

Speech Signal Processing A4 Report

Q1 . Load a speech signal of your interest and select voiced region then Find pitch using LP analysis? Plot the results (Speech, LP-residual, autocorrelation of LP-residual)

A1: - .

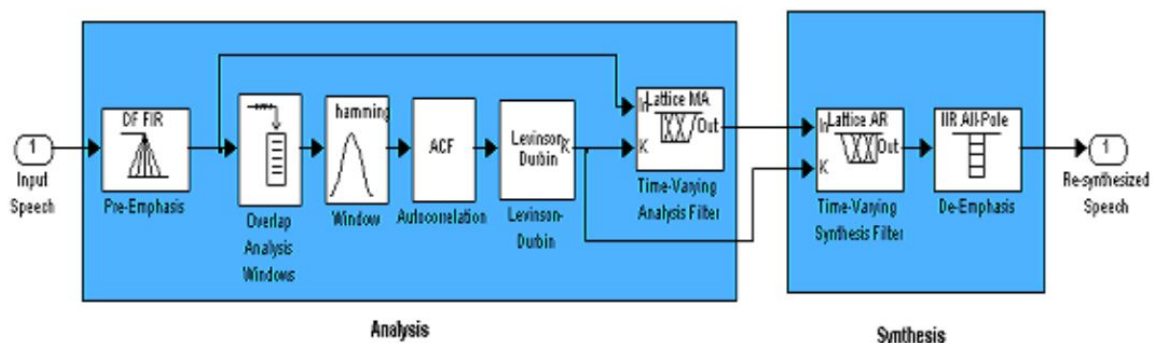
Procedure :

1. Here, SIFT is also used in this procedure along with LP analysis techniques . The first step is to compute the Linear Prediction (LP) residual ,I've used Levinson-Durbin Method for filter coefficients computation (from scratch) .
2. Perform auto-corellation on LP residual signal .
3. Estimation of the pitch period can be computed by finding the time lag corresponding to the second largest peak from the central peak of the autocorrelation sequence.

Here, for file for voiced part = 'a_frame.wav' and for unvoiced frame = 'k_frame.wav' .

Note: I've removed the framing part in this code , since the files are of extremely small duration .

LP Analysis Diagram:



MATLAB Code Snippet : [pitch_estimation.m]

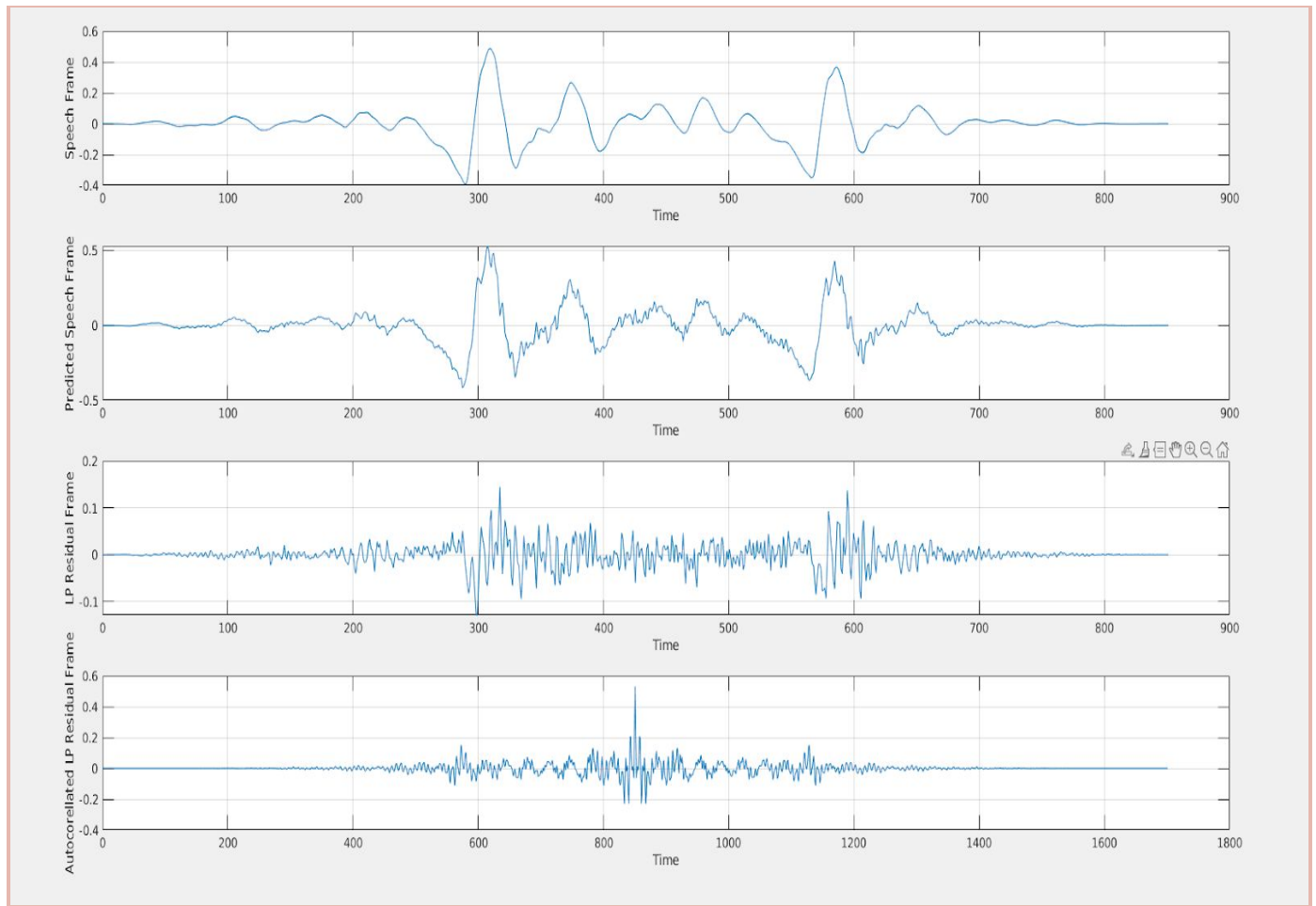
```
function pitch(fileName)
% Read the file ,and output details of the file .
[data,fs]=audioread(fileName);
info=audioinfo(fileName)
flag=1;
% Extracting one channel .
data=data(:,1);

% Length of file is extremely small , hence no framing required .
sigLen=length(data);
orderLPC=24;
window=hanning(sigLen);

% Multiply the signal with hanning window .
data=window.*data;
% Perform Autocorellation
sig_auto=xcorr(data);
%Normalise the data .
sig_auto=sig_auto./abs(max(sig_auto));
% Now, we to create R matrix , we take LPC order number of last elements.
% Past samples
A=sig_auto(1:orderLPC);
% Output signal values for each of the combinations of timestamps.
r=sig_auto(2:orderLPC+1);
% Toeplitz Matrix of coloumn vector-A
A=toeplitz(A);
r_t=transpose(r);
A=(-1).*(inv(A));
% Computing L
L=r_t*A ;
L=transpose(L);
%Extracting the LP Coefficients .
lpcoeffs=[1;L];
% Performing Filtering Operation .
G=1 ;
filtered_sig=filter([0-1*lpcoeffs(2:end)],G,data);
% Error Computation (Residual)
lp_res=data-filtered_sig;
% Autocorrelation of signal .
auto_lpres=xcorr(lp_res);

% Compute the Pitch .
[pks,loc]=findpeaks(auto_lpres);
[pks,ind]=sort(pks,'descend');
loc=loc(ind);
[~,mind]=max(pks);
pitch_period=abs(loc(mind+1)-loc(mind));
%Time conversion .
pitch_ms=pitch_period*1000/fs;
% Estimated Pitch
disp('Pitch in Hz : ')
pitch=1000/pitch_ms
```

Case 1 : Voiced Frame - (a_frame.wav)

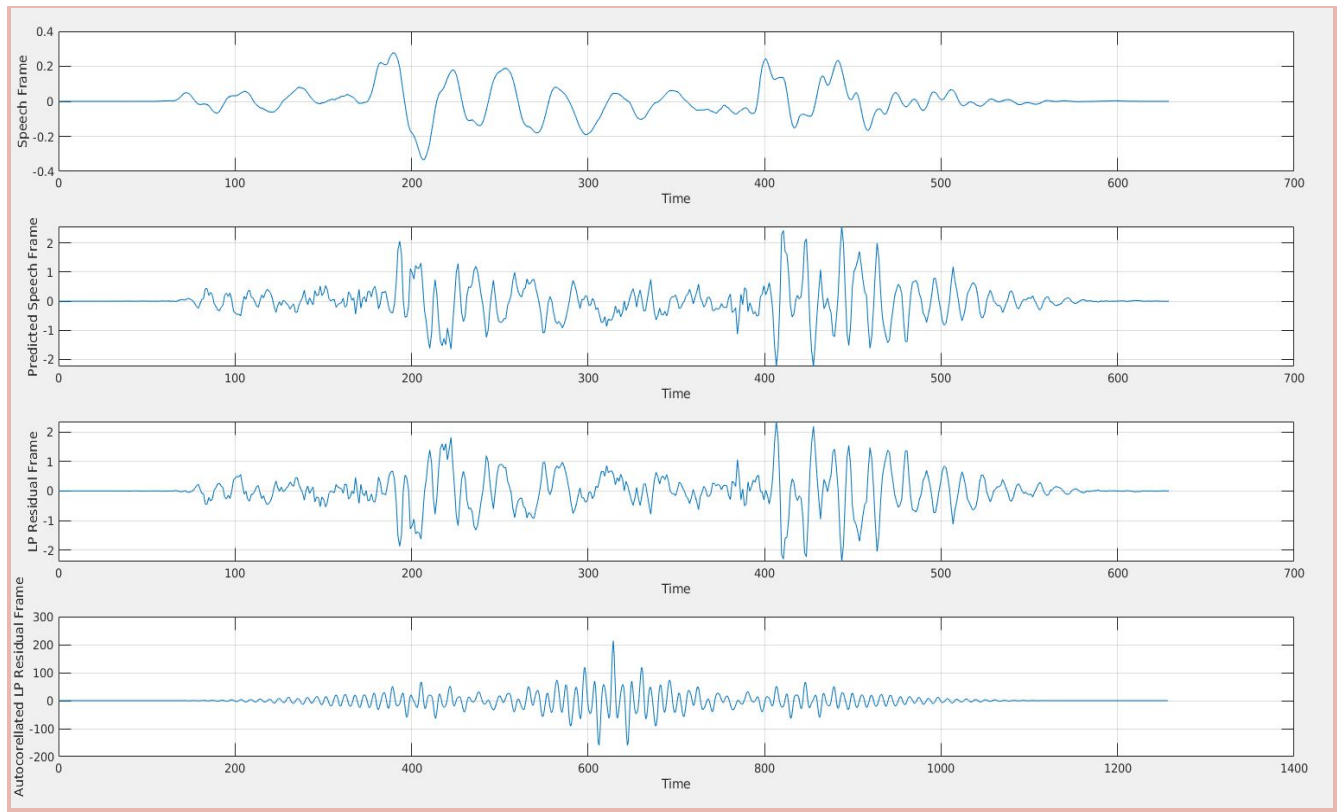


Observation :

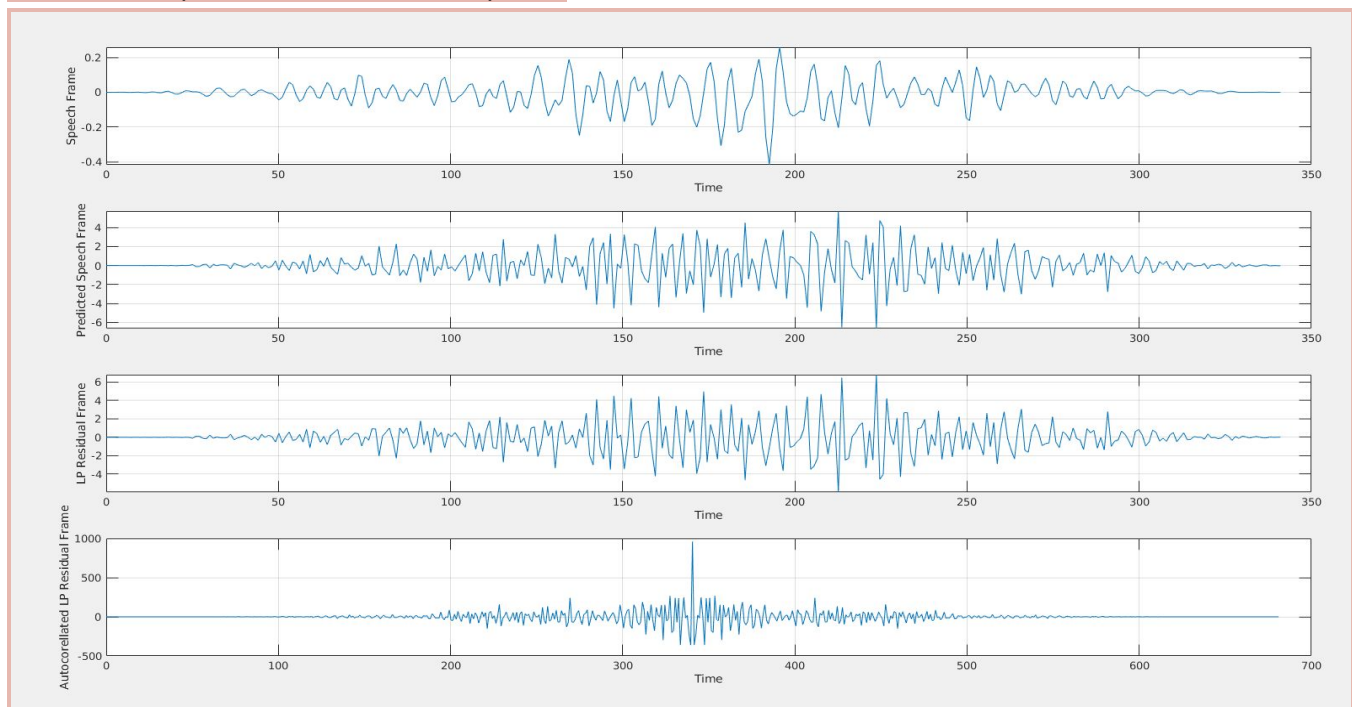
1. We can see that higher order of reconstruction leads to less erroneous reconstructed signal , here lpc order was set to 24 .
2. The 3rd plot is the LP residual , which is the difference between signal and filtered signal , the basic skeleton of the signal is restored here.
3. 4rth plot is the autocorrelation plot and here we need to compute the time difference between the highest peak and second peak , and the inverse of it will give us the pitch value. Here, the value I've got is :

$$A_vowel \text{ pitch} = 6.8571e+03 \text{ Hz}$$

Case 2_a) : Unvoiced Frame - (k_frame)



Case 2_b) Unvoiced Sample2



Observations:

Both of the above plots correspond to random unvoiced regions , though it may appear to have some periodic structures. The pitch estimation won't give the correct answer since the data is not periodic in nature .It has to be noted that for unvoiced speech signals the residual will be like random noise without any periodicity.

Question 2 :

a) Analysis-Synthesis based on Linear Prediction

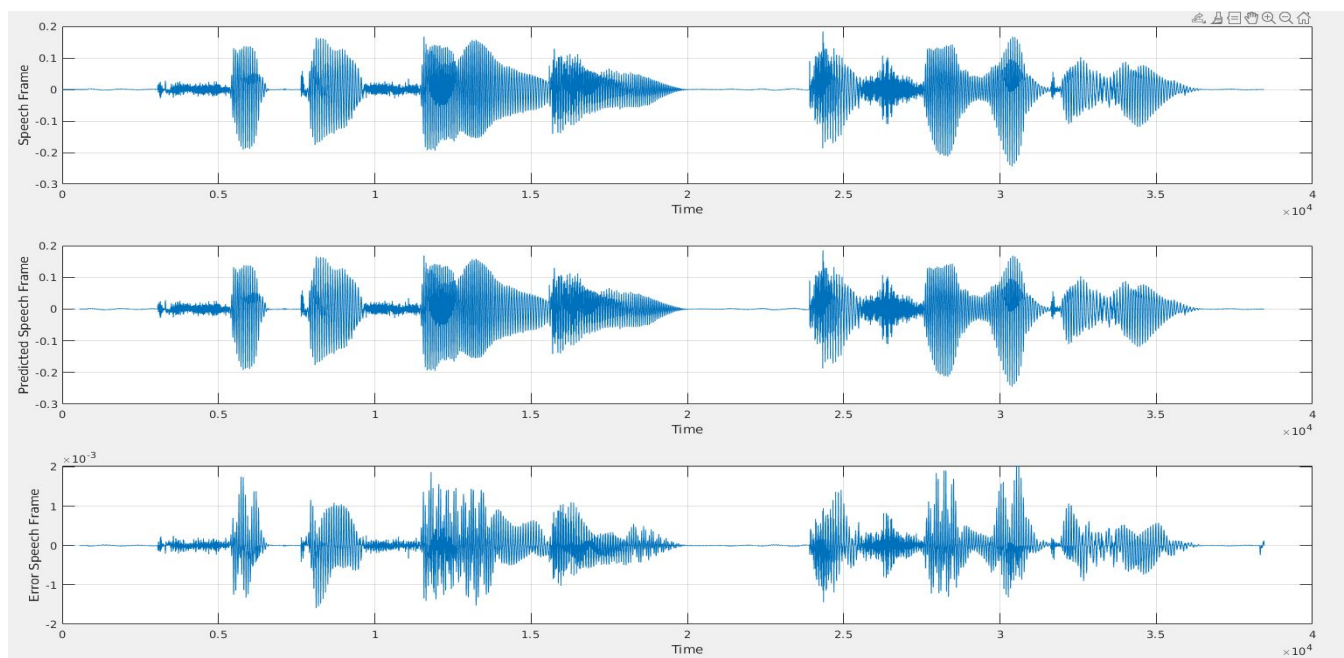
So, when we compute LP coefficients , we are computing the all-pole filter coefficients of speech and the LP residual provides us with glottal excitation .

For this question my input is - '**chris.wav**', sampled at 16KHz and its time duration is 2.4032 secs . The LPC Order is set to 24 and horizon is 10ms . *CODE FILE : lpc_somi.m*

I've plotted the entire speech signal , and reconstructed part and the error between these signals .

Output file is : **result_chris.wav** (PFA) [Fs=16000]

Analysis Graphs



Observation :

- Listening to 'result_chris.wav' and 'chris.wav' file , they sound very similar , and the difference is not detectable to me .
- Hence, I've created plots for speech signal , reconstructed signal and the error signal . The error signal is definitely non-zero and is very peaky .
- To understand more , I have created 'chris_err.wav' , to detect the missing sounds . But , after chris_err.wav , we can observe that the entire skeleton of the speech is preserved but it's a little noisy (static like noise) and less amplified. (Can be observed in the graph as well)
- Since , the order of LPC is high and we are performing short frame analysis and overlapping them , we are achieving a good reconstruction .

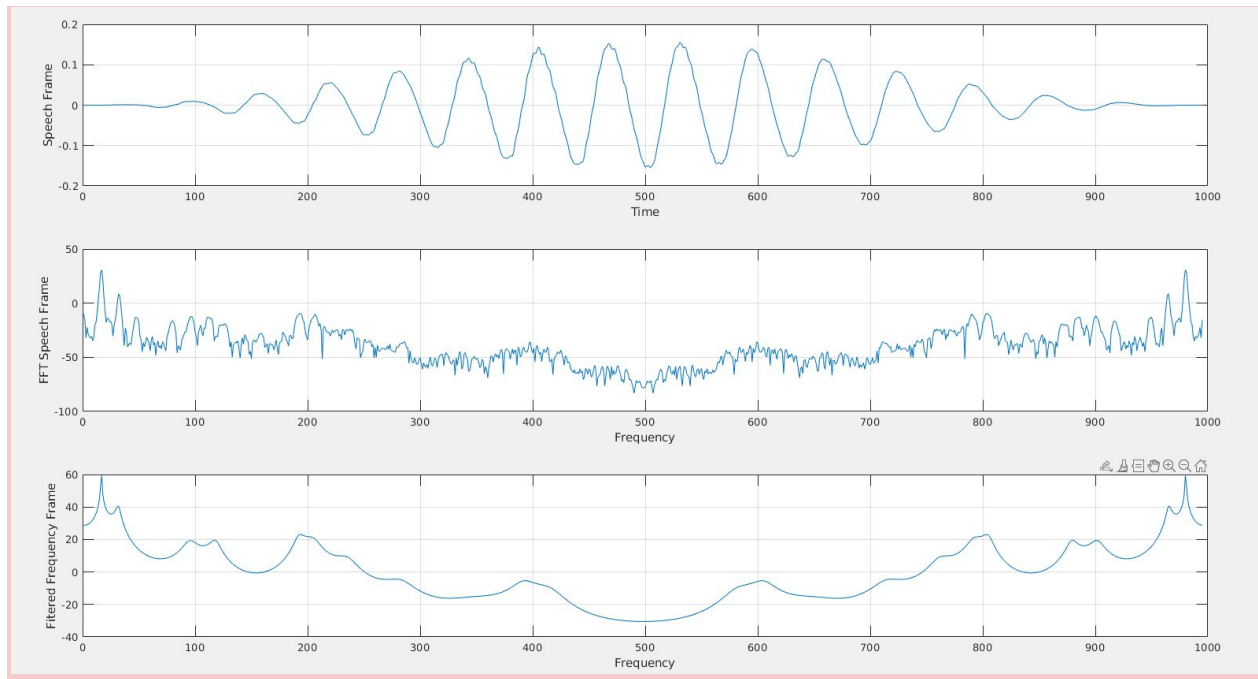
Q 2 b) In each analysis frame, compare the magnitude of the frequency response of the estimated linear prediction filter (using Matlab command `freqz`) with the magnitude of the Fourier Transform (Matlab command `fft`) of the speech frame.

A: --I've segmented out 2 small speech samples from - chris.wav , and they are chris_voiced.wav and chris_unvoiced.wav . We shall now compare their frequency responses and observe its behavior in unvoiced and voiced cases .

Note : All the FFT plots are plotted on dB scale .
(PFA `lpc_freq.m`)

Note wrt source code : Same as 2(a) , but framing parts and for loop + overlap code blocks are removed , and new plots

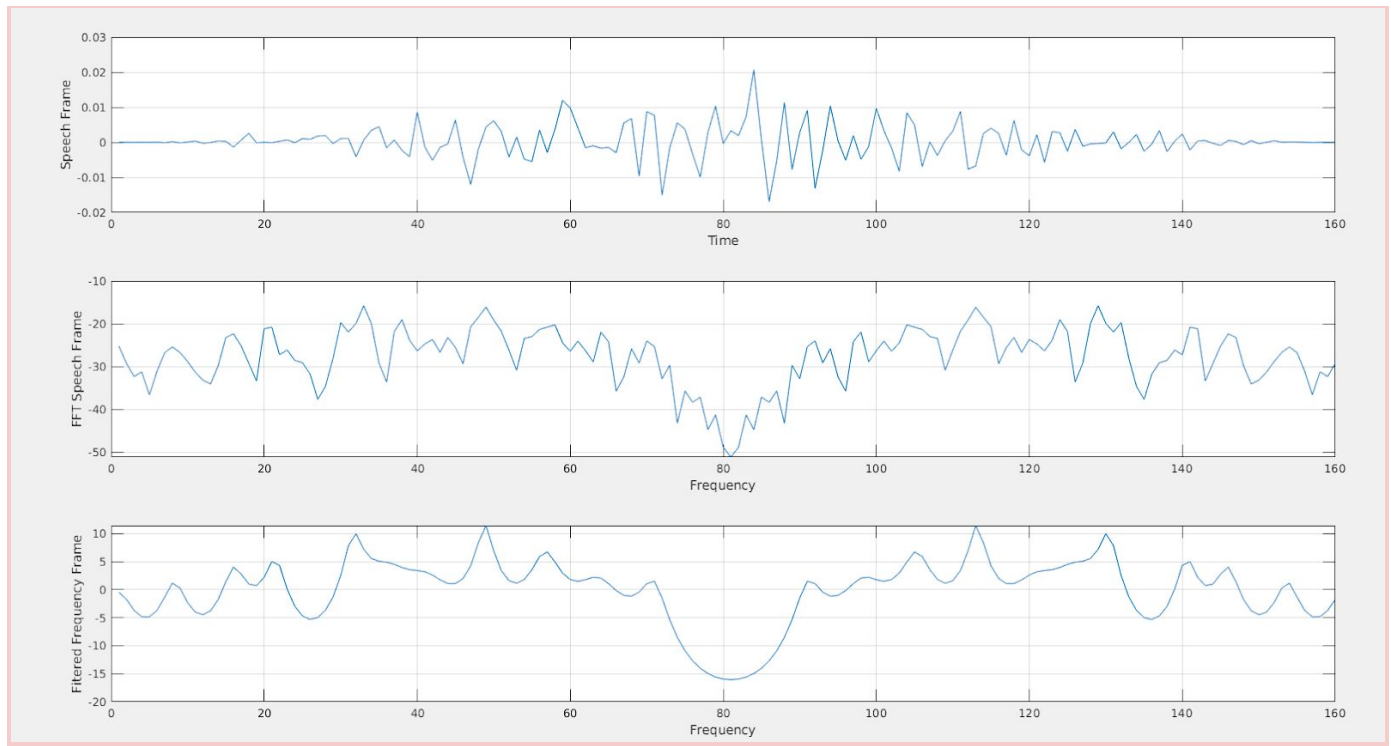
Case 1 : Voiced Part



Observation:

- First observation is that the first plot is a very smoothened version of the fft of speech signal , this matches with the theory --as LPC coefficients are supposed to act as glottal excitation . So, the speech part is noisy since it has other components with random noise(various sources) embedded into it . The 3rd plot acts like the backbone / outline of the 2nd graph .

Case 2 : Unvoiced Part



Observation:

1. We see that the output responses are similar to voiced speech sample case. The frequency response of the filtered signal is very smooth, and only the major peaks are highlighted here.

Q2 (b)

Vary the Order of LPC and write down observations .

File : lpc_plots.m

A: To quantify the quality of the synthesized signals, SNR can be used as the metric for understanding the impact number of LPC coefficients in the reconstruction process. SNR is defined as the ratio between the signal's energy and the noise's energy

in dB. The larger the SNR ratio, the better the sound quality. The SNR is given as :

$$SNR = 10 \log_{10} \left(\frac{\sum_n (x[n])^2}{\sum_n (x[n] - y[n])^2} \right),$$

Where $x[n]$ is the actual signal , and $y[n]$ is the reconstructed signal .

Again , we would be using the entire 'chris_voiced.wav' for analysis. (Smaller window)

Additional line in lpc_somi.m would be :

```
r = snr(X,Y)
```

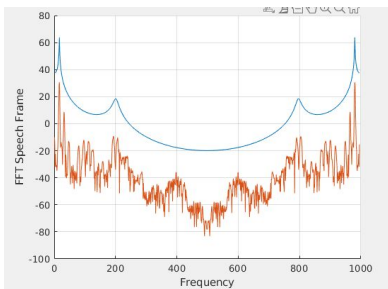
MATLAB , has an inbuilt function where it directly computes the squared ratio of signal (x) and noise(y) . Here , it would be :

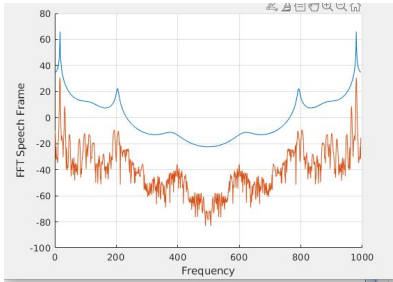
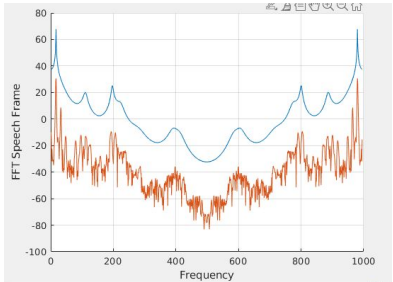
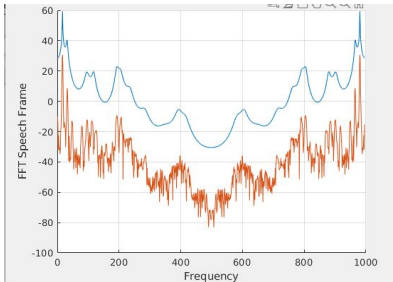
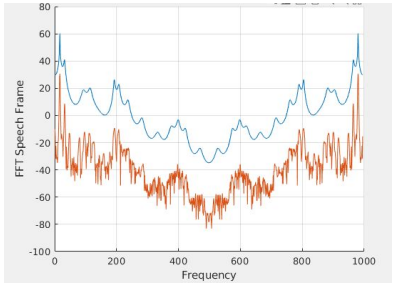
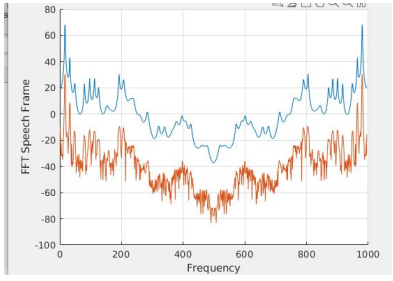
```
X=sig.
```

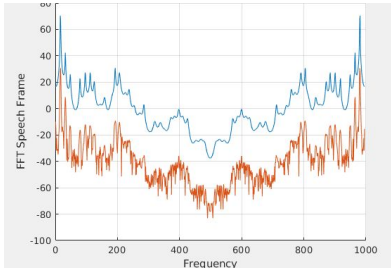
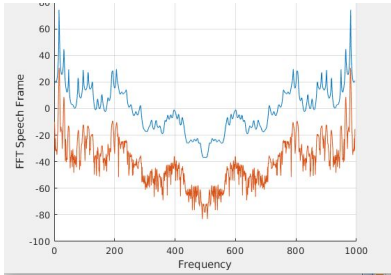
```
Y=sig-out .
```

For, this experiments I would vary the LPC in the order

[4 , 8 , 16 , (24) , 32 ,56, 72 ,100] --and tabulate the SNR .

LPC Order	Plot	SNR Value
4		0.8578

8	 <p>FFT Speech Frame</p> <p>Frequency</p>	1.1829
16	 <p>FFT Speech Frame</p> <p>Frequency</p>	1.0556
24	 <p>FFT Speech Frame</p> <p>Frequency</p>	2.7404
32	 <p>FFT Speech Frame</p> <p>Frequency</p>	2.5966
56	 <p>FFT Speech Frame</p> <p>Frequency</p>	2.7147

72		2.6485
100		2.5974

Observations :

1. Major observation is that the LPC Order dips after an optimal value , this is explainable as considering the combination of very old samples would increase the variance . It would add noise like components to the reconstruction as these samples as there is less correlation . Here , LPCorder = 56 is optimal choice , as the SNR value is peak and keeps decreasing from there .
2. Observing this frame plot , we know that LPC order is supposed to get glottal excitation , but over the increase in LPC order , we can see that it tries to mimic the speech segment . So, instead of extracting the envelope we will get noisy signal itself, its case of overfitting .