Step 1: Notch Filter

• Read two audio files, Bambi1.wav (y1 and fs1) and freqSweep.wav (y2 and fs2).

```
afile = 'freqSweep.wav';
[y2,fs2]=audioread(afile) ;
sound(y2,fs2);
```

• Verify the signals using sound. Note the two different sampling frequencies, so filters designed to have frequency selective behavior will be different coefficients for the two signals. If the sampling rate for your own selected audio file is different, you will need to create different coefficients for its sampling rate.

• For each audio signal sampling frequency, do the following:

o Get filter coefficients for the IIR 1000Hz notch filters with r = 0.9 and r = 0.6 from your prelab that you have already tested by using zplane to plot the poles and zeros for each filter and using freqz to plot the DTFT of the filters.

o Also plot the magnitude of the frequency response in dB by plotting 20*log10(abs(H)+0.0001) instead of plotting abs(H). You can also use the MATLAB function mag2db. The addition of a small number to abs(H) controls the dynamic range of the plot and prevents a really large negative number if abs(H) is actually 0.

o Use MATLAB's filter function to generate the filtered audio signals for each of the three audio files using the two different values of r. Listen to the outputs using sound. How do the filters compare for the audio signals?

o Use spectrogram to observe the frequency content of signal y as a function of time using:

```
sp_win = 1024; sp_ovr = 512; sp_fftN = 1024;
spectrogram(y, sp_win, sp_ovr, sp_fftN, fs, 'yaxis')
```

o Look carefully at the spectrogram for the filtered freqSweep signal. Can you see the different frequency ranges that are attenuated by the different values of r? What is the impact of the filters on the speech spectrogram?

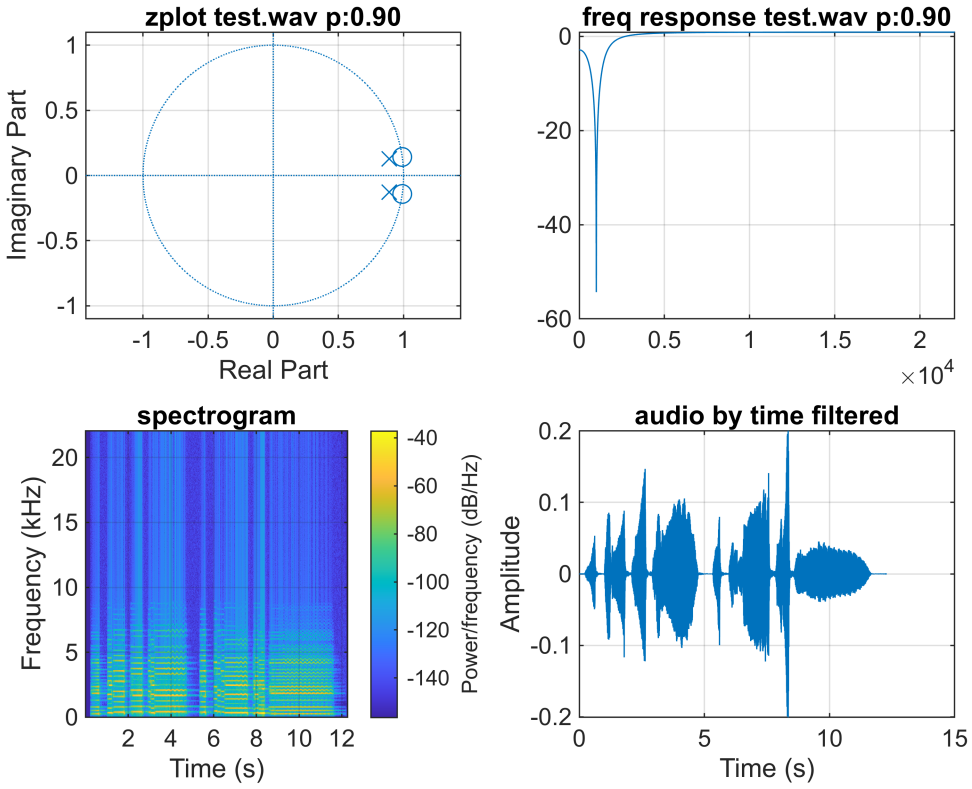
o Create a time vector to plot the output signals, y, individually as a function of time.

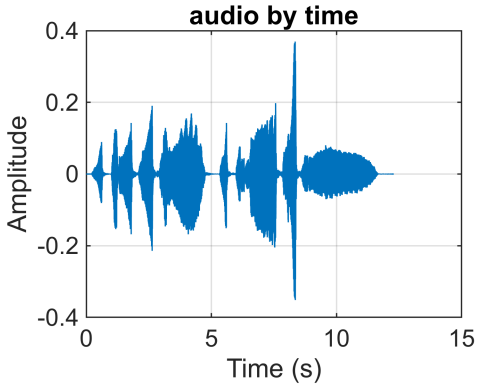
▪ In these plots of signal amplitude vs time, the freqSweep signal has more localized limited frequency content than the speech signal. For freqSweep, note the times when the plots show the highest amplitude and lowest amplitude output? From the spectrogram, determine what frequency corresponded to that time.

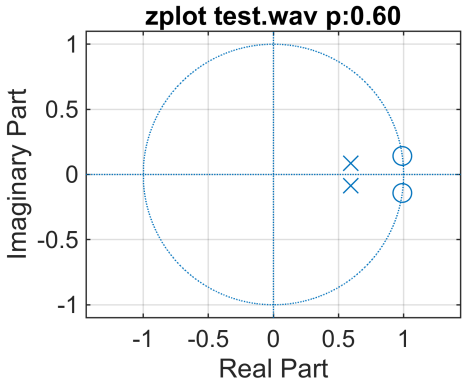
▪ For the speech signal, is there an overall difference in amplitudes? Do any parts of the time waveforms look different?

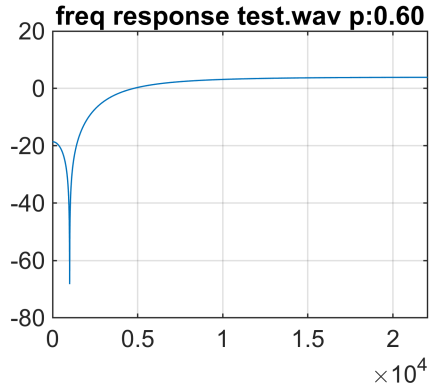
```
pt=1;
f0=1000;
p=[0.9,0.6];
n=2048;
afile=['test.wav',"Bambi1.wav",'freqSweep.wav'];
for i=1:length(afile)
   for  j=1:length(p)
    [x,fs]=audioread(afile(i));
    theta=2*pi*f0/fs;
    player = audioplayer(x, fs);
    play(player);
    pause(pt);
    stop(player);
    % sound(x,fs);
    % pause(length(x)/fs + 0.5); this is for easy sound bites for it.
    b=[1,-2*cos(theta),1];
    a=[1,-2*p(j)*cos(theta),p(j)^2];
    figure;
    subplot(2,2,1);
    zplane(b,a);
    title(sprintf("zplot %s p:%.2f",afile(i),p(j)))
    grid on;
    [H,F] = freqz(b,a,n,fs);
    subplot(2,2,2);
    plot(F,20*log10(abs(H)+0.0001))
    title(sprintf("freq response %s p:%.2f",afile(i),p(j)))
    grid on;
    y=filter(b,a,x);
    player = audioplayer(y, fs);
    play(player);
    pause(pt);
    stop(player);
    % sound(y,fs);
    % pause(length(y)/fs + 0.5);
    sp_win = 1024; sp_ovr = 512; sp_fftN = 1024;
    subplot(2,2,3);
    spectrogram(y, sp_win, sp_ovr, sp_fftN, fs, 'yaxis');
    title("spectrogram");
```

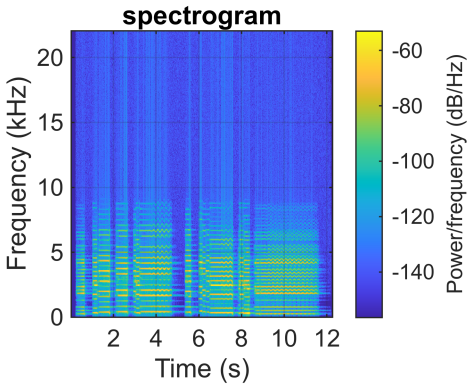
```
    %title(sprintf("spectrogram %s p:%.2f",afile(i),p(j)))
    grid on;
    subplot(2,2,4);
    t = (0:length(y)-1) / fs;
    plot(t, y);
    title("audio by time filtered");
    %title(sprintf("audiotime %s p:%.2f",afile(i),p(j)))
    xlabel('Time (s)');
    ylabel('Amplitude');
    grid on;
    figure;
    subplot(2,2,4);
    t = (0:length(x)-1) / fs;
    plot(t, x);
    title("audio by time");
    %title(sprintf("audiotime %s p:%.2f",afile(i),p(j)))
    xlabel('Time (s)');
    ylabel('Amplitude');
    grid on;
    figure;
   end
end
```

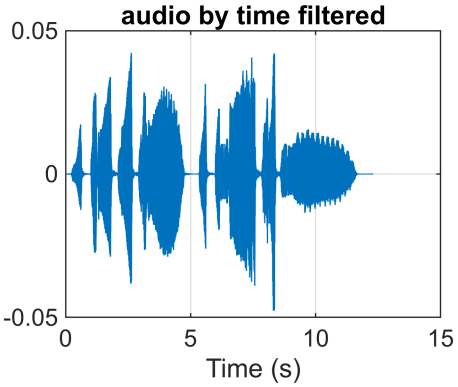


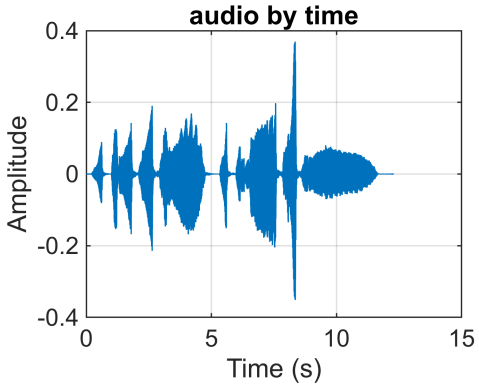

audio by time
Amplitude
Time (s)
0.4
0.2
0
-0.2
-0.4
0
5
10
15

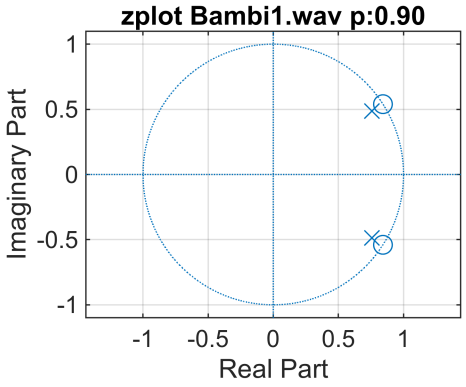

zplot test.wav p:0.60
Imaginary Part
Real Part
1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1

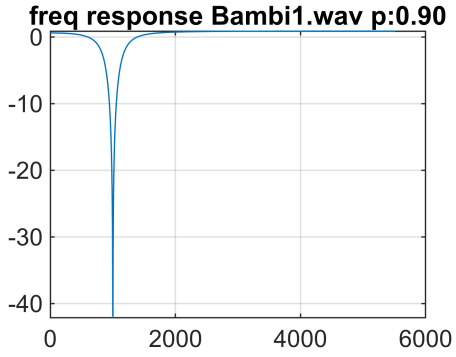

freq response test.wav p:0.60
20
0
-20
-40
-60
-80
0
0.5
1
1.5
2
×10^4

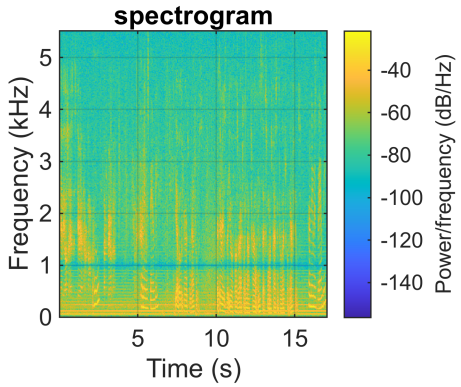

spectrogram
Frequency (kHz)
Time (s)
Power/frequency (dB/Hz)
20
15
10
5
0
2
4
6
8
10
12
-60
-80
-100
-120
-140

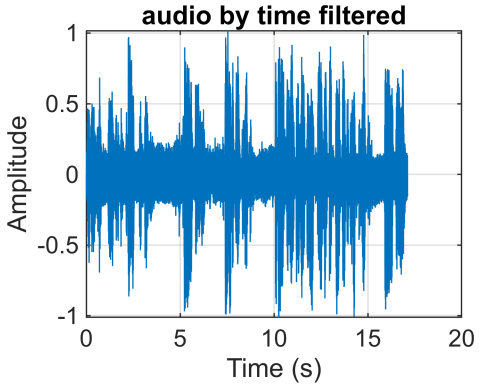

audio by time filtered
Amplitude
Time (s)
0.05
0
-0.05
0
5
10
15

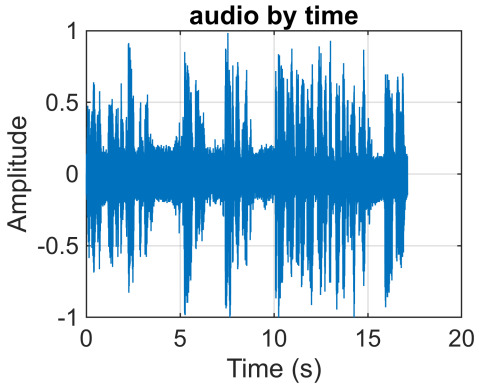

audio by time
Amplitude
Time (s)

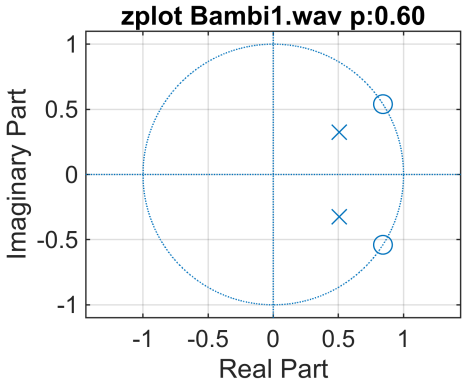

zplot Bambi1.wav p:0.90
Imaginary Part
Real Part

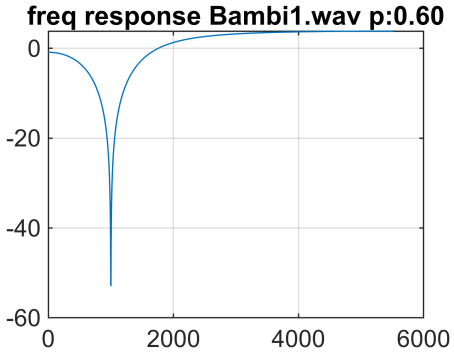

freq response Bambi1.wav p:0.90

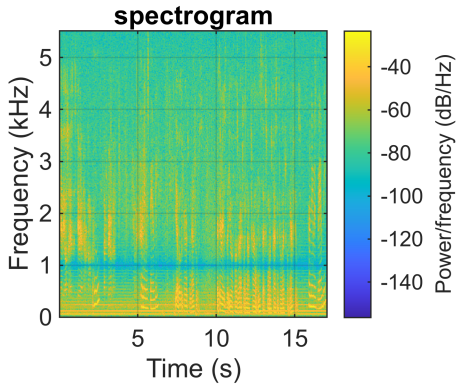

spectrogram
Frequency (kHz)
Time (s)
Power/frequency (dB/Hz)

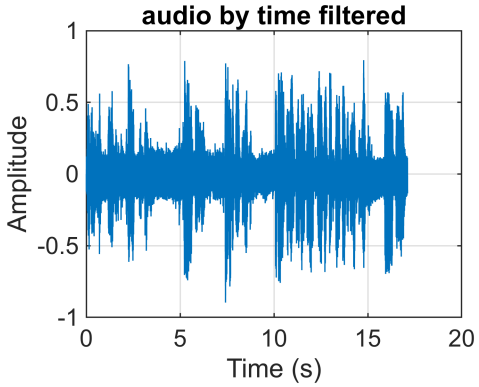

audio by time filtered
Amplitude
Time (s)

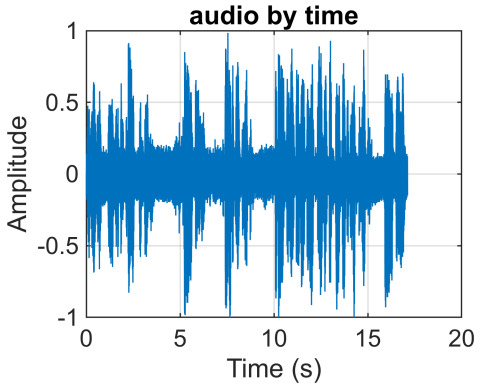

audio by time
Amplitude
Time (s)

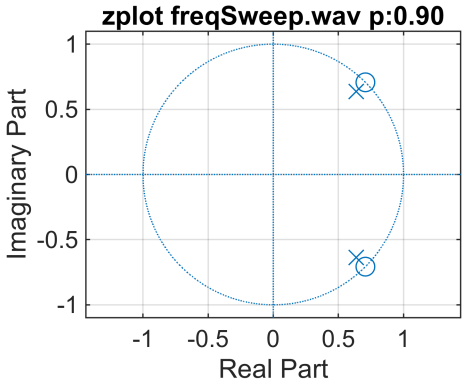

zplot Bambi1.wav p:0.60
Imaginary Part
Real Part

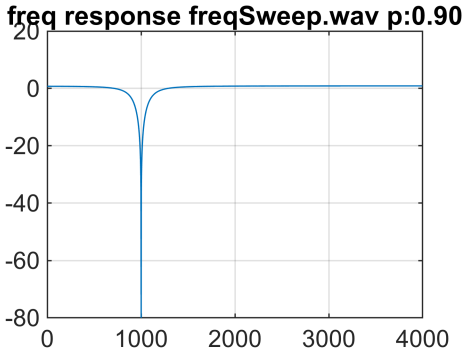

freq response Bambi1.wav p:0.60

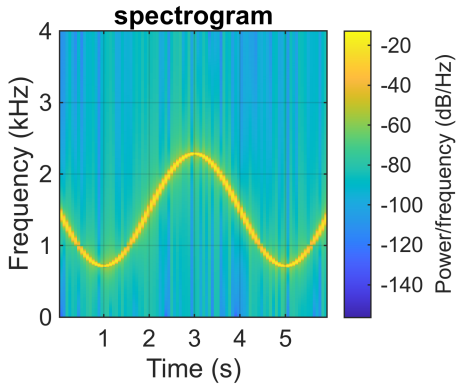

spectrogram
Frequency (kHz)
Time (s)
Power/frequency (dB/Hz)

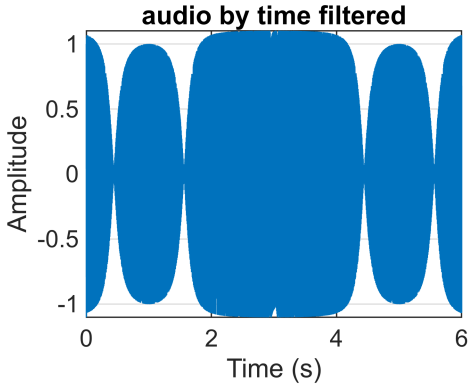

audio by time filtered
Amplitude
Time (s)

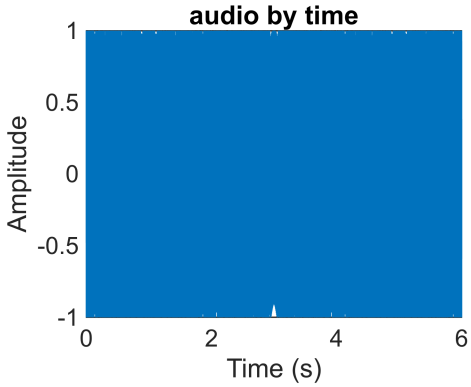

audio by time
Amplitude
Time (s)
1
0.5
0
-0.5
-1
0
5
10
15
20

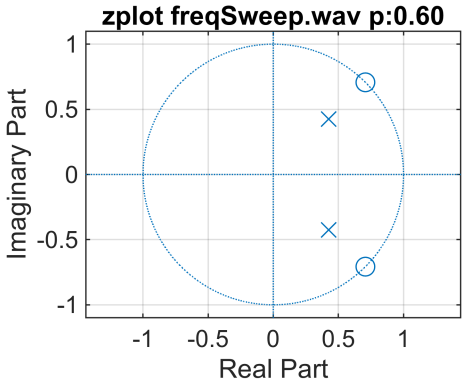

zplot freqSweep.wav p:0.90
Imaginary Part
Real Part
1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1

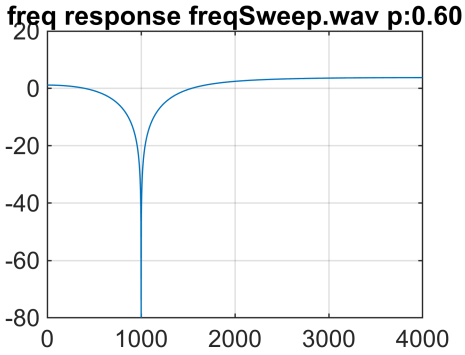

freq response freqSweep.wav p:0.90
20
0
-20
-40
-60
-80
0
1000
2000
3000
4000

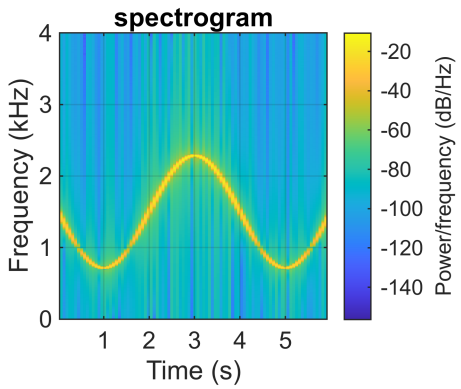

spectrogram
Frequency (kHz)
Time (s)
Power/frequency (dB/Hz)
4
3
2
1
0
1
2
3
4
5
-20
-40
-60
-80
-100
-120
-140

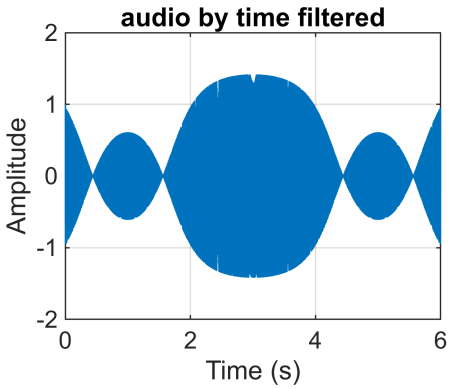

audio by time filtered
Amplitude
Time (s)
1
0.5
0
-0.5
-1
0
2
4
6

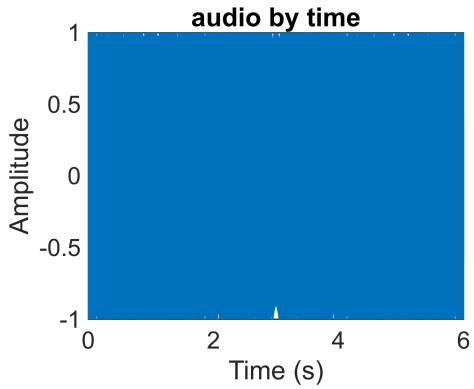

audio by time
Amplitude
Time (s)
1
0.5
0
-0.5
-1
0
2
4
6


zplot freqSweep.wav p:0.60
Imaginary Part
Real Part
1
0.5
0
-0.5
-1
-1
-0.5
0
0.5
1


freq response freqSweep.wav p:0.60
20
0
-20
-40
-60
-80
0
1000
2000
3000
4000


spectrogram
Frequency (kHz)
Time (s)
Power/frequency (dB/Hz)
4
3
2
1
0
1
2
3
4
5
-20
-40
-60
-80
-100
-120
-140


audio by time filtered
Amplitude
Time (s)
2
1
0
-1
-2
0
2
4
6



This is a bandstop filter because it has a full gain for it and then has a dip that cause it to go down.Whenever it has a dip in the ammount of frequencies that happen for it.

The audio for filterd and non filtered outputs sound exactly the same and there is little to no diffrence when using a filter for it.

You can clearly see which values are being filtered out and which ones are being kept from the spectrogram.As the spectrogram has some values especially with the sin graphs as that one is clearly seen right there.

I would say that the filter would help out with the speecs spectrogram because individual frequencies are easiear to see through the filter from it,and you can have it work for it.

For freqsweep time 2 -4 had the highest amplitude for ploting the time signal. The frequencies from spectogram is the 1.5khz -2.3khz

Lowest amplitude for it is 1.5 and 4.5 seconds for it and frequencies are from 750hz.

Depending on the P values there are diffrent amplitudes through the filters. And the time waveform doesn't look diffrent.

Step 2: Bandpass and bandstop FIR filters designed with windowed IDTFT method

Use fir1 to create a bandpass and bandstop filter for each of the audio signals. The band pass filter

will use frequency band edges 400 Hz and 1200 Hz. The narrower band stop frequency band edges are 800Hz and 1200Hz. (Be sure to divide them by fs/2 when using fir1.) For this lab we will use the default Hamming window, so no window specification is needed for fir1.

• Use fir1 to create the bandpass and the bandstop filters of order 64 for the freqSweep signal. Verify that filters are correct using zplane and freqz. Plot the magnitude of the frequency response and also plot the magnitude in dB.

• Filter the freqSweep signal with each filter and listen to the output.

• Plot the output signal as a function of time. Over what time interval is the output highly attenuated in the plot for the bandpass filtered output? For the bandstop filtered output?

• Use the spectrogram plot to determine what frequency range is highly attenuated. Is this consistent with your filter specification? If not, check the inputs for your fir1 function. In the frequency ranges where the signal is highly attenuated (i.e. where the amplitude looks like it is zero in the time plot) can you still see the frequency differences of the signal in the spectrogram? Use the colorbar to estimate the amplitude of the attenuated frequency.

• Repeat the procedure above for the speech signal and note differences in the plots of the two outputs as a function of time.

• Repeat the procedure above for your selected audio signal and note differences in the plots of the two outputs as a function of time.

Checkpoint 2: Show your results from Step 2 to your lab assistant.

```
pt=1;
sp_win = 1024; sp_ovr = 512; sp_fftN = 1024;
afile=["Bambi1.wav",'freqSweep.wav','test.wav']
```

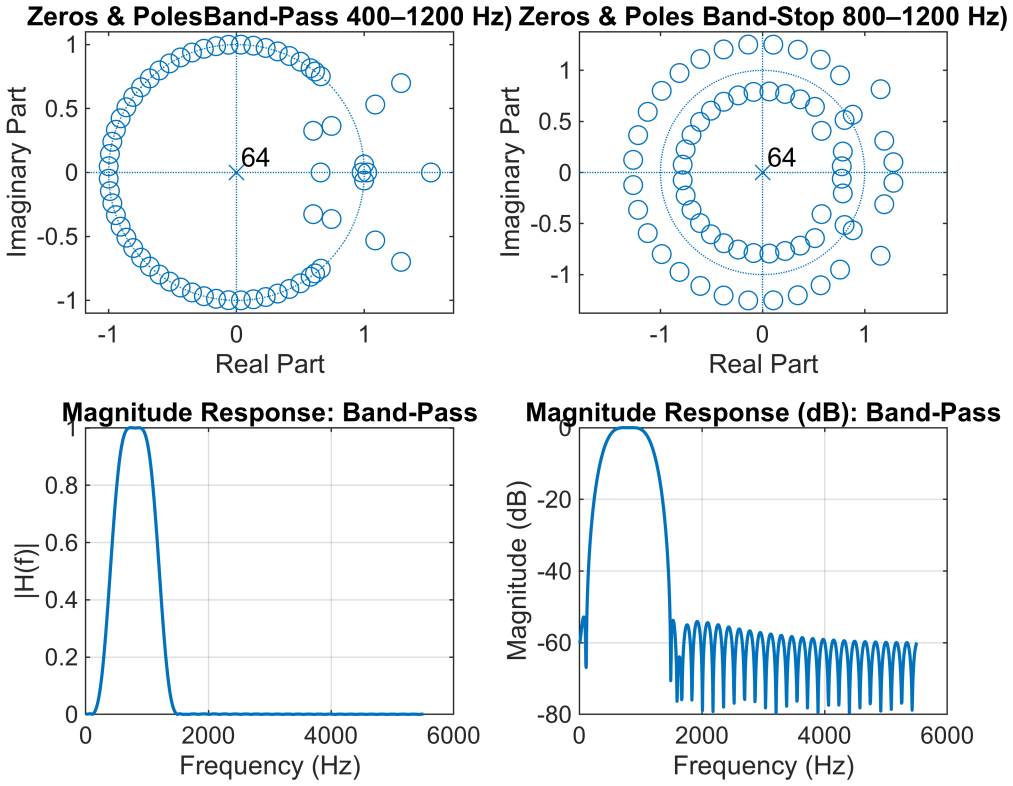
```
afile = 1×3 string
"Bambi1.wav" "freqSweep…  "test.wav"
```

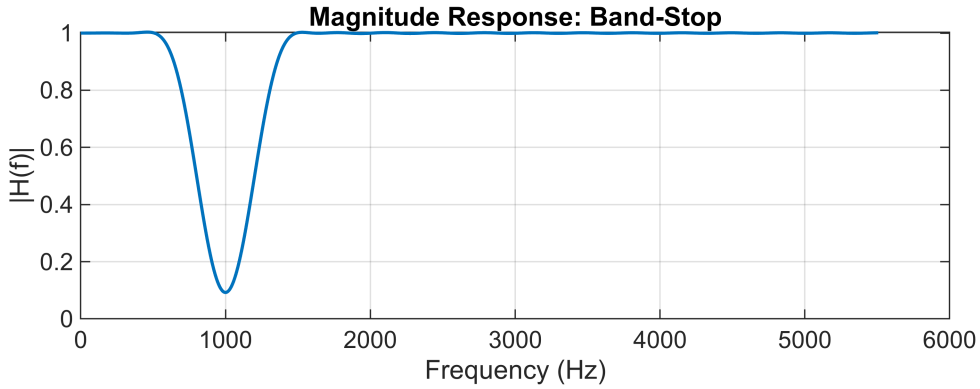
```
n=64;
for i=1:length(afile)
    sprintf("%s",afile(i))
    [x, fs] = audioread(afile(i));
    bp_edges = [400 1200] / (fs/2);   % band-pass: 400–1200 Hz
    bs_edges = [800 1200] / (fs/2);   % band-stop: 800–1200 Hz
    % Design FIR filters (Hamming window by default)
    b_bp = fir1(n, bp_edges, 'bandpass');
    b_bs = fir1(n, bs_edges, 'stop');
    %--- Verify with zplane ---
    figure;
```

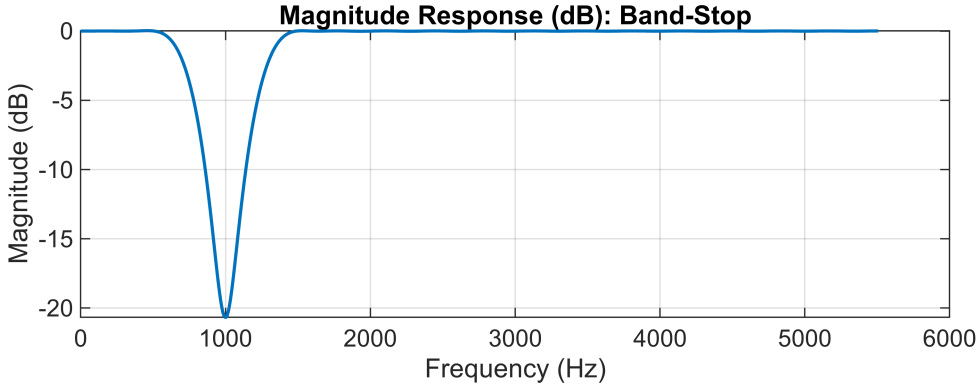
```
subplot(2,2,1);
zplane(b_bp, 1);
title('Zeros & PolesBand-Pass 400–1200 Hz)');
subplot(2,2,2);
zplane(b_bs, 1);
title('Zeros & Poles Band-Stop 800–1200 Hz)');
%--- Verify with freqz ---
nfft = 1024;
subplot(2,2,3);
[Hbp, F] = freqz(b_bp, 1, nfft, fs);
plot(F, abs(Hbp), 'LineWidth',1.2);
xlabel('Frequency (Hz)');
ylabel('|H(f)|');
title('Magnitude Response: Band-Pass');
grid on;
subplot(2,2,4);
plot(F, 20*log10(abs(Hbp)+1e-4), 'LineWidth',1.2);
xlabel('Frequency (Hz)');
ylabel('Magnitude (dB)');
title('Magnitude Response (dB): Band-Pass');
grid on;
%--- Plot the band-stop magnitude too, for completeness ---
figure;
subplot(2,1,1);
[Hbs, F] = freqz(b_bs, 1, nfft, fs);
plot(F, abs(Hbs), 'LineWidth',1.2);
xlabel('Frequency (Hz)');
ylabel('|H(f)|');
title('Magnitude Response: Band-Stop');
grid on;

subplot(2,1,2);
plot(F, 20*log10(abs(Hbs)+1e-4), 'LineWidth',1.2);
xlabel('Frequency (Hz)');
ylabel('Magnitude (dB)');
title('Magnitude Response (dB): Band-Stop');
grid on;
%--- Filter the signal and listen ---
y_bp = filter(b_bp, 1, x);
y_bs = filter(b_bs, 1, x);
fprintf('Playing band-pass output...\n');
player = audioplayer(y_bp, fs);
play(player);
pause(pt);
stop(player);
fprintf('Playing band-stop output...\n');
player = audioplayer(y_bs, fs);
play(player);
pause(pt);
stop(player);
```

```
    t = (0:length(y_bp)-1) / fs;
    figure;
    subplot(3,1,1);
    plot(t, y_bp);
    title(sprintf("audio signal by time  band-stop%s p:%.2f",afile(i)))
    t = (0:length(y_bs)-1) / fs;
    subplot(3,1,2);
    plot(t, y_bs);
    title(sprintf("audio signal by time  band-pass%s p:%.2f",afile(i)))

    subplot(3,1,3);
    t = (0:length(x)-1) / fs;
    plot(t,x);
     title(sprintf("audio signal by time no filter %s p:%.2f",afile(i)))
    figure;
    subplot(2,1,1);
    spectrogram(y_bp, sp_win, sp_ovr, sp_fftN, fs, 'yaxis');
    title(sprintf("spectrogram band pass %s p:%.2f",afile(i)));
    subplot(2,1,2);
    spectrogram(y_bs, sp_win, sp_ovr, sp_fftN, fs, 'yaxis');
    title(sprintf("spectrogram band stop %s p:%.2f",afile(i)));
    clear y_bp y_bs
end
```

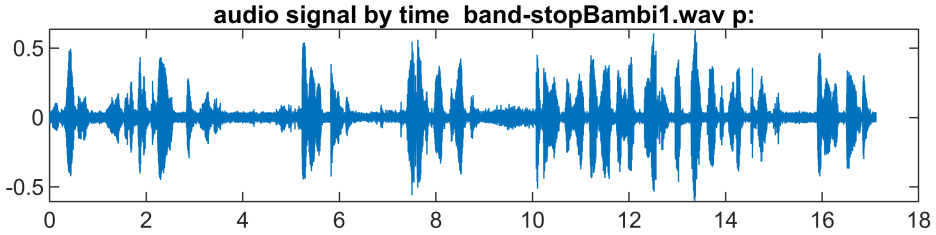
```
ans =
"Bambi1.wav"
```

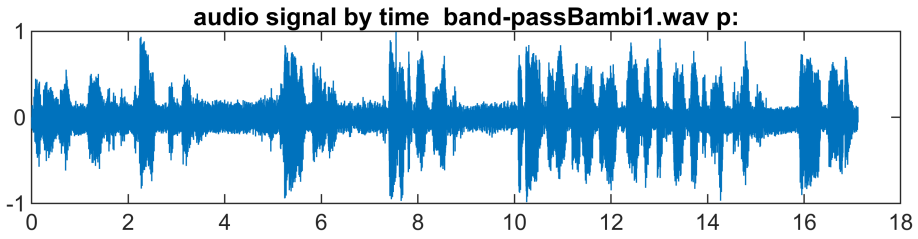


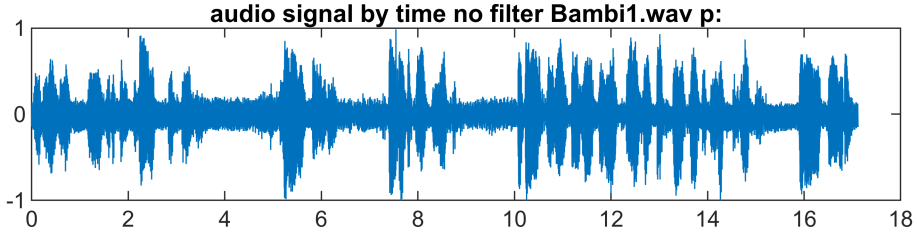


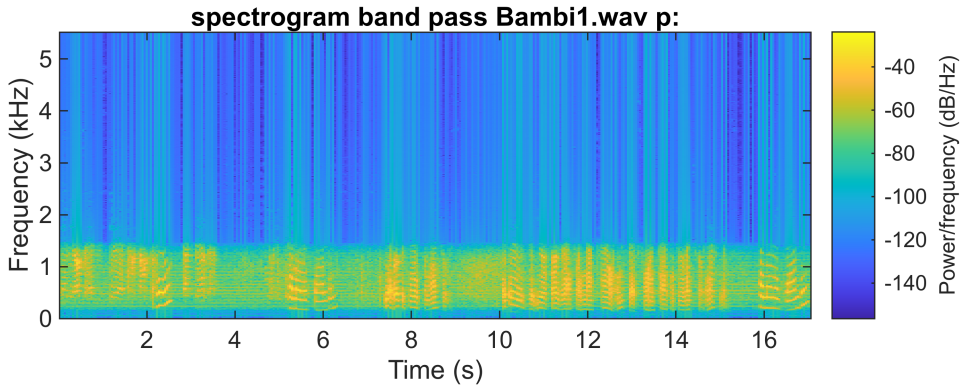


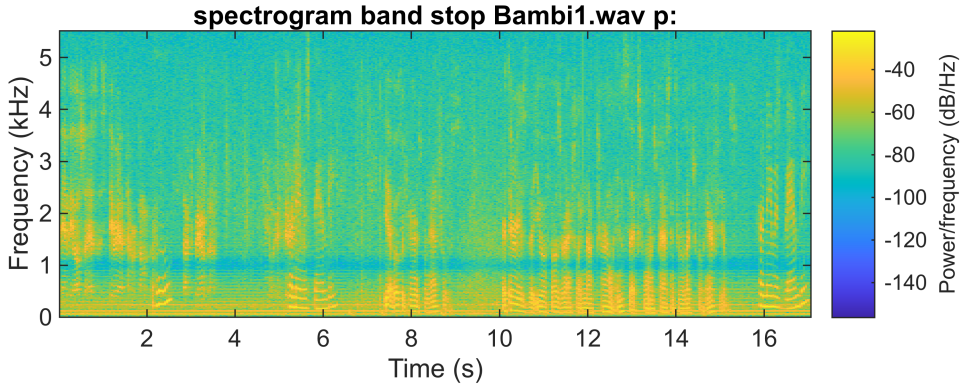

```
Playing band-pass output...
Playing band-stop output...
```



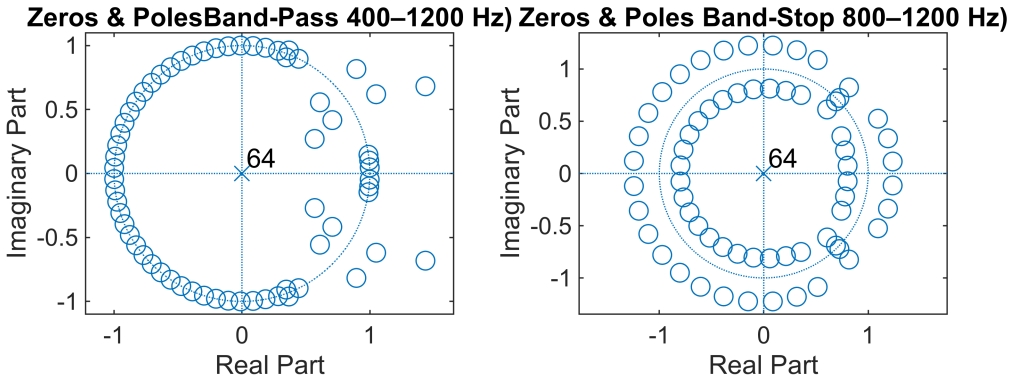


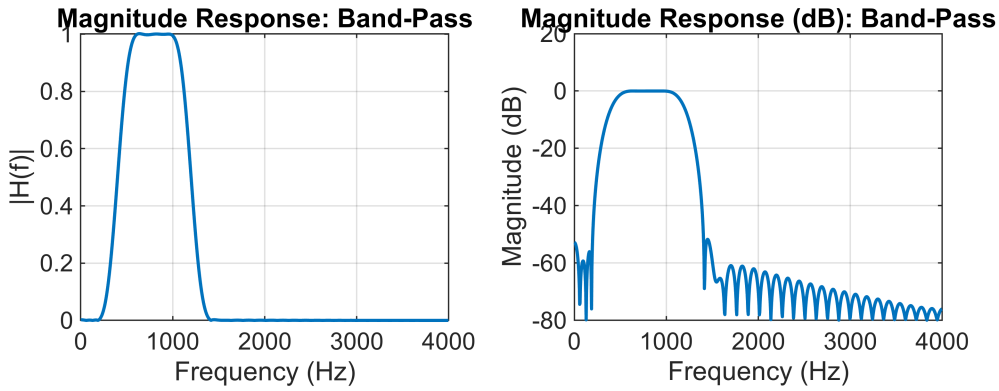


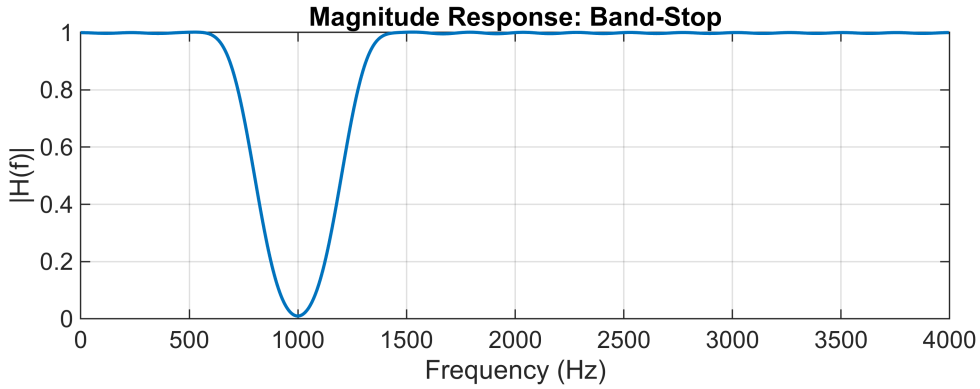


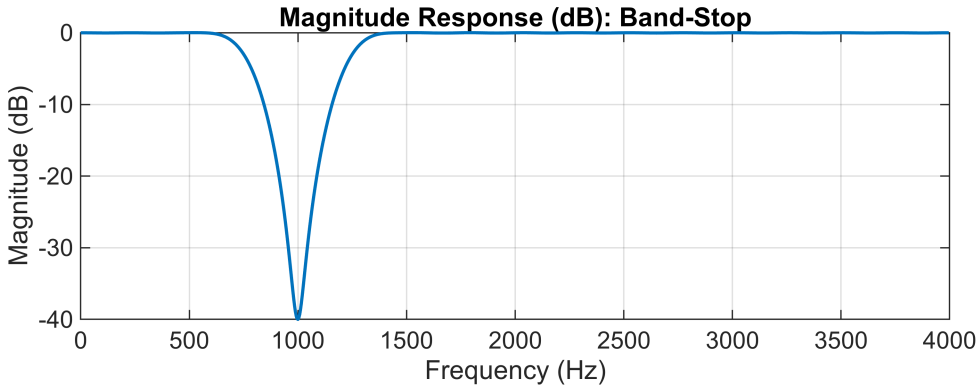



```
ans =
"freqSweep.wav"
```

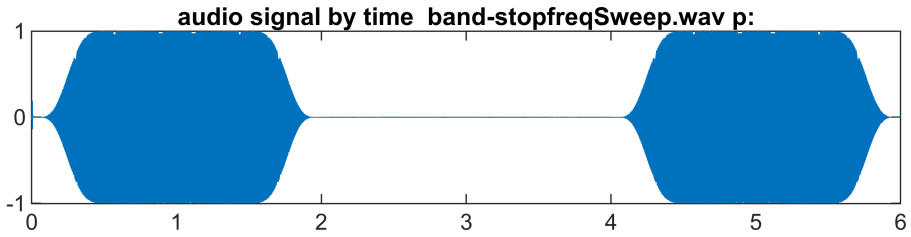


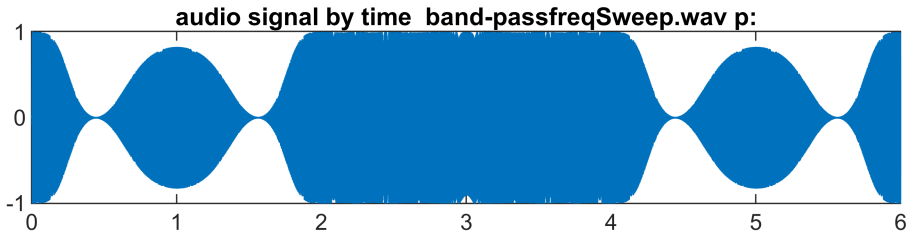


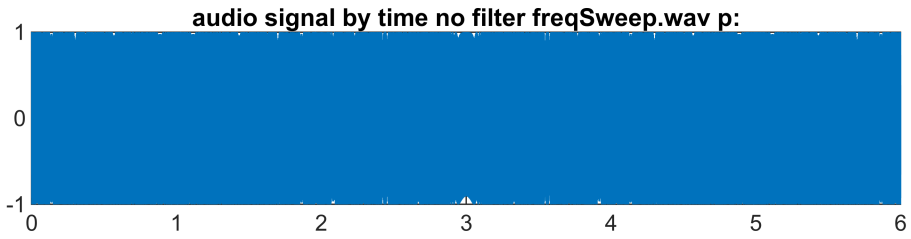


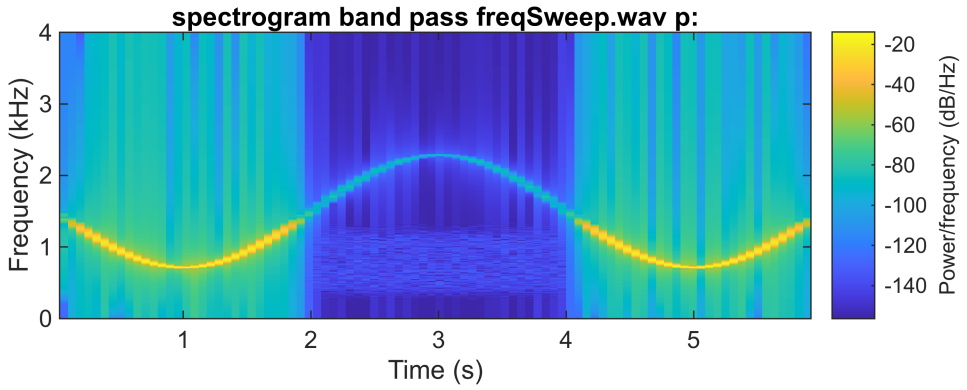


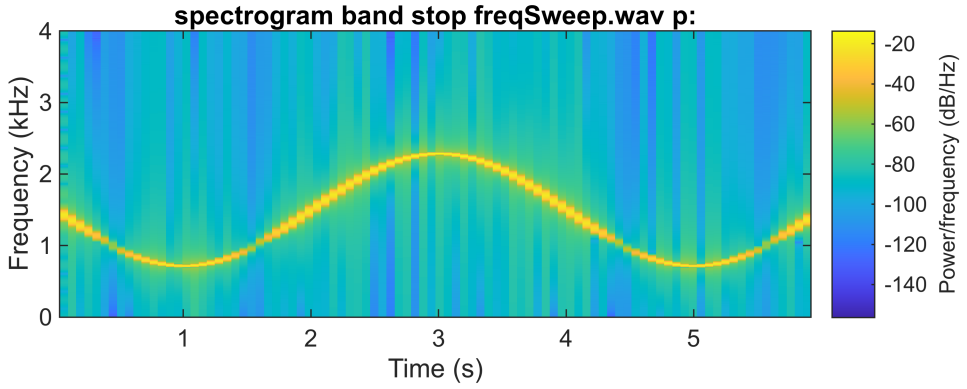

```
Playing band-pass output...
Playing band-stop output...
```



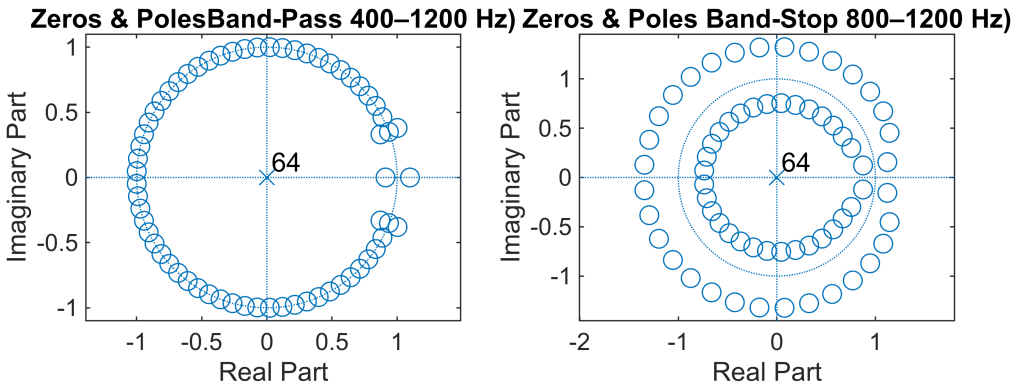


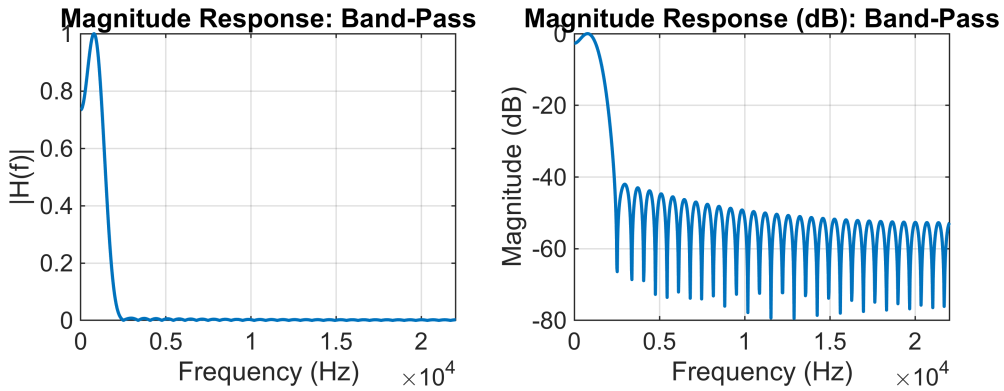


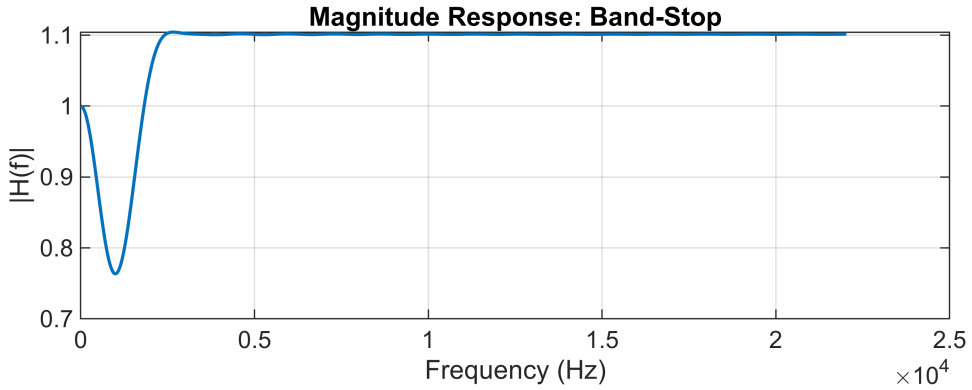

spectrogram band pass freqSweep.wav p:
Frequency (kHz)
Time (s)
Power/frequency (dB/Hz)

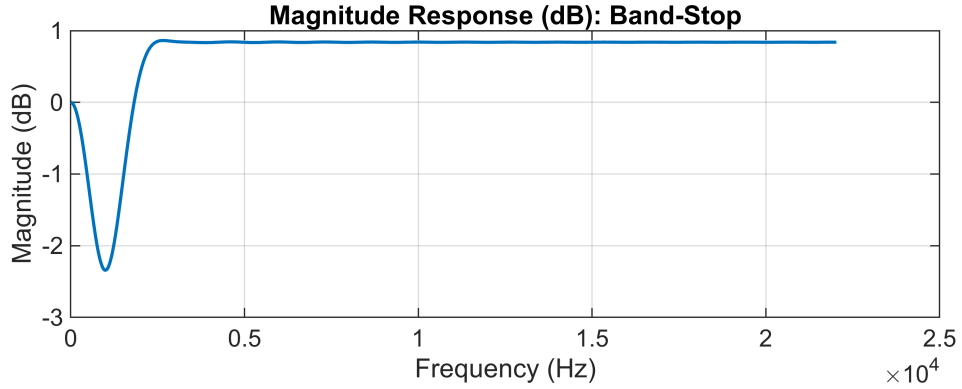

spectrogram band stop freqSweep.wav p:
Frequency (kHz)
Time (s)
Power/frequency (dB/Hz)

```
ans =
"test.wav"
```

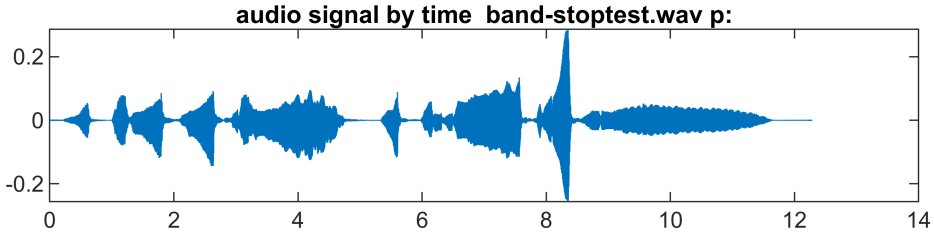

Zeros & PolesBand-Pass 400–1200 Hz)
Zeros & Poles Band-Stop 800–1200 Hz)
Imaginary Part
Real Part
64

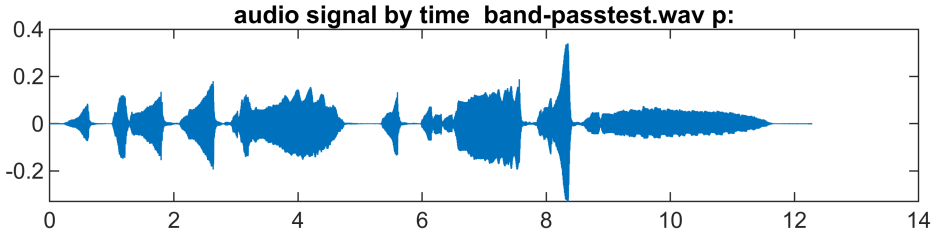

Magnitude Response: Band-Pass
Magnitude Response (dB): Band-Pass
|H(f)|
Magnitude (dB)
Frequency (Hz)

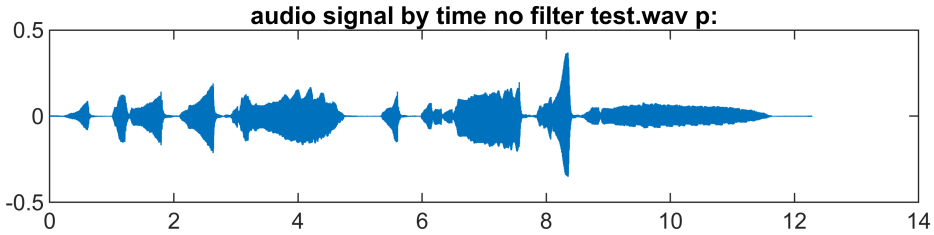


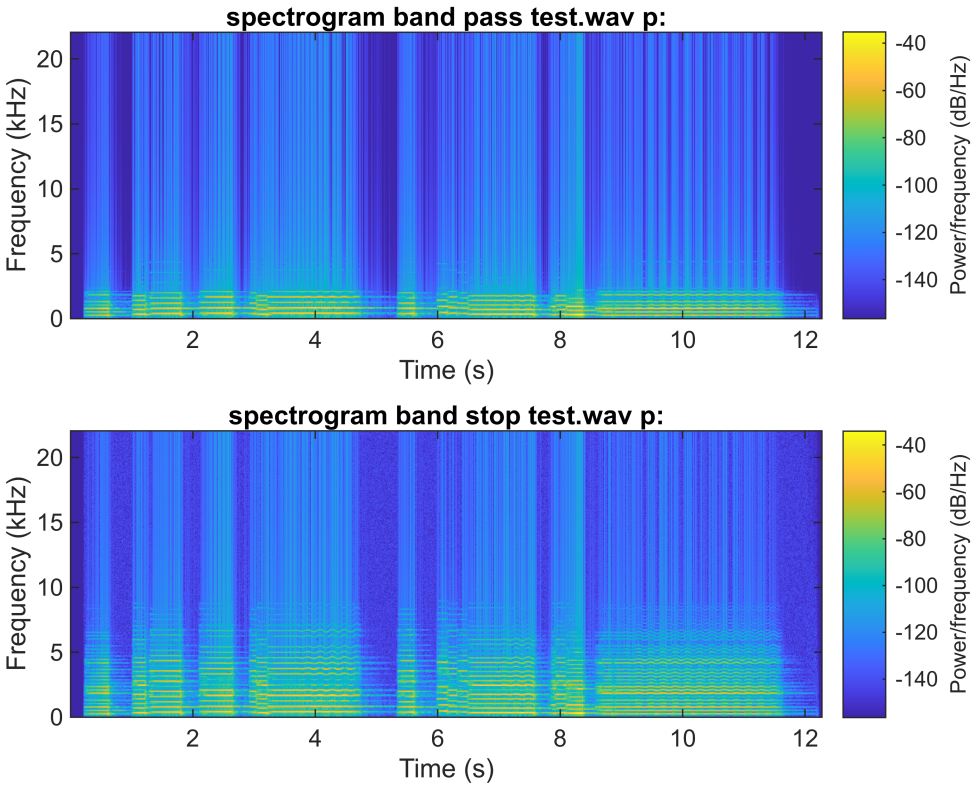



```
Playing band-pass output...
Playing band-stop output...
```









For the freqsweep signal we can see that the signal gets highlty attentuated at around 1.5-1.75 and 0.5 to 0.75 for the signal frequencies for it for bandpass.

For the freqsweep for banstop we can see that banstop frequencies at around 2-4 seconsd for it to be highly attenuated.

For the freqsweep signal we can see that 1.5-2.3khz is the time where it gets atttenuated. Yeahs its consistent for the filter specification.

We can see the diiffrences for freqsweep.

For bambi signal we can see that for bandpass 1.5-5.5 khz there is no signal passing through to it at all.

for bansstop we can see that 1khz is being highly attenuated.

Its pretty hard to see since there is a very large range of audio signal because it goes from 0-20khz which is much larger range then any of the other filters.

Step 3: Bandpass and bandstop FIR filters designed with optimizing PM method

Repeat Step2 using firpm to create FIR Parks-McClellan filters. Type help firpm in the command

window. For both the bandpass and band stop filters, the function input f and a vectors will be of

length 6 to specify three segments in frequency. The vectors below allow a 100Hz separation

between frequency segments. IMPORTANT NOTE: The “A” input for firpm input specification i.e.”apm” below, is NOT the vector A of filter coefficients for the filter input specification.

For the bandpass filter, the frequency segments should be fpm = [0, 350, 450, 1150, 1250, Fs/2] and the amplitude segments should be apm = [0 0 1 1 0 0].

For the bandstop filter, the frequency segments should be fpm = [0, 750, 850, 1150, 1250, Fs/2] and the amplitude segments should be apm= [1 1 0 0 1 1].

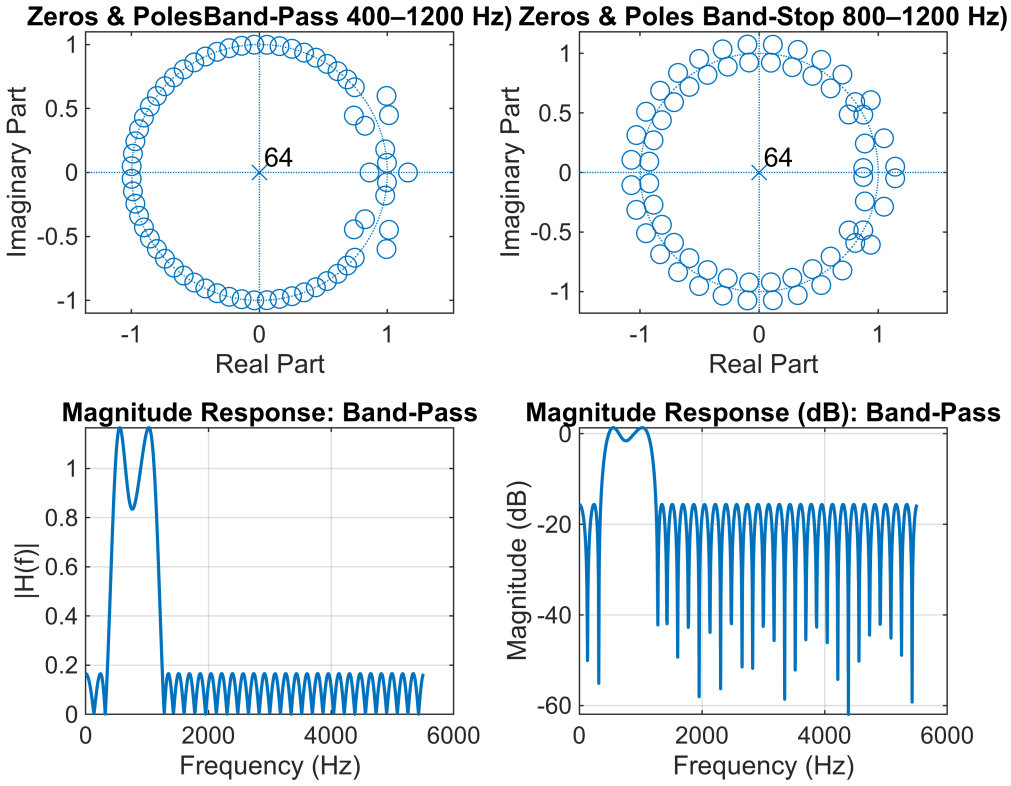
```
pt=1;
sp_win = 1024; sp_ovr = 512; sp_fftN = 1024;
afile=["Bambi1.wav",'freqSweep.wav','test.wav']
```

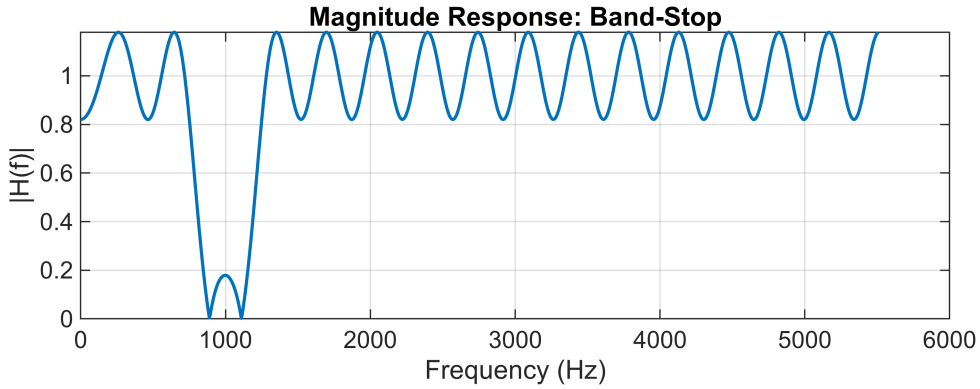
```
afile = 1×3 string
"Bambi1.wav" "freqSweep…  "test.wav"
```

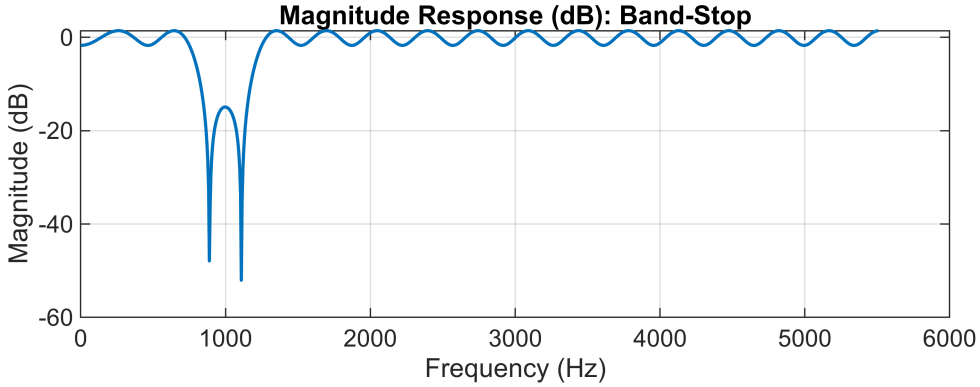
```
n=64;
for i=1:length(afile)
    sprintf("%s",afile(i))
    [x, fs] = audioread(afile(i));
    % Design FIR filters (Hamming window by default)
    f_bp_hz = [0, 350, 450, 1150, 1250, fs/2]
    a_bp = [0 0 1 1 0 0]
    f_bs_hz=[0, 750, 850, 1150, 1250, fs/2]
    a_bs=[1 1 0 0 1 1];
    f_bp = f_bp_hz/(fs/2);
    f_bs = f_bs_hz/(fs/2);
    b_bp = firpm(n, f_bp, a_bp);
    b_bs = firpm(n, f_bs, a_bs);
    %--- Verify with zplane ---
    figure;
    subplot(2,2,1);
    zplane(b_bp, 1);
    title('Zeros & PolesBand-Pass 400-1200 Hz)');
    subplot(2,2,2);
    zplane(b_bs, 1);
    title('Zeros & Poles Band-Stop 800-1200 Hz)');
    %--- Verify with freqz ---
    nfft = 1024;
    subplot(2,2,3);
    [Hbp, F] = freqz(b_bp, 1, nfft, fs);
    plot(F, abs(Hbp), 'LineWidth',1.2);
    xlabel('Frequency (Hz)');
    ylabel('|H(f)|');
    title('Magnitude Response: Band-Pass');
    grid on;
    subplot(2,2,4);
    plot(F, 20*log10(abs(Hbp)+1e-4), 'LineWidth',1.2);
    xlabel('Frequency (Hz)');
    ylabel('Magnitude (dB)');
```

```
    title('Magnitude Response (dB): Band-Pass');
    grid on;
    %--- Plot the band-stop magnitude too, for completeness ---
    figure;
    subplot(2,1,1);
    [Hbs, F] = freqz(b_bs, 1, nfft, fs);
    plot(F, abs(Hbs), 'LineWidth',1.2);
    xlabel('Frequency (Hz)');
    ylabel('|H(f)|');
    title('Magnitude Response: Band-Stop');
    grid on;

    subplot(2,1,2);
    plot(F, 20*log10(abs(Hbs)+1e-4), 'LineWidth',1.2);
    xlabel('Frequency (Hz)');
    ylabel('Magnitude (dB)');
    title('Magnitude Response (dB): Band-Stop');
    grid on;
    %--- Filter the signal and listen ---
    y_bp = filter(b_bp, 1, x);
    y_bs = filter(b_bs, 1, x);
    fprintf('Playing band-pass output...\n');
    player = audioplayer(y_bp, fs);
    play(player);
    pause(pt);
    stop(player);
    fprintf('Playing band-stop output...\n');
    player = audioplayer(y_bs, fs);
    play(player);
    pause(pt);
    stop(player);
    t = (0:length(y_bp)-1) / fs;
    figure;
    subplot(2,1,1);
    plot(t, y_bp);
    title(sprintf("audio signal by time  band-stop%s p:%.2f",afile(i)))
    t = (0:length(y_bs)-1) / fs;
    subplot(2,1,2);
    plot(t, y_bs);
    title(sprintf("audio signal by time  band-pass%s p:%.2f",afile(i)))
    figure;
    subplot(2,1,1);
    spectrogram(y_bp, sp_win, sp_ovr, sp_fftN, fs, 'yaxis');
    title(sprintf("spectrogram band pass %s p:%.2f",afile(i)));
    subplot(2,1,2);
    spectrogram(y_bs, sp_win, sp_ovr, sp_fftN, fs, 'yaxis');
    title(sprintf("spectrogram band stop %s p:%.2f",afile(i)));
    clear y_bp y_bs
end
```

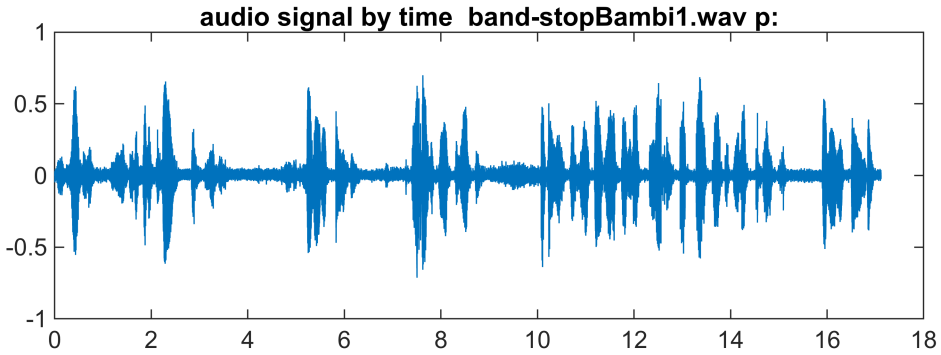
```
ans =
"Bambi1.wav"
f_bp_hz = 1×6
10^3 ×
         0    0.3500    0.4500    1.1500    1.2500    5.5125
a_bp = 1×6
     0     0     1     1     0     0
f_bs_hz = 1×6
10^3 ×
         0    0.7500    0.8500    1.1500    1.2500    5.5125
```

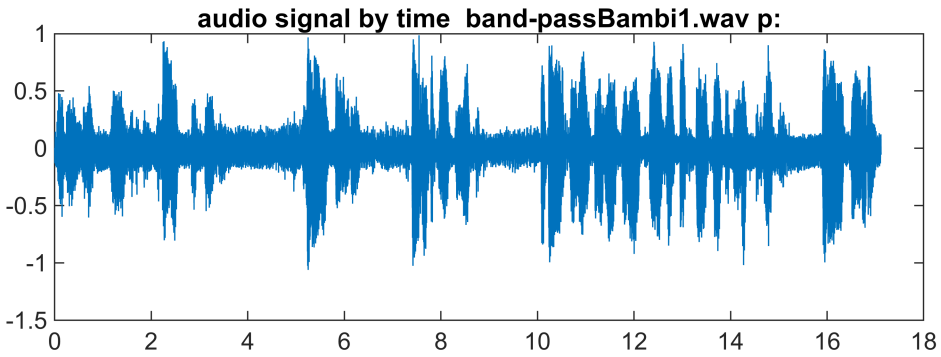


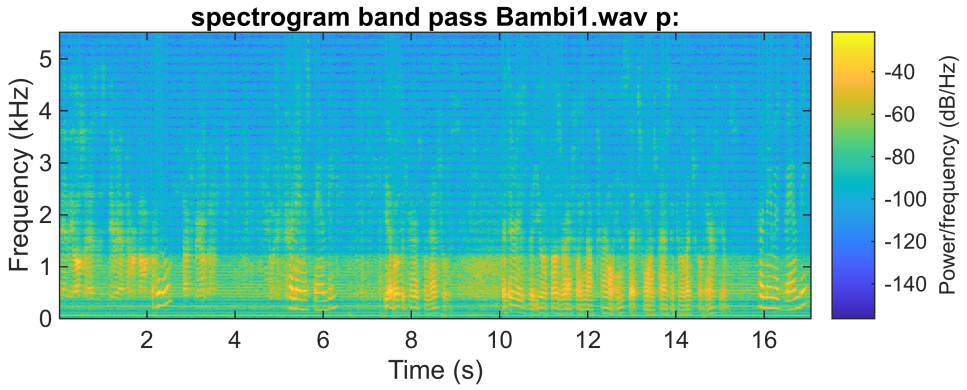


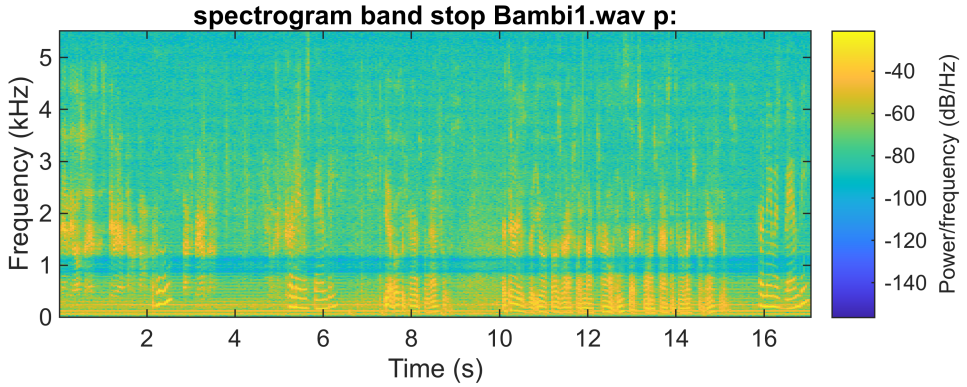



```
Playing band-pass output...
Playing band-stop output...
```

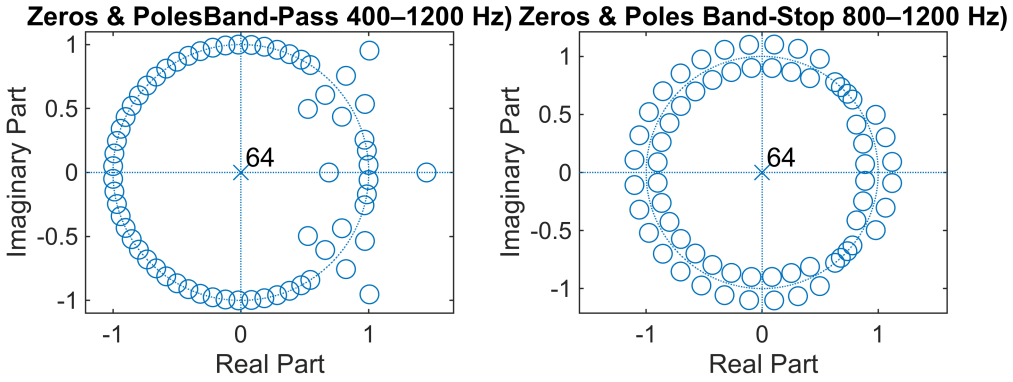


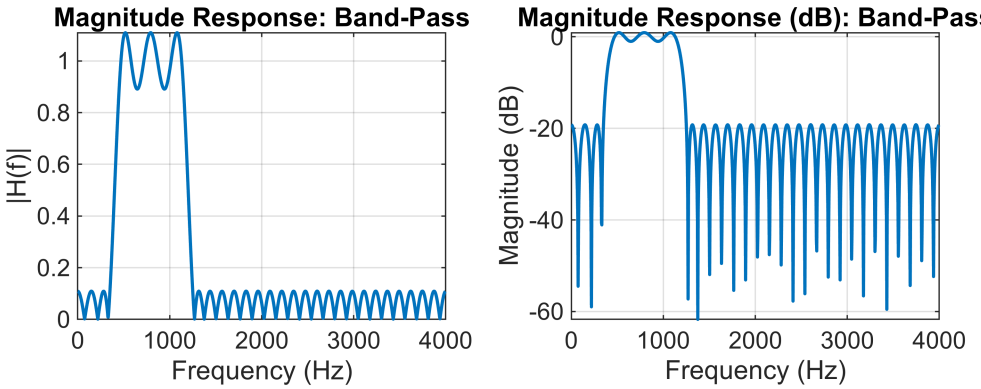


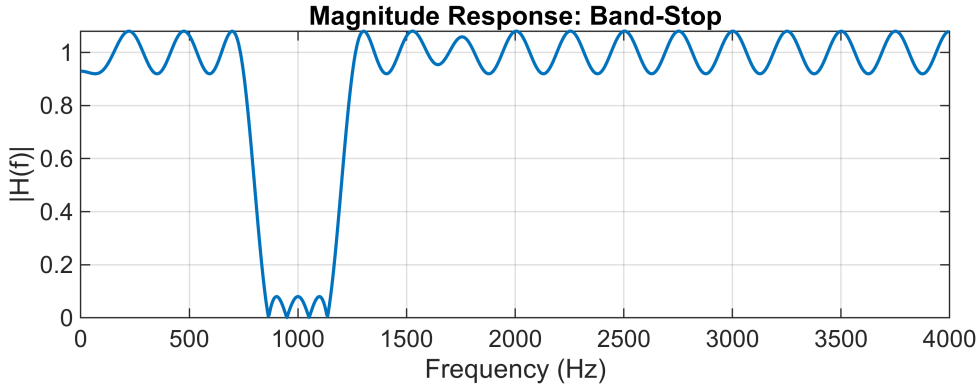


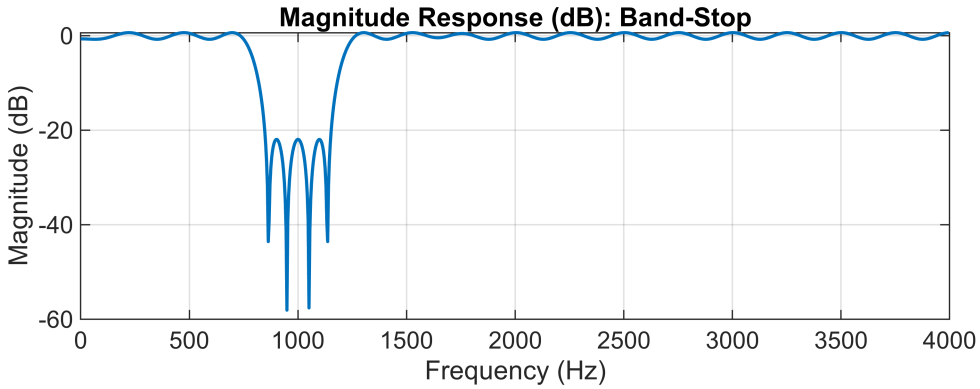



```
ans =
"freqSweep.wav"
f_bp_hz = 1×6
         0         350         450        1150        1250        4000
a_bp = 1×6
     0     0     1     1     0     0
f_bs_hz = 1×6
         0         750         850        1150        1250        4000
```

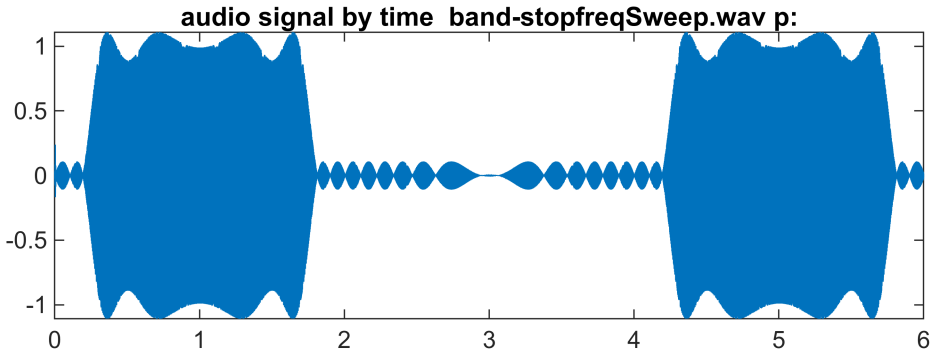


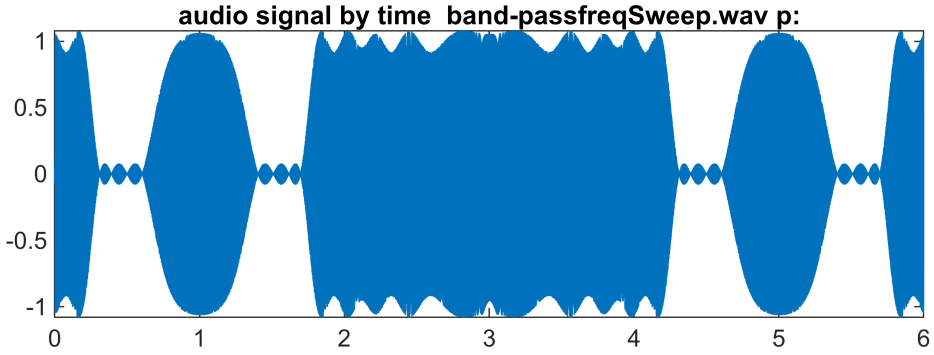


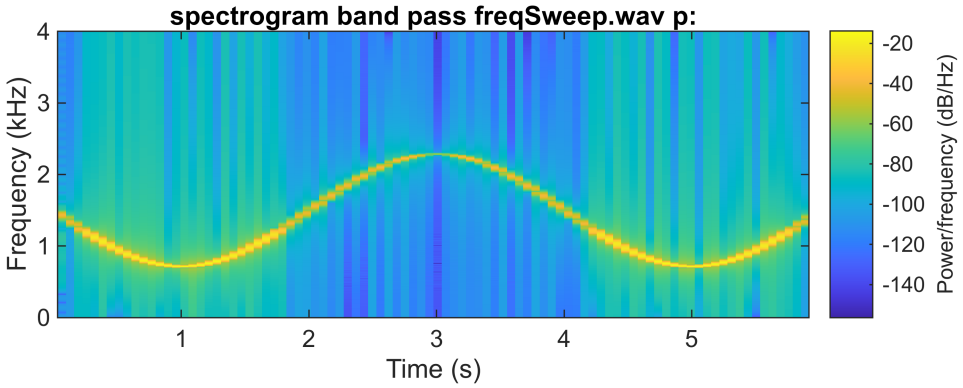


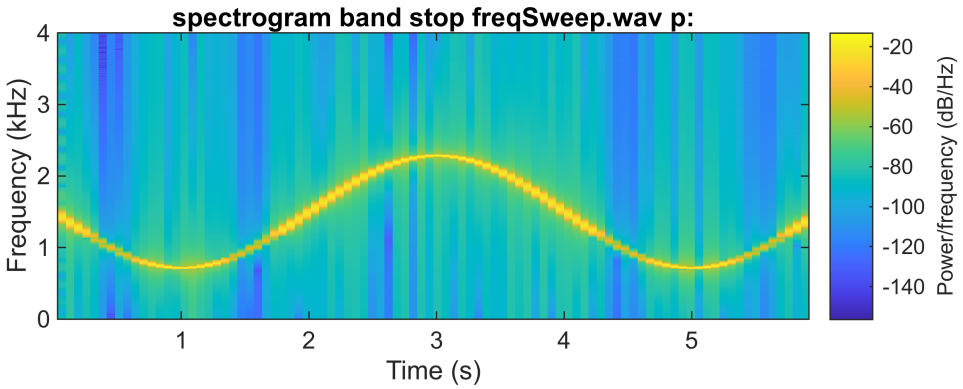



```
Playing band-pass output...
Playing band-stop output...
```



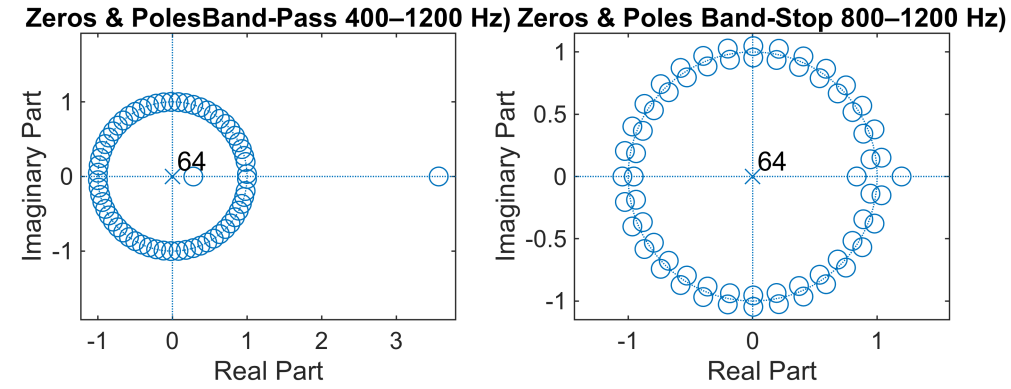


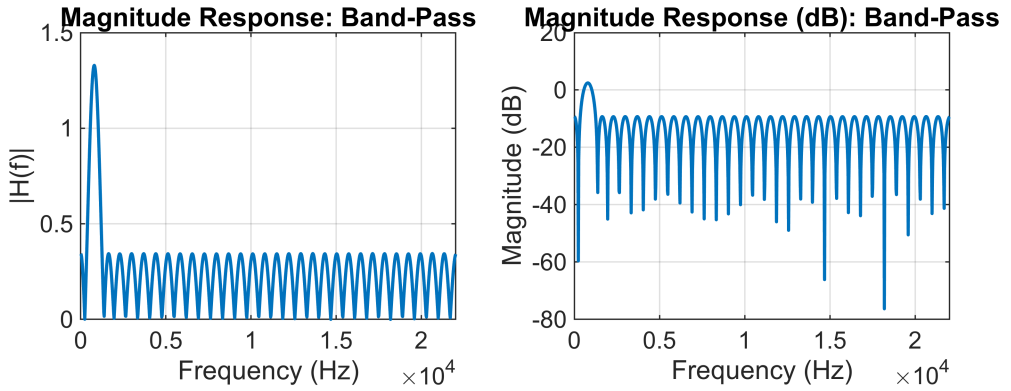


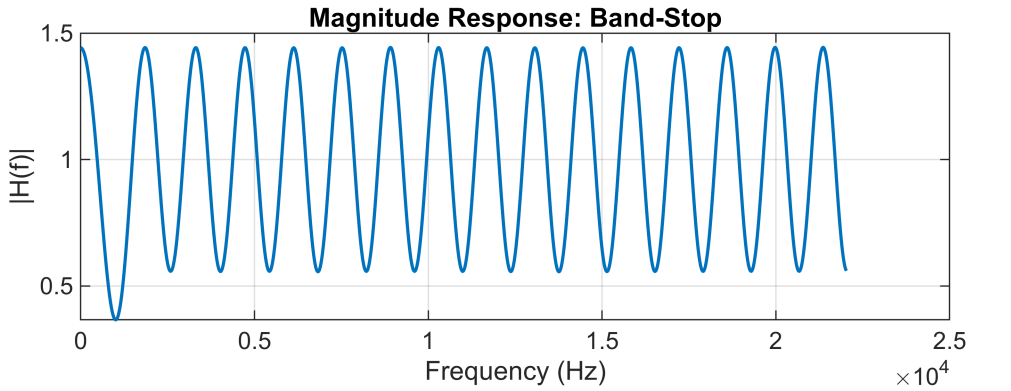


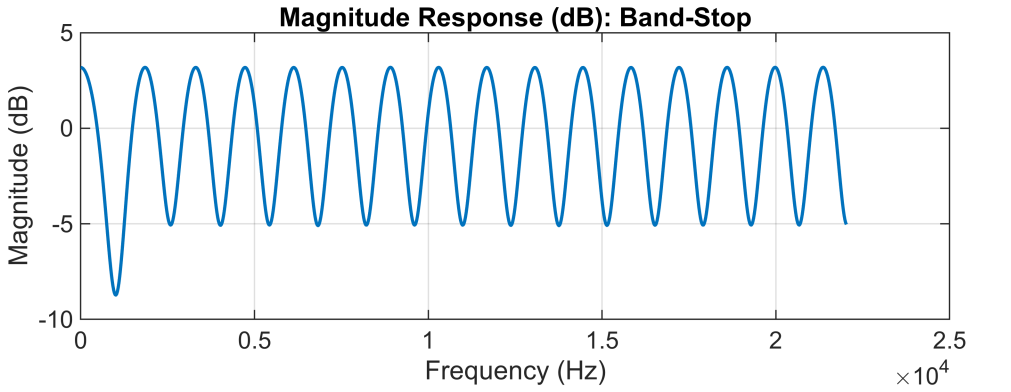

```
ans =
"test.wav"
f_bp_hz = 1×6
         0         350         450        1150        1250       22050
a_bp = 1×6
```

```
     0     0     1     1     0     0
f_bs_hz = 1×6
         0         750         850        1150        1250       22050
```

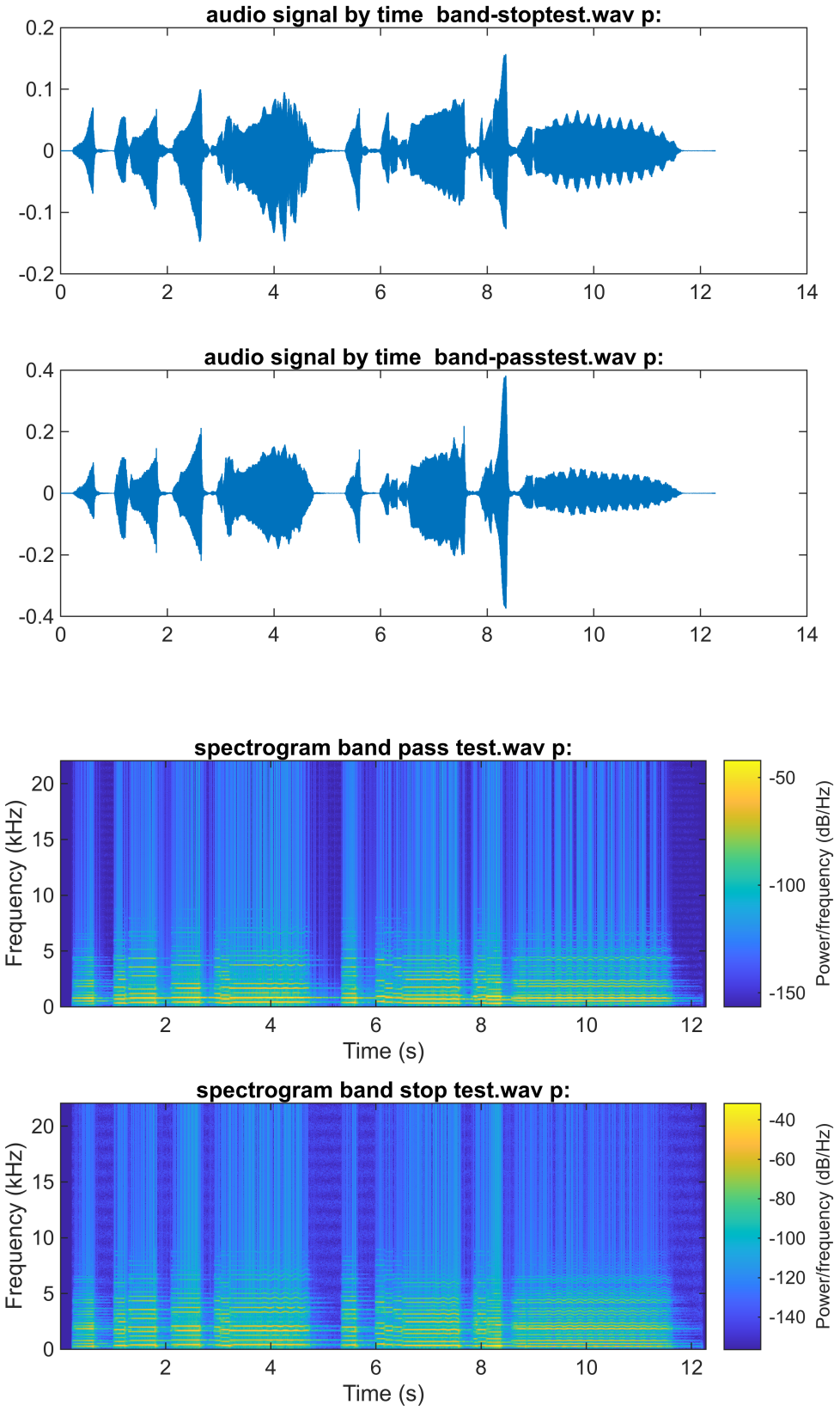









```
Playing band-pass output...
Playing band-stop output...
```



Step 4: Bandpass and bandstop IIR filters designed with Butterworth method

Repeat Step2 using butter to create IIR Butterworth filters. Use the same frequency band edges used in Step 2 and specify the order as 5. This will create 10th order filters. Observe the difference in the zplane plots for the Butterworth filter compared to the fir1 filter. How does the freqz frequency response of the 10th order filters compare to the 64th order FIR filters?

```
pt=1;
sp_win = 1024; sp_ovr = 512; sp_fftN = 1024;
afile=["Bambi1.wav",'freqSweep.wav','test.wav']
```

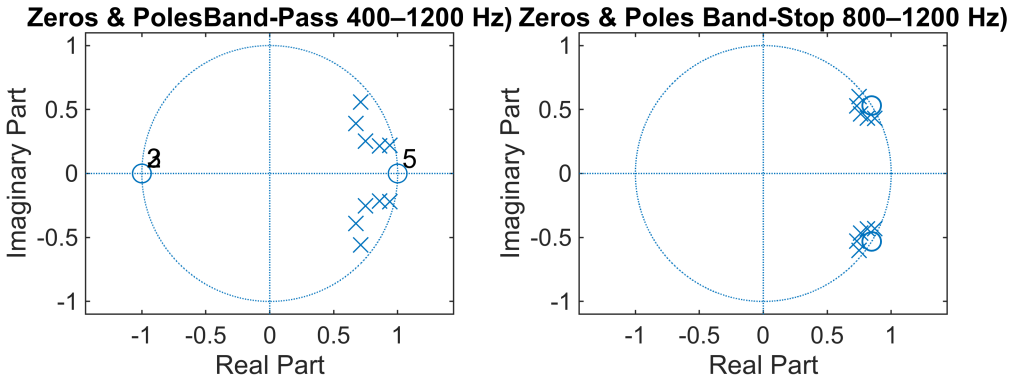
```
afile = 1×3 string
"Bambi1.wav" "freqSweep…  "test.wav"
```

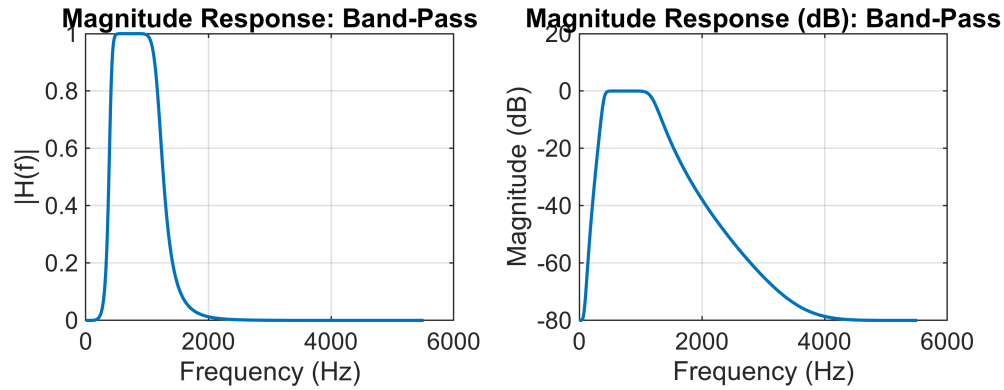
```
n=64;
for i=1:length(afile)
    sprintf("%s",afile(i))
    [x, fs] = audioread(afile(i));
    % Design FIR filters (Hamming window by default)

    f1_bp = 400;           % BP lower edge  (Hz)
    f2_bp = 1200;          % BP upper edge  (Hz)
    f1_bs = 800;           % BS lower edge  (Hz)
    f2_bs = 1200;          % BS upper edge  (Hz)
    N   = 5;               % Butterworth order
    % Normalize to Nyquist:
    Wp_bp = [f1_bp f2_bp]/(fs/2);
    Wp_bs = [f1_bs f2_bs]/(fs/2);

    % Design:
    [b_bp, a_bp] = butter(N, Wp_bp, 'bandpass');
    [b_bs, a_bs] = butter(N, Wp_bs, 'stop');
    %--- Verify with zplane ---
    figure;
    subplot(2,2,1);
    zplane(b_bp, a_bp);
    title('Zeros & PolesBand-Pass 400–1200 Hz)');
    subplot(2,2,2);
    zplane(b_bs, a_bs);
    title('Zeros & Poles Band-Stop 800–1200 Hz)');
    %--- Verify with freqz ---
    nfft = 1024;
    subplot(2,2,3);
    [Hbp, F] = freqz(b_bp, a_bp, nfft, fs);
    plot(F, abs(Hbp), 'LineWidth',1.2);
    xlabel('Frequency (Hz)');
    ylabel('|H(f)|');
    title('Magnitude Response: Band-Pass');
    grid on;
    subplot(2,2,4);
    plot(F, 20*log10(abs(Hbp)+1e-4), 'LineWidth',1.2);
```

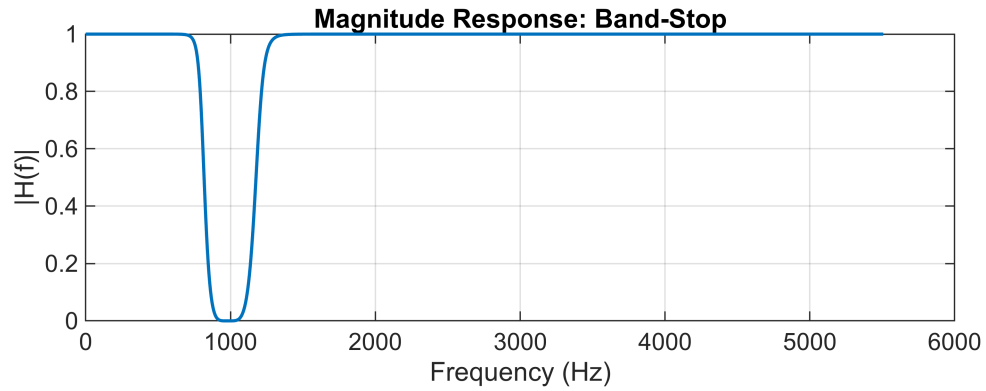
```
xlabel('Frequency (Hz)');
ylabel('Magnitude (dB)');
title('Magnitude Response (dB): Band-Pass');
grid on;
%--- Plot the band-stop magnitude too, for completeness ---
figure;
subplot(2,1,1);
[Hbs, F] = freqz(b_bs, a_bs, nfft, fs);
plot(F, abs(Hbs), 'LineWidth',1.2);
xlabel('Frequency (Hz)');
ylabel('|H(f)|');
title('Magnitude Response: Band-Stop');
grid on;

subplot(2,1,2);
plot(F, 20*log10(abs(Hbs)+1e-4), 'LineWidth',1.2);
xlabel('Frequency (Hz)');
ylabel('Magnitude (dB)');
title('Magnitude Response (dB): Band-Stop');
grid on;
%--- Filter the signal and listen ---
y_bp = filter(b_bp, a_bp, x);
y_bs = filter(b_bs, a_bs, x);
fprintf('Playing band-pass output...\n');
player = audioplayer(y_bp, fs);
play(player);
pause(pt);
stop(player);
fprintf('Playing band-stop output...\n');
player = audioplayer(y_bs, fs);
play(player);
pause(pt);
stop(player);
t = (0:length(y_bp)-1) / fs;
figure;
subplot(2,1,1);
plot(t, y_bp);
title(sprintf("audio signal by time  band-stop%s p:%.2f",afile(i)))
t = (0:length(y_bs)-1) / fs;
subplot(2,1,2);
plot(t, y_bs);
title(sprintf("audio signal by time  band-pass%s p:%.2f",afile(i)))
figure;
subplot(2,1,1);
spectrogram(y_bp, sp_win, sp_ovr, sp_fftN, fs, 'yaxis');
title(sprintf("spectrogram band pass %s p:%.2f",afile(i)));
subplot(2,1,2);
spectrogram(y_bs, sp_win, sp_ovr, sp_fftN, fs, 'yaxis');
title(sprintf("spectrogram band stop %s p:%.2f",afile(i)));
clear y_bp y_bs
```

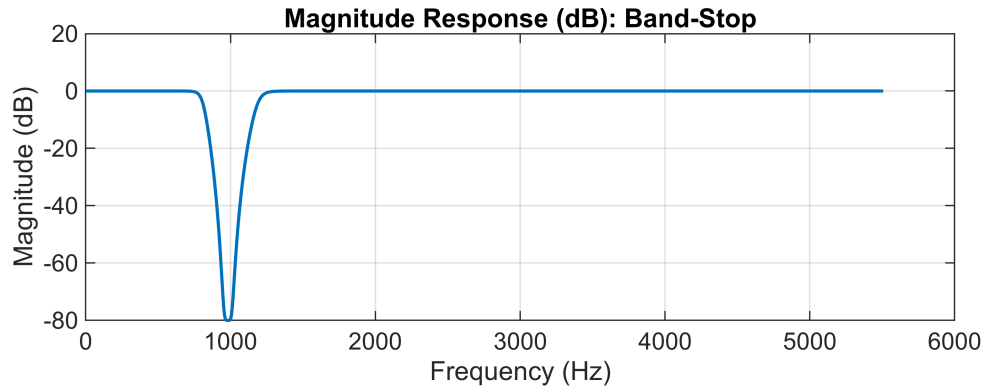
```
end
```

```
ans =
"Bambi1.wav"
```



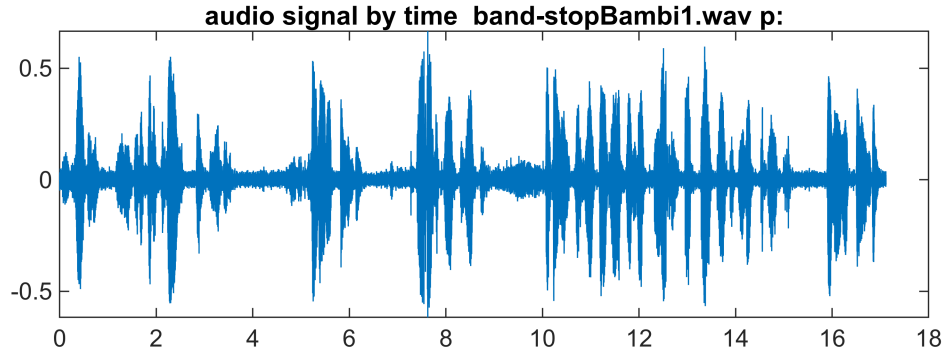


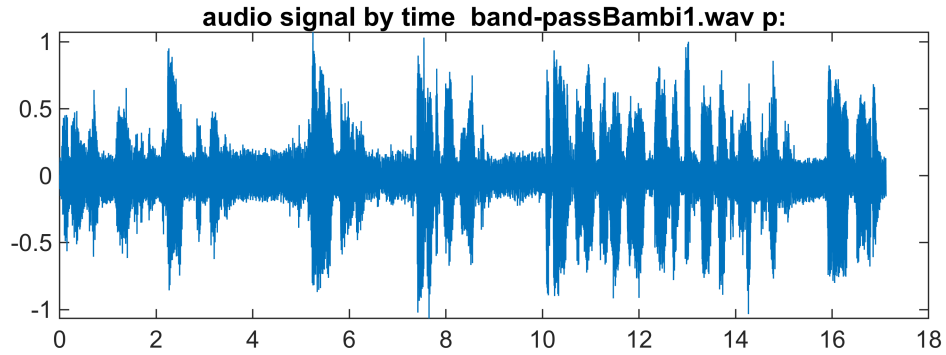


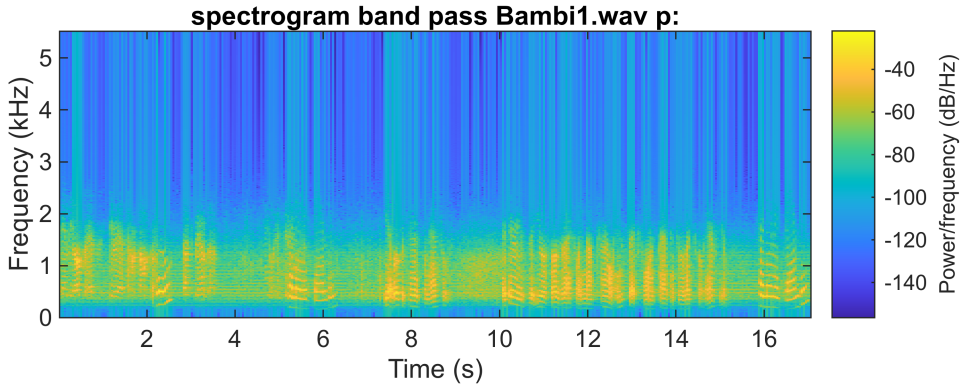


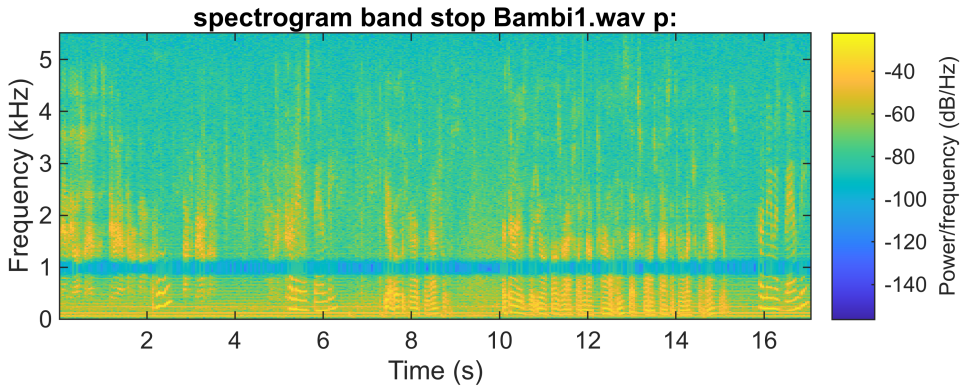

```
Playing band-pass output...
```

```
Playing band-stop output...
```

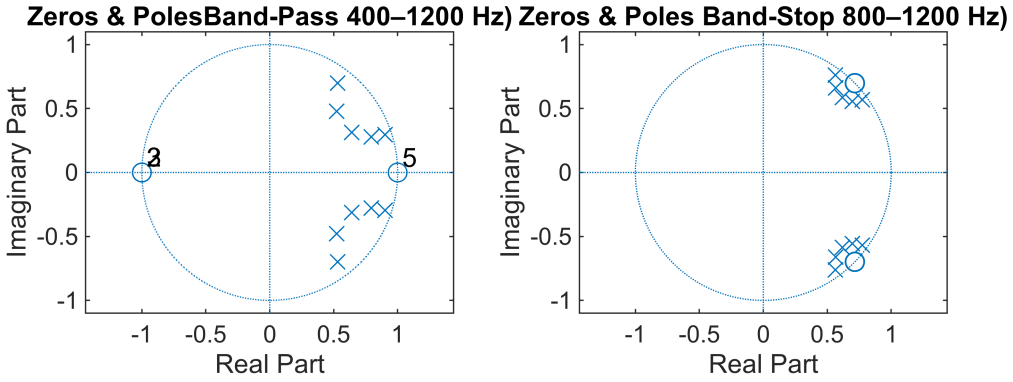


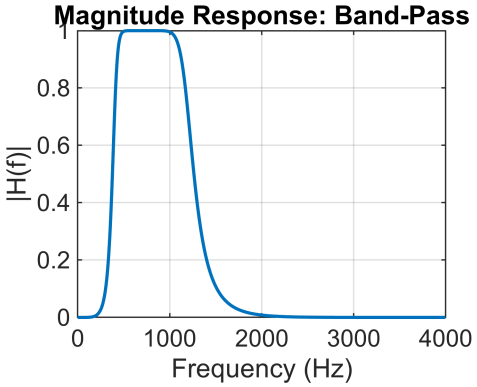


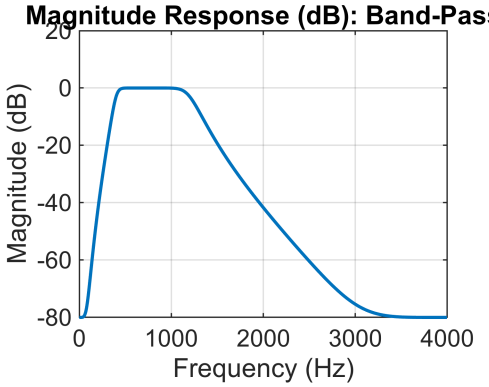


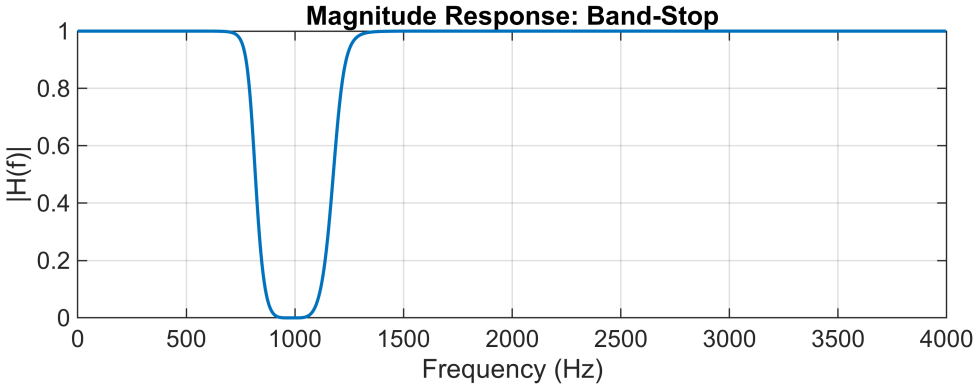


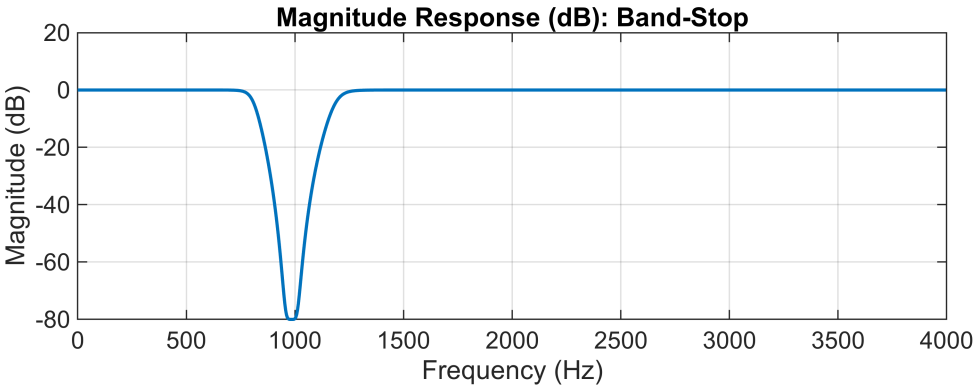

```
ans =
"freqSweep.wav"
```



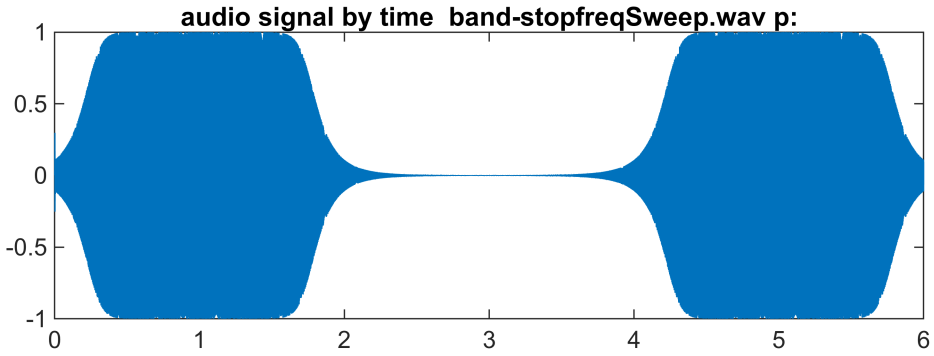


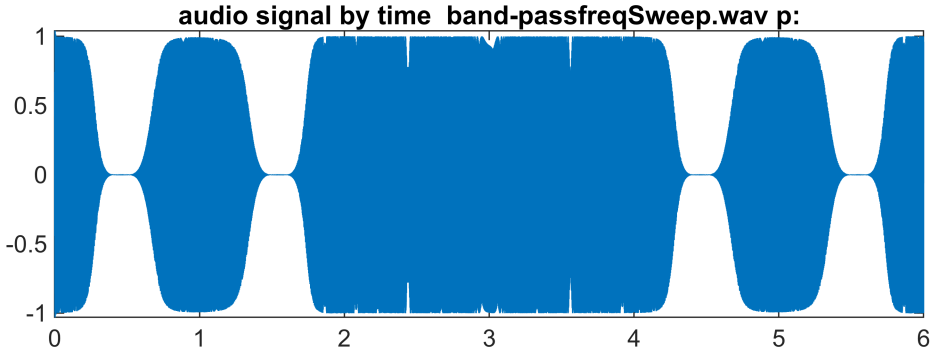


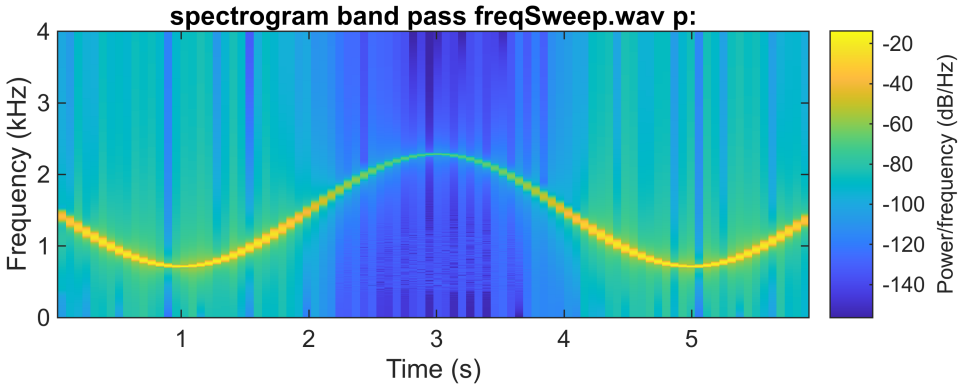


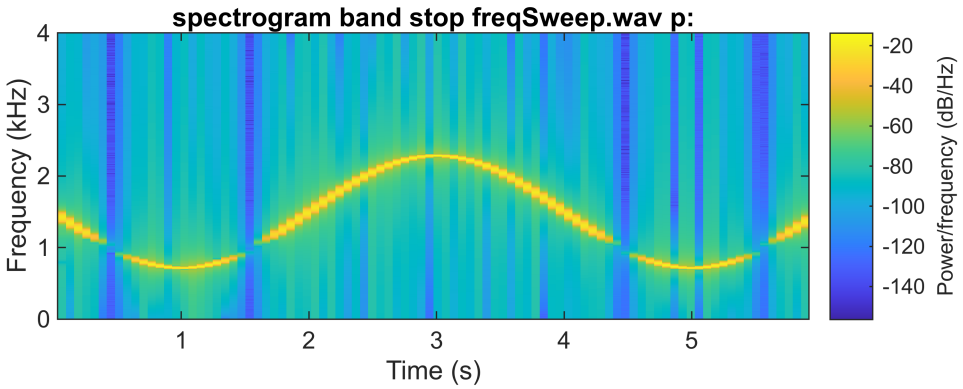



```
Playing band-pass output...
Playing band-stop output...
```

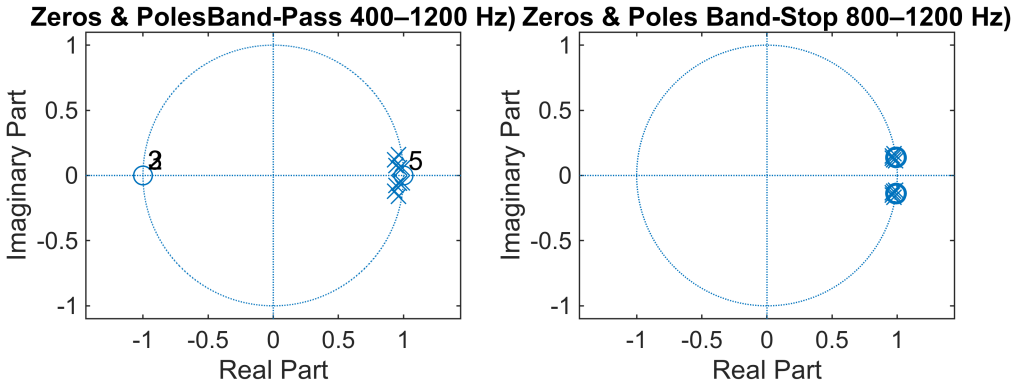

audio signal by time band-stopfreqSweep.wav p:
1
0.5
0
-0.5
-1
0
1
2
3
4
5
6

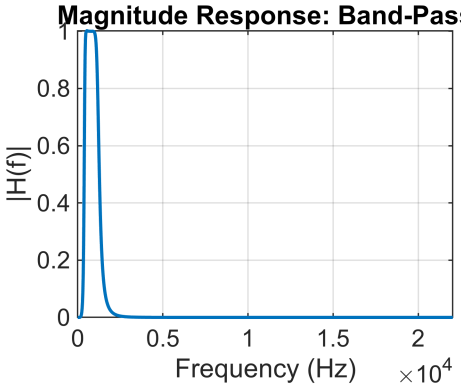

audio signal by time band-passfreqSweep.wav p:
1
0.5
0
-0.5
-1
0
1
2
3
4
5
6

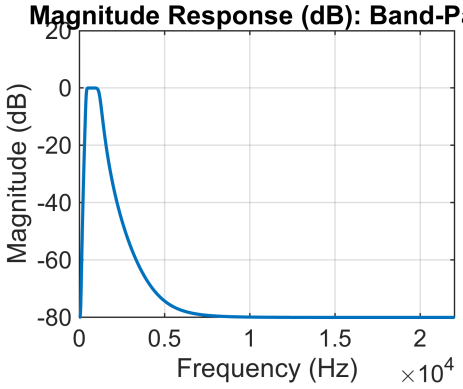

spectrogram band pass freqSweep.wav p:
Frequency (kHz)
Time (s)
Power/frequency (dB/Hz)
-20
-40
-60
-80
-100
-120
-140

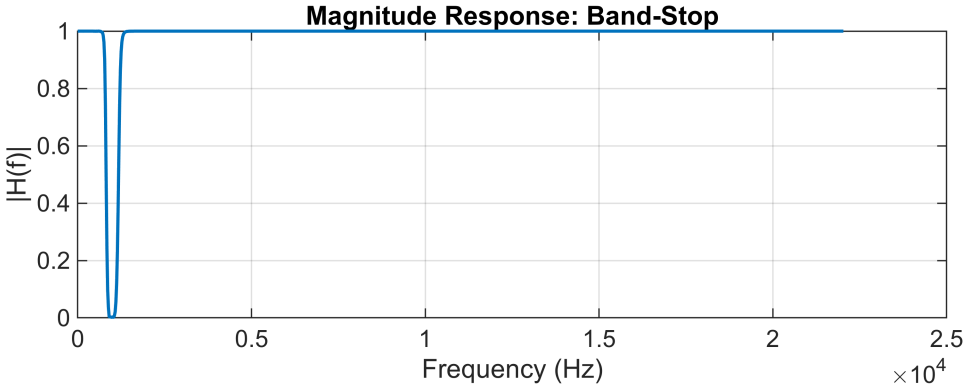

spectrogram band stop freqSweep.wav p:
Frequency (kHz)
Time (s)
Power/frequency (dB/Hz)
-20
-40
-60
-80
-100
-120
-140

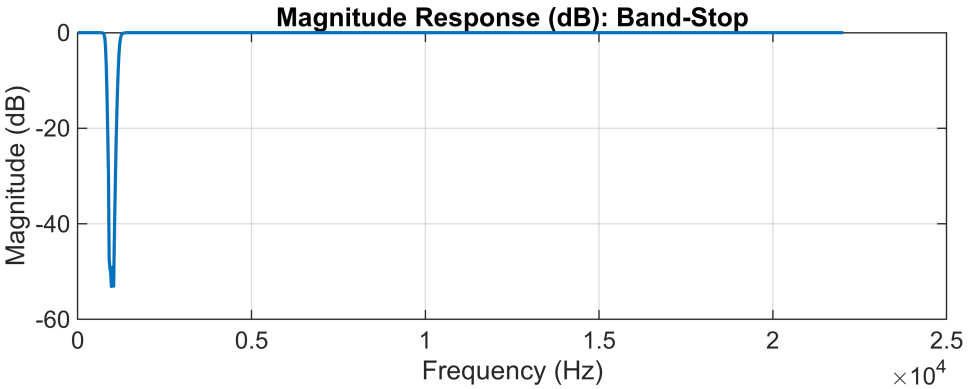
```
ans =
"test.wav"
```



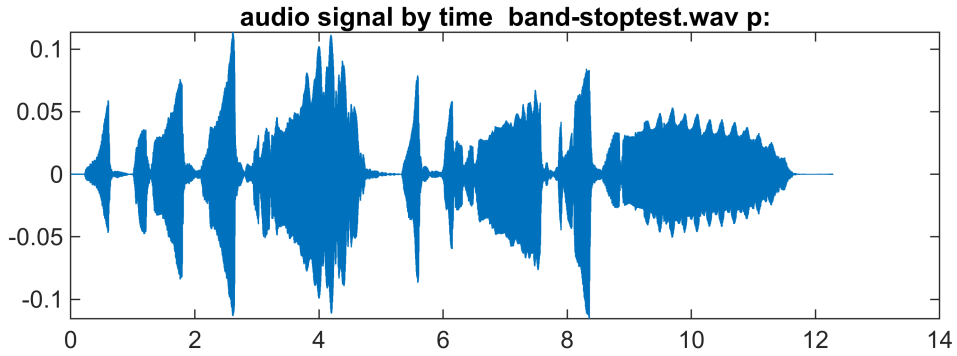


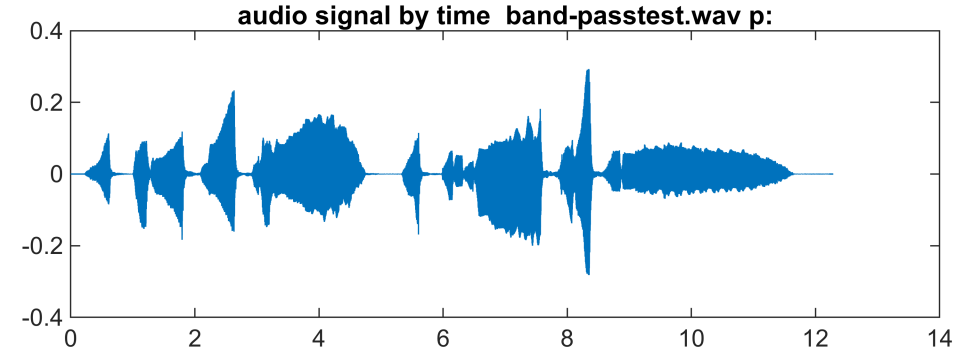


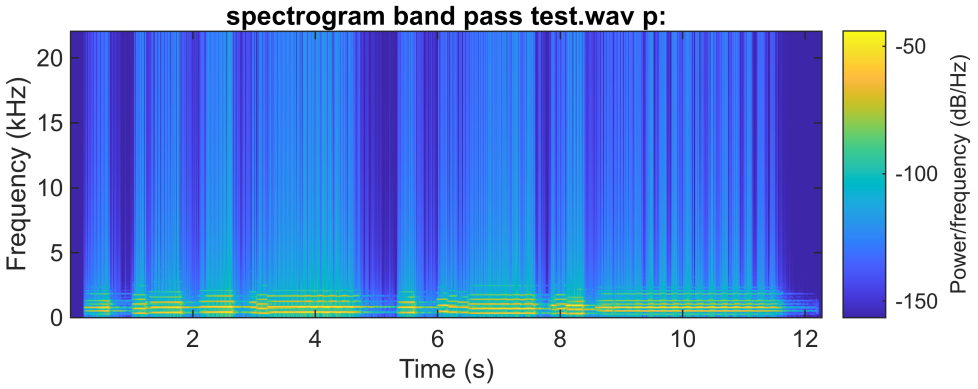


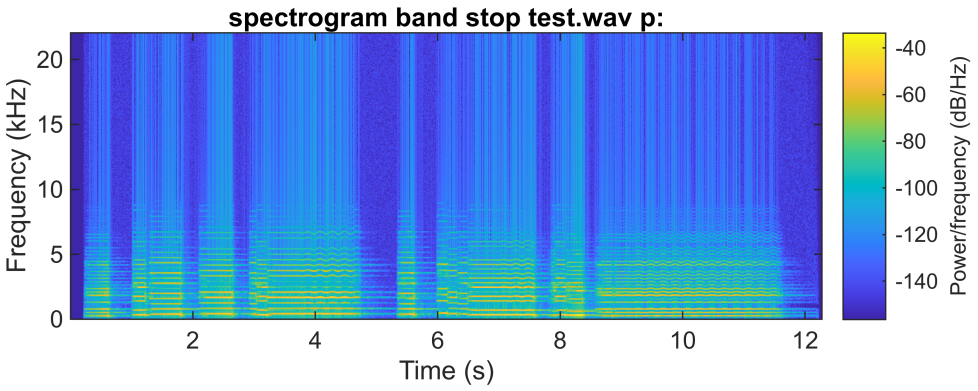



```
Playing band-pass output...
Playing band-stop output...
```


audio signal by time band-stoptest.wav p:
0.1
0.05
0
-0.05
-0.1
0
2
4
6
8
10
12
14


audio signal by time band-passtest.wav p:
0.4
0.2
0
-0.2
-0.4
0
2
4
6
8
10
12
14


spectrogram band pass test.wav p:
Frequency (kHz)
20
15
10
5
0
2
4
6
8
10
12
Time (s)
-50
-100
-150
Power/frequency (dB/Hz)


spectrogram band stop test.wav p:
Frequency (kHz)
20
15
10
5
0
2
4
6
8
10
12
Time (s)
-40
-60
-80
-100
-120
-140
Power/frequency (dB/Hz)

We can see that the db and the actual width of the butterworth is signfactnly better with less computation from it.As butterworth is -80db compared to the other filter where the other ones are around -20db-50db for it. Meaning the butterworth is probably better for these types of signals.