EE-414 Speech Processing Lab

Lab-7 - 180020002

Aim:

● To understand the limitation of DTFT for the spectral analysis of speech.

● To understand the development of short-term Fourier transform (STFT) representation.

● To understand the difference in the nature of linear and log magnitude spectra.

● To understand the difference among the spectra of voiced, unvoiced and silence regions of speech.

● To plot the STFT of a speech signal.

● To understand the difference between true and convolved spectra.

● To understand the effect of rectangular, Hamming and Hanning window functions on short term spectral analysis.

● To understand the effect of frame size on short term spectral analysis

Theory:

The different frequency or spectral components that are present in the speech signal are not directly apparent in the time domain. Hence we need to go for frequency domain representation using Fourier representation. The conventional Fourier representation is inadequate to provide information about the time varying nature of spectral information present in speech. Hence the need for short term version of Fourier transform, termed more commonly as Short term Fourier Transform (STFT) .

DTFT

If X(w) is the discrete time Fourier Transform(DTFT) of x(n), a discrete time signal, then its DTFT is given by:

$$X(w) = \sum_{n=-\infty}^{\infty} x(n) e^{-jwn}$$

X(w) is continuous function of frequency and hence cannot be computed on a digital signal processor or machine. To make it possible discrete version of the DTFT termed as discrete Fourier transform (DFT) is defined where DFT is obtained by uniformly sampling X(w) is given by

$$X(w) = X(w_k)$$

where w_k=2*pi*k , k=0,1,....(N-1), where 'N' is the number of samples of X(w). The value of 'N' is chosen such that, the reconstructed sequence x(n) in the time domain is free from aliasing. For this the condition is 'N' should be greater than or equal to the length of x(n).

Limitation of DTFT

Speech is made of time-varying sequence of spectra. Thus one spectrum for the entire signal will not help us to understand the different frequency components present in the speech signal. Therefore for speech we need a representation that will give time varying spectral information.

Need for STFT

To take care of time varying spectral information, the short term processing approach is employed. In short term processing, speech is processed is blocks of 10-30 ms with a shift of 10 ms. For instance, using a block size of 20 ms, the DTFT is computed using DFT for that block. Their process is repeated for all the blocks of speech signal and all the spectra computed are stacked together as a function of time and frequency to observe the time varying spectra. To accommodate the time varying nature of this spectra, the DTFT equation is defined as given below

$$X(w,n) = \sum_{m=-\infty}^{\infty} x(m)w(n-m)e^{-jwm}$$

where **W(n)** is the window function for short term processing. Now the spectral amplitude and phase are function of both frequency and time where as it was only function of frequency in the earlier case of DTFT. **x(m).w(n-m)** represents the window segment around the time instant **'n'.** Hence **X(w,n)** at **'n'** represents the spectrum of the speech segment present around it. When we shift **'n'**, then correspondent **X(w,n)** also changes. Thus giving visualization of the time varying spectra of speech. Since such a time-spectral is computed using short term processes, **X(w,n)** is termed as Short Term Fourier Transform (STFT).

Concept of True vs Convolved spectra

STFT need not accurately represent the true spectral information of speech.

We can consider a segment of sinewave, say 30 ms somewhere along its length by windowing and compute its spectrum. In this case, the resulting spectrum is not an impulse at f Hz, but a sinc function centered around f Hz . This is because, by multipliying it with a rectangular window in the time domain, leads to the convolution of their spectra in the frequency domain. The true spectrum of sine wave is impulse at f HZ and that of rectangular window is sinc function. As a result, the convolved spectrum is a sinc function around f HZ .

Window size for short term spectral analysis:

If the window duration is of 10-30 ms, then the spectral information may be affected to a minimum extent. Alternatively, if the window size is too less, then the window effect may be too severe. If the window is too large , say 100 msec, then even though windowing effect is negligable compared to 10-30 ms case, we cannot use such a long window due to the non-stationary nature of speech.

Effect of windowing function on short term spectral analysis:

The width of main lose in the frequency domain is less for rectangular window compared to the other two window function. As a result the resolution offered by the rectangular window function is better. Alternatively, the peak-to-side lobe ratio of rectangular window is significantly poor compared to the other two window functions. This results in relatively more spectral leakage in case of rectangular window which is not desirable. Thus from

the resolution point of view, rectangular window is preferable and from spectral leakage point of view other two window functions are preferable. The effect of spectral leakage is severe and hence in most of the speech analysis applications, either Hamming or Hanning window function is employed.

Procedure:

Record (16kHz, 16bit) the word “speech signal”; truncate long silence regions.

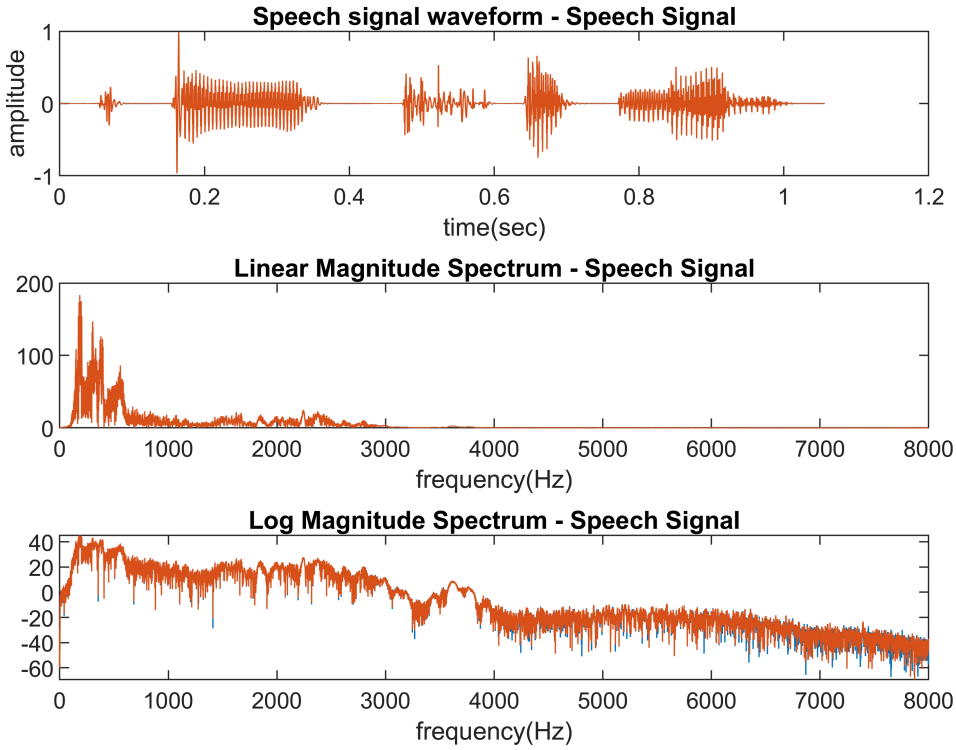
A. DTFT of and its limitation:

a. Plot the linear and log magnitude spectrum for the entire speech.

b. Plot log-magnitude spectrum of voiced, unvoiced and silence regions in the recorded speech and explain the difference between log-spectrum of all three cases.

c. Comment on the limitation of DTFT.

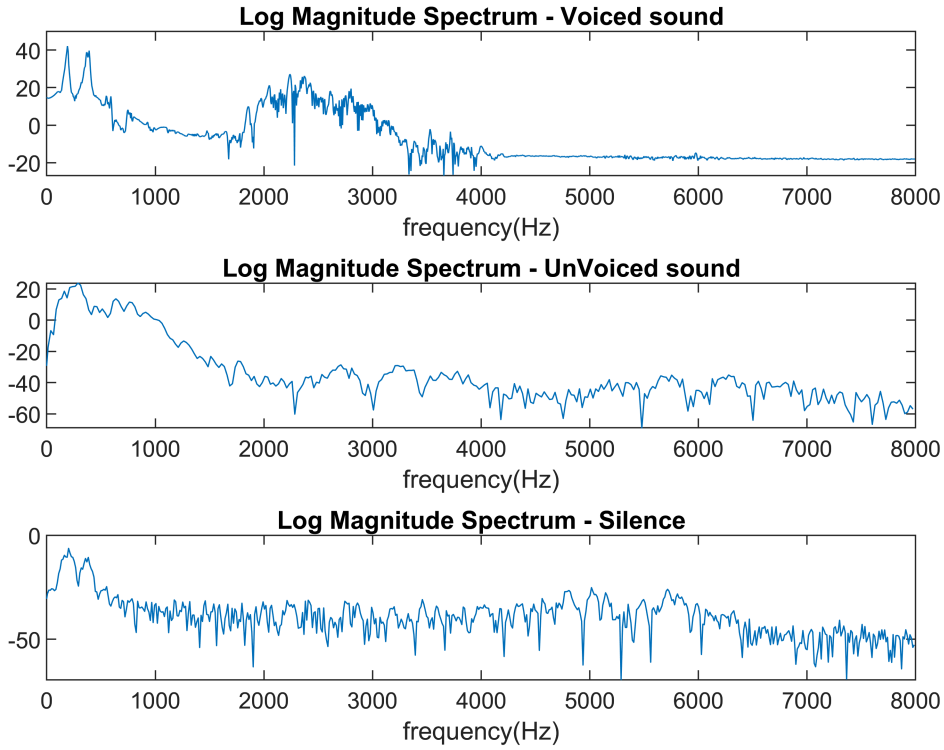
```
% A-a

%Matlab program to load and plot waveform of speech signal stored in
% wav file format, linear and log magnitude spectrum for the entire speech.
%file name is l7_speech_signal.wav and full path is given
[y,fs]=audioread('l7_speech_signal.wav');

%normalising the signal amplitudes to be in -1 to 1
y=y./(1.01*abs(max(y)));
%plotting waveform of the speech signal
t = 0 : 1 / fs : (length(y) - 1) / fs;
figure;
subplot(311);
plot(t, y);
xlabel('time(sec)');
ylabel('amplitude');
title('Speech signal waveform - Speech Signal');

%plotting Linear Magnitude Spectrum of the speech signal
Y = fftshift(fft(y));
F = -fs/2 : fs/length(Y) : fs/2 - fs/length(Y);

subplot(312);
plot(F, abs(Y));
xlim([0, fs/2]);
xlabel('frequency(Hz)');
title('Linear Magnitude Spectrum - Speech Signal');

%plotting Log Magnitude Spectrum of the speech signal
subplot(313);
plot(F, 20*log10(abs(Y)));
xlim([0, fs/2]);
xlabel('frequency(Hz)');
title('Log Magnitude Spectrum - Speech Signal');
```



..

```
% A-b

%/ee/- Voiced sound
y_v = y(ceil(0.169*fs) : floor(0.334*fs));
Y_v = fftshift(fft(y_v));
F_v = -fs/2 : fs/length(Y_v) : fs/2 - fs/length(Y_v);

%/s/ - Unvoiced sound
y_uv = y(ceil(0.053*fs) : floor(0.093*fs));
Y_uv = fftshift(fft(y_uv));
F_uv = -fs/2 : fs/length(Y_uv) : fs/2 - fs/length(Y_uv);

% silence between /ch/ & /s/
y_s = y(ceil(0.364*fs) : floor(0.441*fs));
Y_s = fftshift(fft(y_s));
F_s = -fs/2 : fs/length(Y_s) : fs/2 - fs/length(Y_s);


%plotting Log Magnitude Spectrum of the voiced sound
figure;
subplot(3,1,1);
plot(F_v,20*log10(abs(Y_v)));
xlim([0, fs/2]);
xlabel('frequency(Hz)');
title('Log Magnitude Spectrum - Voiced sound');
```

```
%plotting Log Magnitude Spectrum of the unvoiced sound
subplot(3,1,2);
plot(F_uv,20*log10(abs(Y_uv)));
xlim([0, fs/2]);
xlabel('frequency(Hz)');
title('Log Magnitude Spectrum - UnVoiced sound');

%plotting Log Magnitude Spectrum of the silence
subplot(3,1,3);
plot(F_s,20*log10(abs(Y_s)));
xlim([0, fs/2]);
xlabel('frequency(Hz)');
title('Log Magnitude Spectrum - Silence');
```



Observations:

Log spectrum of Voiced region has a nature of downward sloping.

Log spectrum of Unvoiced region has a nature of non-decreasing sloping.

Log spectrum of silence region maintains the same level.

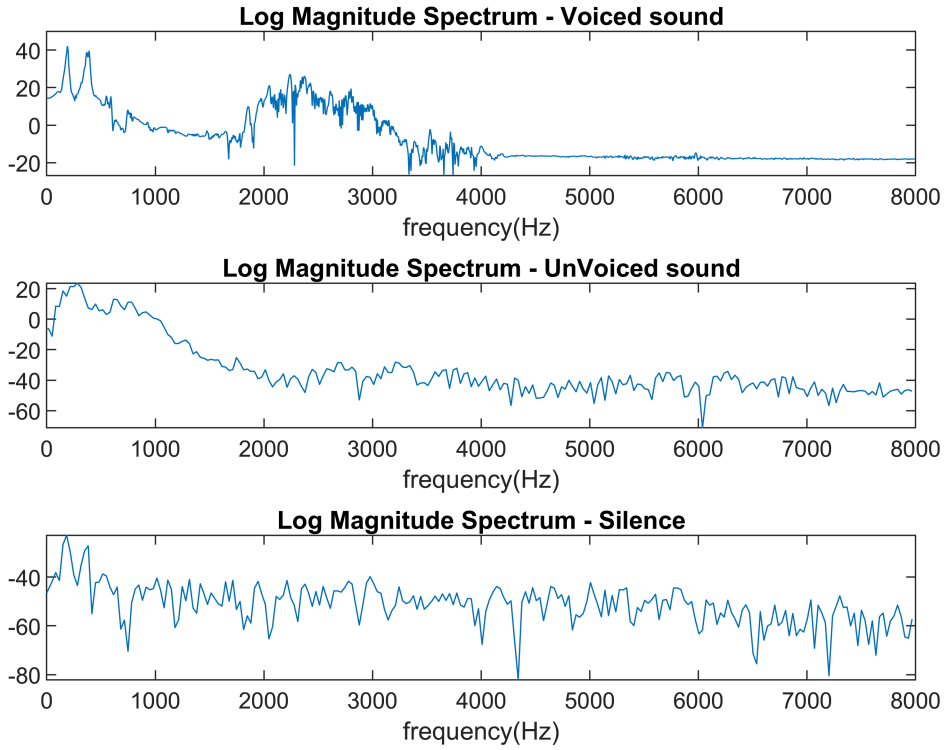
Here the speech regions are made of time-varying sequence of spectra. Thus one spectrum for the entire signal of the speech regions will not help us to understand the different frequency components present in the speech signal. Therefore for speech we need a representation that will give time varying spectral information.

B. Need for Short Time Fourier Transform(STFT):

a. How can you solve the above problem using STFT? Plot the STFT of one speech frame at the centre of above three regions. Write your observations.

Here, speech is processed in blocks of 20-30 ms. Using a block size of 30 ms, the DTFT is computed using DFT for that block. Their process is repeated for all the blocks of speech signal and all the spectra computed are stacked together as a function of time and frequency to observe the time varying spectra.

```
%/ee/ - Voiced sound
Y_v = y(ceil(0.222*fs) : floor(0.252*fs));
Y_v = fftshift(fft(y_v));
F_v = -fs/2 : fs/length(Y_v) : fs/2 - fs/length(Y_v);

%/s/ - Unvoiced sound
y_uv = y(ceil(0.054*fs) : floor(0.084*fs));
Y_uv = fftshift(fft(y_uv));
F_uv = -fs/2 : fs/length(Y_uv) : fs/2 - fs/length(Y_uv);

% silence between /ch/ & /s/
y_s = y(ceil(0.387*fs) : floor(0.417*fs));
Y_s = fftshift(fft(y_s));
F_s = -fs/2 : fs/length(Y_s) : fs/2 - fs/length(Y_s);


%plotting Log Magnitude Spectrum of the voiced sound
figure;
subplot(3,1,1);
plot(F_v,20*log10(abs(Y_v)));
xlim([0, fs/2]);
xlabel('frequency(Hz)');
title('Log Magnitude Spectrum - Voiced sound');

%plotting Log Magnitude Spectrum of the unvoiced sound
subplot(3,1,2);
plot(F_uv,20*log10(abs(Y_uv)));
xlim([0, fs/2]);
xlabel('frequency(Hz)');
title('Log Magnitude Spectrum - UnVoiced sound');

%plotting Log Magnitude Spectrum of the silence
subplot(3,1,3);
plot(F_s,20*log10(abs(Y_s)));
xlim([0, fs/2]);
xlabel('frequency(Hz)');
title('Log Magnitude Spectrum - Silence');
```



Observations:

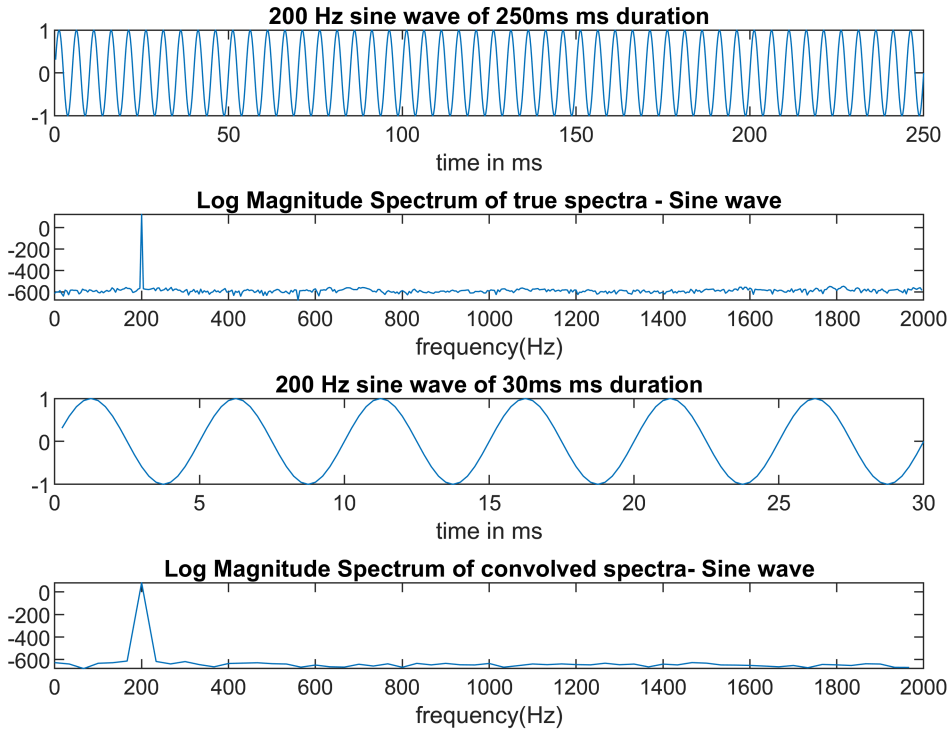
The spectral amplitude associated with a frequency component varies as a function of time. This is what needs to be observed in case of time varying spectra available in speech and hence the usefulness of STFT.

C. Concept of true and convolved spectra:

a. Create a sine wave of 200Hz and compute DTFT of the whole signal. Now also compute the STFT by taking a 20-30ms short term segment of the sine wave. Explain your observation and difference between the true and convolved spectrum.

```
Fs=4000; %sampling frequency
f=200; %frequency of the sine wave
t=1/Fs:1/Fs:.25;
y_sin=sin(2*pi*f*t);
figure;
subplot(411);
t=t*1000;
plot(t,y_sin);
title('200 Hz sine wave of 250ms ms duration');
xlabel('time in ms');

%DTFT of sine wave
Y_sin = fftshift(fft(y_sin));
F_sin = -Fs/2 : Fs/length(Y_sin) : Fs/2 - Fs/length(Y_sin);
```

```
subplot(412);
plot(F_sin, 20*log(abs(Y_sin)));
xlim([0, Fs/2]);
xlabel('frequency(Hz)');
title('Log Magnitude Spectrum of true spectra - Sine wave');



t=1/Fs:1/Fs:.03;
y_sin=sin(2*pi*f*t);
t=t*1000;
subplot(413);
plot(t,y_sin);
title('200 Hz sine wave of 30ms ms duration');
xlabel('time in ms');

%DTFT of sine wave
Y_sin = fftshift(fft(y_sin));
F_sin = -Fs/2 : Fs/length(Y_sin) : Fs/2 - Fs/length(Y_sin);

subplot(414);
plot(F_sin, 20*log(abs(Y_sin)));
xlim([0, Fs/2]);
xlabel('frequency(Hz)');
title('Log Magnitude Spectrum of convolved spectra- Sine wave');
```



Observations:

We can observe that the true spectra is represented by an impulse function at 200 Hz and the convolved spectrum is represented by the sinc function centered around 200 Hz . This is because, by multipliying it with a rectangular window in the time domain, leads to the convolution of their spectra(i.e sinc function) in the frequency domain.

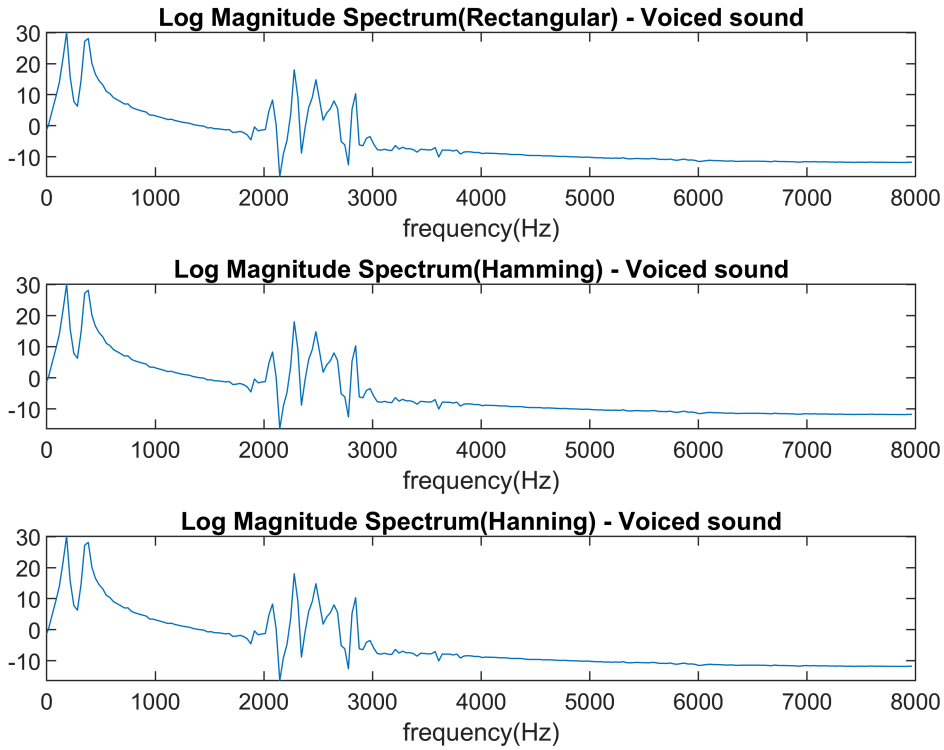
D. Effect of windowing function and window size on short term spectral analysis:

a. Plot short term log magnitude spectra of a 30 ms voiced speech segment using a rectangular, hamming and hanning window functions. Compare and write your observations in all the three cases.

b. Plot a short term log magnitude spectra of the voiced segment using a frame size of 3 ms, 30 ms and 300 ms. Compare and write your observations in all the three cases.

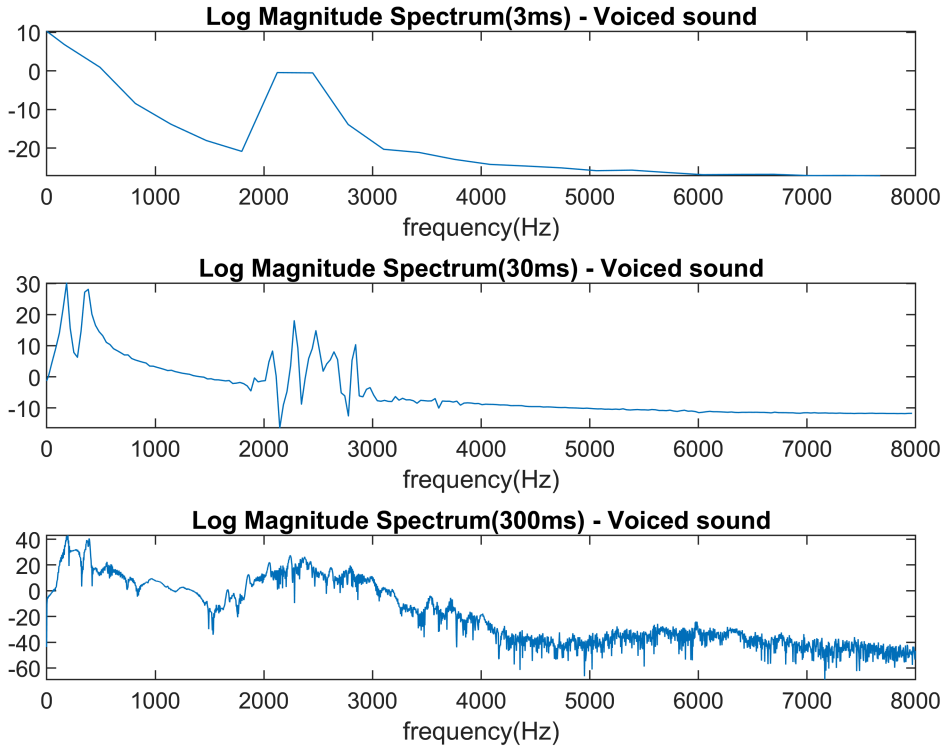
```
% D-a

%% Rectangular
y_v = y(ceil(0.222*fs) : floor(0.252*fs));
Y_v = fftshift(fft(y_v));
F_v = -fs/2 : fs/length(Y_v) : fs/2 - fs/length(Y_v);

%plotting Log Magnitude Spectrum of the voiced sound
figure;
subplot(311);
plot(F_v,20*log10(abs(Y_v)));
xlim([0, fs/2]);
xlabel('frequency(Hz)');
title('Log Magnitude Spectrum(Rectangular) - Voiced sound');

%% Hamming
win_hamm = dsp.Window('Hamming');
y_v_hamm = win_hamm(y_v);
Y_v_hamm = fftshift(fft(y_v_hamm));
F_v_hamm = -fs/2 : fs/length(Y_v_hamm) : fs/2 - fs/length(Y_v_hamm);

%plotting Log Magnitude Spectrum of the voiced sound
subplot(312);
plot(F_v_hamm,20*log10(abs(Y_v_hamm)));
xlim([0, fs/2]);
xlabel('frequency(Hz)');
title('Log Magnitude Spectrum(Hamming) - Voiced sound');

%% Hanning
win_hann = dsp.Window('Hanning');
y_v_hann = win_hann(y_v);
Y_v_hann = fftshift(fft(y_v_hann));
F_v_hann = -fs/2 : fs/length(Y_v_hann) : fs/2 - fs/length(Y_v_hann);
```

```
% plotting Log Magnitude Spectrum of the voiced sound
subplot(313);
plot(F_v_hann,20*log10(abs(Y_v_hann)));
xlim([0, fs/2]);
xlabel('frequency(Hz)');
title('Log Magnitude Spectrum(Hanning) - Voiced sound');
```




```
%D-b

%3ms
y_v = y(ceil(0.239*fs) : floor(0.242*fs));
Y_v = fftshift(fft(y_v));
F_v = -fs/2 : fs/length(Y_v) : fs/2 - fs/length(Y_v);

%plotting Log Magnitude Spectrum of the voiced sound
figure;
subplot(311);
plot(F_v,20*log10(abs(Y_v)));
xlim([0, fs/2]);
xlabel('frequency(Hz)');
title('Log Magnitude Spectrum(3ms) - Voiced sound');

%30ms
y_v = y(ceil(0.222*fs) : floor(0.252*fs));
Y_v = fftshift(fft(y_v));
F_v = -fs/2 : fs/length(Y_v) : fs/2 - fs/length(Y_v);

%plotting Log Magnitude Spectrum of the voiced sound
```

```
subplot(312);
plot(F_v,20*log10(abs(Y_v)));
xlim([0, fs/2]);
xlabel('frequency(Hz)');
title('Log Magnitude Spectrum(30ms) - Voiced sound');

%300ms
y_v = y(ceil(0.108*fs) : floor(0.408*fs));
Y_v = fftshift(fft(y_v));
F_v = -fs/2 : fs/length(Y_v) : fs/2 - fs/length(Y_v);

%plotting Log Magnitude Spectrum of the voiced sound
subplot(313);
plot(F_v,20*log10(abs(Y_v)));
xlim([0, fs/2]);
xlabel('frequency(Hz)');
title('Log Magnitude Spectrum(300ms) - Voiced sound');
```



Observations:

We observed that, the STFT using the rectangular spectra is found to be more noisy compared to STFT spectra due to other window functions. This is due to the higher spectral leakage in rectangular window compared to other window functions. The width of main lose in the frequency domain is less for rectangular window compared to the other two window function. As a result the resolution offered by the rectangular window function is better. Alternatively, the peak-to-side lobe ratio of rectangular window is significantly poor compared to the other two window functions. This results in relatively more spectral leakage in case of rectangular window

which is not desirable. Thus from the resolution point of view, rectangular window is preferable and from spectral leakage point of other two window functions are preferable.

Here we can observe that the poor spectral resolution due to smaller window size in case of 3 ms window. Also poor time resolution can be observed in case of 300 ms window.The log magnitude spectrum of the 30 ms windowed speech segment shows vocal tract spectral envelope and excitation information in terms of pitch and its harmonics.

```
function [STFT, f, t] = stft(x, win, hop, nfft, fs)
% function: [STFT, f, t] = stft(x, win, hop, nfft, fs)
%
% Input:
% x - signal in the time domain
% win - analysis window function
% hop - hop size
% nfft - number of FFT points
% fs - sampling frequency, Hz
%
% Output:
% STFT - STFT-matrix (only unique points, time
%        across columns, frequency across rows)
% f - frequency vector, Hz
% t - time vector, s
% representation of the signal as column-vector
x = x(:);
% determination of the signal length
xlen = length(x);
% determination of the window length
wlen = length(win);
% stft matrix size estimation and preallocation
NUP = ceil((1+nfft)/2);     % calculate the number of unique fft points
L = 1+fix((xlen-wlen)/hop); % calculate the number of signal frames
STFT = zeros(NUP, L);       % preallocate the stft matrix
% STFT (via time-localized FFT)
for l = 0:L-1
    % windowing
    xw = x(1+l*hop : wlen+l*hop).*win;

    % FFT
    X = fft(xw, nfft);

    % update of the stft matrix
    STFT(:, 1+l) = X(1:NUP);
end
% calculation of the time and frequency vectors
t = (wlen/2:hop:wlen/2+(L-1)*hop)/fs;
f = (0:NUP-1)*fs/nfft;
end
```