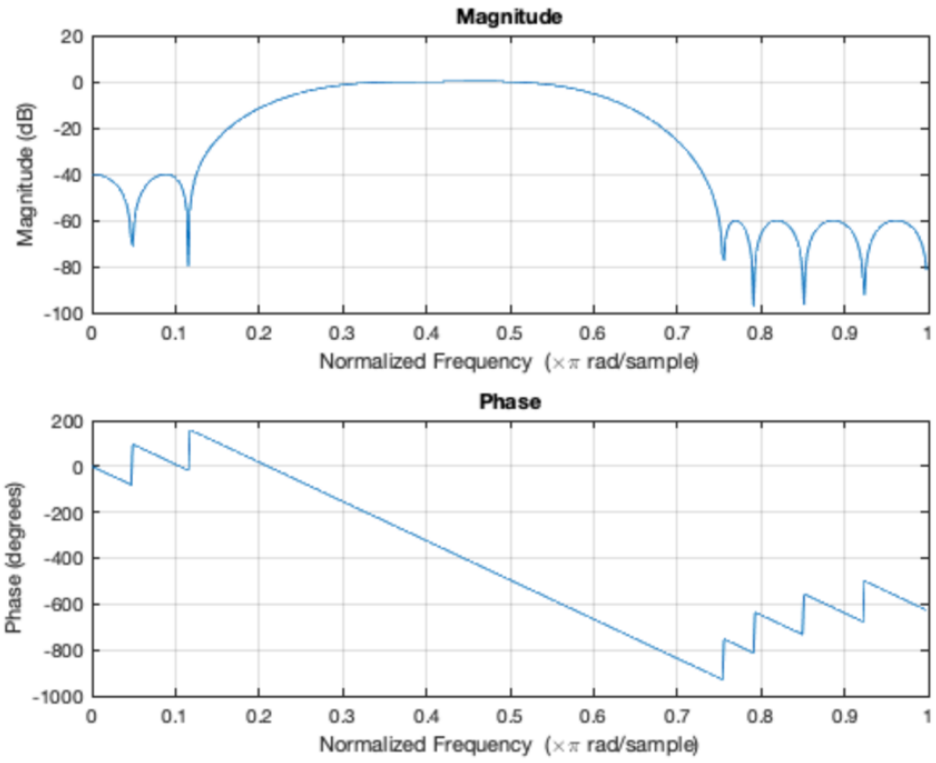
Results and Discussion

Question 1:

```
clear;
close all;

fs=8000; % Sampling frequency
f=[500, 1500, 2000, 3000];
a=[0,1,0]; % Stopband/Passband order for a bandpass filter
dev=[0.01,0.01,0.001];

[N,Fi,Ai,W]=firpmord(f,a,dev,fs); % get
h=firpm(N,Fi,Ai,W); % Find coefficients
freqz(h,1);
```

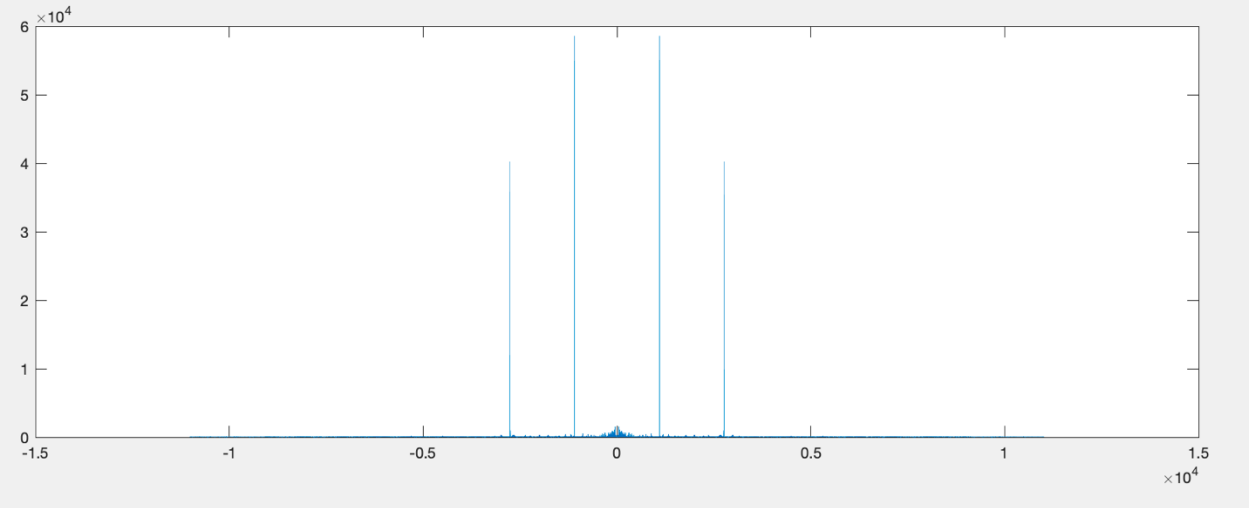


Filter length: 19

Filter coefficients:
h(0) = -0.089 = h(19)
h(1) = -0.0039 = h(18)
h(2) = 0.0275 = h(17)
h(3) = 0.0187 = h(16)
h(4) = -0.0025 = h(15)
h(5) = 0.0512 = h(14)
h(6) = -0.0068 = h(13)
h(7) = -0.2256 = h(12)
h(8) = -0.1293 = h(11)
h(9) = 0.2845 = h(10)

Question 2:

I have identified the noisy signals by first plotting the FFT of the sampled data and shifting it. Upon hearing the audio, the magnitude of the noise is very high as it is very loud. Therefore, noisy data is represented by the long vertical lines as shown below.



Filter specifications:

time-domain method:
lowpass:

- Passband frequency: 1000 Hz
- Stopband frequency: 1100 Hz
- Passband/Stopband ripple: 0.0001

bandpass:

- Stopband 1 frequency: 1100 Hz
- Passband 1 frequency: 1500 Hz

- Passband 2 frequency: 2000 Hz
- Stopband 2 frequency: 2700 Hz
- Passband ripple: 0.001
- Stopband ripple:0.0001

highpass:

- Stopband frequency: 2760 Hz
- Passband frequency: 2780 Hz
- Passband/Stopband ripple: 0.001

frequency-domain method:

lowpass:

- Cutoff frequency: 1000
- Order: 100

bandpass:

- Stopband 1 frequency: 1000 Hz
- Passband 1 frequency: 1150 Hz
- Passband 2 frequency: 2720 Hz
- Stopband 2 frequency: 2770 Hz
- Passband ripple: 1
- Stopband1 ripple:20
- Stopband2 ripple: 80

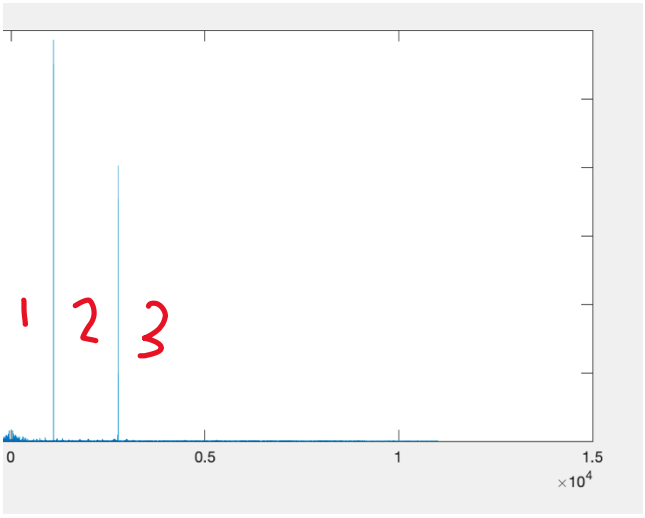
highpass:

- Stopband frequency: 2750 Hz
- Passband frequency: 2820 Hz
- Passband ripple: 50
- Stopband ripple: 4

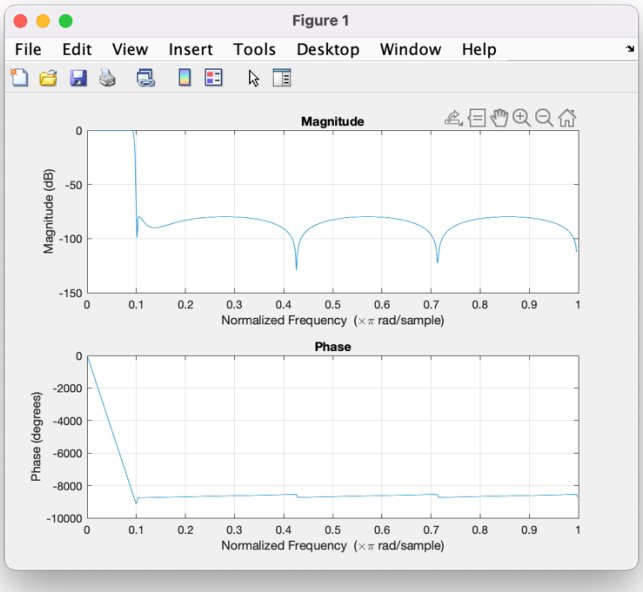
Analysis:

The differentiation is between the time domain method and frequency domain method is how the coefficient is filtered. The time domain method uses to convolution function in matlab (conv) and the frequency domain uses the filter function in matlab (filter).

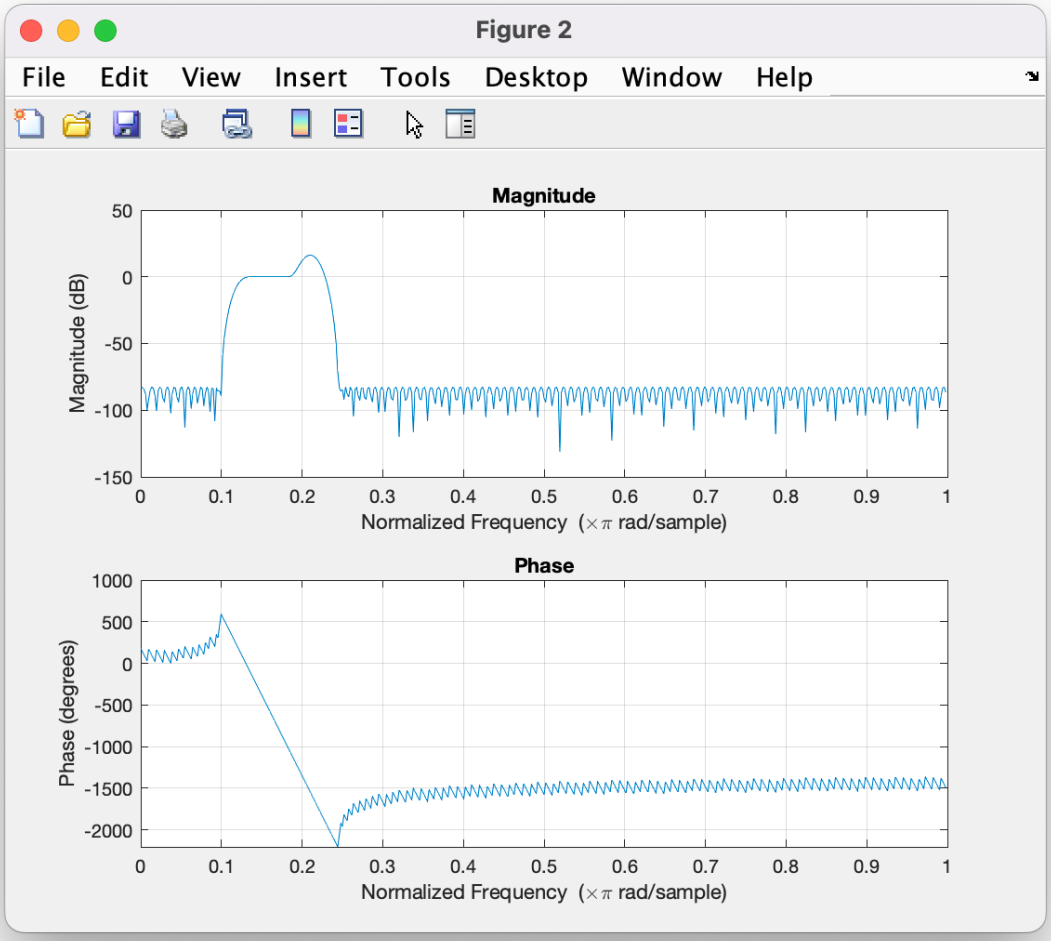
The design of the filters required multiple types of filters to get the audio. As shown in the plot, there are three regions of frequency in which we want to keep. This can be done by using a lowpass filter followed by a bandpass filter followed by a high pass filter. Then we add those results together to form the final result.



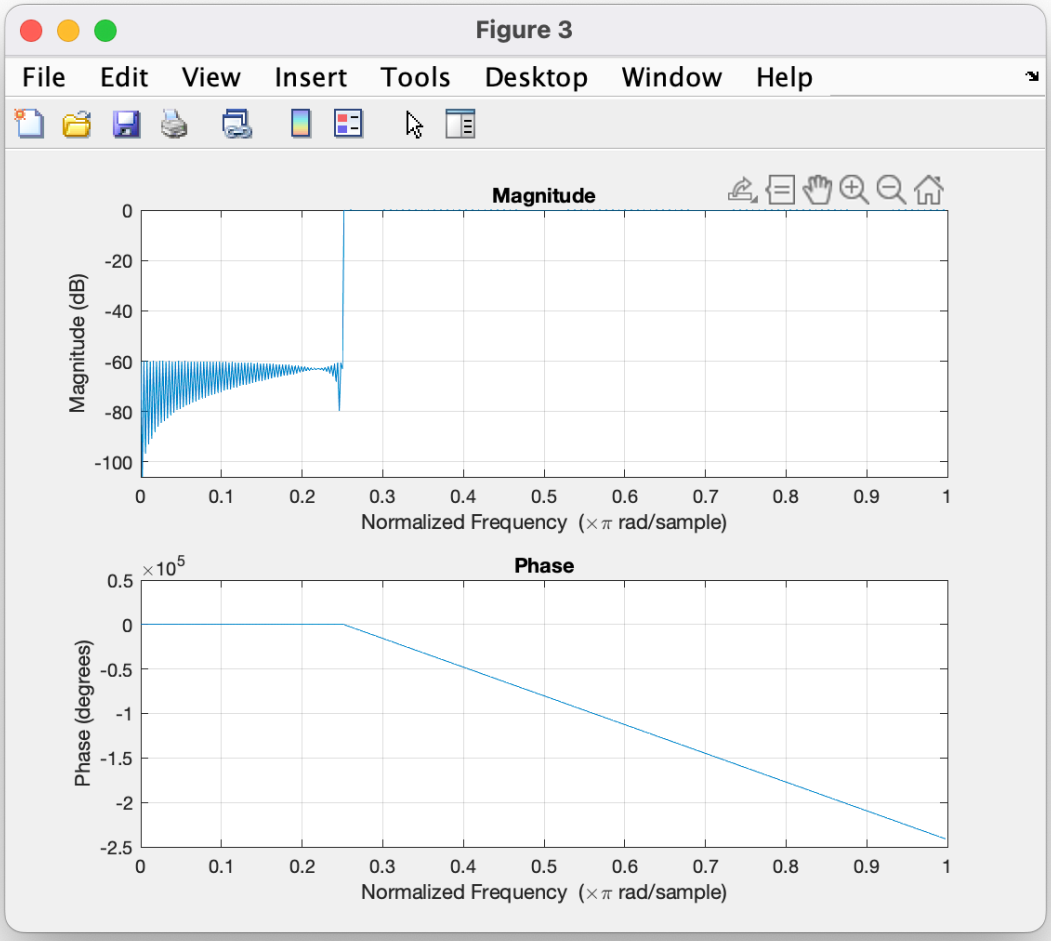
Splitting the graph into 3 regions.



Above is the lowpass filter. As you can see it will only allow the lower frequencies (lower frequency than region 1) to pass and exclude the first abnormal noise. In the song, there are some high frequencies that we want to keep.

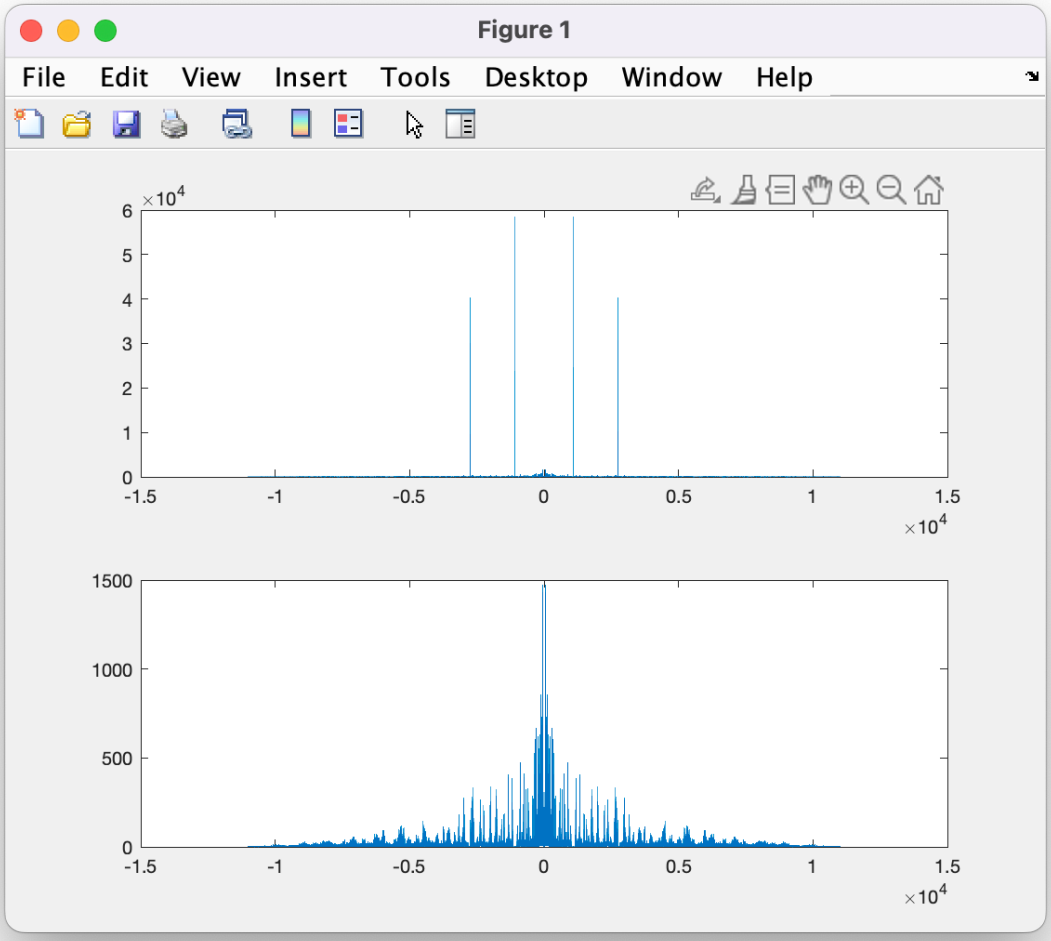


Above is the bandpass filter. This includes all of region 2 and excludes both abnormal noises



Above is the highpass filter to include all of region 3 and exclude the second abnormal noise.

Below is the resulting graph showing that the noises are removed in the processed signal (next page)



Conclusion

The objectives of this lab was to design a filter according to specifications and design two filters to remove abnormal noise in the given wave filer using frequency domain analysis and time domain analysis. I have learned how to use various functions to design filters and that I can combine the results of these filters to create a custom filter. There were many difficulties such as figuring out the accuracy of the filters. I have found a useful tool offered by matlab called "filterDesigner" which implements butterworth filters for me to use in the frequency domain. This allowed me to quickly change the frequencies found by analyzing the graph of the nonfiltered audio and to use trial and error to determine exact frequencies.

Appendix (Question 2 only as code for Question 1 is in results/discussion)

Time Domain Code:

```
clear;
close all;

% read samples and sampling frequency
[y,fs] = audioread('music_noisy.wav');

%
% Low pass filter
%
fLow=[1000,1100]; % define passband edge and stopband edge
aLow=[1,0]; % order (passband, stopband)
devLow=[0.0001,0.0001]; % define stopband and passband ripples

[NL,FiL,AiL,WL]=firpmord(fLow,aLow,devLow,fs);
bLow=firpm(NL,FiL,AiL,WL); %coefficient

%
% Band pass filter
%
fBand=[1100,1500,2000,2700]; % define passband edges and stopband edges
aBand=[0,1,0]; % order (stopband,passband,stopband)
devBand=[0.0001,0.001,0.0001]; % define stopband and passband ripples

[NB,FiB,AiB,WB]=firpmord(fBand,aBand,devBand,fs);
bBand=firpm(NB,FiB,AiB,WB); %coefficient

%
% Band pass filter
%
fHigh=[2760,2780]; % define passband edges and stopband edges
aHigh=[0,1]; % order (stopband,passband)
devHigh=[0.001,0.001]; % define stopband and passband ripples

[NH,FiH,AiH,WH]=firpmord(fHigh,aHigh,devHigh,fs);
bHigh=firpm(NH,FiH,AiH,WH); %coefficient

% compute convolution of each filter's coefficient and samples
Low = conv(y,bLow,'same');
Med = conv(y,bBand,'same');
High = conv(y,bHigh,'same');

res = Low + Med + High; % Combine results of convolutions
audiowrite('testfir.wav', res, fs); % write to file using new samples

[test] = audioread('testfir.wav'); % read samples of file
testz = fft(test); % Fast fourier transform of samples
z = fftshift(testz); % Shift
[N,P] = size(test);
f=-fs/2:fs/(N-1):fs/2;
figure;
plot(f,abs(z)); % analysis of plots
```

Frequency Domain Code: (next page)

```
close all;
clear;
[y,fs] = audioread('music_noisy.wav');

[N,P] = size(y);
f=-fs/2:fs/(N-1):fs/2;
x = fft(y);
z = fftshift(x);
hh = filter(lowpass_butter,y);
yy = filter(bandpass,y);
xx = filter(highpass,y);

res = hh+yy+xx;

audiowrite('filtered.wav',res,fs);
[y2,fs2] = audioread('filtered.wav');


[N2,P2] = size(y2);
f2=-fs2/2:fs2/(N-1):fs2/2;
x2 = fft(y2);
z2 = fftshift(x2);
subplot(2,1,1);
plot(f,abs(z));
subplot(2,1,2);
plot(f2,abs(z2));


function Hd = lowpass_butter
%LOWPASS_BUTTER Returns a discrete-time filter object.

% MATLAB Code
% Generated by MATLAB(R) 9.13 and Signal Processing Toolbox 9.1.
% Generated on: 04-Dec-2022 01:04:44

% Butterworth Lowpass filter designed using FDESIGN.LOWPASS.

% All frequency values are in Hz.
Fs = 22050;  % Sampling Frequency

N  = 100;     % Order
Fc = 1000;  % Cutoff Frequency

% Construct an FDESIGN object and call its BUTTER method.
h  = fdesign.lowpass('N,F3dB', N, Fc, Fs);
Hd = design(h, 'butter');
end
% [EOF]

function Hd = bandpass
%BANDPASS Returns a discrete-time filter object.

% MATLAB Code
% Generated by MATLAB(R) 9.13 and Signal Processing Toolbox 9.1.
% Generated on: 05-Dec-2022 23:09:43

% Butterworth Bandpass filter designed using FDESIGN.BANDPASS.

% All frequency values are in Hz.
Fs = 22050;  % Sampling Frequency

Fstop1 = 1000;        % First Stopband Frequency
Fpass1 = 1150;        % First Passband Frequency
Fpass2 = 2720;        % Second Passband Frequency
Fstop2 = 2770;        % Second Stopband Frequency
Astop1 = 20;          % First Stopband Attenuation (dB)
Apass  = 1;           % Passband Ripple (dB)
Astop2 = 80;          % Second Stopband Attenuation (dB)
match  = 'stopband';  % Band to match exactly

% Construct an FDESIGN object and call its BUTTER method.
h  = fdesign.bandpass(Fstop1, Fpass1, Fpass2, Fstop2, Astop1, Apass, ...
                      Astop2, Fs);
Hd = design(h, 'butter', 'MatchExactly', match);
end

function Hd = highpass
%HIGHPASS Returns a discrete-time filter object.

% MATLAB Code
% Generated by MATLAB(R) 9.13 and Signal Processing Toolbox 9.1.
% Generated on: 06-Dec-2022 20:37:11

% Butterworth Highpass filter designed using FDESIGN.HIGHPASS.

% All frequency values are in Hz.
Fs = 22050;  % Sampling Frequency

Fstop = 2750;        % Stopband Frequency
Fpass = 2820;        % Passband Frequency
Astop = 50;          % Stopband Attenuation (dB)
Apass = 4;           % Passband Ripple (dB)
match = 'stopband';  % Band to match exactly

% Construct an FDESIGN object and call its BUTTER method.
h  = fdesign.highpass(Fstop, Fpass, Astop, Apass, Fs);
Hd = design(h, 'butter', 'MatchExactly', match);
end
% [EOF]
```