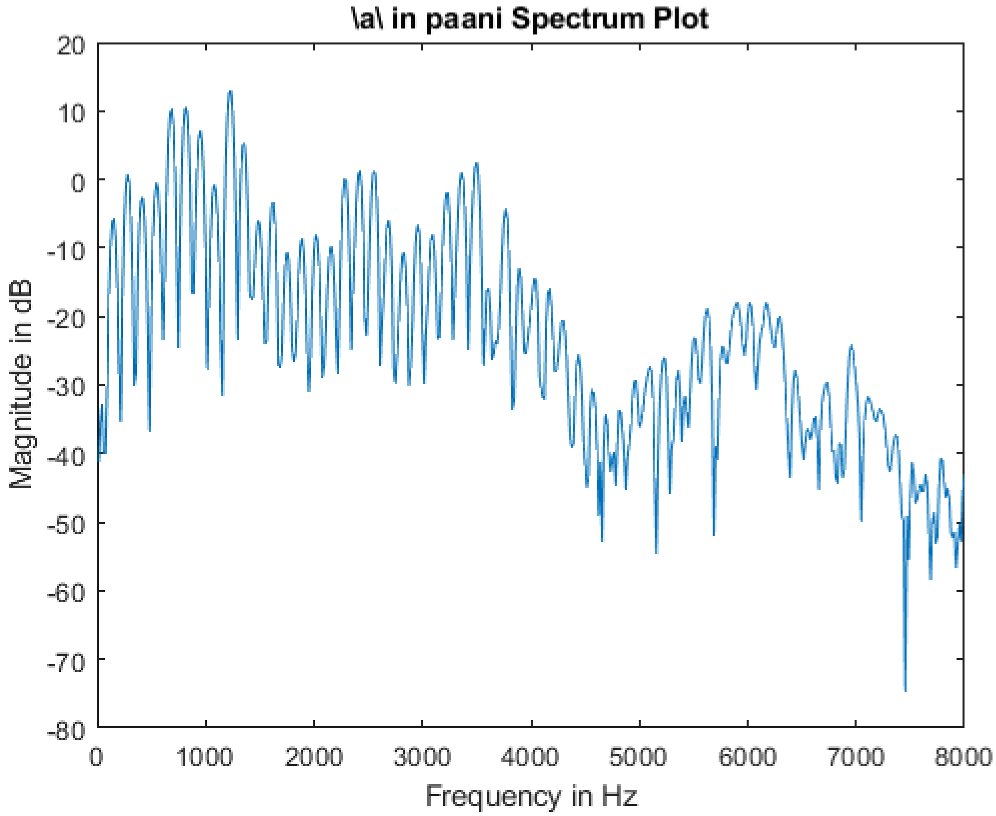
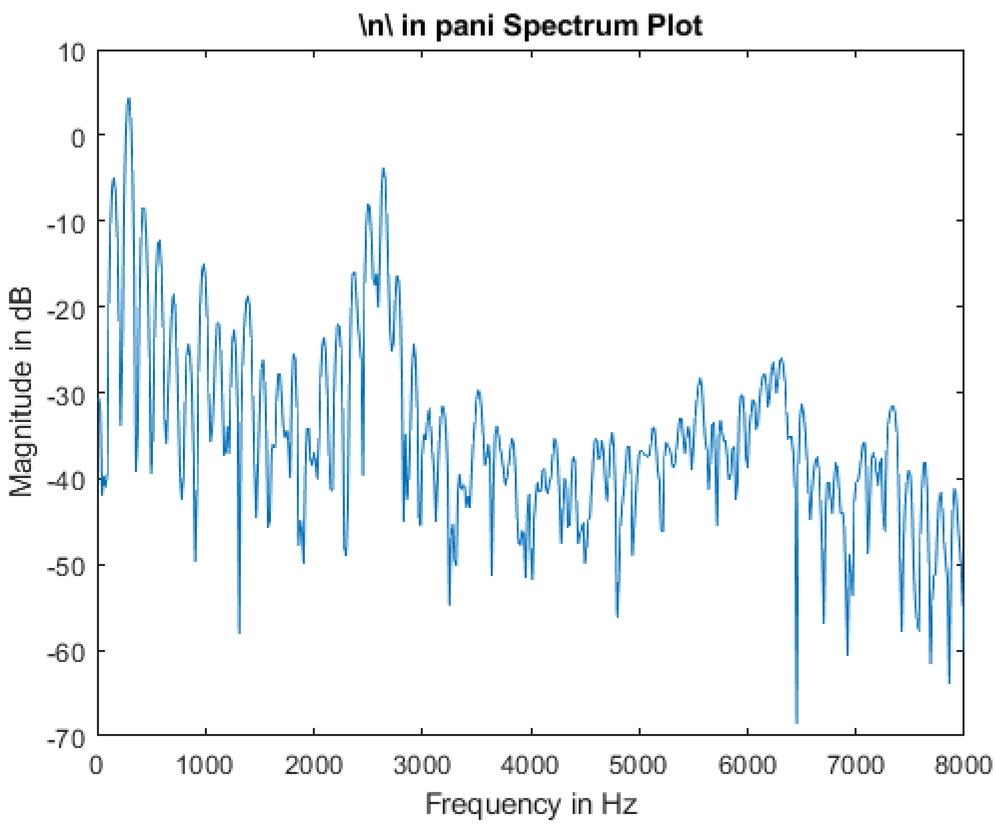
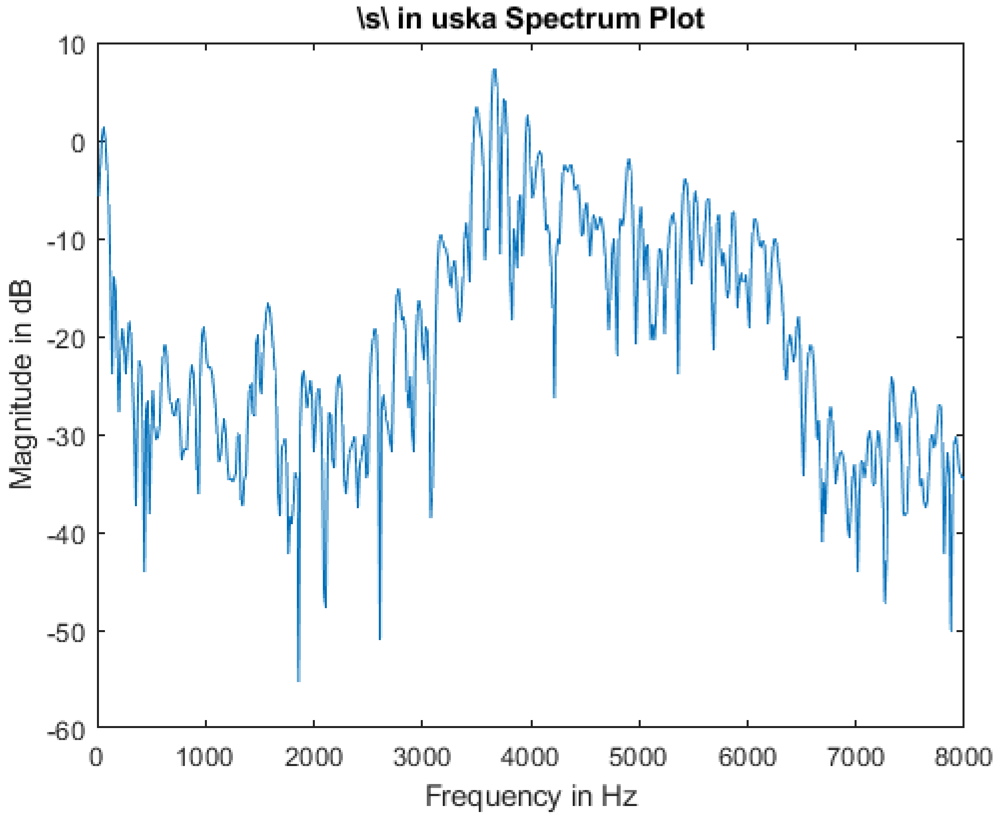
Table of Contents

Q1

```
% For \a\ in pani
output_a = nb_spectrum('a_in_pani.wav', 1); %Voiced
figure
plot((1:512)*8000/512, output_a)
title('\a\ in paani Spectrum Plot')
xlabel('Frequency in Hz')
ylabel('Magnitude in dB')

% For \n\ in pani
output_n = nb_spectrum('n_in_pani.wav', 1); %Voiced
figure
plot((1:512)*8000/512, output_n)
title('\n\ in pani Spectrum Plot')
xlabel('Frequency in Hz')
ylabel('Magnitude in dB')

% For \s\ in uska
output_s = nb_spectrum('s_in_uska.wav', 0); %Unvoiced
figure
plot((1:512)*8000/512, output_s)
title('\s\ in uska Spectrum Plot')
xlabel('Frequency in Hz')
ylabel('Magnitude in dB')
```


\a\ in paani Spectrum Plot
Magnitude in dB
Frequency in Hz
20
10
0
-10
-20
-30
-40
-50
-60
-70
-80
0
1000
2000
3000
4000
5000
6000
7000
8000

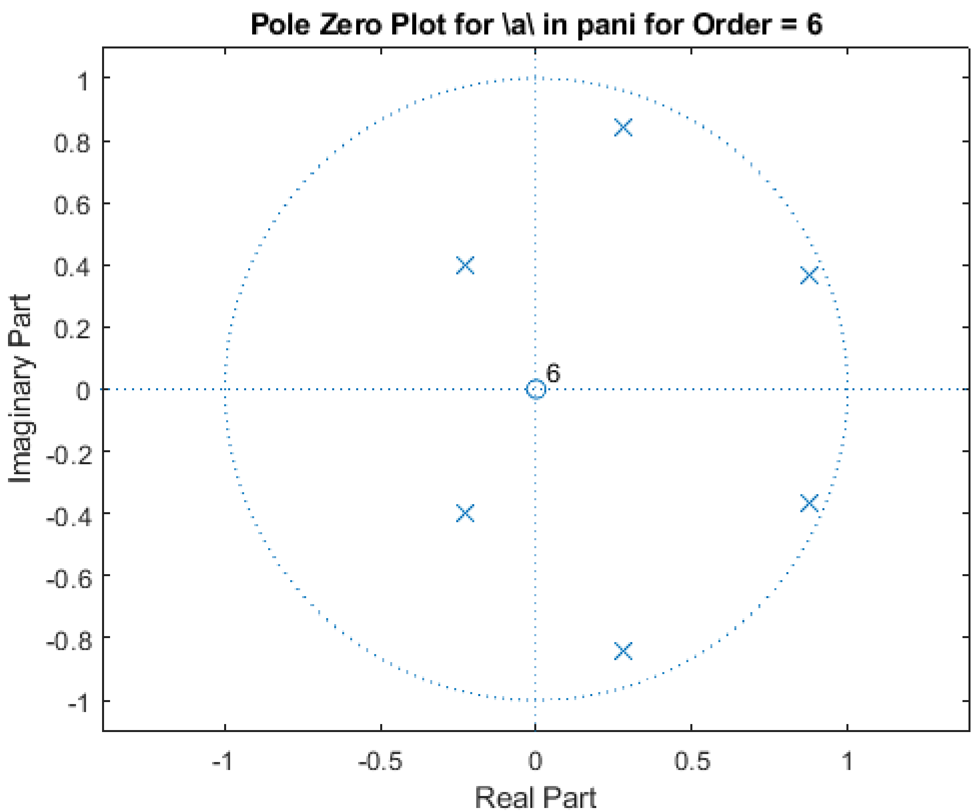

\n\ in pani Spectrum Plot
Magnitude in dB
Frequency in Hz
10
0
-10
-20
-30
-40
-50
-60
-70
0
1000
2000
3000
4000
5000
6000
7000
8000

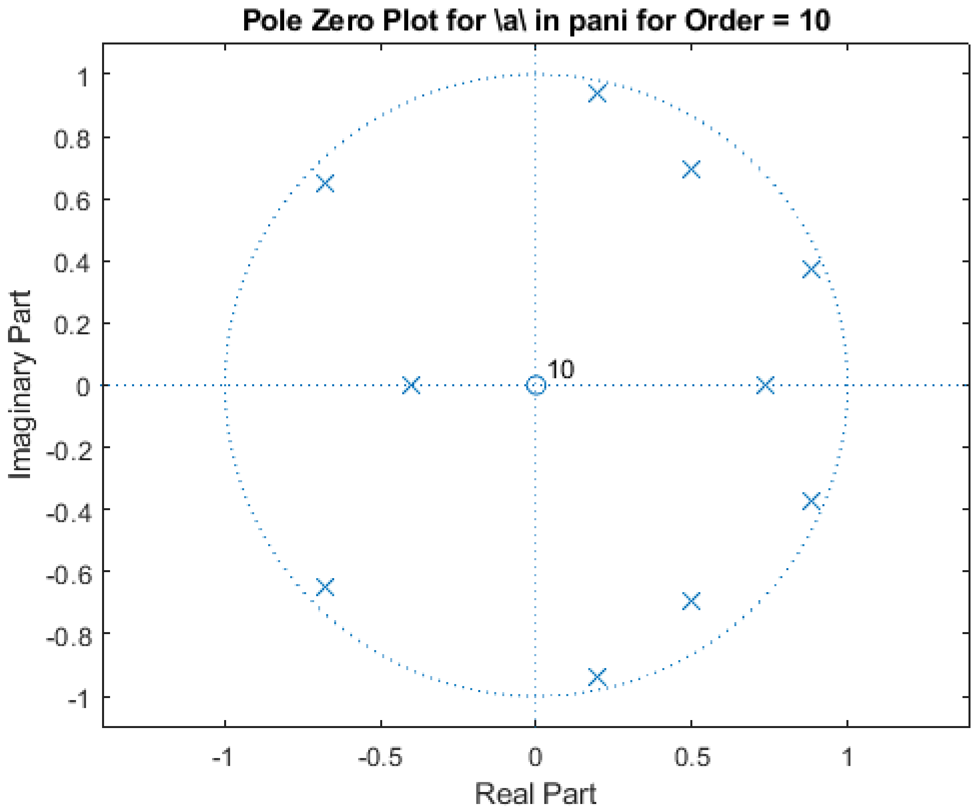


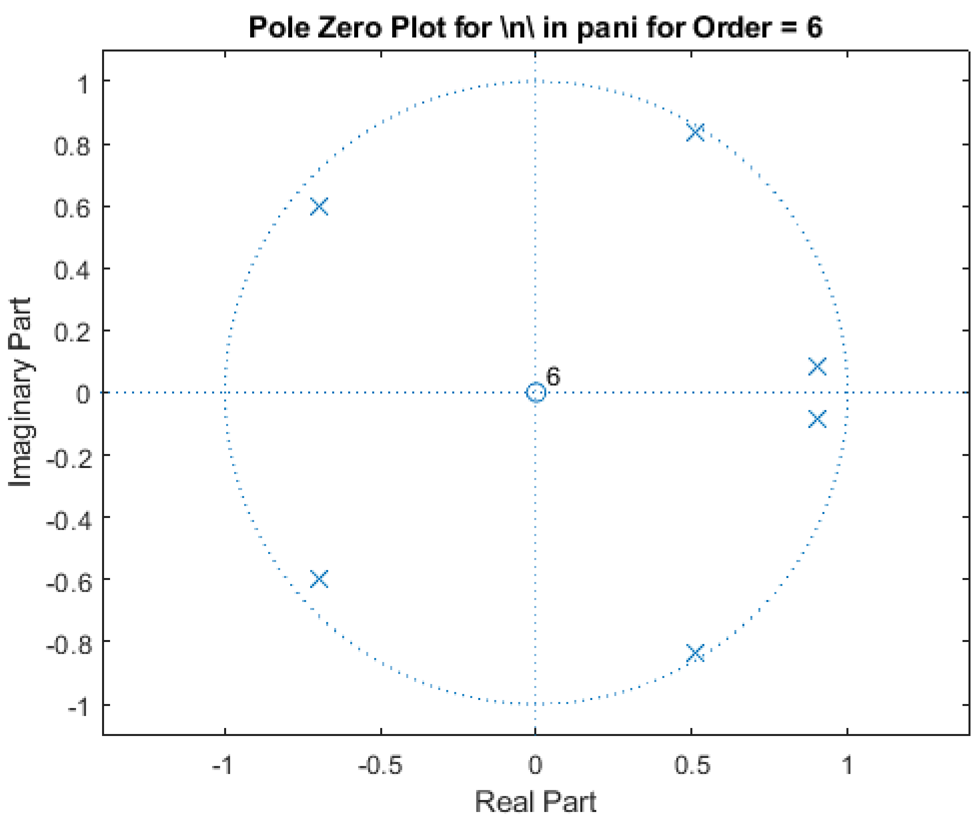
Q2 Part A

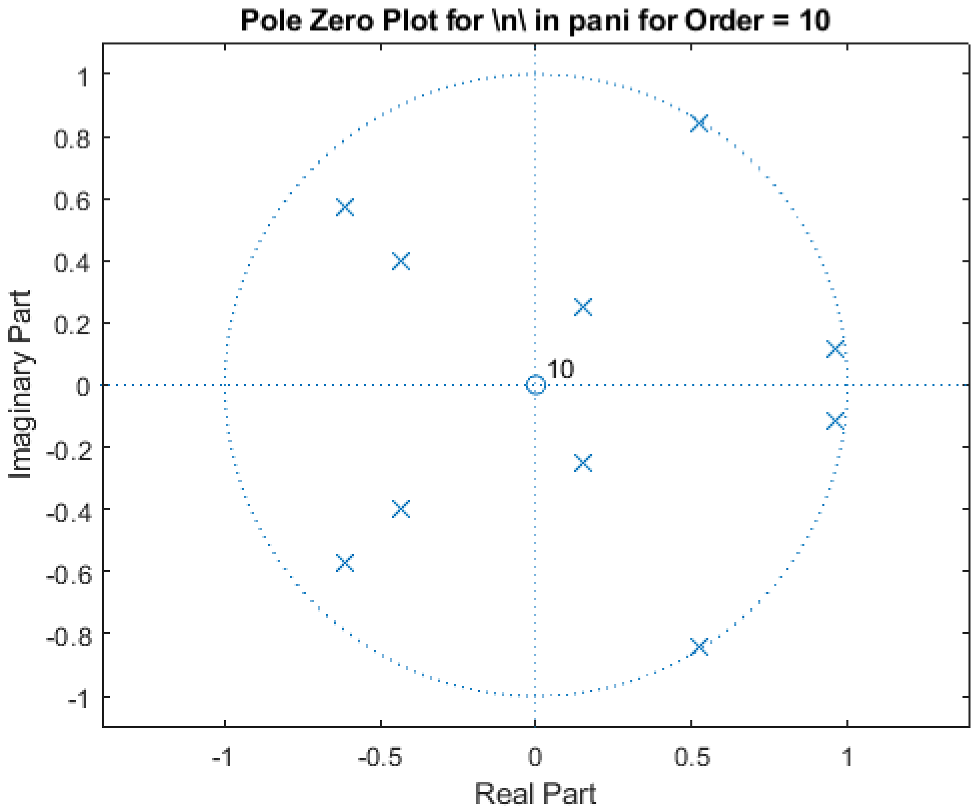
```
% For \a\ in pani
%   For p=6
[~, num_a_6, den_a_6] = LP_Q2('a_in_pani.wav', 6, 1);
figure
zplane(num_a_6, den_a_6)
title('Pole Zero Plot for \a\ in pani for Order = 6')
%   For p=10
[~, num_a_10, den_a_10] = LP_Q2('a_in_pani.wav', 10, 1);
figure
zplane(num_a_10, den_a_10)
title('Pole Zero Plot for \a\ in pani for Order = 10')

% For \n\ in pani
%   For p=6
[~, num_a_6, den_a_6] = LP_Q2('n_in_pani.wav', 6, 1);
figure
zplane(num_a_6, den_a_6)
title('Pole Zero Plot for \n\ in pani for Order = 6')
%   For p=10
[~, num_a_10, den_a_10] = LP_Q2('n_in_pani.wav', 10, 1);
figure
zplane(num_a_10, den_a_10)
title('Pole Zero Plot for \n\ in pani for Order = 10')
```

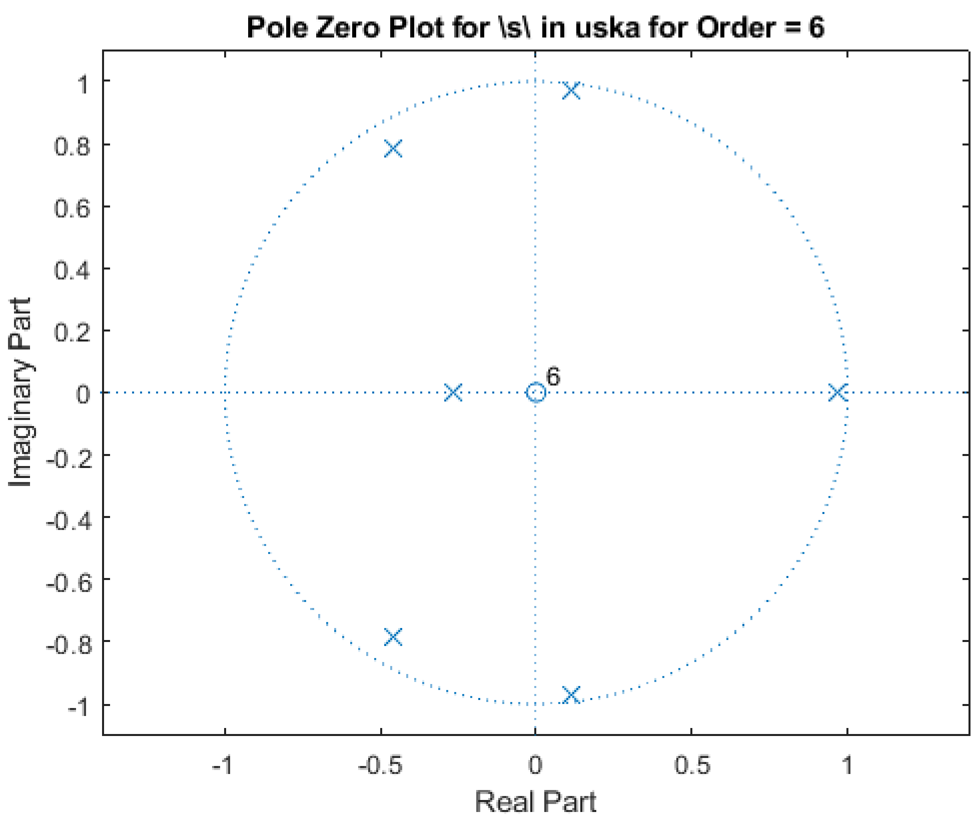
```
% For \s\ in uska
%   For p=6
[~, num_a_6, den_a_6] = LP_Q2('s_in_uska.wav', 6, 0);
figure
zplane(num_a_6, den_a_6)
title('Pole Zero Plot for \s\ in uska for Order = 6')
%   For p=10
[~, num_a_10, den_a_10] = LP_Q2('s_in_uska.wav', 10, 0);
figure
zplane(num_a_10, den_a_10)
title('Pole Zero Plot for \s\ in uska for Order = 10')
```

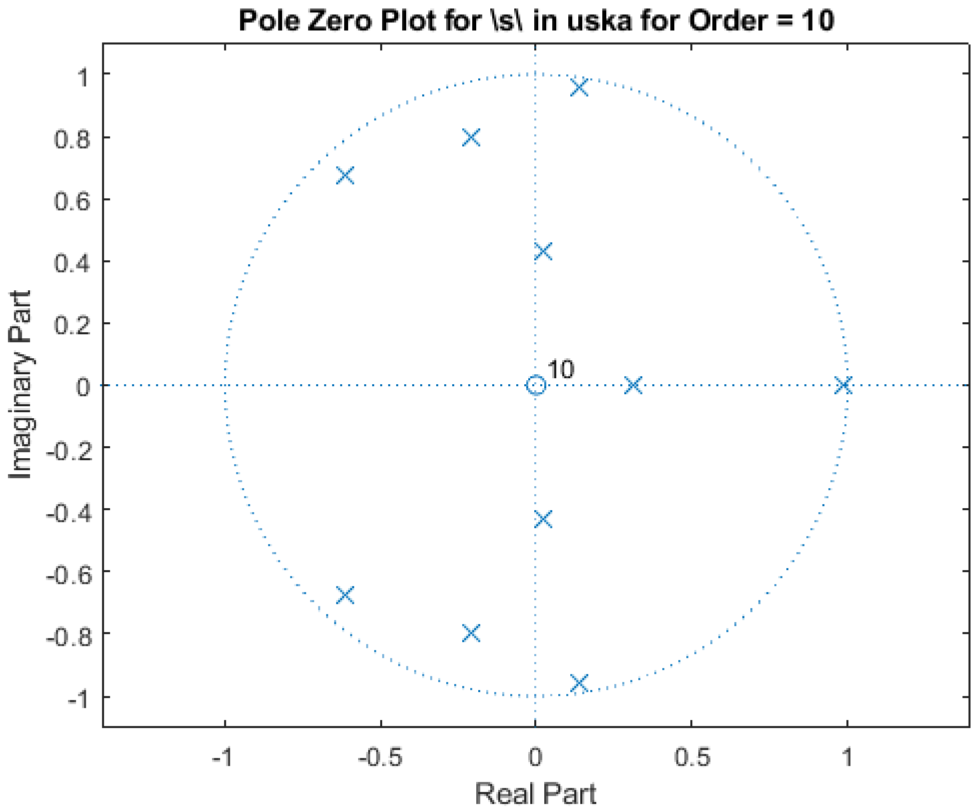



Pole Zero Plot for \a\ in pani for Order = 10
Imaginary Part
Real Part
10
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1
-0.5
0
0.5
1


Pole Zero Plot for \n\ in pani for Order = 6
Imaginary Part
Real Part
6
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1
-0.5
0
0.5
1


Pole Zero Plot for \n\ in pani for Order = 10
Imaginary Part
Real Part
10
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.5
0.5


Pole Zero Plot for \s\ in uska for Order = 6
Imaginary Part
Real Part
6
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-0.5
0.5

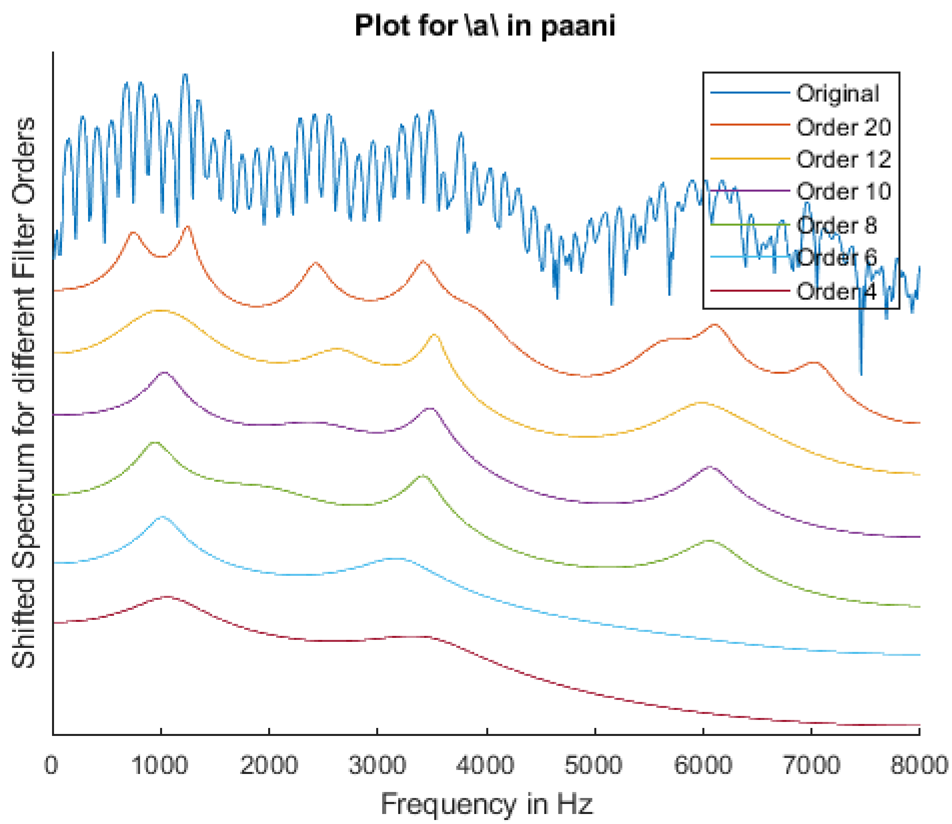


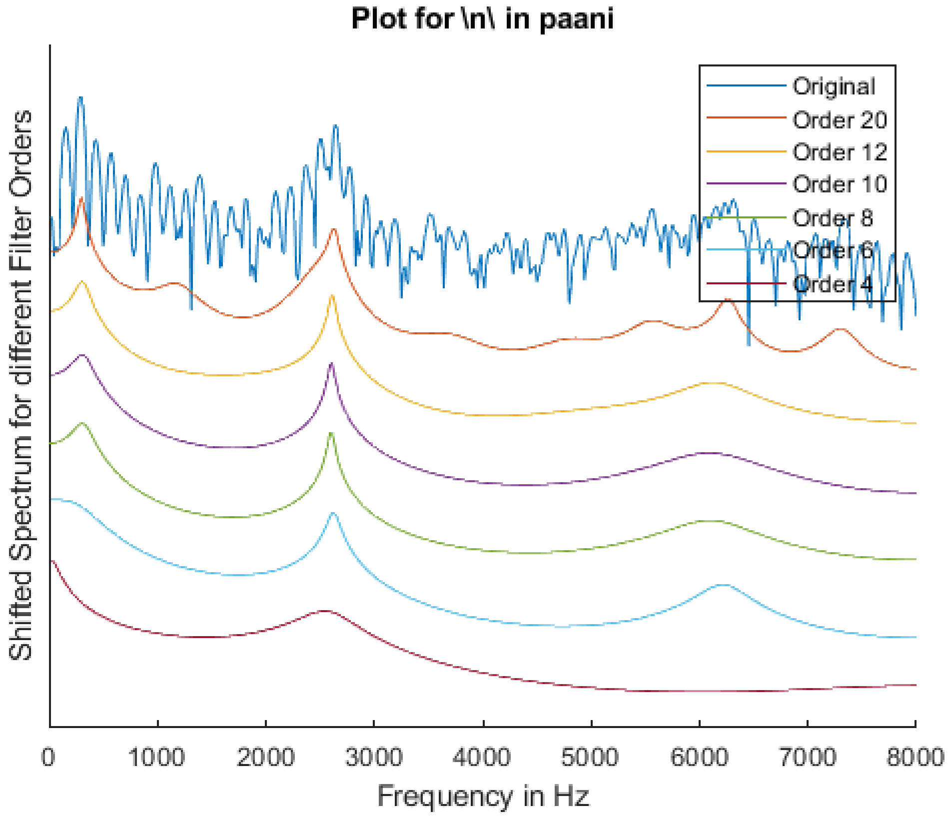
Q2 Part B

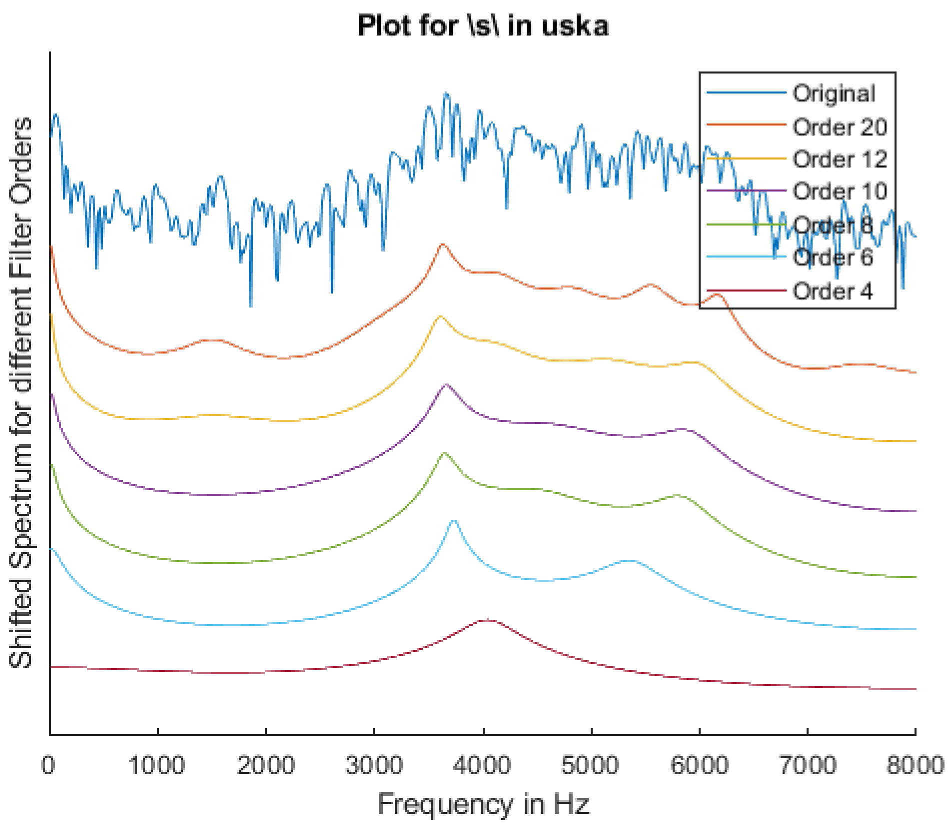
```
% For /a/
[magnitude_a_4, ~, ~, ~] = LP_Q2('a_in_pani.wav', 4, 1);
[magnitude_a_6, ~, ~, ~] = LP_Q2('a_in_pani.wav', 6, 1);
[magnitude_a_8, ~, ~, ~] = LP_Q2('a_in_pani.wav', 8, 1);
[magnitude_a_10, ~, ~, ~] = LP_Q2('a_in_pani.wav', 10, 1);
[magnitude_a_12, ~, ~, ~] = LP_Q2('a_in_pani.wav', 12, 1);
[magnitude_a_20, ~, ~, ~] = LP_Q2('a_in_pani.wav', 20, 1);
figure
hold on
plot((1:512)*8000/512, 10*16 + output_a)
plot((1:length(magnitude_a_20))*8000/length(magnitude_a_20), 10*10 +
 magnitude_a_20)
plot((1:length(magnitude_a_12))*8000/length(magnitude_a_12), 10*8 +
 magnitude_a_12)
plot((1:length(magnitude_a_10))*8000/length(magnitude_a_10), 10*6 +
 magnitude_a_10)
plot((1:length(magnitude_a_8))*8000/length(magnitude_a_8), 10*4 +
 magnitude_a_8)
plot((1:length(magnitude_a_6))*8000/length(magnitude_a_6), 10*2 +
 magnitude_a_6)
plot((1:length(magnitude_a_4))*8000/length(magnitude_a_4),
 magnitude_a_4)
```

```
lgnd = legend('Original', 'Order 20','Order 12', 'Order 10', 'Order
 8', 'Order 6', 'Order 4');
set(lgnd,'color','none');
hold off
title('Plot for \a\ in paani')
xlabel('Frequency in Hz')
ylabel('Shifted Spectrum for different Filter Orders')
set(gca,'YTick', [])

% For /n/
[magnitude_n_4, ~, ~, ~] = LP_Q2('n_in_pani.wav', 4, 1);
[magnitude_n_6, ~, ~, ~] = LP_Q2('n_in_pani.wav', 6, 1);
[magnitude_n_8, ~, ~, ~] = LP_Q2('n_in_pani.wav', 8, 1);
[magnitude_n_10, ~, ~, ~] = LP_Q2('n_in_pani.wav', 10, 1);
[magnitude_n_12, ~, ~, ~] = LP_Q2('n_in_pani.wav', 12, 1);
[magnitude_n_20, ~, ~, ~] = LP_Q2('n_in_pani.wav', 20, 1);
figure
hold on
plot((1:512)*8000/512, 10*16 + output_n)
plot((1:length(magnitude_n_20))*8000/length(magnitude_n_20), 10*10 +
 magnitude_n_20)
plot((1:length(magnitude_n_12))*8000/length(magnitude_n_12), 10*8 +
 magnitude_n_12)
plot((1:length(magnitude_n_10))*8000/length(magnitude_n_10), 10*6 +
 magnitude_n_10)
plot((1:length(magnitude_n_8))*8000/length(magnitude_n_8), 10*4 +
 magnitude_n_8)
plot((1:length(magnitude_n_6))*8000/length(magnitude_n_6), 10*2 +
 magnitude_n_6)
plot((1:length(magnitude_n_4))*8000/length(magnitude_n_4),
 magnitude_n_4)
lgnd = legend('Original', 'Order 20','Order 12', 'Order 10', 'Order
 8', 'Order 6', 'Order 4');
set(lgnd,'color','none');
hold off
title('Plot for \n\ in paani')
xlabel('Frequency in Hz')
ylabel('Shifted Spectrum for different Filter Orders')
set(gca,'YTick', [])

% For /s/
[magnitude_s_4, ~, ~, ~] = LP_Q2('s_in_uska.wav', 4, 0);
[magnitude_s_6, ~, ~, ~] = LP_Q2('s_in_uska.wav', 6, 0);
[magnitude_s_8, ~, ~, ~] = LP_Q2('s_in_uska.wav', 8, 0);
[magnitude_s_10, ~, ~, ~] = LP_Q2('s_in_uska.wav', 10, 0);
[magnitude_s_12, ~, ~, ~] = LP_Q2('s_in_uska.wav', 12, 0);
[magnitude_s_20, ~, ~, ~] = LP_Q2('s_in_uska.wav', 20, 0);
figure
hold on
plot((1:512)*8000/512, 10*16 + output_s)
plot((1:length(magnitude_s_20))*8000/length(magnitude_s_20), 10*10 +
 magnitude_s_20)
plot((1:length(magnitude_s_12))*8000/length(magnitude_s_12), 10*8 +
 magnitude_s_12)
```

```
plot((1:length(magnitude_s_10))*8000/length(magnitude_s_10), 10*6 +
 magnitude_s_10)
plot((1:length(magnitude_s_8))*8000/length(magnitude_s_8), 10*4 +
 magnitude_s_8)
plot((1:length(magnitude_s_6))*8000/length(magnitude_s_6), 10*2 +
 magnitude_s_6)
plot((1:length(magnitude_s_4))*8000/length(magnitude_s_4),
 magnitude_s_4)
lgnd = legend('Original', 'Order 20','Order 12', 'Order 10', 'Order
 8', 'Order 6', 'Order 4');
set(lgnd,'color','none');
hold off
title('Plot for \s\ in uska')
xlabel('Frequency in Hz')
ylabel('Shifted Spectrum for different Filter Orders')
set(gca,'YTick', [])
```



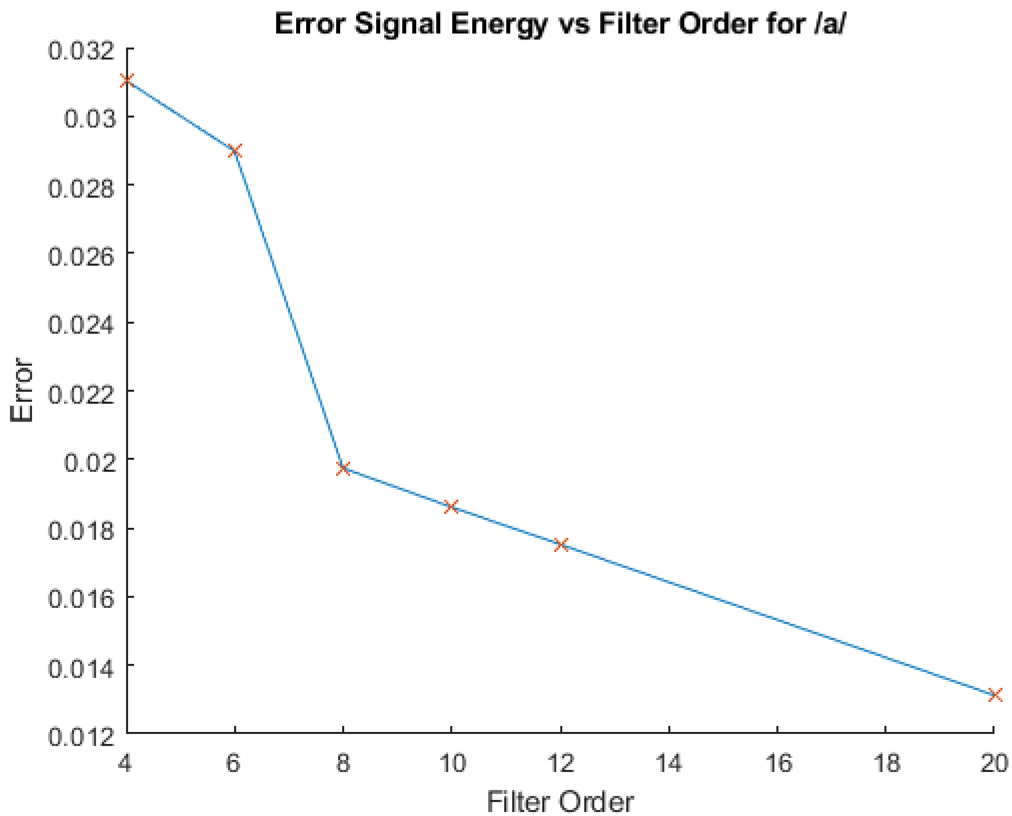

Plot for \n\ in paani
Original
Order 20
Order 12
Order 10
Order 8
Order 6
Order 4
Shifted Spectrum for different Filter Orders
0
1000
2000
3000
4000
5000
6000
7000
8000
Frequency in Hz

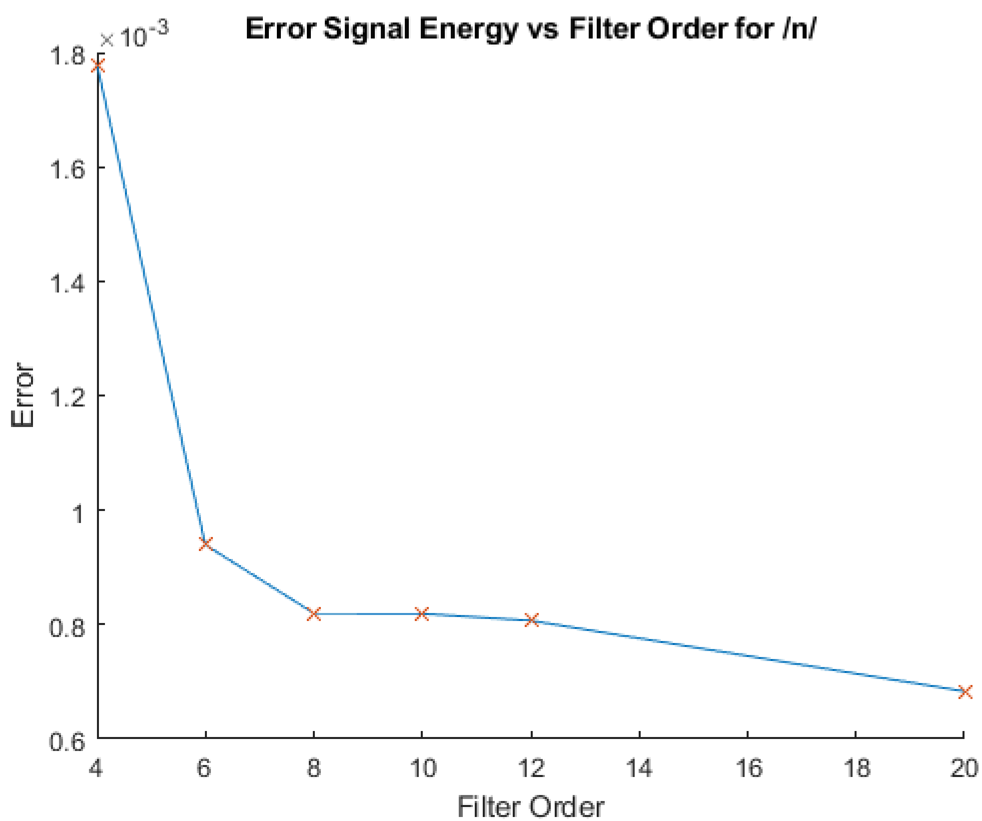

Plot for \s\ in uska
Original
Order 20
Order 12
Order 10
Order 8
Order 6
Order 4
Shifted Spectrum for different Filter Orders
0
1000
2000
3000
4000
5000
6000
7000
8000
Frequency in Hz

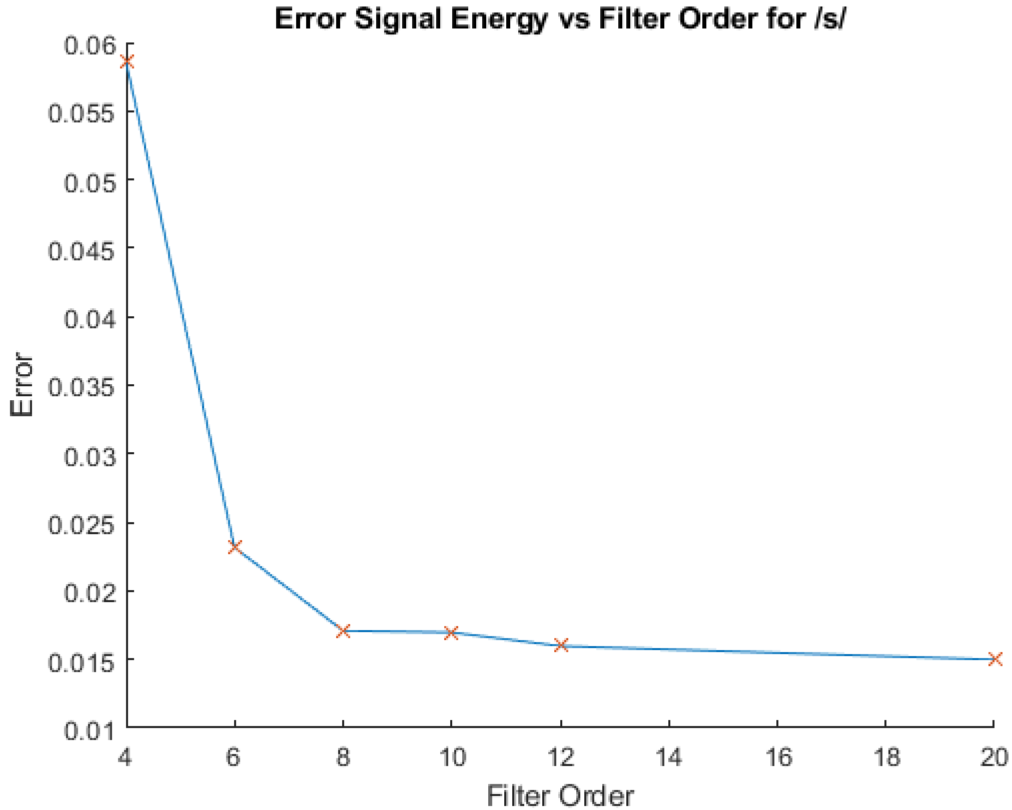
We see that the approximated LP filter is following the trend of the original waveform. As the order of the filter increases, the appoximated waveform becomes more close to the original waveform. This is also evident from the fact that LP estimate tries to minimize the error, hence more number of coefficients is better for approximating.

Q2 Part C

```
G_a = zeros(1, 6);
G_n = zeros(1, 6);
G_s = zeros(1, 6);
filter_orders = [4, 6, 8, 10, 12, 20];

for iter=1:6
    [~, ~, ~, G_a(iter)] = LP_Q2('a_in_pani.wav', filter_orders(iter),
 1);
    [~, ~, ~, G_n(iter)] = LP_Q2('n_in_pani.wav', filter_orders(iter),
 1);
    [~, ~, ~, G_s(iter)] = LP_Q2('s_in_uska.wav', filter_orders(iter),
 0);
end
figure
hold on
plot(filter_orders, G_a)
scatter(filter_orders, G_a, 'x')
hold off
title('Error Signal Energy vs Filter Order for /a/')
xlabel('Filter Order')
ylabel('Error')

figure
hold on
plot(filter_orders, G_n)
scatter(filter_orders, G_n, 'x')
hold off
title('Error Signal Energy vs Filter Order for /n/')
xlabel('Filter Order')
ylabel('Error')

figure
hold on
plot(filter_orders, G_s)
scatter(filter_orders, G_s, 'x')
hold off
title('Error Signal Energy vs Filter Order for /s/')
xlabel('Filter Order')
ylabel('Error')
```


Error Signal Energy vs Filter Order for /a/
0.032
0.03
0.028
0.026
0.024
0.022
0.02
0.018
0.016
0.014
0.012
Error
4
6
8
10
12
14
16
18
20
Filter Order


Error Signal Energy vs Filter Order for /n/
×10^-3
1.8
1.6
1.4
1.2
1
0.8
0.6
Error
4
6
8
10
12
14
16
18
20
Filter Order



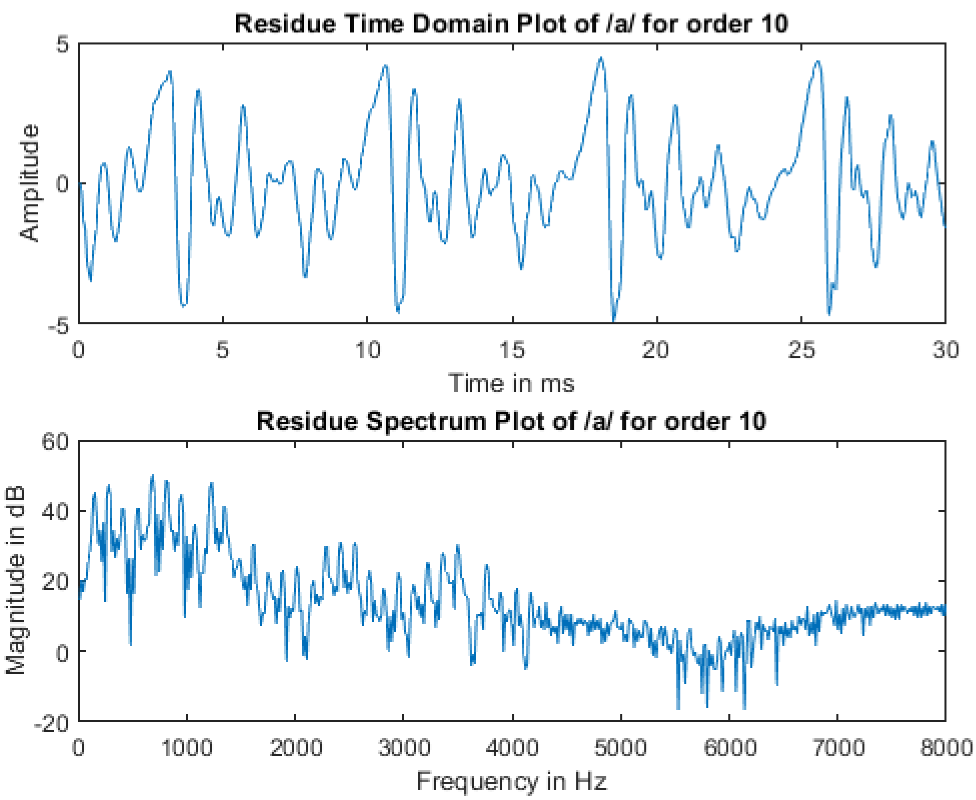
Q3

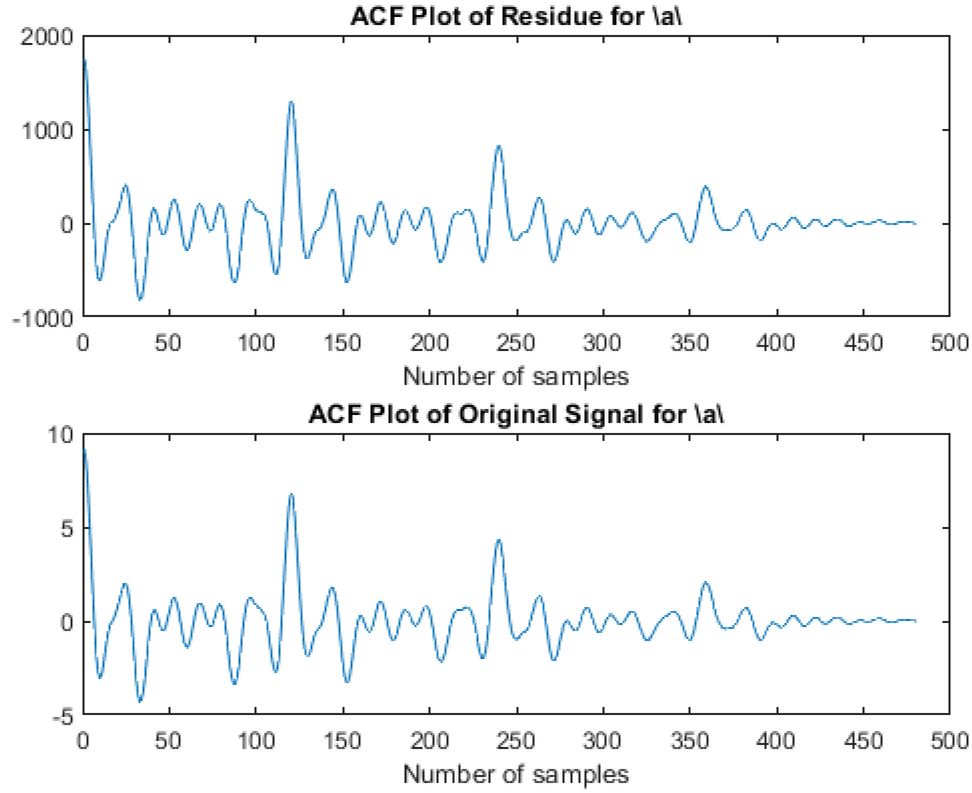
For /a/ in paani at Filter order 10

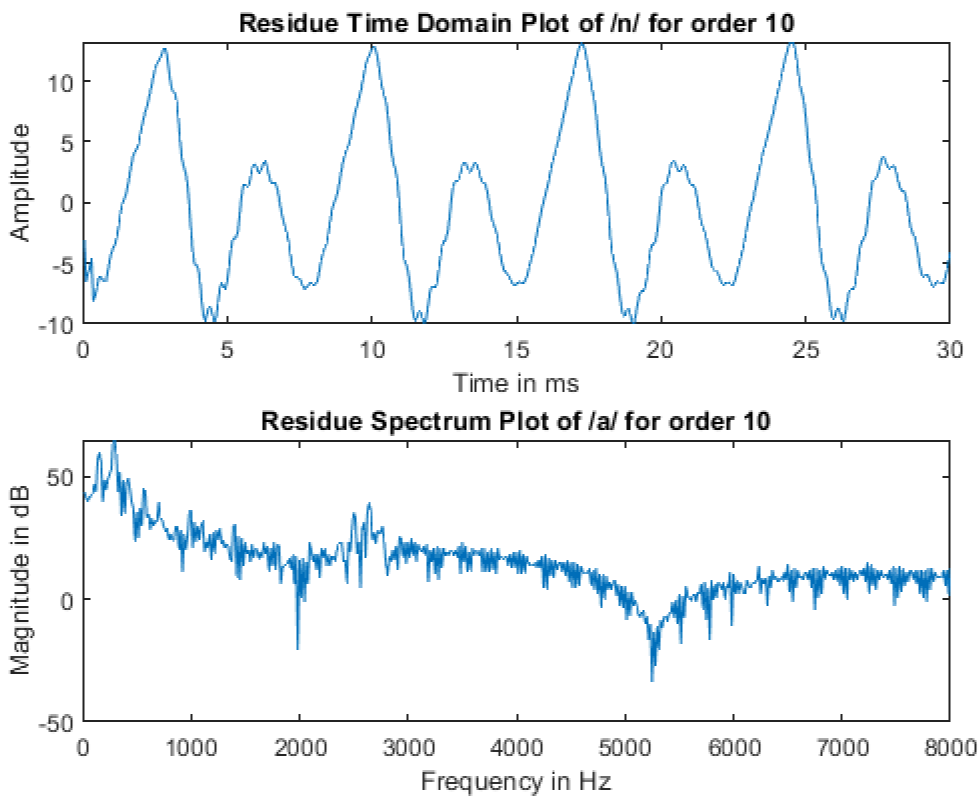
```
samp_freq = 16e3;
[residue_10_a, data_orig] = residue_signal('a_in_pani.wav', 10, 1);
figure
subplot(2, 1, 1)
plot((1:length(residue_10_a(1:480)))/(samp_freq/1e3),
 residue_10_a(1:480))
title('Residue Time Domain Plot of /a/ for order 10')
xlabel('Time in ms')
ylabel('Amplitude')

residue_10_a_fft = abs(fft(residue_10_a, 1024));
residue_10_a_fft = 20*log10(residue_10_a_fft(1:512));
subplot(2, 1, 2)
plot((1:512)*8000/512, residue_10_a_fft)
title('Residue Spectrum Plot of /a/ for order 10')
xlabel('Frequency in Hz')
ylabel('Magnitude in dB')

acf_output = ACF(residue_10_a);
data_acf = ACF(data_orig);
figure
subplot(2, 1, 1)
```

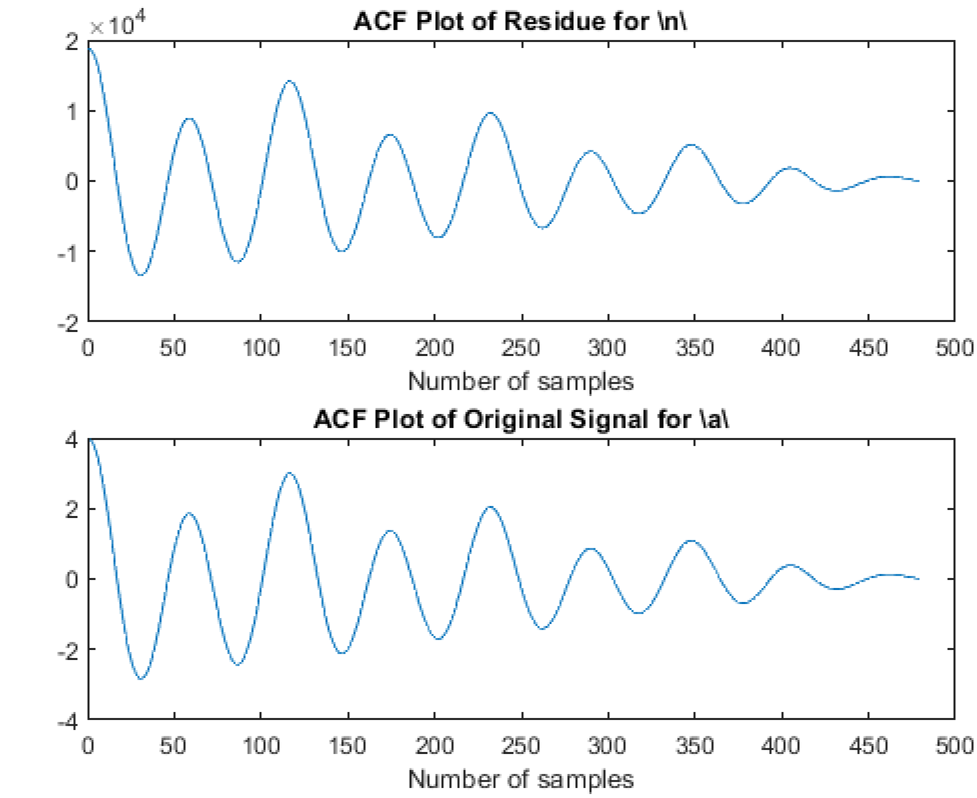
```
plot(acf_output((1:480)))
title('ACF Plot of Residue for \a\')
xlabel('Number of samples')
subplot(2, 1, 2)
plot(data_acf)
title('ACF Plot of Original Signal for \a\')
xlabel('Number of samples')

% *For /a/ peak of ACF occurs at 121th sample. Hence Pitch Period =
 121*16e3
% = 7.6 ms and Pitch frequency = 131.57 Hz*

% For /n/ in paani at Filter order 10
samp_freq = 16e3;
[residue_10_n, data_orig] = residue_signal('n_in_pani.wav', 10, 1);
figure
subplot(2, 1, 1)
plot((1:length(residue_10_n(1:480)))/(samp_freq/1e3),
 residue_10_n(1:480))
title('Residue Time Domain Plot of /n/ for order 10')
%ylim([-0.5 0.5])
xlabel('Time in ms')
ylabel('Amplitude')

residue_10_n_fft = abs(fft(residue_10_n, 1024));
residue_10_n_fft = 20*log10(residue_10_n_fft(1:512));
subplot(2, 1, 2)
plot((1:512)*8000/512, residue_10_n_fft)
title('Residue Spectrum Plot of /a/ for order 10')
xlabel('Frequency in Hz')
ylabel('Magnitude in dB')

acf_output = ACF(residue_10_n);
data_acf = ACF(data_orig);
figure
subplot(2, 1, 1)
plot(acf_output((1:480)))
title('ACF Plot of Residue for \n\')
xlabel('Number of samples')
subplot(2, 1, 2)
plot(data_acf)
title('ACF Plot of Original Signal for \a\')
xlabel('Number of samples')

% *For /n/ peak of ACF occurs at 116th sample. Hence Pitch Period =
 116*16e3
% = 7.3 ms and Pitch frequency = 137.93 Hz*

% For /s/ in paani at Filter order 10
samp_freq = 16e3;
[residue_10_s, data_orig] = residue_signal('s_in_uska.wav', 10, 0);
figure
subplot(2, 1, 1)
```

```
plot((1:length(residue_10_s(1:480)))/(samp_freq/1e3),
 residue_10_s(1:480))
title('Residue Time Domain Plot of /s/ for order 10')
xlabel('Time in ms')
ylabel('Amplitude')

residue_10_s_fft = abs(fft(residue_10_s, 1024));
residue_10_s_fft = 20*log10(residue_10_s_fft(1:512));
subplot(2, 1, 2)
plot((1:512)*8000/512, residue_10_s_fft)
title('Residue Spectrum Plot of /s/ for order 10')
xlabel('Frequency in Hz')
ylabel('Magnitude in dB')

acf_output = ACF(residue_10_s);
data_acf = ACF(data_orig);
figure
subplot(2, 1, 1)
plot(acf_output((1:480)))
title('ACF Plot of Residue for \s\')
xlabel('Number of samples')
subplot(2, 1, 2)
plot(data_acf)
title('ACF Plot of Original Signal for \s\')
xlabel('Number of samples')
```

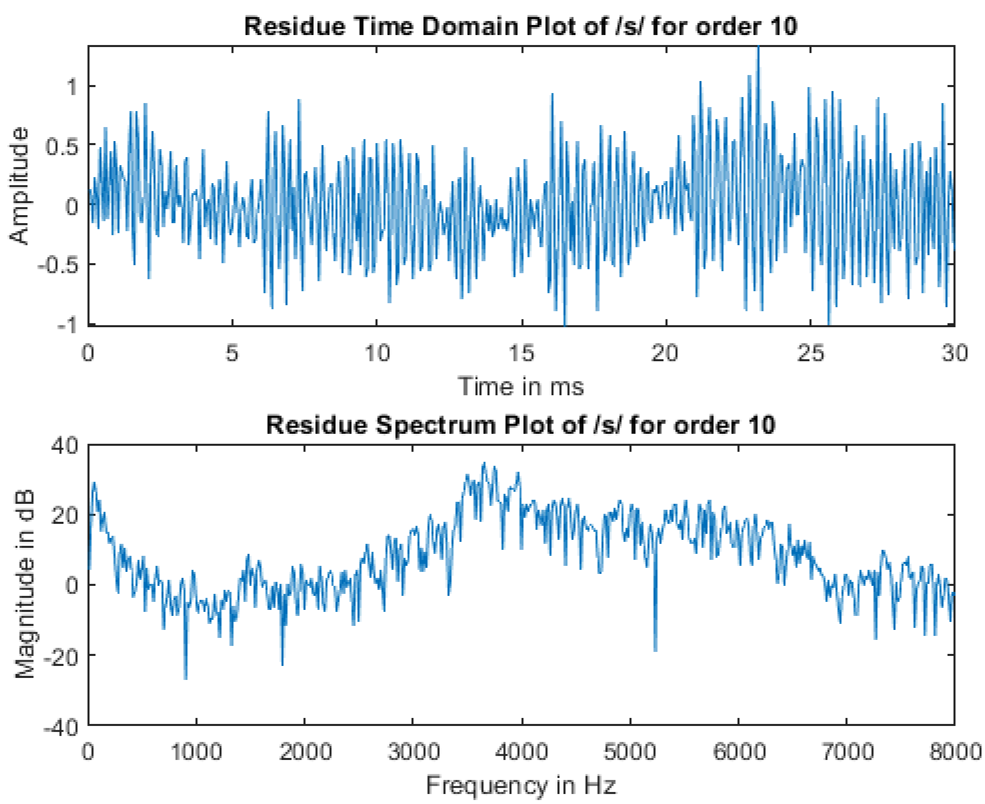



ACF Plot of Residue for \a\
2000
1000
0
-1000
0 50 100 150 200 250 300 350 400 450 500
Number of samples
ACF Plot of Original Signal for \a\
10
5
0
-5
0 50 100 150 200 250 300 350 400 450 500
Number of samples

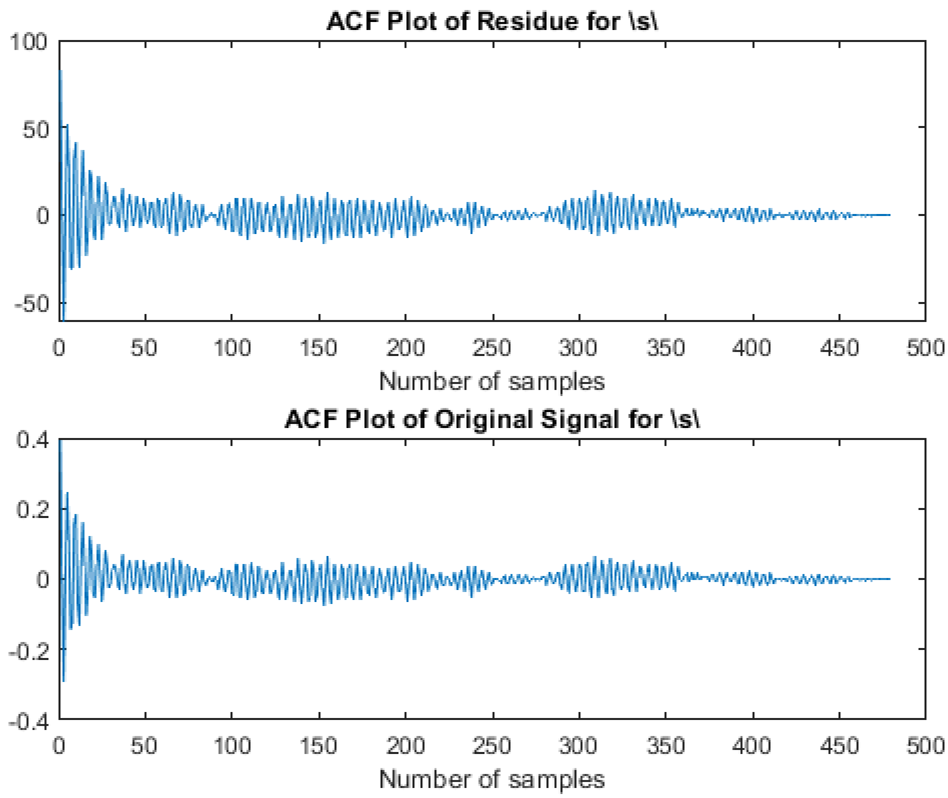

Residue Time Domain Plot of /n/ for order 10
Amplitude
10
5
0
-5
-10
0 5 10 15 20 25 30
Time in ms
Residue Spectrum Plot of /a/ for order 10
Magnitude in dB
50
0
-50
0 1000 2000 3000 4000 5000 6000 7000 8000
Frequency in Hz



ACF Plot of Residue for \n\
×10^4
Number of samples
ACF Plot of Original Signal for \a\
Number of samples



Residue Time Domain Plot of /s/ for order 10
Amplitude
Time in ms
Residue Spectrum Plot of /s/ for order 10
Magnitude in dB
Frequency in Hz



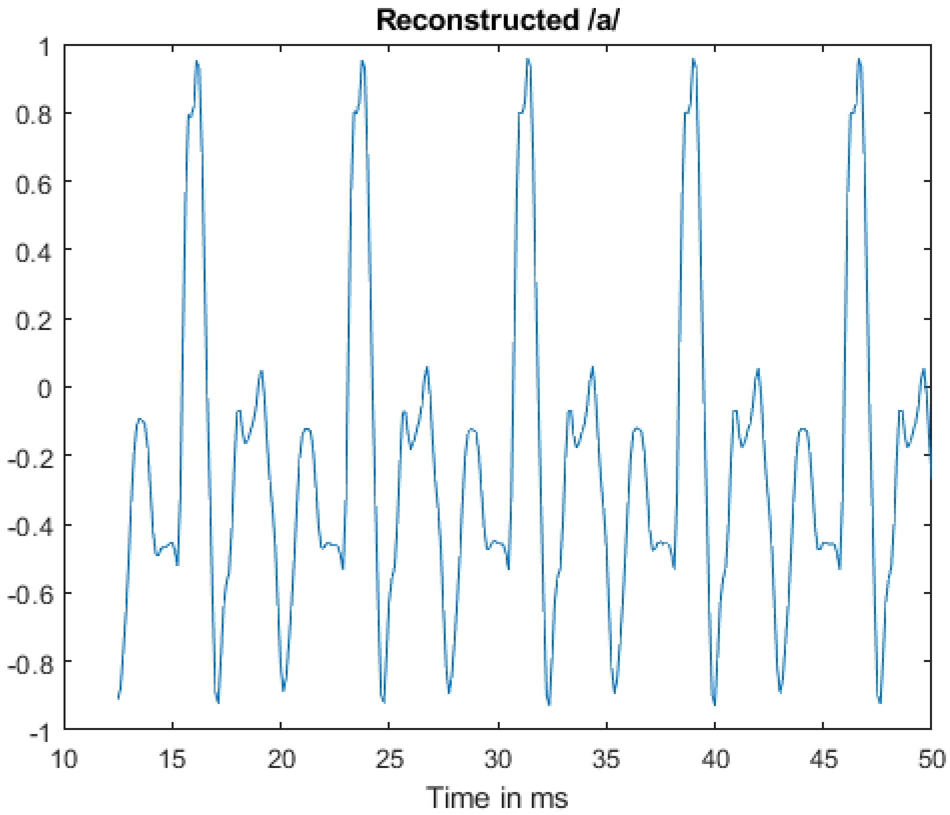
Comparison of ACF Plot - It is evident from theory as well as the above demonstration that Residue is a good approximation of the glottal waveform. Therefore for voiced the output will be periodic, whereas for unvoiced, it would not reveal any significant pattern. As we see above, for /s/ the output is non perioidc and does not contain any inference. We would also expect the magnitude response of /s/ to be flat, which is visible in the plot as well.

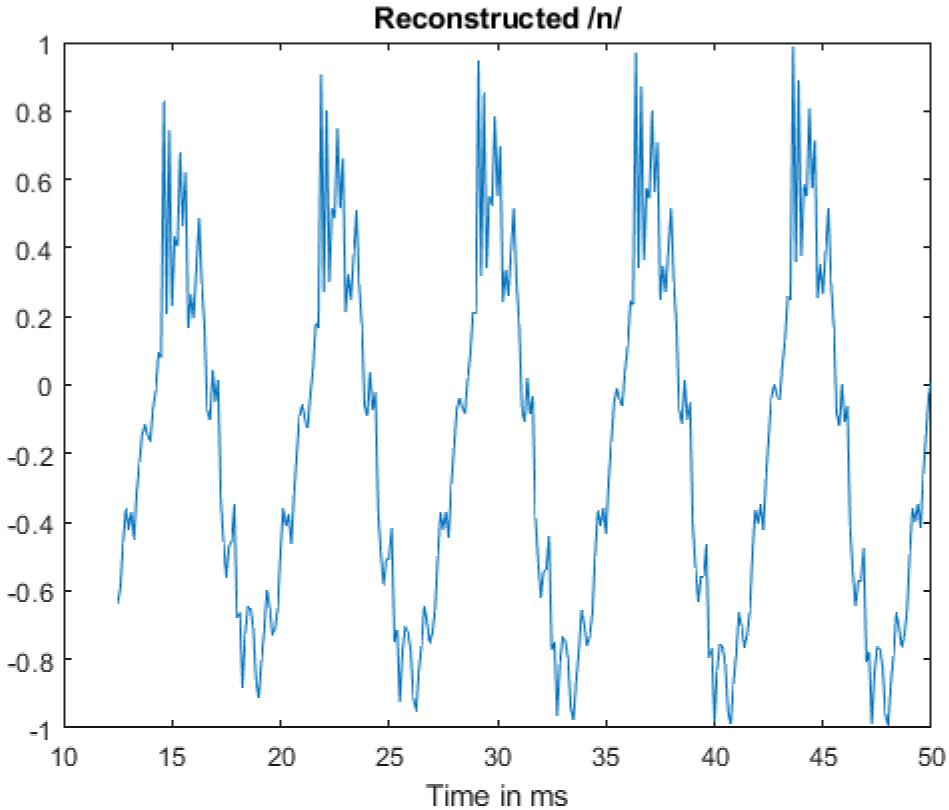
Q4

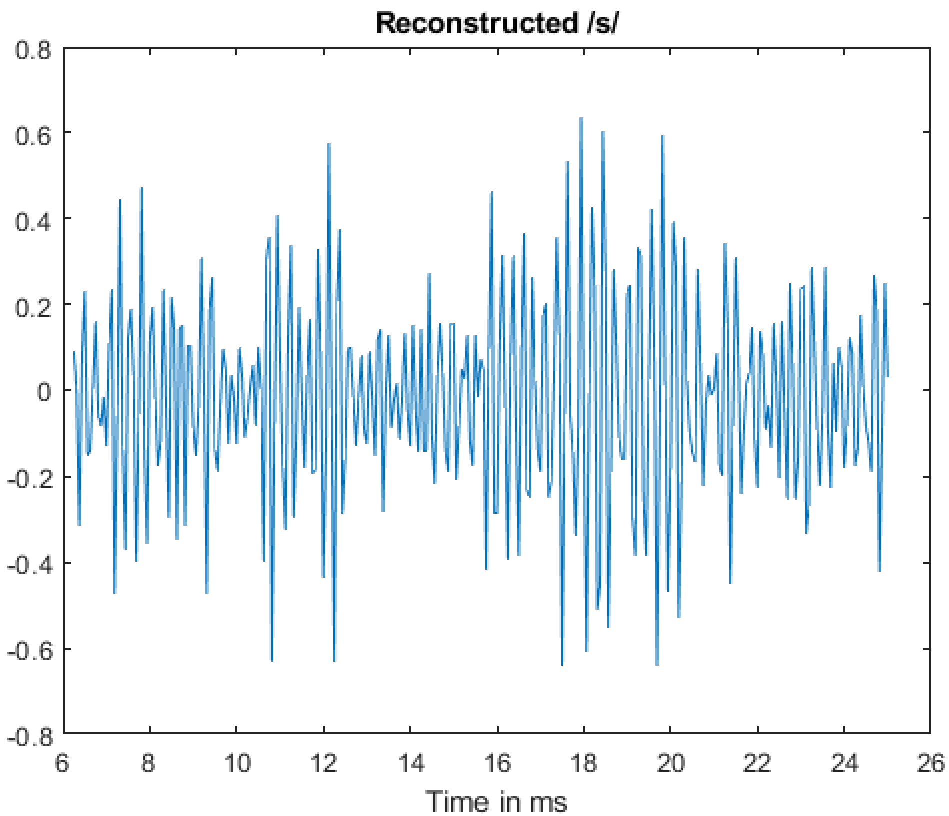
```
%%%%%
% *For /a/ peak of ACF occurs at 121th sample. Hence Pitch Period =
 121*16e3
% = 7.6 ms and Pitch frequency = 131.57 Hz
%
% For /n/ peak of ACF occurs at 116th sample. Hence Pitch Period =
 116*16e3
% = 7.3 ms and Pitch frequency = 137.93 Hz*

time_duration = 5000e-3;
%/a/ Re-generation of sound using LP analysis
samp_freq = 8e3;
filter_order = 20;
num_samples = samp_freq*time_duration;
input = zeros(1, num_samples);
a_period = 121;
a_period_downsampled = round(a_period/2); %Since we need to save at 8k
input(1:a_period_downsampled:end) = 1;
```

```
[~, num_coeffs, den_coeffs, ~] = LP_Q2('a_in_pani.wav', filter_order,
 1);
output_filter = filter(num_coeffs, den_coeffs, input);
%output_filter = filter_signal(input, coeffs, filter_order);
output_deemphasis = filter(1, [1 -0.95], output_filter);
scaled_output = (2/(max(output_deemphasis(:))-
min(output_deemphasis(:))))*(output_deemphasis -
 min(output_deemphasis(:))) - 1;
figure
plot((100:400)/(samp_freq/1e3), scaled_output(100:400))
xlabel('Time in ms')
title('Reconstructed /a/')
audiowrite('a-reconstructed.wav', scaled_output, samp_freq);

%/n/ Re-generation of sound using LP analysis
samp_freq = 8e3;
filter_order = 20;
num_samples = samp_freq*time_duration;
input = zeros(1, num_samples);
n_period = 116;
n_period_downsampled = round(n_period/2); %Since we need to save at 8k
input(1:n_period_downsampled:end) = 1;
[~, num_coeffs, den_coeffs, ~] = LP_Q2('n_in_pani.wav', filter_order,
 1);
output_filter = filter(num_coeffs, den_coeffs, input);
%output_filter = filter_signal(input, coeffs, filter_order);
%output_deemphasis = filter(1, [1 -0.95], output_filter);
output_deemphasis = filter_signal(output_filter, [1 -0.95], 1);
scaled_output = (2/(max(output_deemphasis(:))-
min(output_deemphasis(:))))*(output_deemphasis -
 min(output_deemphasis(:))) - 1;
figure
plot((100:400)/(samp_freq/1e3), scaled_output(100:400))
xlabel('Time in ms')
title('Reconstructed /n/')
audiowrite('n-reconstructed.wav', scaled_output, samp_freq);


%/s/ Re-generation of sound using LP analysis
samp_freq = 16e3;
filter_order = 20;
num_samples = samp_freq*time_duration;
input = wgn(1, num_samples, 0); %WGN with var 1. Required
 Communication Toolbox
[~, num_coeffs, den_coeffs, ~] = LP_Q2('s_in_uska.wav', filter_order,
 1);
output_filter = filter(num_coeffs, den_coeffs, input);
%output_filter = filter_signal(input, coeffs, filter_order);
scaled_output = (2/(max(output_filter(:))-
min(output_filter(:))))*(output_filter - min(output_filter(:))) - 1;
figure
plot((100:400)/(samp_freq/1e3), scaled_output(100:400))
xlabel('Time in ms')
title('Reconstructed /s/')
```

```
audiowrite('s-reconstructed.wav', scaled_output, samp_freq);
```




Reconstructed /n/
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
10
15
20
25
30
35
40
45
50
Time in ms


Reconstructed /s/
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
6
8
10
12
14
16
18
20
22
24
26
Time in ms

The Audio seems to resemble the sounds discussed. Thus through this exercise, we established that Linear Predictive Analysis is a good way to approximate a sound signal with very few coefficients. This has several important application. For example, we only need to store the coefficients in order to reconstruct an approximate signal, thus saving storage space. With reduced storage space, it is advantageous to send the coefficients instead of the whole sound signal.

The difference between Actual and LP filtered sound is due to several facts. One major reason is the use of impulse train. As we know that our glotal vibration is different than an ideal impulse response. Hence, the audio sounds monotonic. This could have been further improved by adding aspiration and jitter. Also since in our analysis, we approximate the sound using finite coefficients, the output is naturally prone to errors. Also, for the unvoiced sound, it is noisy because of the mismatch in the glottal waveform.

Published with MATLAB® R2019a