EEEM030 – Speech & Audio Processing

Implementation and Analysis of a Linear Predictive Speech Synthesizer

By Anisha Jubitha Kadathur Thameemul Ansari

Coursework for the Master of Science in

Computer Vision, Robotics and Machine Learning

from the

University of Surrey

Department of Electronic Engineering

Faculty of Engineering and Physical Sciences

University of Surrey

Guildford, Surrey, GU2 7JG, UK

Table of Contents

1 INTRODUCTION

Speech and Audio Processing is a case where digital processing methods are carried out on speech signals. The output obtained will be the digital representation of these signals and this process is called as Speech Synthesis. The process of speech synthesis involves the analysis and reconstruction of speech signals.

1.1 Description

This report focuses on the techniques for digital signal processing that can be used for speech synthesis. These are applied to sample vowel sounds which in turns estimates the respective synthesised vowel.

2 IMPLEMENTATIONS

The assignment was implemented on MATLAB with the help of the signal processing toolbox [1] [2]. Two vowel samples 'heed_f.wav' and 'hood_m.wav' were used. A quasi-stationary segment of 100ms length was clipped from both the vowels for further signal processing (refer Figure 1).

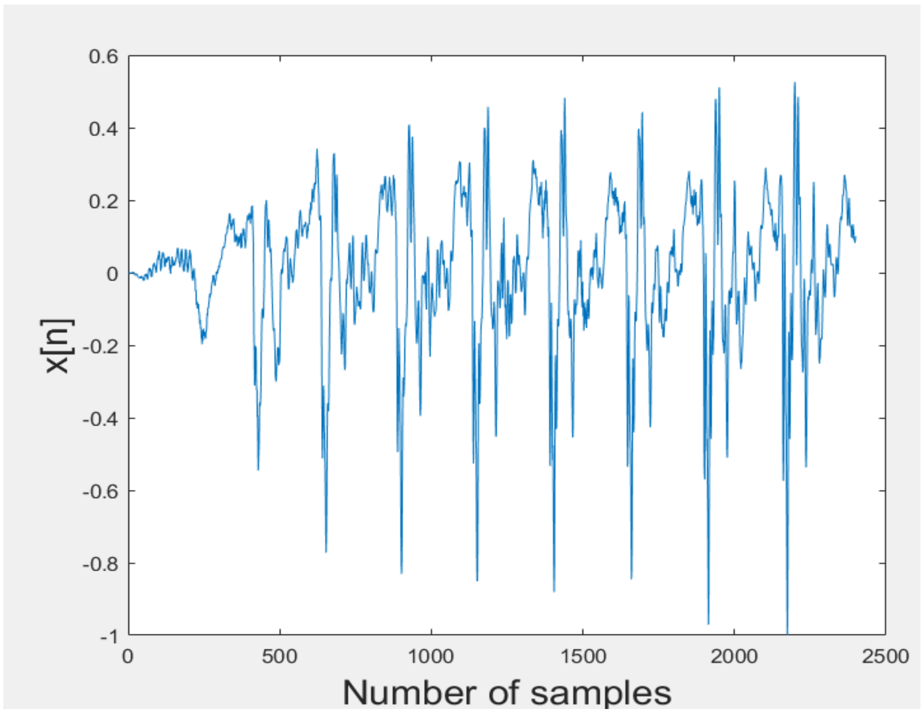


Figure 1: Segmented Speech Signal of hood_m.wav

2.1 Model Estimation

The vocal tract transfer function,

$$S(z) = H(z)\,U(z) \qquad \textbf{(1)}$$

where H(z) is the filter and U(z) is the input signal which is of periodic impulses. Using Linear predictive coding, the model estimation of H(z) was performed.

2.1.1 LPC Coefficient Estimation

For the estimation of the LPC coefficients from the provided speech waveforms, a function of AR (Autoregressive) modelling in MATLAB, namely **'lpc (speech segment, model order)'** [4] was chosen. The Fast Fourier transform function **'fft ()'** [3] was used in order to find the frequency responses of the segment and **'freqz ()'** [6] to plot the response of the LPC filter and the original speech signal. Table 1 below shows the LPC coefficients for both the vowel segments.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
F	1	-1.6797	1.0678	-0.2298	-0.2231	0.1260	-0.2231	-0.0826	0.1252	-0.0438	0.0485
M	1	-1.9301	0.9430	0.1416	0.5798	-0.9453	-0.4494	0.9662	0.1166	-0.4818	0.1734

	12	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**
F	0.0019	0.2076	-0.0899	0.2269	-0.2376	-0.0093	0.2569	-0.0501	-0.3132	0.1399
M	-0.6813	0.7972	0.0380	-0.2342	-0.2244	-0.0337	0.6025	-0.3188	-0.2282	0.2166

Table 1: LPC coefficients of heed_f (represented by F) and hood_m (M) for a Model Order of 20.

2.1.2 LPC spectrum & Model Order

The model order usually represents the formants or number of poles in the filter. It should be low so that it contains only spectral envelope and no spectral details. After careful consideration from a value of 1 to 50, a model order of 20 is chosen. Figure 2 illustrates the comparison of frequency response of the LPC Filter and the amplitude spectrum of the extracted vowel segment for an order of 20.

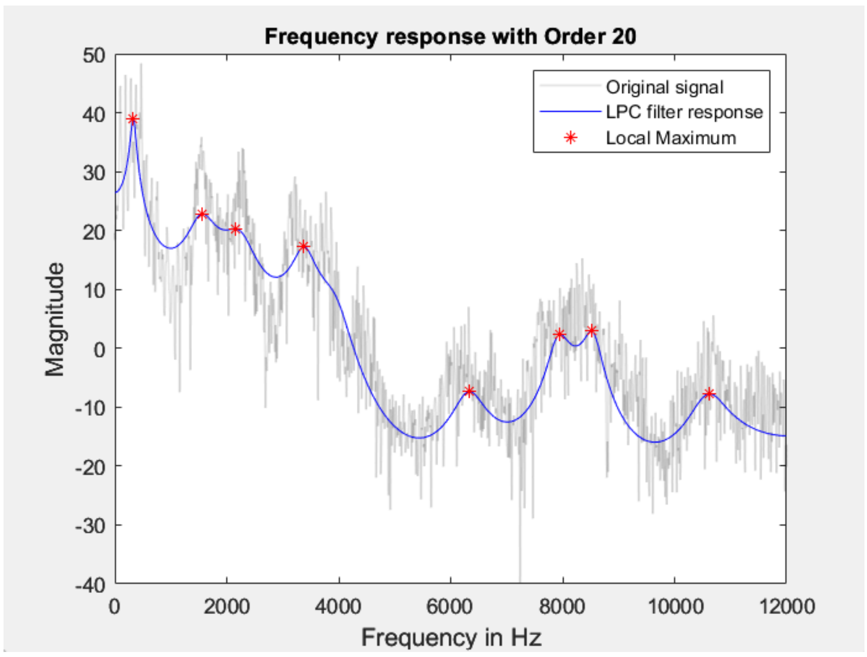


Figure 2: Comparison of Original signal and LPC filter response (in dB) with Order 20 for hood_m.wav

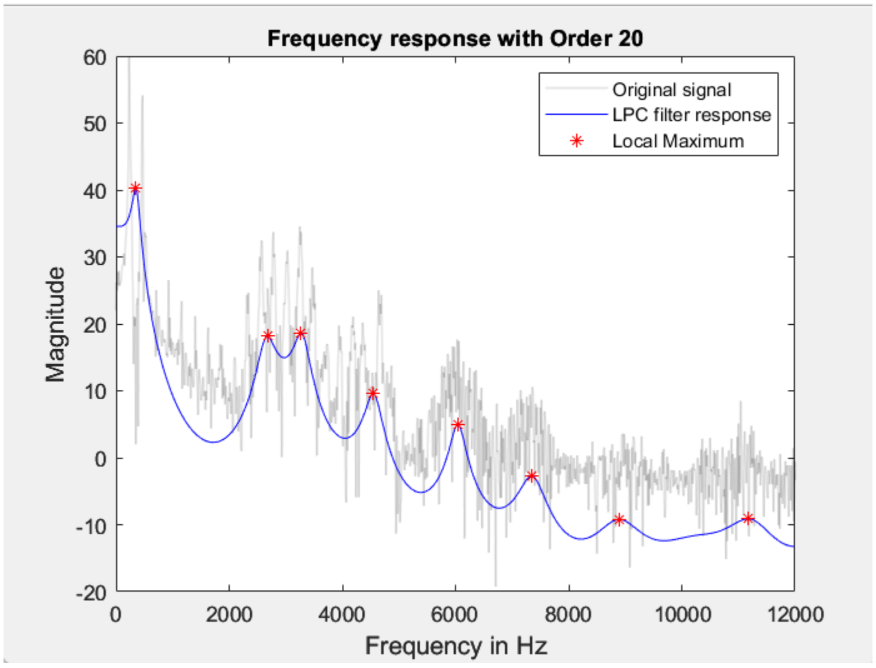


Figure 2: Comparison of Original signal and LPC filter response (in dB) with Order 20 for heed_f.wav

Here the formants frequencies are pointed out as the local maximum, or the peaks achieved by the

LPC filter frequency response spectrum.

From Figure 4, we can also conclude that the similarity of the original signal and the LPC filter response increases as the model order increases. When the model order is low, the filter does not identify the formants and so results in missing information which is mainly due to inadequate information on the spectral envelope. If the model order is very high, it results in the collection of spectral detail. This may result in a noisy output. Hence, the model order should be at an optimal level.

This can be better understood from Figure 4 that shows the frequency response of LPC filter during order variation. The order variation from 1 to 50 of ‘hood_m.wav’ is shown.

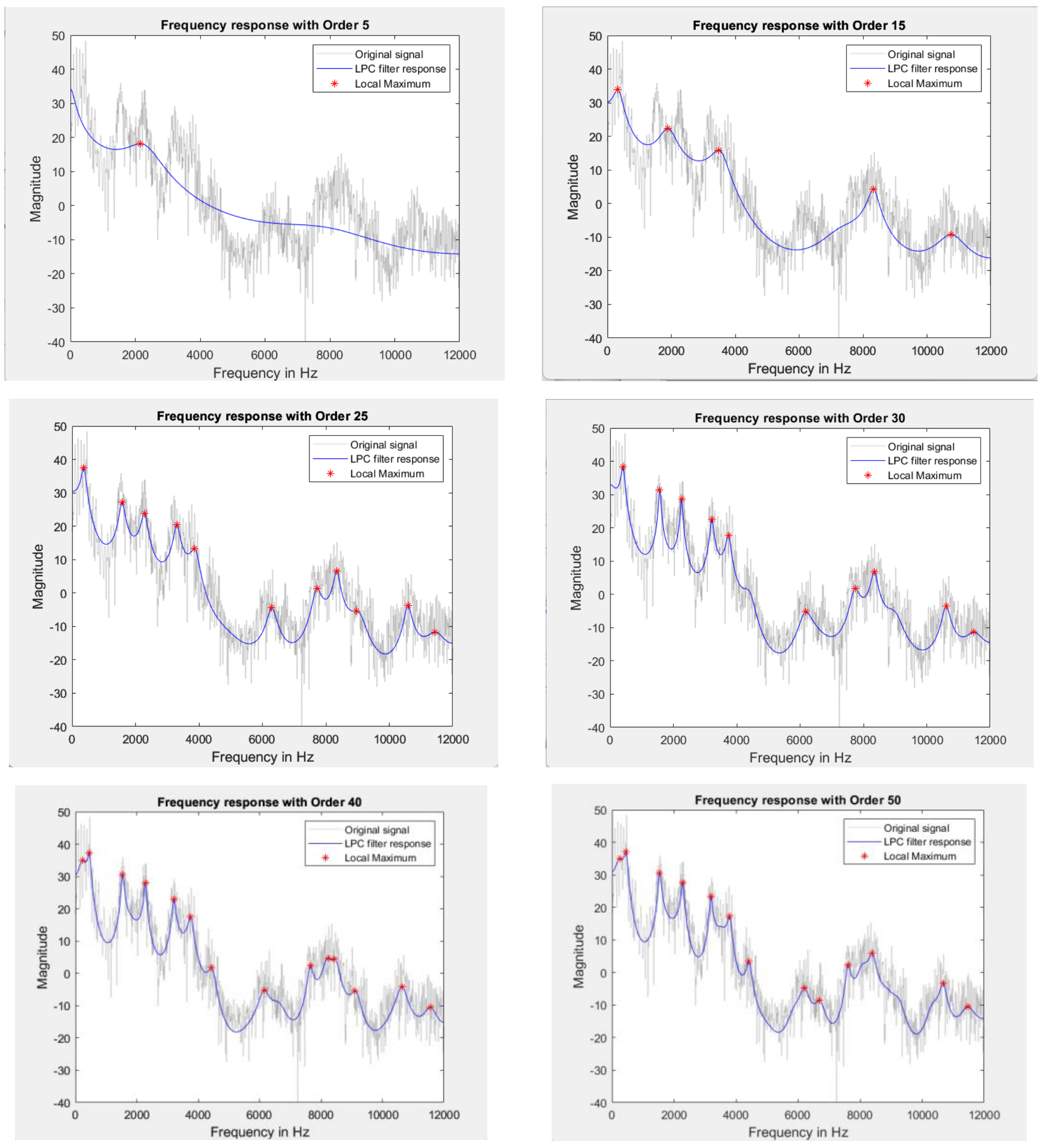


Figure 4: Illustration of increasing the model order on hood_m.wav

2.1.3 Formant Frequencies

Formant frequencies are high energy points. The first three formant frequencies are usually considered as the most important. We can also notice from Table 2 that the formant frequencies are less for the male while compared to the female sample. The first three formant frequencies of the male and female samples are noted below.

	f_1	**f_2**	**f_3**
heed_f	349.7086	2677.7685	3267.2773
hood_m	329.7252	1568.6928	2148.2098

Table 2: First 3 formant frequencies (in Hz) at model order 20.

2.1.4 Mean Fundamental Frequency

The mean fundamental frequency of the vowel segments can be calculated as the average difference between two local maximums of a speech signal which can then be converted to the time-period.

	F_0
heed_f	230
hood_m	100

Table 3: Mean fundamental frequencies (in Hz)

2.1.5 Segment Length Variation

In this section, we explore the change in the reconstructed speech segment for varying segment lengths (Figure 5). From the figure, we can observe that as the segment length rises, peaks in the signal increases mostly, in the middle section of the spectrum while low segment lengths exhibit a smooth spectrum. The below experiment is made with the vowel segment ‘heed_f.wav’.

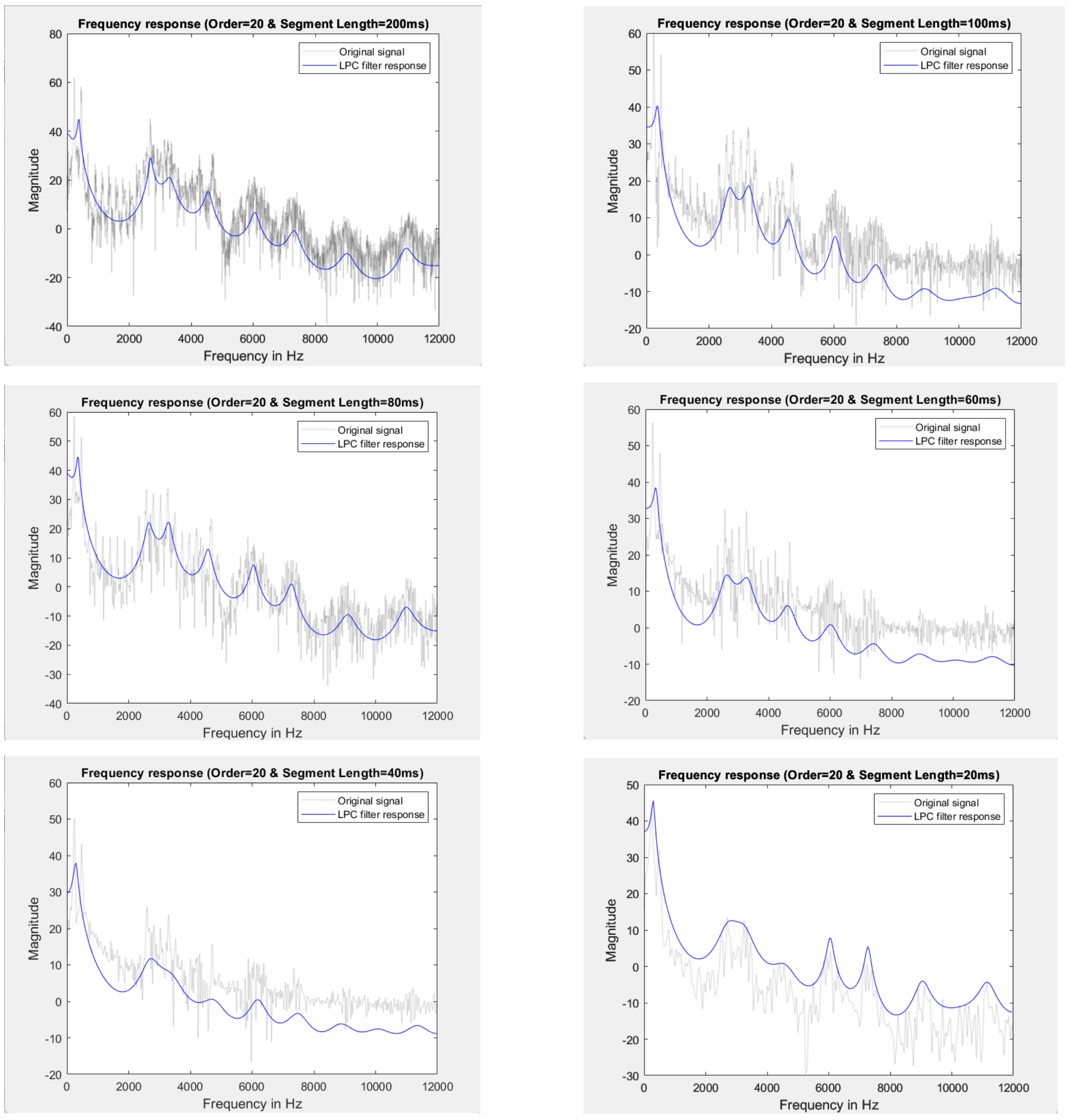


Figure 5: Variation in spectrum of LPC filter and original signal at different segment lengths.

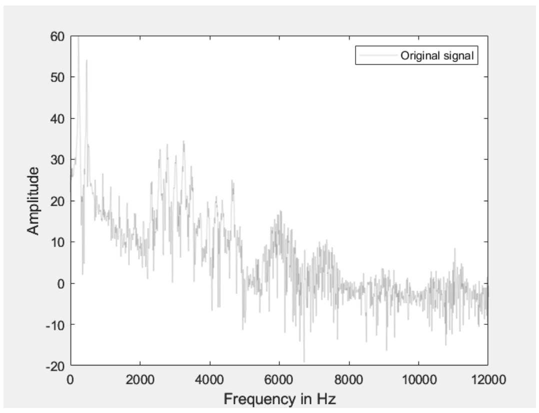
2.2 Synthesis

The synthesised speech signal is obtained using the fundamental frequency calculated in the previous section 2.1.5. With the aid of the following equation, this value is subsequently transformed to a time-period.

$$T = \frac{1}{F_s} \quad \textbf{(2)}$$

where F_s is the sampling frequency.

An impulse train is first generated by referencing the obtained fundamental frequency and time period followed by the speech synthesis using the LPC filter function, **'filter ( )'** [5]in MATLAB. Here the convolution of the LPC filter and the impulse train happens. The output generated from the LPC filter is synthesized such that the spectral envelope is project by limiting the spectral details in the spectrum. Since a model order of 20 is considered, smaller peaks are removed to accommodate the required number of frequency components which is 20. Even though, some of the smaller peaks are ignored, the reconstructed signal will not be affected much and is similar to human speech.



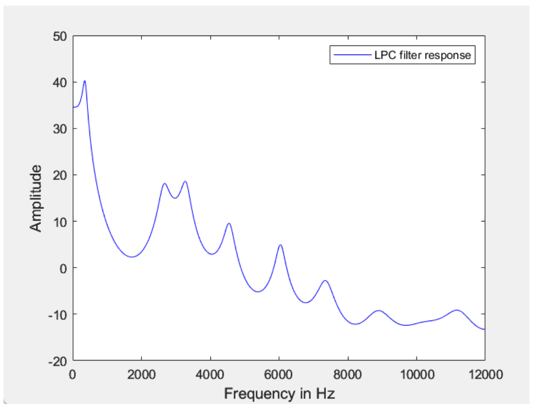


Figure 6: Spectrogram of the original and synthesised speech segment of hood_m for model order 20.

By listening to the audio clip of the synthesized speech sound, the similarity can be observed. But when the model order is increased more than 20, there is little change in the spectrum but does not resemble the original speech quality.

3 Informal Subjective Assessment

Based on the observations from the experiment conducted, a synthesized speech signal can be produced by the convolution of an impulse signal referencing the input frequency and the LPC filter output.

The formant frequencies in the speech spectrums are high energy peaks produced. The first three of them are highly important. This formant frequency gets modified when the LPC model order is changed. The model order of the LPC filter should be at an optimal point such that the speech details of the vowel segment does not exist. This gives preference to the speech envelope present in the signal. As the model order rises, the formants also increase depending on the speech segment. If the model order is very high, the peaks or formants will be high. This can also display some spectral details. Low model order shows lesser peaks thus losing important data with a smooth spectrum. The segment length of a spectrum can also be varied. Larger lengths exhibit more peaks.

The below table compares the results of the synthesised speech signal with varying model order. The assessment is based on a rating of 1 to 5 on the speech produced where 1 stand for poor quality and 5 is of good quality.

	Segment Length	Order 10	Order 20	Order 30	Order 50	Order 200
hood_m	100 ms	1	3	5	5	4
	60 ms	1	3	4	5	3
heed_f	200 ms	1	4	4	3	2
	100 ms	2	5	5	4	3
	60 ms	1	3	4	5	4

Table 4: Subjective Assessment of Synthesized signal by varying model order & segment length

4 Conclusion

In this assignment, various speech samples were analysed and reconstructed. The synthesis of the speech signal was used to minimise the spectral details and project the spectral envelope in order to get a good quality signal. For this purpose, a source filter model for speech was explored using the Linear predictive coding filter.

Required parameters like LPC coefficients, Formant and Fundamental Frequency etc were observed with the help of plots and by varying model order of the filter and length of the segment. An optimal point model order and other parameters were presented.

5 References

[1] https://www.phon.ucl.ac.uk/courses/spsci/matlab/lect10.html

[2] https://uk.mathworks.com/help/signal/index.html?s_tid=CRUX_lftnav

[3] https://uk.mathworks.com/help/matlab/ref/fft.html#d124e442489

[4] https://uk.mathworks.com/help/signal/ref/lpc.html?s_tid=doc_ta

[5] https://uk.mathworks.com/help/matlab/ref/filter.html?s_tid=doc_ta

[6] https://uk.mathworks.com/help/signal/ref/freqz.html

6 Appendix

```
clc; close all; clear all;


%% Signal Segmentation

[input, Fs] = audioread('D:\OneDrive\Personal\Surrey\SurreyLearn\Sem2_SPAP\speech
samples\speech\heed_f.wav');
samples = [1,0.1*Fs];
[input, Fs] = audioread('D:\OneDrive\Personal\Surrey\SurreyLearn\Sem2_SPAP\speech
samples\speech\heed_f.wav',samples);

plot(input);
xlabel('Number of samples','FontSize', 18);
ylabel('x[n]','FontSize', 18);
sound(input,Fs);


%% Speech signal FFT

signal_length = length(input);
f = Fs*(0:(signal_length/2))/signal_length;
fft_output = fft(input);
P1 = abs(fft_output);
P = P1(1:signal_length/2+1);
P(2:end-1) = 2*P(2:end-1);

plot(f,P)
title("Single-Sided amplitude spectrum")
xlabel("frequency")
ylabel("|P(f)|")

freq = Fs*(0:(signal_length/2))/signal_length;
values = P;


%% LPC Filter

lpc_output = lpc(input, 20);  % LPC Coeffecients
[h_out, filter_frequency] = freqz(1, lpc_output, length(freq), Fs);

[filter_value, filter_frequency] = freqz(1, lpc_output, length(freq), Fs);
filter_dbvalue = 20*log10(abs(filter_value));

original_plot = plot(freq, 20*log10(abs(values)), 'black');
original_plot.Color(4) = 0.1;
original_plot.LineWidth = 1;
hold on
```

```
lpc_plot = plot(filter_frequency, filter_dbvalue, 'b');
hold off;


%% Estimation of the Formant Frequency

i = 1;
n = 3;
formant_freq = zeros(1,3);  %Formant Frequencies
formant_amplitude = zeros(1,3);
while(i<=3)
   if(filter_dbvalue(n-1)<filter_dbvalue(n) && filter_dbvalue(n)>filter_dbvalue(n+1))
       formant_freq(i) = filter_frequency(n);
       formant_amplitude(i) = filter_dbvalue(n);
       i = i+1;
   end
   if(n<length(filter_frequency))
       n = n+1;
   end
end

original_plot = plot(freq, 20*log10(abs(values)), 'black');
original_plot.Color(4) = 0.1;
original_plot.LineWidth = 1;
hold on

lpc_plot = plot(filter_frequency, filter_dbvalue, 'b');
hold on

plot(formant_freq,formant_amplitude,'r*');
text(formant_freq(1),formant_amplitude(1),{num2str(formant_freq(1))});
text(formant_freq(2),formant_amplitude(2),{num2str(formant_freq(2))});
text(formant_freq(3),formant_amplitude(3),{num2str(formant_freq(3))});
hold off


%% Estimation of the Fundamental frequencies

i = 1;
n = 3;
fundamental_freq = 0;
fundamental_amplitude = 0;
for r = (1:length(freq))
   if(values(r)<80)
       values(r) = 0;
   end
end

while(i<2)
```

```
    if(values(n-1)<values(n) && values(n)>values(n+1))
        fundamental_freq = freq(n);  % Fundamental Frequencies
        fundamental_amplitude = values(n);
        i = i+1;
    end

    if(n<length(freq))
        n = n+1;
    end
end


plot(freq,values);
title('Segmented sound representation in Frequency Domain','FontSize', 20)
xlabel('Frequency','FontSize', 12)
ylabel('Amplitude','FontSize', 12)
hold on

plot(fundamental_freq,fundamental_amplitude,'r*');
text(fundamental_freq,fundamental_amplitude,{'   ',num2str(fundamental_freq)});
disp('fundamental_vector');
disp(fundamental_freq);
hold off;


% Generating a Periodicc Impulse Train

Synth_length = 0.2;
Ts = 1/Fs;
impulse = zeros(1,Fs*Synth_length)
Timpluse = 1/fundamental_freq;
impulse(1:round(Timpluse/Ts):end)=1;    % Periodic Impulse signal
vector = (1:Fs*Synth_length)*Ts;

disp('Impulse_train');
disp(round(Timpluse/Ts));


% Speech Synthesis

Synthesised_speech = filter(1,lpc_output,impulse); % Synthesised Speech Output

figure(5)
plot(vector,Synthesised_speech)
title('Synthesised Speech Signal','FontSize', 20)
sound(Synthesised_speech,Fs); %play the speech synthesis
audiowrite('D:\OneDrive\Personal\Surrey\SurreyLearn\Sem2_SPAP\Coursework_Output\Au
dio Output\heed_f.wav',Synthesised_speech,Fs);
```