EE6133 Multirate Digital Signal Processing
Experiment # 2

Amizhthni P R K, EE21B015

November 27, 2023

Contents

List of Figures

1 Designing H^0z

The Type 2 Linear Phase LPF magnitude and frequency response are as shown in the following figures. We first design a filter for the given specifications and then change the order of the filter such that it is odd. We then verify if the filter is of type 2 by using the command `firtype(b)`. We change the specifications of the filter to get a reasonably high order which is **odd** in this case.

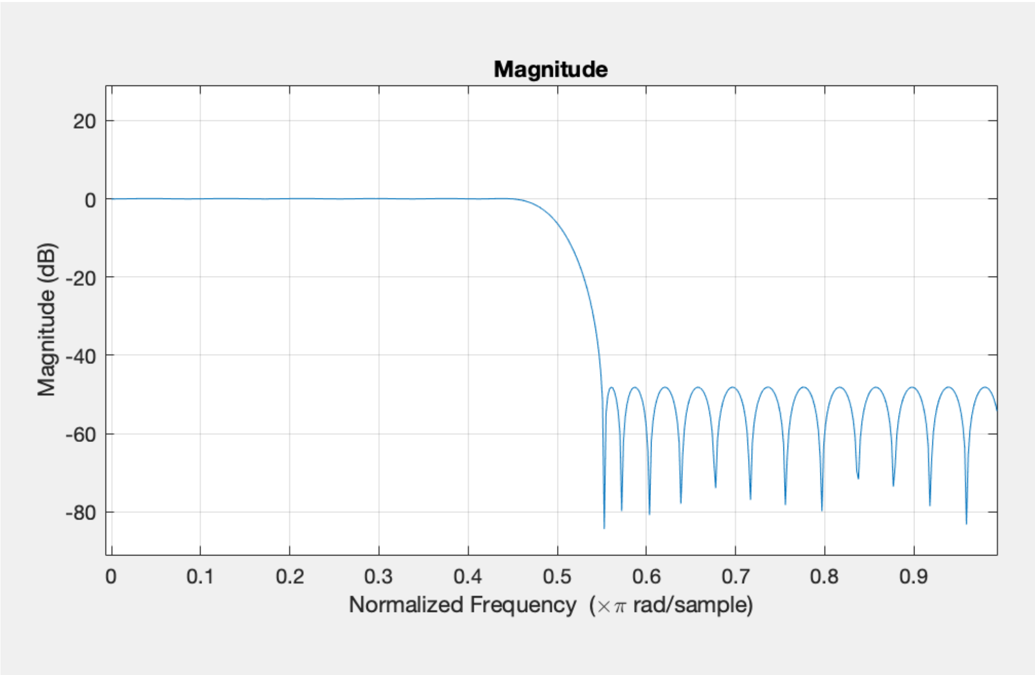


Figure 1: The magnitude response of the Filter thus designed

```
n =

    49


typeoffilter =

     2
```

Figure 2: Order and type of filter

The filter designed by us has an order of **49**, is **causal** and has $\omega_p = 0.45\pi$ and $\omega_s = 0.55\pi$ and from the linear nature of the phase response it is evident that the filter is an LPF.

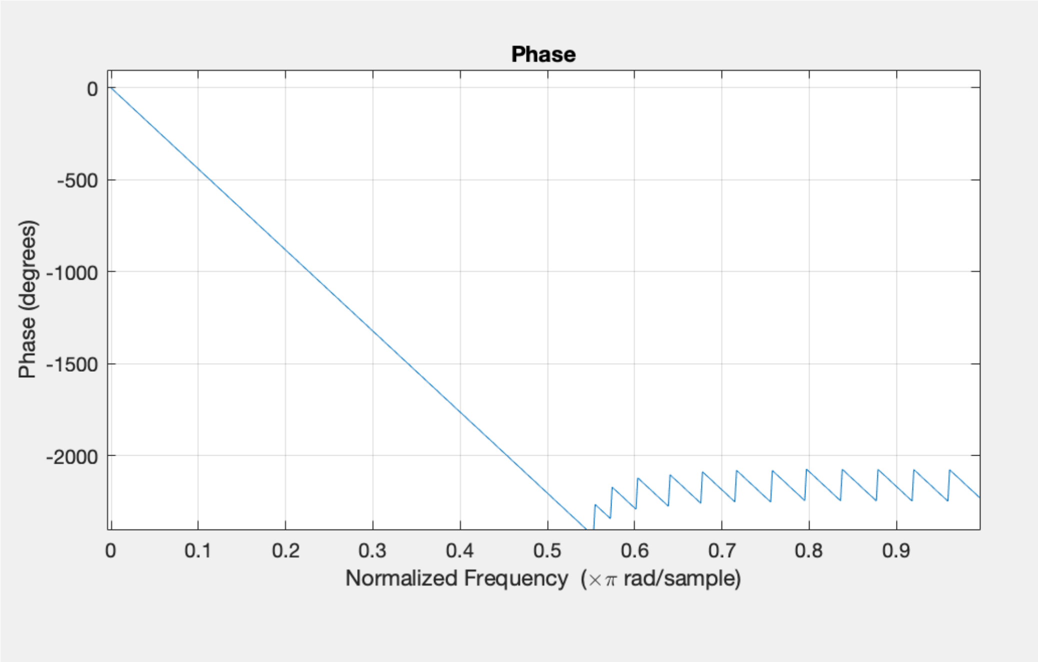


Figure 3: The frequency response of the Filter thus designed

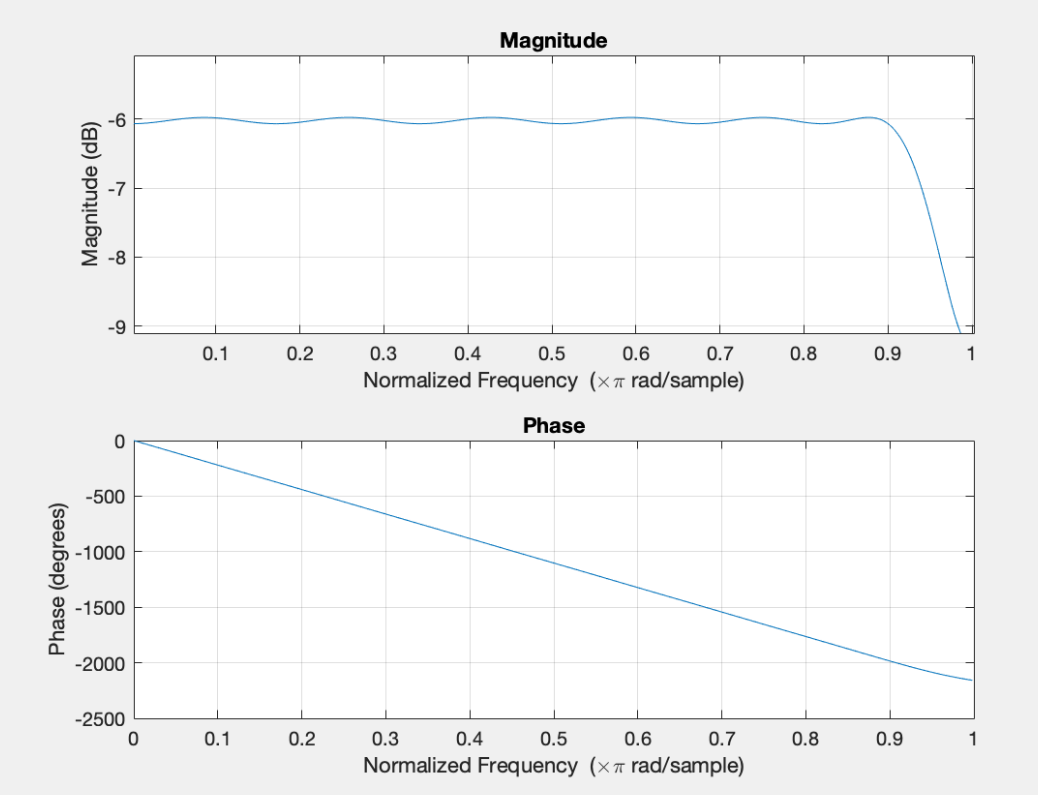


Figure 4: The frequency response of the first polyphase component H_0^0 thus designed

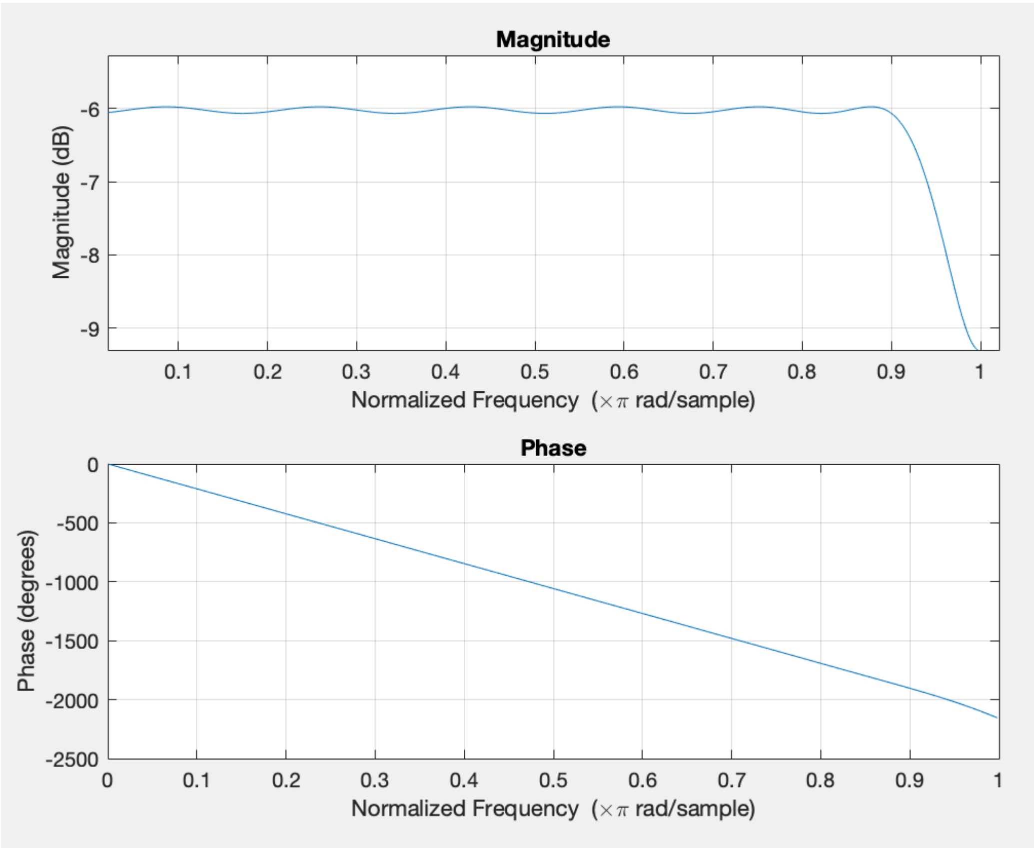


Figure 5: The frequency response of the second polyphase component H_1^0 thus designed

2 $T_{zp}(\omega)$

We have used the formula given in Lecture 18 to plot the Zero Phase response of $T(\omega)$ where ω varies from 0 to π.

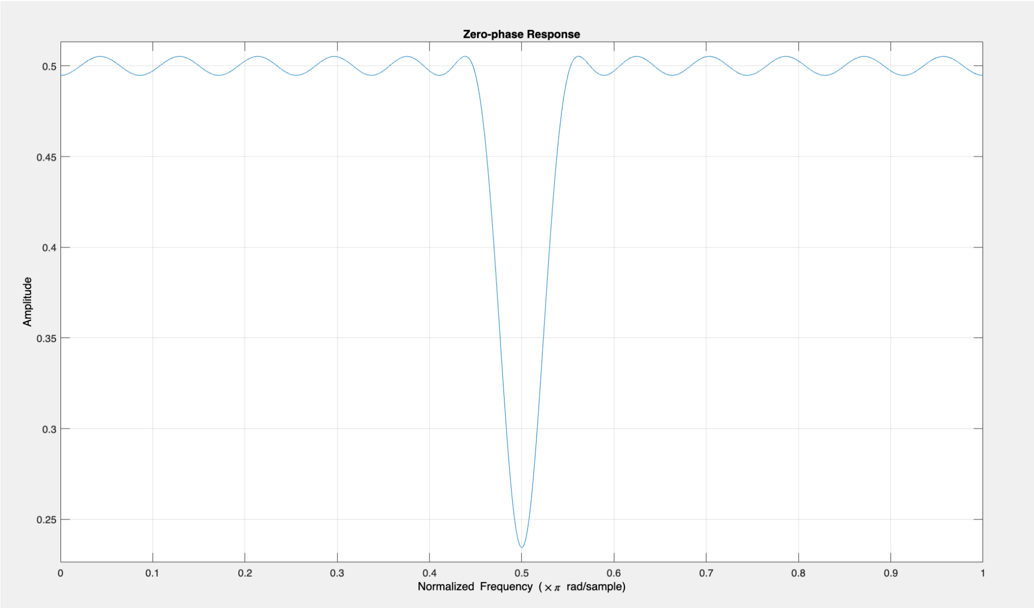


Figure 6: The zero phase response of the filter outputs

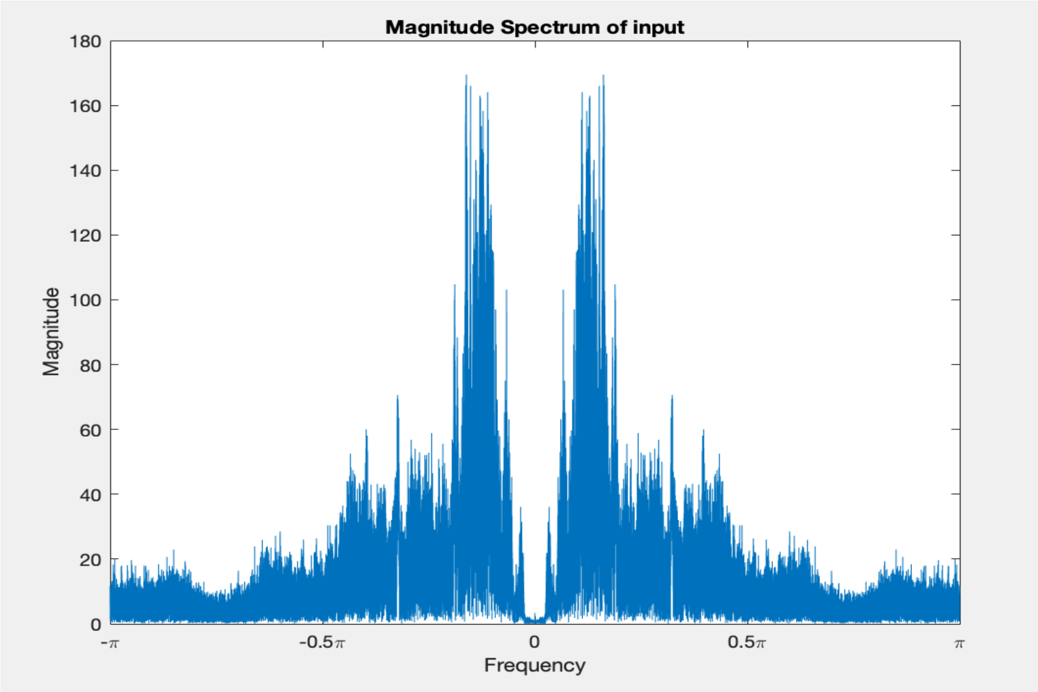
→ For Odd N, ie $H^0(z)$ is Type-2, $T_{zp}(\omega) = \frac{1}{2}\left[(H_{zp}(\omega))^2 + (H_{zp}(\omega-\pi))^2\right]$

Figure 7: The formula used for plotting this

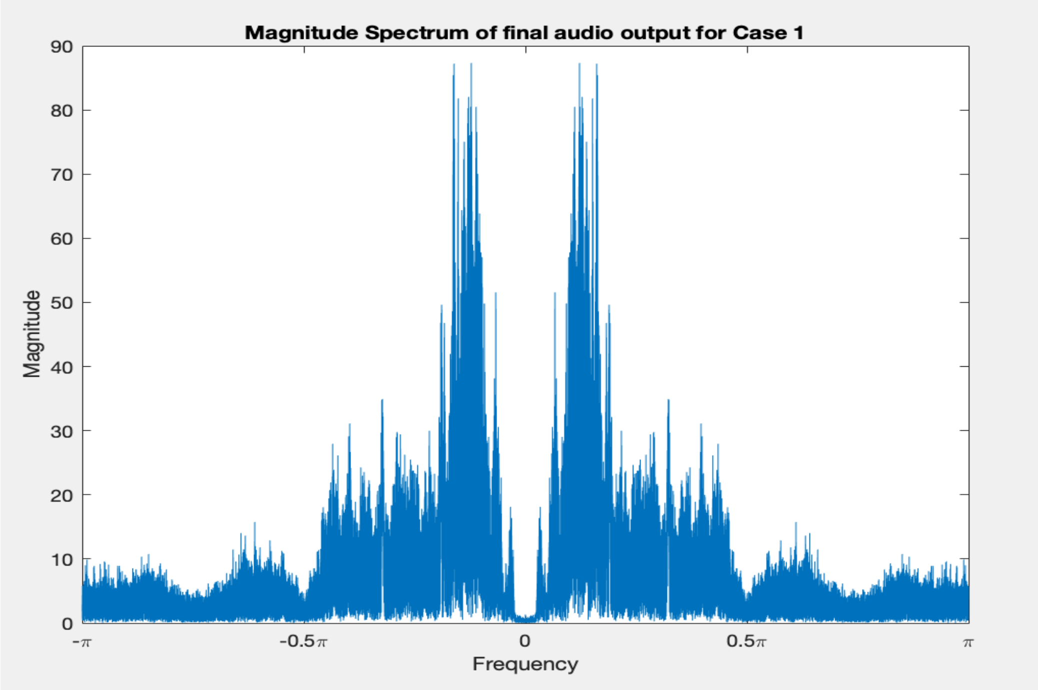
There is no aliasing here, however, this is **not Perfect Reconstruction** since there is one dip at 0.5π. Ideally we would have wanted the whole response to be flat and equal to 0.5. This dip leads to data distortion and loss.

3 Magnitude spectrum of two audio output signals for both the configurations

3.1 For the speech audio signal for Case 1



(a) Magnitude Spectrum of input `speech8khz.wav`



(b) The magnitude spectrum of the final output signal $y(.)$

Figure 8: Comparing final FB output and input

3.2 For the speech audio signal for Case 2

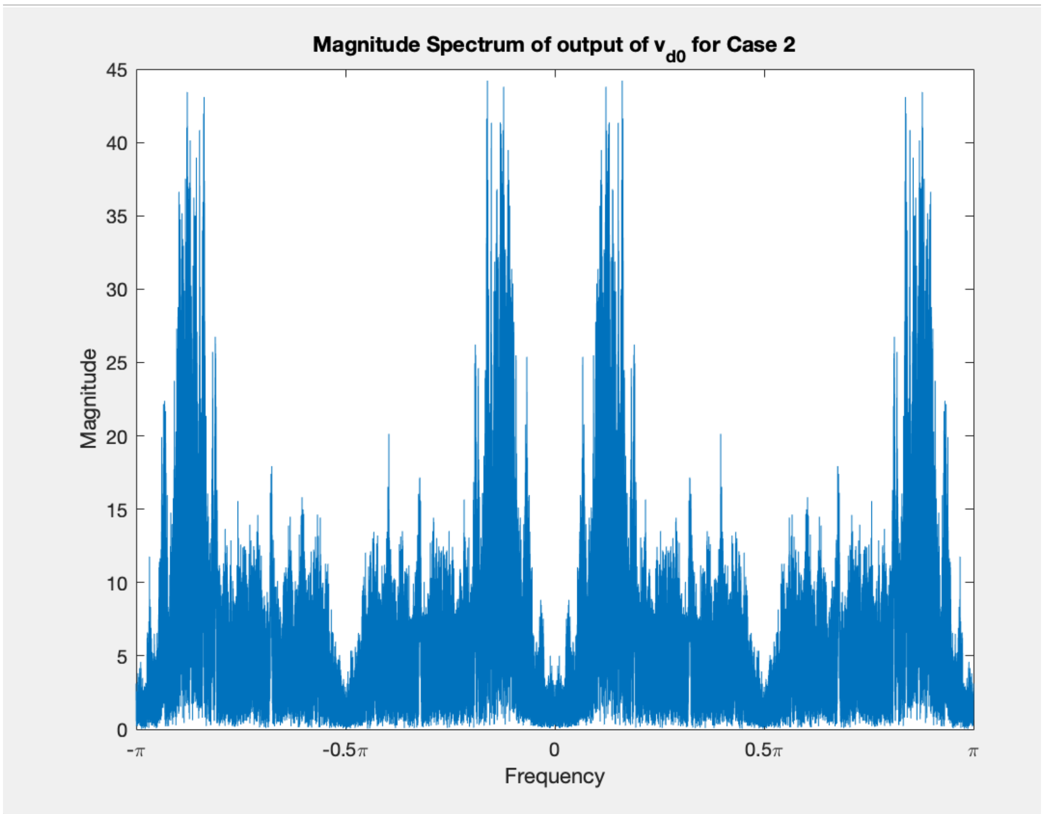


Figure 9: The magnitude spectrum of the output signal of V_d^0

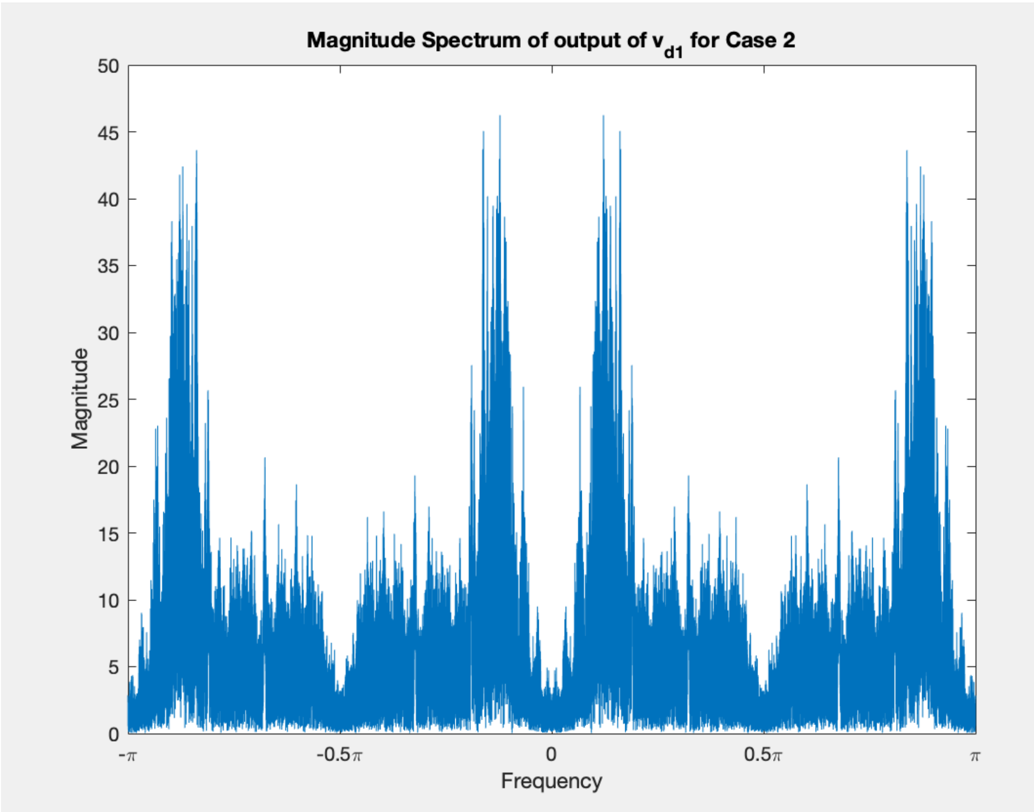
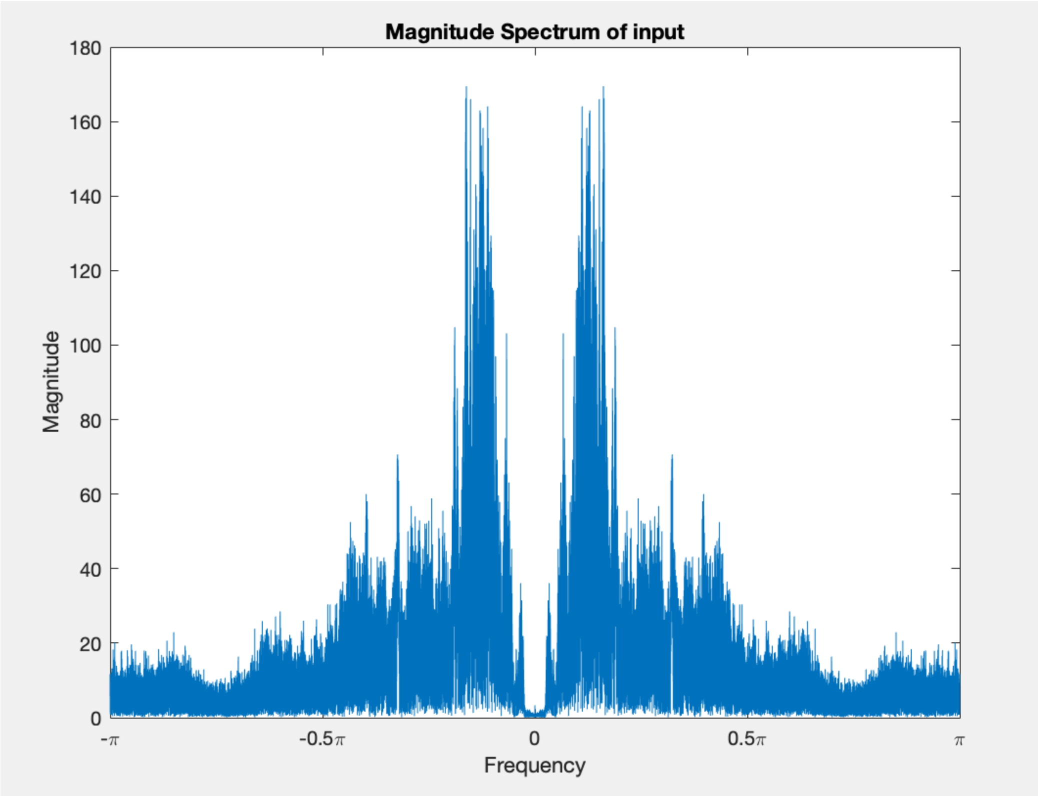
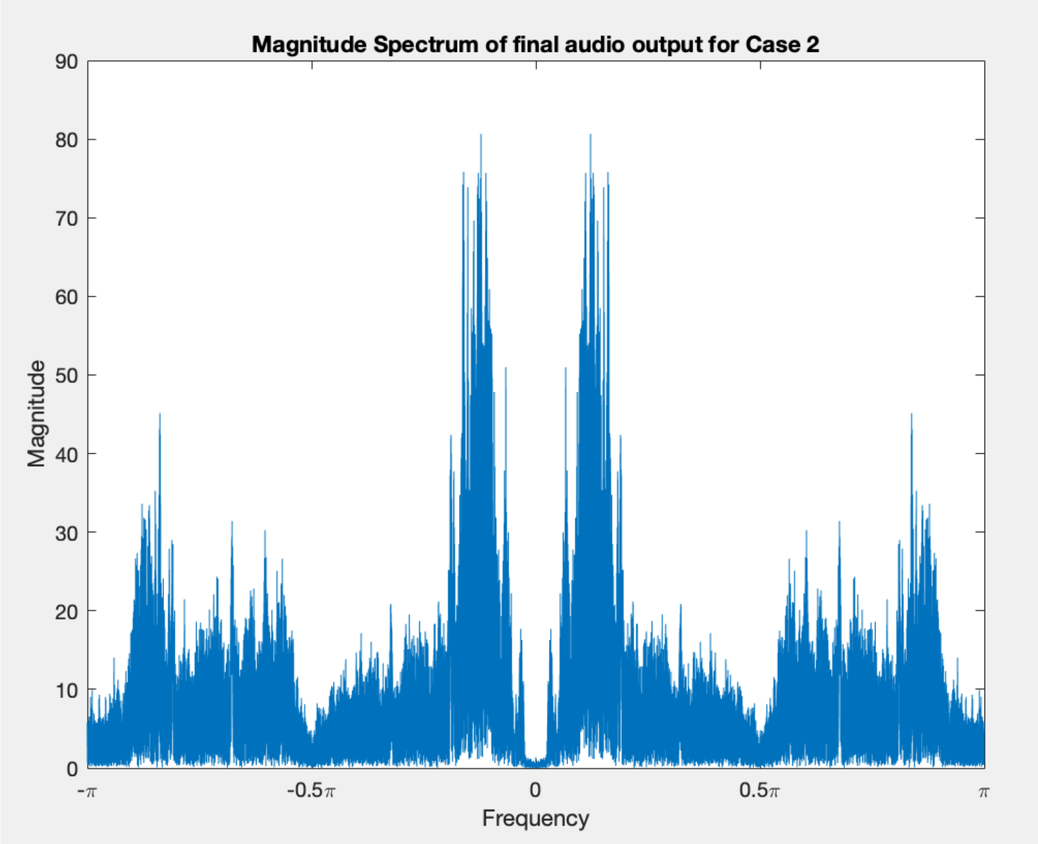


Figure 10: The magnitude spectrum of the output signal of V_d^1



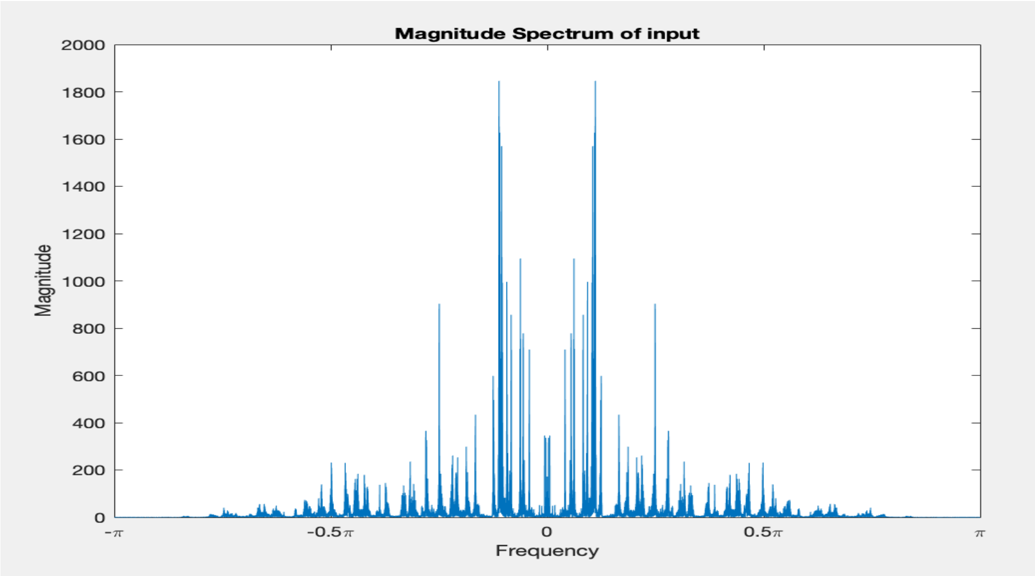
(a) Magnitude Spectrum of input `speech8khz.wav`



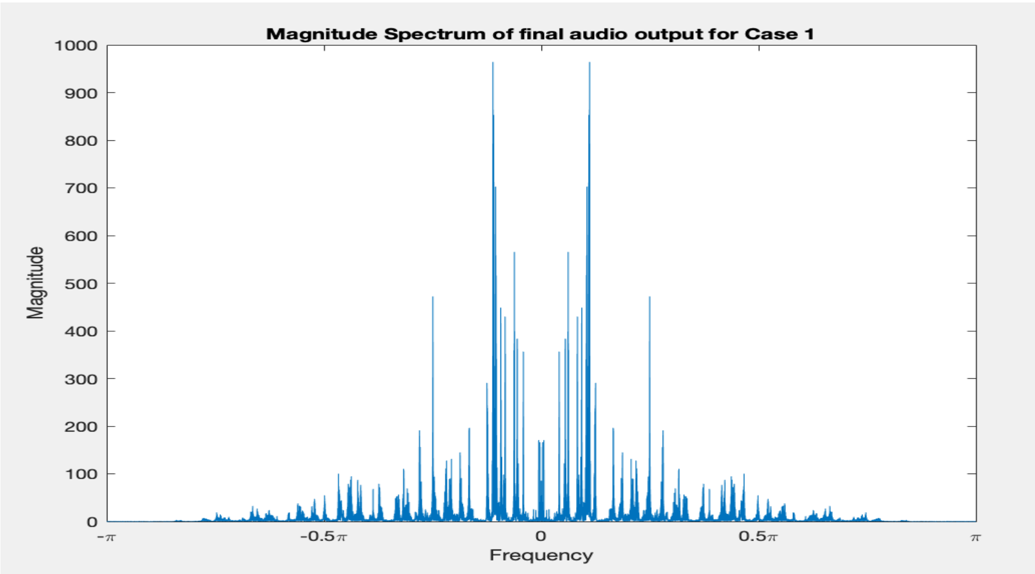
(b) The magnitude spectrum of the final output signal $y(.)$

Figure 11: Comparing final FB output and input

3.3 For the music audio signal for Case 1



(a) Magnitude Spectrum of input `music16khz.wav`



(b) The magnitude spectrum of the final output signal $y(.)$

Figure 12: Comparing final FB output and input

3.4 For the music audio signal for Case 2

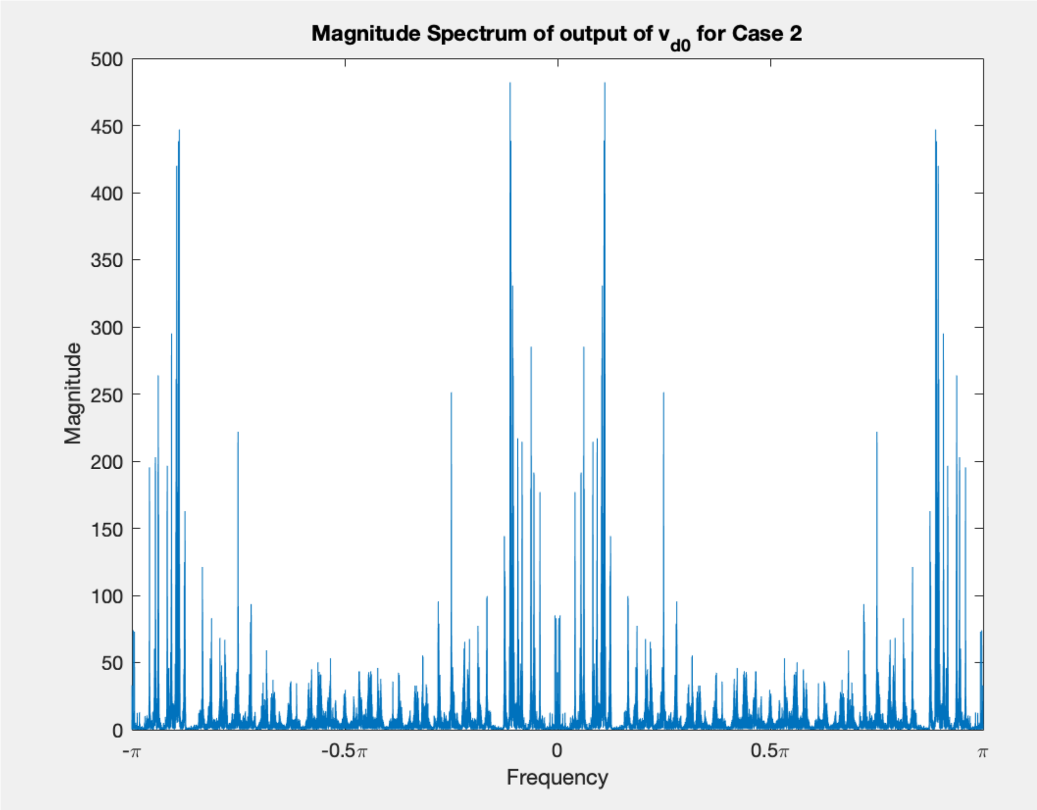


Figure 13: The magnitude spectrum of the output signal of V_d^0

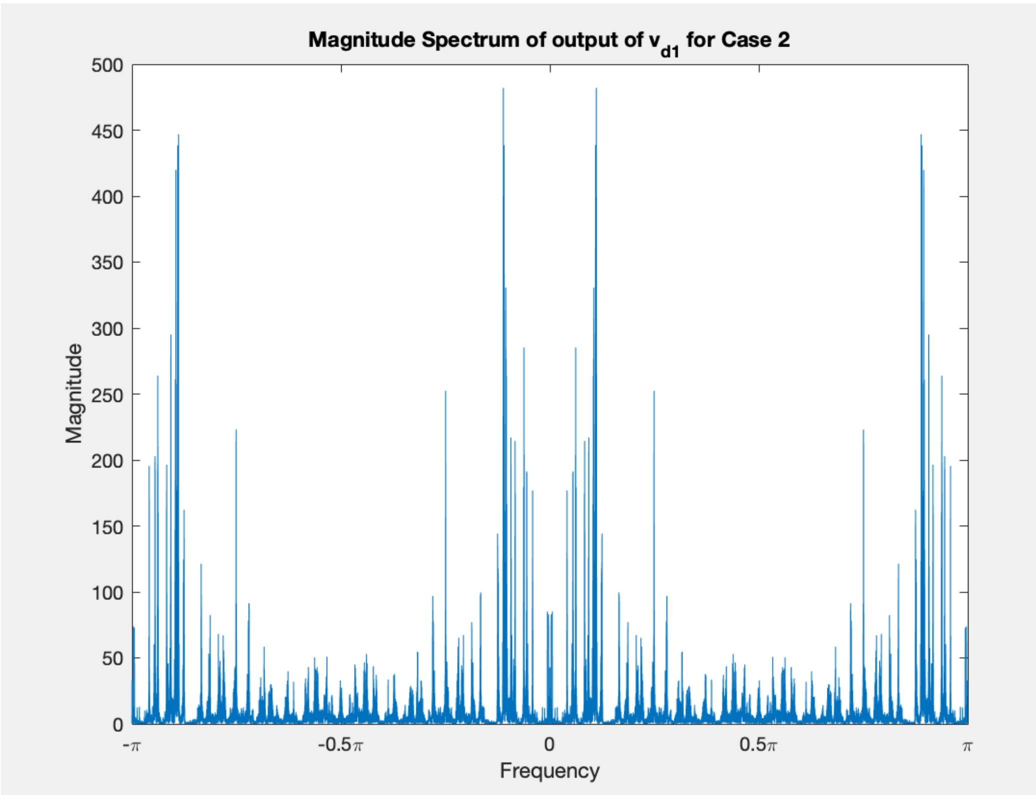
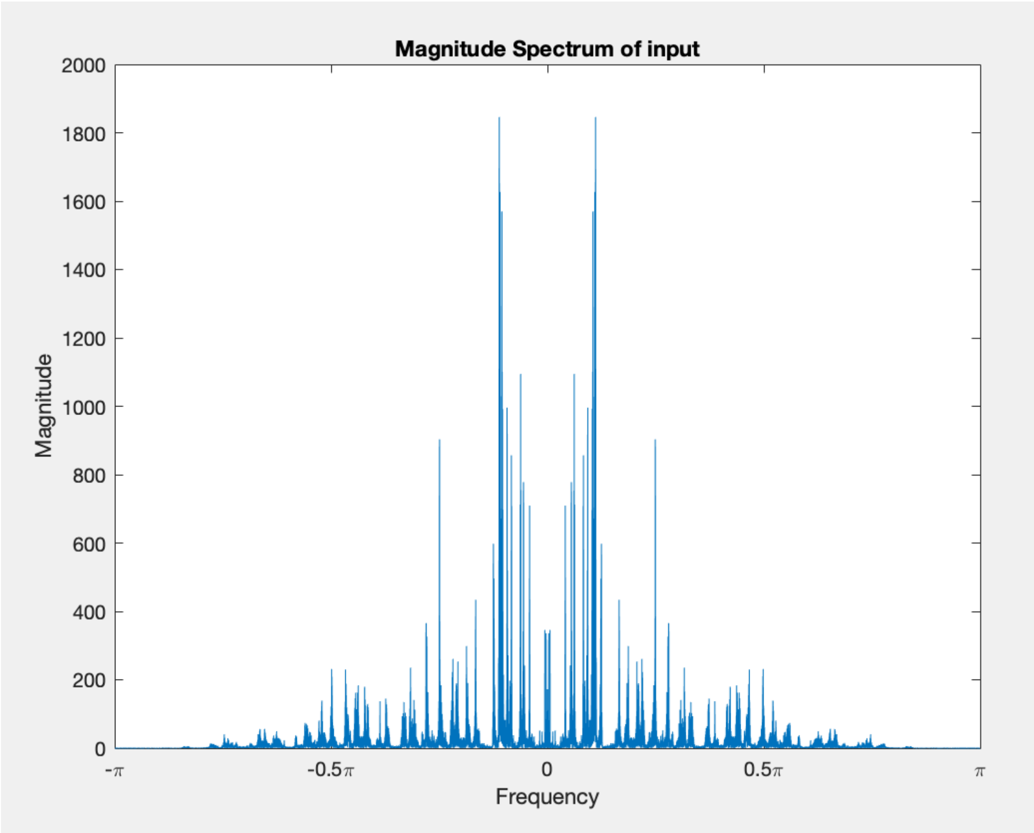
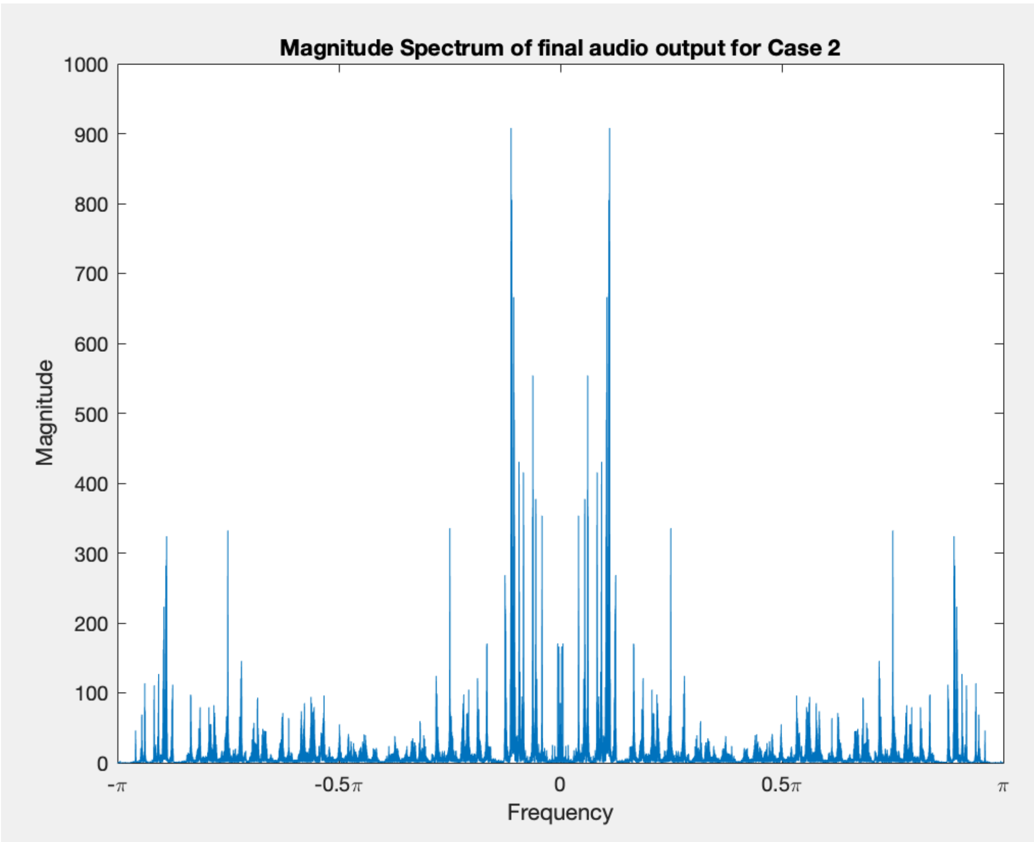


Figure 14: The magnitude spectrum of the output signal of V_d^1



(a) Magnitude Spectrum of input `music16khz.wav`



(b) The magnitude spectrum of the final output signal $y(.)$

Figure 15: Comparing final FB output and input

4 Verbal comparison

4.1 Audio Quality

- In case 1, where there is no aliasing, the output has less noise and it is clear. Though both music and speech outputs were much feeble compared to the input signals.
- In case 2, where aliasing is present, the output is noisy and distorted and some background noise is added. This is especially so in the speech signal where the words are very hard to recognise
- Both the outputs seem to be lower in loudness compared to the original audio sample. However, the distinctive sounds are separate and their distinction is maintained clearly, just like the original.

4.2 Magnitude Spectrum

- The amplitude of the output is nearly half of the amplitude of original input.
- In case 1, the magnitude spectrum of the original input is nearly identical to that of the final output from the filter bank. There is no aliasing in the final output and the audio is therefore very clear. The aliasing terms cancel out to zero in this case. We also notice a very slight dip around 0.5π indicating that it is not PR though the magnitude responses match to a great degree.
- In case 2, however, there are a lot of large amplitude spikes especially near the stop band. These extra signals can be accounted for by aliasing.

5 Matlab code for the entire experiment

```
%Filter Design
[speech,fs]=audioread('/Applications/music16khz.wav');
rp = 0.004;
rs = 0.003;
f = [0.45*(fs/2) 0.55*(fs/2)];
a = [1 0];
dev = [rp rs];
[n,fo,ao,w] = firpmord(f,a,dev,fs);
n=2*floor(n/2)-1;
b = firpm(n,fo,ao,w);
display(n);
typeoffilter=firtype(b);

freqz(b); %filter plots

h= b.* real(exp(1j*-1*pi*(0: n)));
zerophase(dfilt.dffir(conv(b, b) - conv(h,h))); %T_zp plot

H_0_0=b(1:2:end); %polyphase decomposition
H_1_0=b(2:2:end); %polyphase decomposition

%freqz(H_0_0);
```

```
%freqz(H_1_0);

x_d=downsample([speech;0], 2); % ;0 adds one zero to the end.
   This essentially equates the sizes of the two vectors. This
   indirectly starts x from x(-1)
s_d=downsample([zeros(1, size(speech,2));speech],2); %we are
   essentially adding one 0 before the whole vector to delay
   the vector by one sample

t_0=filter(H_0_0,1,x_d);%outputs of the polyphase filters
t_1=filter(H_1_0,1,s_d);

v_d_0=t_0+t_1;%final analysis bank outputs after passing
   through the IDFT matrix of order 2, without the 1/2 factor
v_d_1=t_0-t_1;
%output_v=[v_d_0;v_d_1];

u_0=v_d_0+v_d_1;%The inputs for the K_0 filters are obtained by
    passing the previous outputs  through the DFT matrix of
   order 2
u_1=v_d_0-v_d_1;

%Case1
op1=filter(H_1_0,1,u_0);
op2=filter(H_0_0,1,u_1);

op_com1=upsample(op1,2);
op_com2=upsample(op2,2);
op_com1=[0;op_com1]; %We delay the first output since the
   commutator reaches it finally %this is done solely to see
   the aliasing in individual components
op_com2=[op_com2;0]; %We start with the second output since the
    commutator starts from it

%bottom to top commutator
final_output1=1:length(op1)+length(op2);

for i = 1: length(op2)
final_output1(2*i-1)=op2(i);
end

for i = 1: length(op1)
final_output1(2*i)=op1(i);
end
final_output1=reshape(final_output1,length(final_output1),1) ;


L = length(op_com1); %This is the output after upsampling but
   not delaying
Y = fft(op_com1);
```

```
f_vector = (-L/2:L/2-1)*2*pi/L;
plot(f_vector, fftshift(abs(Y)));
title('Magnitude Spectrum of output of v_d_0 for Case 1');
xlabel('Frequency');
ylabel('Magnitude')
xlim([-pi, pi]);
xticks((-2:2)*pi/2);
xticklabels({ '-\pi','-0.5\pi', '0', '0.5\pi', '\pi'});

figure(2);
L = length(op_com2);
Y = fft(op_com2);
f_vector = (-L/2:L/2-1)*2*pi/L;
plot(f_vector, fftshift(abs(Y)));
title('Magnitude Spectrum of output of v_d_1 for Case 1');
xlabel('Frequency');
ylabel('Magnitude')
xlim([-pi, pi]);
xticks((-2:2)*pi/2);
xticklabels({ '-\pi','-0.5\pi', '0', '0.5\pi', '\pi'});

%Case2
opp1=filter(H_0_0,1,u_0);
opp2=filter(H_1_0,1,u_1);

opp_com1=upsample(opp1,2);
opp_com2=upsample(opp2,2);
opp_com1=[0;opp_com1];
opp_com2=[opp_com2;0];


final_output2=1:length(opp1)+length(opp2);

for i = 1: length(opp2)
final_output2(2*i-1)=opp2(i);
end

for i = 1: length(opp1)
final_output2(2*i)=opp1(i);
end
final_output2=reshape(final_output2,length(final_output2),1) ;


figure(3);
L = length(opp_com1);
Y = fft(opp_com1);
f_vector = (-L/2:L/2-1)*2*pi/L;
plot(f_vector, fftshift(abs(Y)));
title('Magnitude Spectrum of output of v_d_0 for Case 2');
xlabel('Frequency');
```

```
ylabel('Magnitude')
xlim([-pi, pi]);
xticks((-2:2)*pi/2);
xticklabels({ '-\pi','-0.5\pi', '0', '0.5\pi', '\pi'});

figure(4);
L = length(opp_com2);
Y = fft(opp_com2);
f_vector = (-L/2:L/2-1)*2*pi/L;
plot(f_vector, fftshift(abs(Y)));
title('Magnitude Spectrum of output of v_d_1 for Case 2');
xlabel('Frequency');
ylabel('Magnitude')
xlim([-pi, pi]);
xticks((-2:2)*pi/2);
xticklabels({ '-\pi','-0.5\pi', '0', '0.5\pi', '\pi'});

%final output plots

figure(5);
L = length(final_output1);
Y = fft(final_output1);
f_vector = (-L/2:L/2-1)*2*pi/L;
plot(f_vector, fftshift(abs(Y)));
title('Magnitude Spectrum of final audio output for Case 1');
xlabel('Frequency');
ylabel('Magnitude')
xlim([-pi, pi]);
xticks((-2:2)*pi/2);
xticklabels({ '-\pi','-0.5\pi', '0', '0.5\pi', '\pi'});


figure(6);
L = length(final_output2);
Y = fft(final_output2);
f_vector = (-L/2:L/2-1)*2*pi/L;
plot(f_vector, fftshift(abs(Y)));
title('Magnitude Spectrum of final audio output for Case 2');
xlabel('Frequency');
ylabel('Magnitude')
xlim([-pi, pi]);
xticks((-2:2)*pi/2);
xticklabels({ '-\pi','-0.5\pi', '0', '0.5\pi', '\pi'});
```