Speech Signal Processing(EE679) Assignment 1

Akshay Bajpai(193079002)

20th September 2020

1 Report Structure

For each of the questions , I have written the basic algorithm followed to implement the solution , followed by the code and the output graphs. The codes have been written in octave and the corresponding output files(**plots and .wav files**) are in the data folder submitted .

1.1 Question 1

Given the following specification for a single-formant resonator, obtain the transfer function of the filter H(z) from the relation between resonance frequency / bandwidth, and the pole angle / radius. Plot filter magnitude response (dB magnitude versus frequency) and impulse response.
F1 (formant) = 900 Hz
B1(bandwidth) = 200 Hz
Fs (sampling freq) = 16 kHz

Solution :

Algorithm :

1. Given the formant frequency (F1) , Bandwidth (B1) , r and θ are found using :

$$r_1 = exp(\frac{-B_1\pi}{F_s}) \tag{1}$$

$$\theta_1 = \frac{2\pi F_1}{F_s} \tag{2}$$

2. Using the values obtained for r and θ , we calculate the coefficients of the second order system using :

$$H(z) = \frac{1}{1 - 2r\cos\theta z^{-1} + r^2 z^{-2}} \tag{3}$$

3. The impulse response is then found by using the difference equation and $x(n)$ as a discrete time impulse function.

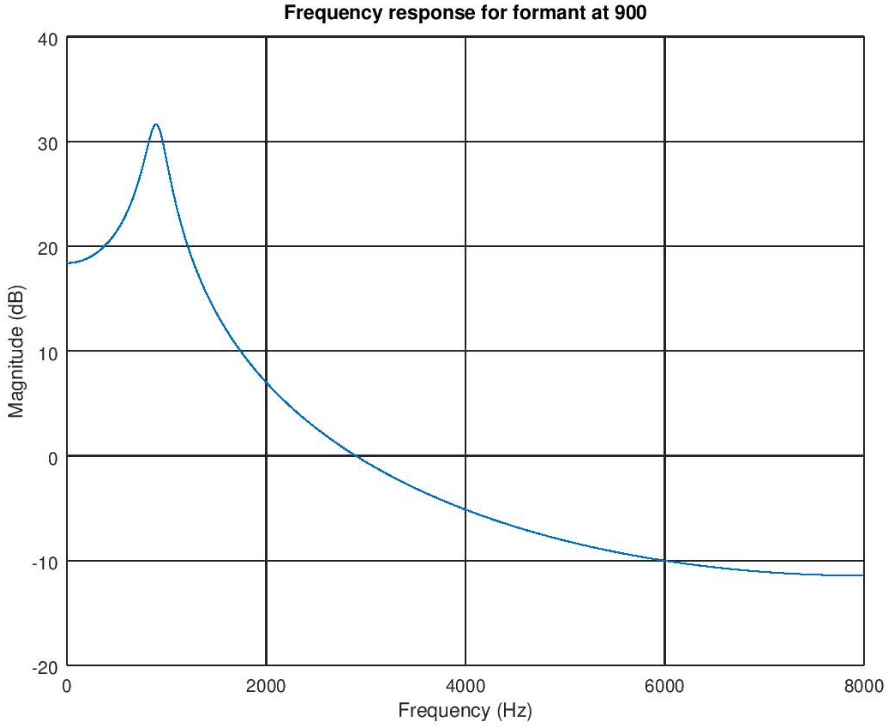
Code :

```
pkg load signal;

%Question 1
f1=900;
b1=200;
fs=16000;
```

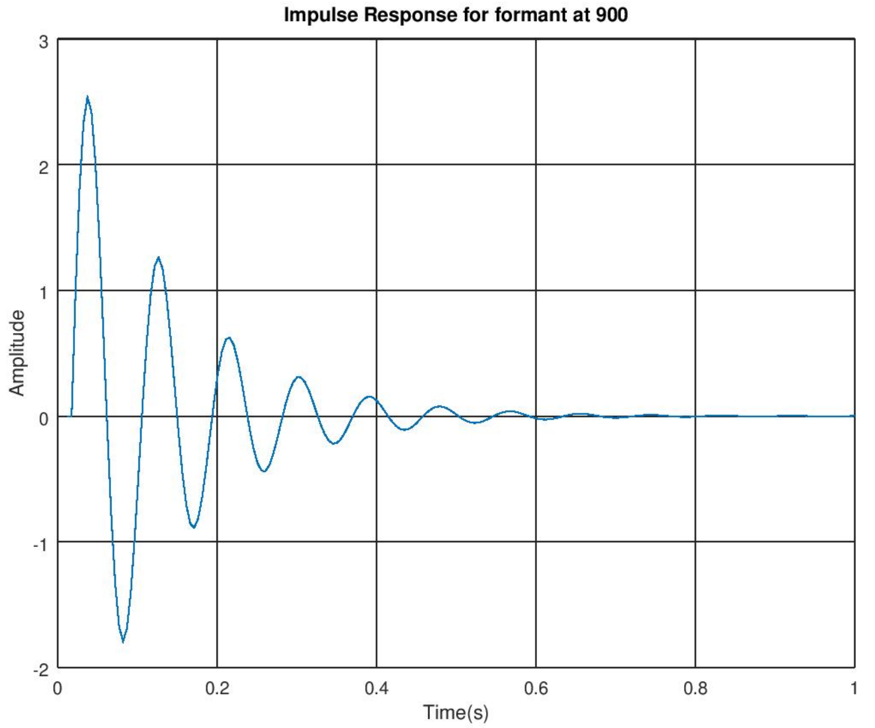
```
 f0=140
 [b,a]=get_coeff(f1,b1,fs);
 h=freq_response(b,a,fs);
```

```
function [b,a]=get_coeff(f1,b1,fs)

    r = exp(-b1*pi*1/fs);
    theta = 2*pi*f1*1/fs;
    poles = [r*exp(1j*theta) , r*exp(-1j*theta)];
    b = [1  ,0,0];
    a= [1,-2*r*cos(theta),r**2];
```

```
function h=freq_response(b,a,fs,f1)

    [h,w] = freqz(b,a) ;
    fullname=['assignment1/frequency_response_f1_',num2str(f1),'.jpg']
    figure
    plot(fs*w/(2*pi),20*log10(abs(h)));
    xlabel('Frequency (Hz)');
    ylabel('Magnitude (dB)');
    title(['Frequency response for formant at ',num2str(f1)]);
    grid on;

    impulse = zeros(200,1);
    impulse (1,1) = 1;
    y = zeros(200,1);
    [ro,col]=size(y);
    [m,number_of_poles]=size(a);
    for i=number_of_poles:ro
      y(i,1) = b(1,1)*impulse(i-2,1) ;
      for j=2:number_of_poles
        y(i,1)=y(i,1)-a(1,j)*y(i-j+1,1);
      end
    end
    figure;
    fullname=['assignment1/filter_response_f1_',num2str(f1),'.jpg']
    time= linspace(200/fs,1,200);
    plot(time,y);
    xlabel('Time(s)');
    ylabel('Amplitude');
    title(['Impulse Response for formant at ',num2str(f1)] );
    grid on;
```



(a)



(b)

Figure 1: (a)Frequency response of the filter having formant frequency at 900Hz (b)Impulse response for the filter

1.2 Question 2

Excite the above resonator ("filter") with a periodic source excitation of F0 = 140 Hz. You can approximate the source signal by narrow-triangular pulse train. Compute the output of the source-filter system over the duration of 0.5 second using the difference equation implementation of the LTI system. Plot the time domain waveform over a few pitch periods so that you can observe waveform characteristics. Play out the 0.5 sec duration sound and comment on the sound quality.

Solution:

Algorithm :

1. We create a triangular impulse train $x(n)$ by approximating the square function generator of the desired length.
2. In the above filter , we have derived the filter coefficients which will be used to form the difference equation :

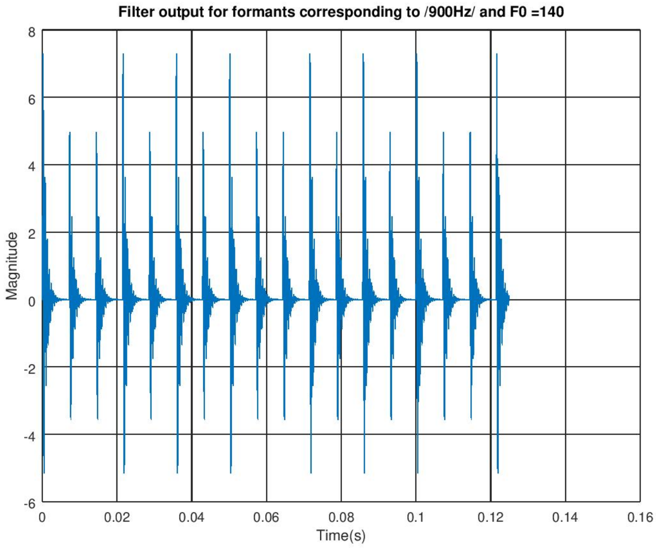
$$y(n) = b_0 x(n) - a_1 y(n-1) - a_2 y(n-2) \tag{4}$$

3. The above equation is looped for all values of n to obtain the discrete time domain output signal $y(n)$

Code :

```
 function input_signal(h,b,a,f0,fs,time_,filename)

t = 0:time_/fs:time_; % 0s to 0.5s with Fs sample freq
[z,time_length]=size(t);
x1 = max(0,(square(2*pi*f0*t,0.01))); % 2*pi* freq * time duration
% figure;
% plot(t,x1);
% title('Triangular pulse train ');
x1=x1';
y=zeros(time_length,1);

[ro,col]=size(y);
[m,number_of_poles]=size(a);

for i=number_of_poles:rows(y)
  y(i,1) = b(1,1)*x1(i-2,1) ;
  for j=2:number_of_poles
    y(i,1)=y(i,1)-a(1,j)*y(i-j+1,1);
  end
end

figure;
t=t';
plot(t(1:4000,1),y(1:4000,1));
xlabel('Time(s)');
ylabel('Magnitude');
title(['Filter output for formants corresponding to /',filename,'/ and
   F0 =', num2str(f0)]);
```

```
grid on;
wavwrite (y,fs, ["assignment1/",filename,"_",num2str(f0),".wav"]);
```



(a)

Figure 2: (a)Filter response for triangular pulse train at $F_0 = 140$Hz

Comments :

1. The output of the filter is periodic having sudden peaks in between
2. The sound is of the same pitch throughout, seems to be noisy and not much pleasing to the ears.

1.3 Question 3

Vary the parameters as indicated below and comment on the differences in waveform and sound quality for the different parameter combinations.

1. F0 = 120 Hz, F1 = 300 Hz, B1 = 100 Hz
2. F0 = 120 Hz, F1 = 1200 Hz, B1 = 200 Hz
3. F0 = 180 Hz, F1 = 300 Hz, B1 = 100 Hz

Solution:
Algorithm :

1. The same function made above is used in loop with varied formant frequency and fundamental frequency values.
2. Using the function made in the first question, we find the values of the frequency response for different values of ω
3. The same difference equation as below is used then to find the output of the filter $y(n)$.

$$H(z) = \frac{1}{1 - 2r\cos\theta z^{-1} + r^2 z^{-2}} \tag{5}$$

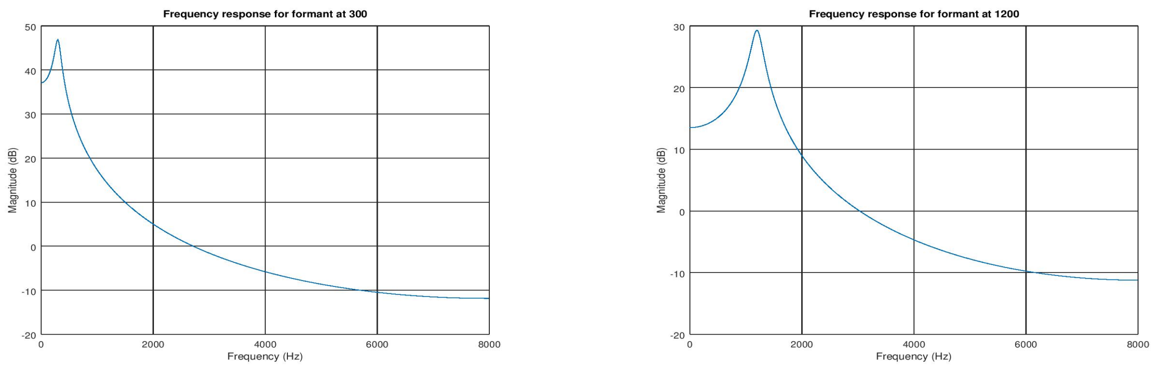
$$y(n) = b_0 x(n) - a_1 y(n-1) - a_2 y(n-2) \tag{6}$$

4. The above equation is looped for all values of n to obtain the discrete time domain output signal $y(n)$

5. **The function used below** $freq\_response()$ **and** $input\_signal()$ **are defined in the code section of question 1 and question 3 respectively.**

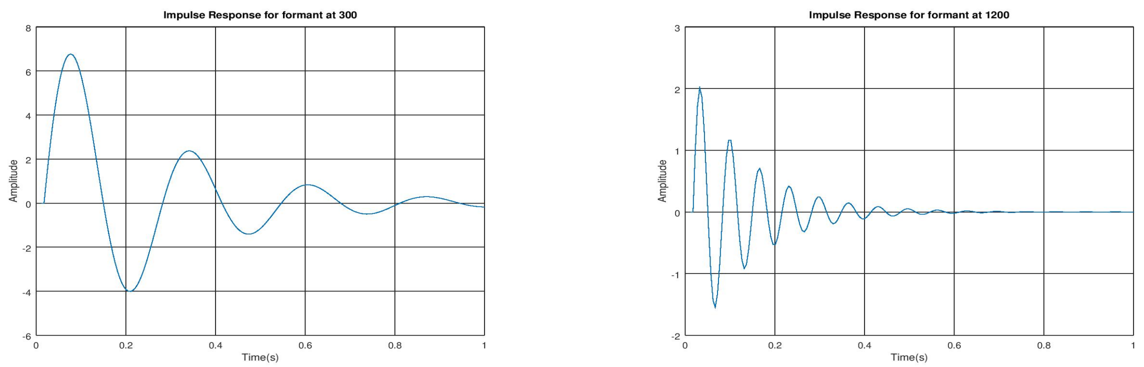
Code :

```
f0=[120,120,180];
f1=[300,1200,300];
b1=[100,200,100];
for  i=1:3
 [b,a]=get_coeff(f1(1,i),b1(1,i),fs);
 h=freq_response(b,a,fs,f1(1,i));
 input_signal(h,b,a,f0(1,i),fs,0.5,num2str(f1(1,i)));
endfor
```



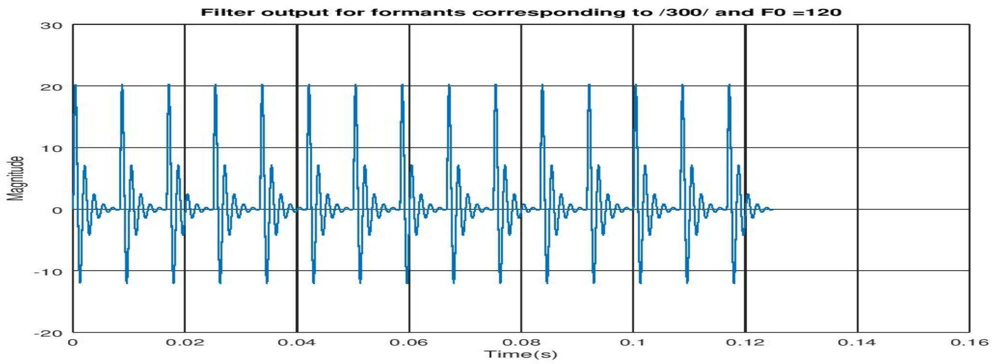
(a) 300 Hz (b) 1200 Hz

Figure 3: Frequency Response

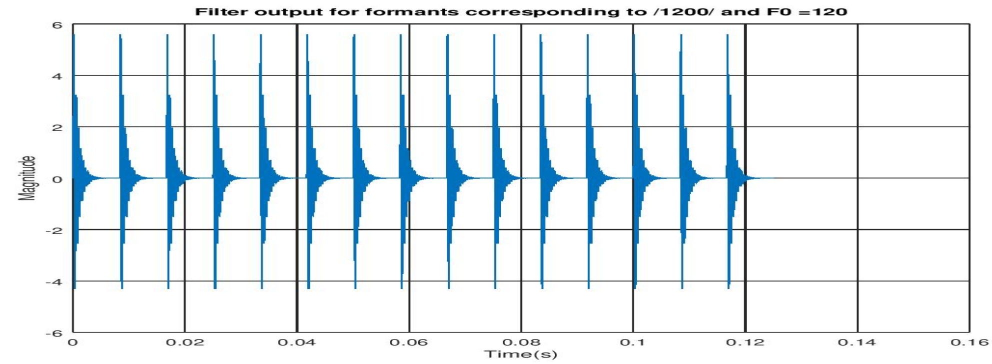


(a) 300 Hz (b) 1200 Hz

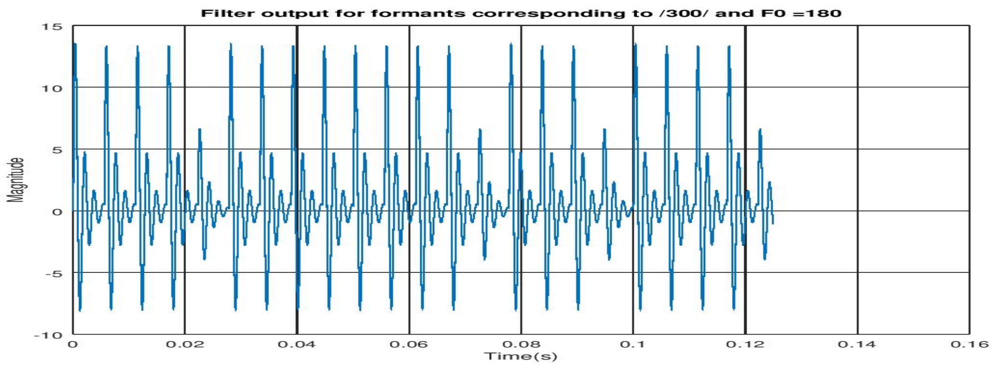
Figure 4: Impulse Response



(a) $\mathrm{F}_1 = 300\mathrm{Hz} F_0 = 120\mathrm{Hz}$



(b) $\mathrm{F}_1 = 1200\mathrm{Hz} F_0 = 120\mathrm{Hz}$



(c) $\mathrm{F}_1 = 300\mathrm{Hz} F_0 = 180\mathrm{Hz}$

Figure 5: Filter Response for the given formant frequency F1 and input fundamental frequency F0

Comments :

1. The frequency of the output waveform from the filter varies in proportion to the fundamental frequency which is expected as we have and LTI filter. (Higher if the input fundamental freq. is higher)

2. For the case (a) and (c) we have the same formant frequency and fundamental frequency is varied. Hence the sound seems to be of same acoustic features with a change in the pitch. (Similar to singing a music notation at two different pitch).

3. Case (a) and (b) have the same fundamental frequency , hence the pitch of the sound is same while different frequency components are amplified , hence the difference in the sound.

1.4 Question 4

In place of the simple single-resonance signal, synthesize the following more realistic vowel sounds at two distinct pitches (F0 = 120 Hz, F0 = 220 Hz). Keep the bandwidths constant at 100 Hz for all formants. Duration of sound: 0.5 sec

1. /a/ 730,1090,2440
2. /i/ 270,2290,3010
3. /u/ 300,870,2240

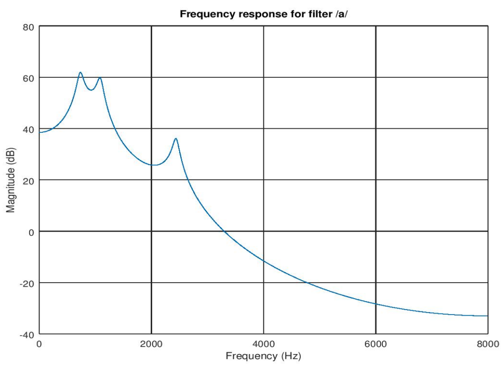
Solution:
Algorithm :

1. Find the coefficients of the frequency response for each of the formant frequencies.
2. Calculate the coefficients of the combined frequency response by convolution of coefficients of the individual functions.
3. Use the same function as earlier to obtain the frequency response.
4. Generate triangular impulse train of the given formant frequency and calculate the output in discrete time domain using the difference equation, same as earlier.
5. **As before , the function used below** *freq_response*() **and** *input_signal*() **are defined in the code section of question 1 and question 2 respectively.**

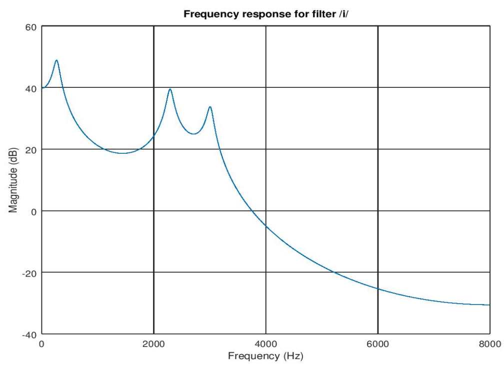
Code :

```
    f1=[730,270,300];
f2=[1090,2290,870];
f3=[2440,3010,2240];
filename=["a","i","u"];
b0=100;
fs=16000;
f0=[120,220];

for j=1:columns(f0)
  for i=1:columns(f1)
    r1 = exp(-b0*pi*1/fs);
    theta1 = 2*pi*f1(1,i)*1/fs;
    r2 = exp(-b0*pi*1/fs);
    theta2 = 2*pi*f2(1,i)*1/fs;
    r3 = exp(-b0*pi*1/fs);
    theta3 = 2*pi*f3(1,i)*1/fs;

    poles1 = [r1*exp(1j*theta1) , r1*exp(-1j*theta1)];
    poles2 = [r2*exp(1j*theta2) , r2*exp(-1j*theta2)];
    poles3 = [r3*exp(1j*theta3) , r3*exp(-1j*theta3)];

    b1 = [1  ,0,0];
```

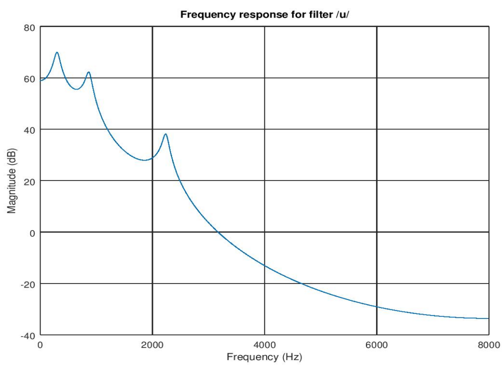
```
        a1= [1,-2*r1*cos(theta1),r1**2];
        b2 = [1  ,0,0];
        a2= [1,-2*r2*cos(theta2),r2**2];
        b3 = [1  ,0,0];
        a3= [1,-2*r3*cos(theta3),r3**2];
        b_temp=conv(b1,b2);
        b = conv(b_temp,b3);
        a_temp=conv(a1,a2);
        a = conv(a_temp,a3);
        [h,w] = freqz(b,a) ;
        figure;
        plot(fs*w/(2*pi),20*log10(abs(h)));
        xlabel('Frequency (Hz)');
        ylabel('Magnitude (dB)');
        title(['Frequency response for filter /',filename(1,i),'/']);
        grid on;
        input_signal(h,b,a,f0(1,j),fs,0.5,filename(1,i))

    endfor
endfor
```



(a) /a/ 730Hz,1090Hz,2440Hz

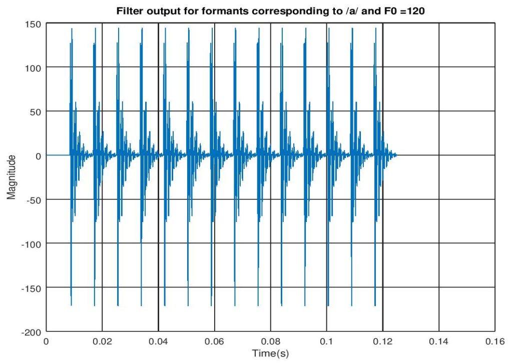


(b) /i/ 270Hz,2290Hz,3010Hz

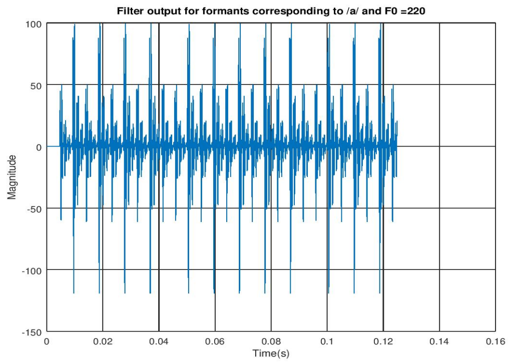


(c) /u/ 300Hz,870Hz,2240Hz

Figure 6: Frequency Response - 3 formants

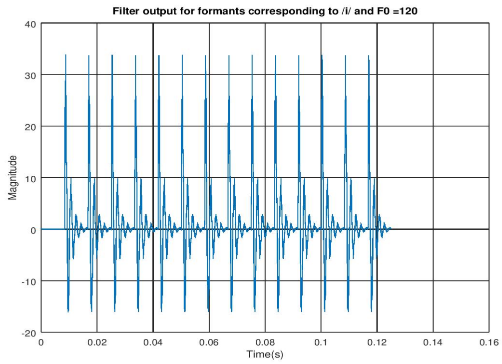


(a) /a/ F0 = 120Hz

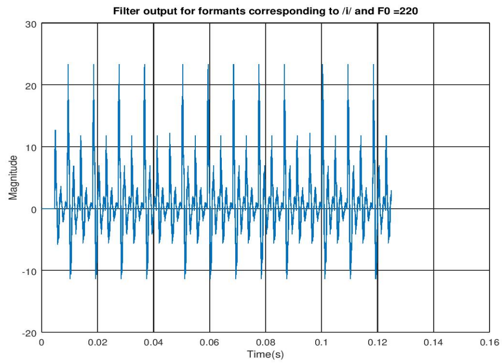


(b) /a/ F0 = 220Hz

Figure 7: Filter Response for filter corresponding to vowel /a/ at two different fundamental frequencies

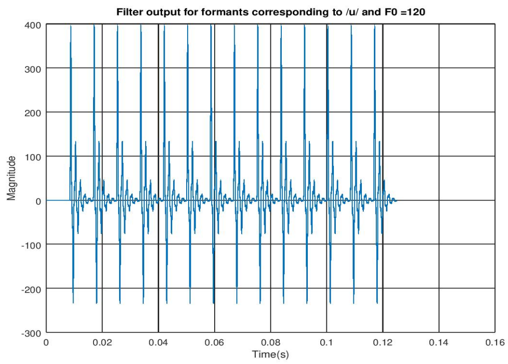


(a) /i/ F0 = 120Hz

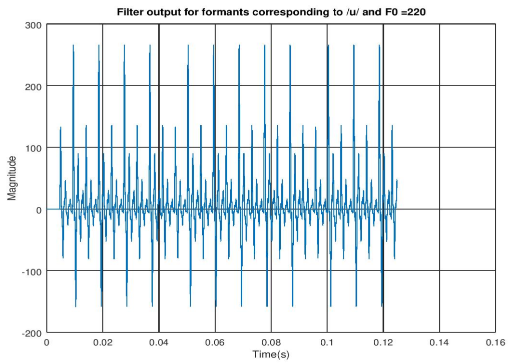


(b) /i/ F0 = 220Hz

Figure 8: Filter Response for filter corresponding to vowel /i/ at two different fundamental frequencies



(a) /u/ F0 = 120Hz



(b) /u/ F0 = 220Hz

Figure 9: Filter Response for filter corresponding to vowel /u/ at two different fundamental frequencies

Comments :

1. The output sounds actually do sound like the vowels /a/ , /i/ ,/u/ but it is still far from the actual sound which we are used to , due to lack of change in pitch/aspiration/lip radiation etc.