EL 7133 (DSP II)
Spring 2016
Homework Assignment - Week 05

Name: **Amitesh Kumar Sah**
NYU ID: **N19714360**

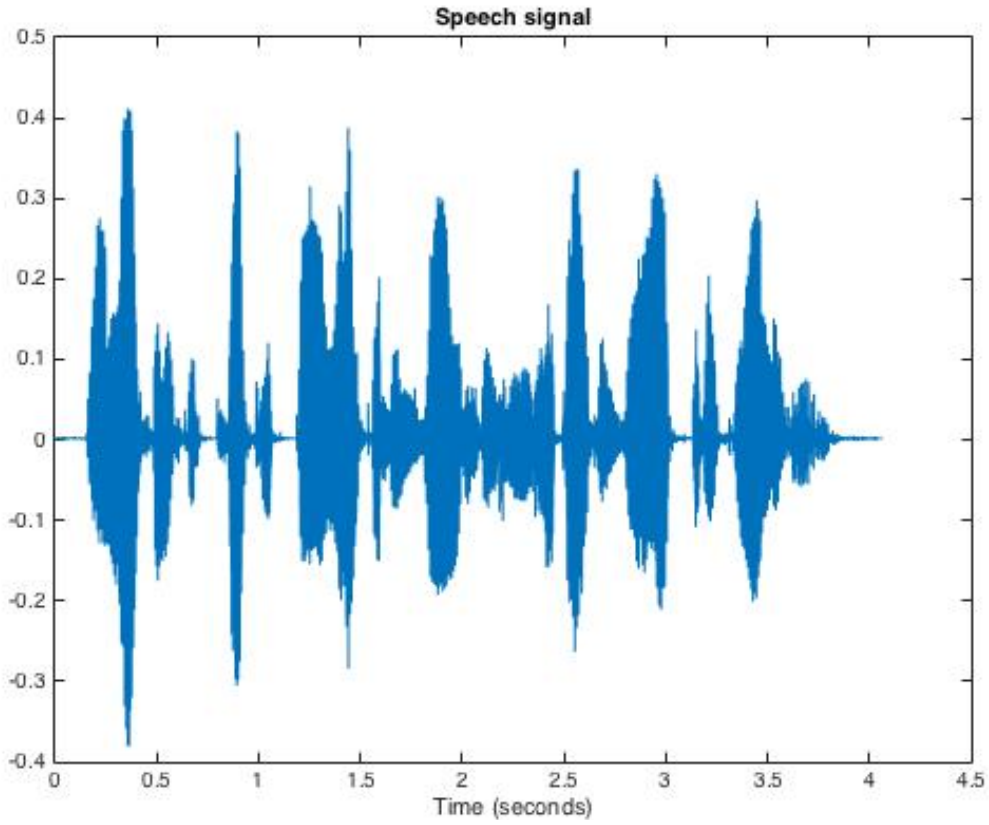
Question 1:

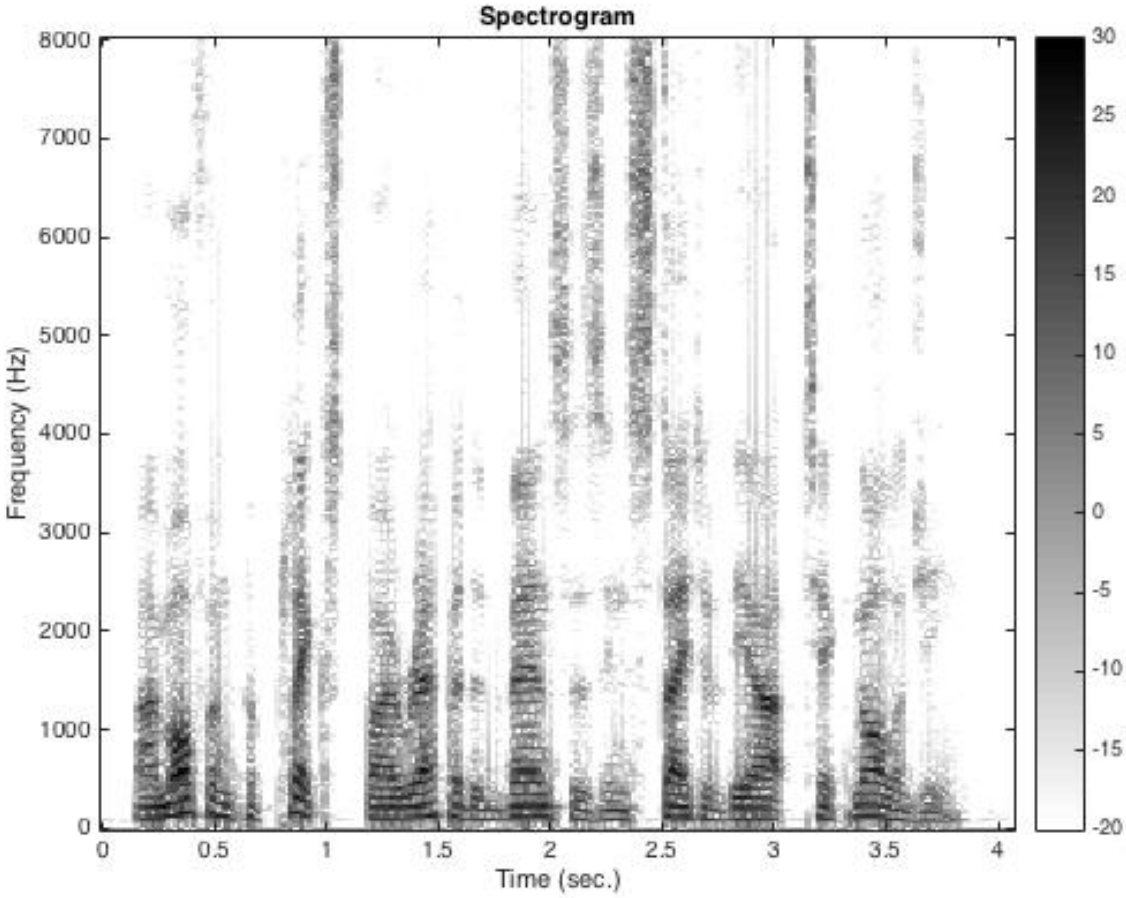
Write a MATLAB program to implement the STFT with 50% overlapping and a second program to implement its inverse. Verify numerically that the inverse program reconstructs a test signal.

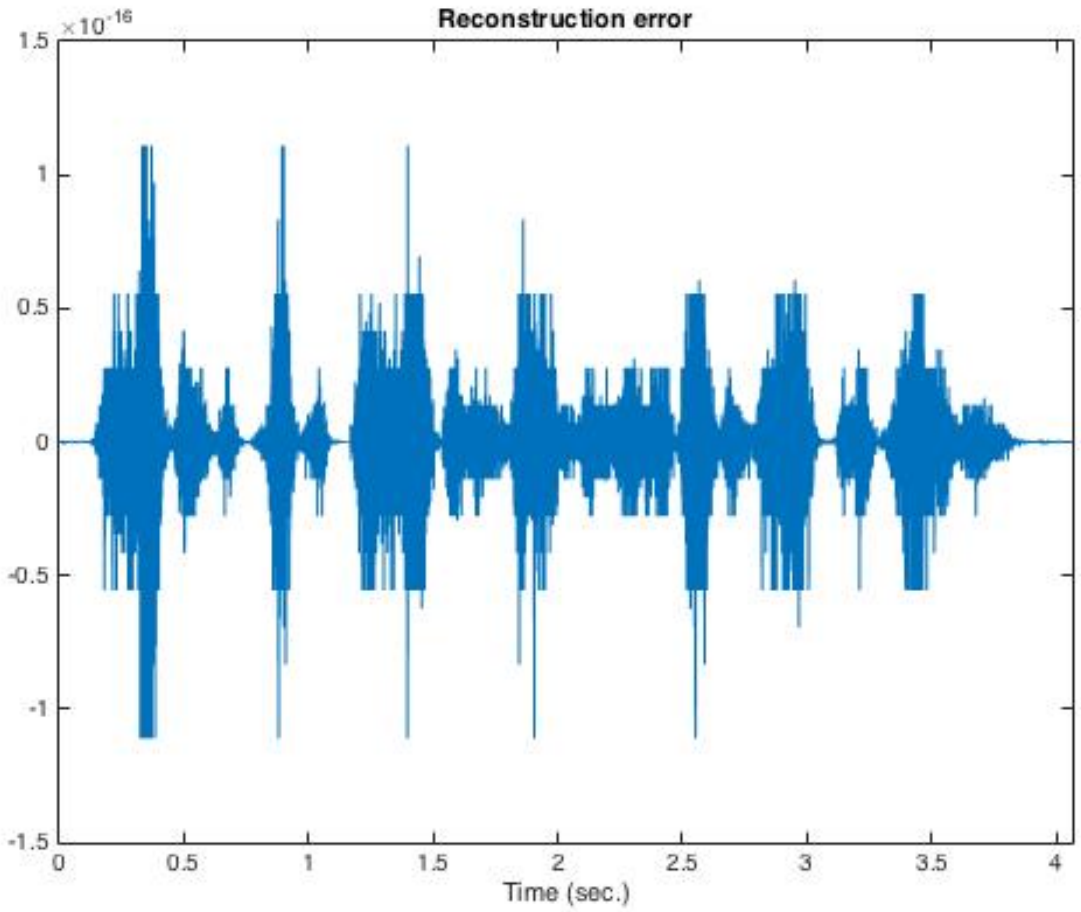
Matlab Code:

```
% Read a signal or audio file
[s,fs]=audioread('sp1.wav');

% Length of audio signal
N=length(s);

% Block length
R=512;

% Overlapping is 50% , Number of block calculation
Nb=floor(N/(R/2))-1;

% Plotting the speech signal
x=s(1:(Nb+1)*(R/2));
N = length(x);
figure,plot((1:N)/fs, x)
title('Speech signal')
xlabel('Time (seconds)')

soundsc(x, fs);    % Play the audio file

%  FORWARD STFT
X = zeros(R, Nb);
i = 0;
for k = 1:Nb
   X(:,k) = x(i + (1:R));
   i = i + R/2;
end
X = fft(X);              % Compute the FFT of each block
Tr = R/fs;               % Duration of each block (in seconds)

% plot spectrogram
figure,imagesc([0 N/fs], [0 fs/2], 20*log10(abs(X(1:R/2, :))));
cmap = flipud(gray);
```

```
colormap(cmap);
colorbar
axis xy
xlabel('Time (sec.)')
ylabel('Frequency (Hz)')
title('Spectrogram')
caxis([-20 30])

% INVERSE Stft

Y = ifft(X);       % inverse FFT of each column of X
y = zeros(size(x));
i = 0;
for k = 1:Nb
   y(i+(1:R),1)=y(i+(1:R),1)+0.5*Y(:, k);
   i = i + R/2;
end
% Take care of first half-block
y(1:R/2) = 2*y(1:R/2);
% Take care of last half-block
y(end-R/2+(1:R/2)) = 2*y(end-R/2+(1:R/2));
z=x-y;
% Display difference signal to verify perfect reconstruction
figure,plot((1:N)/fs,z)
title('Reconstruction error')
xlabel('Time (sec.)')
xlim([0 N/fs])
% ylim([-1 1])
```


Speech signal
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
Time (seconds)


Spectrogram
8000
7000
6000
5000
4000
3000
2000
1000
0
Frequency (Hz)
0
0.5
1
1.5
2
2.5
3
3.5
4
Time (sec.)
30
25
20
15
10
5
0
-5
-10
-15
-20



Conclusion

Here, I took a speech signal as input and implemented STFT with 50 % overlapping and plotted the spectrogram. I also implemented the inverse STFT. Further, I took the difference of original signal and reconstructed signal. We get error that tends to zero. So there is a perfect reconstruction.

Question 2

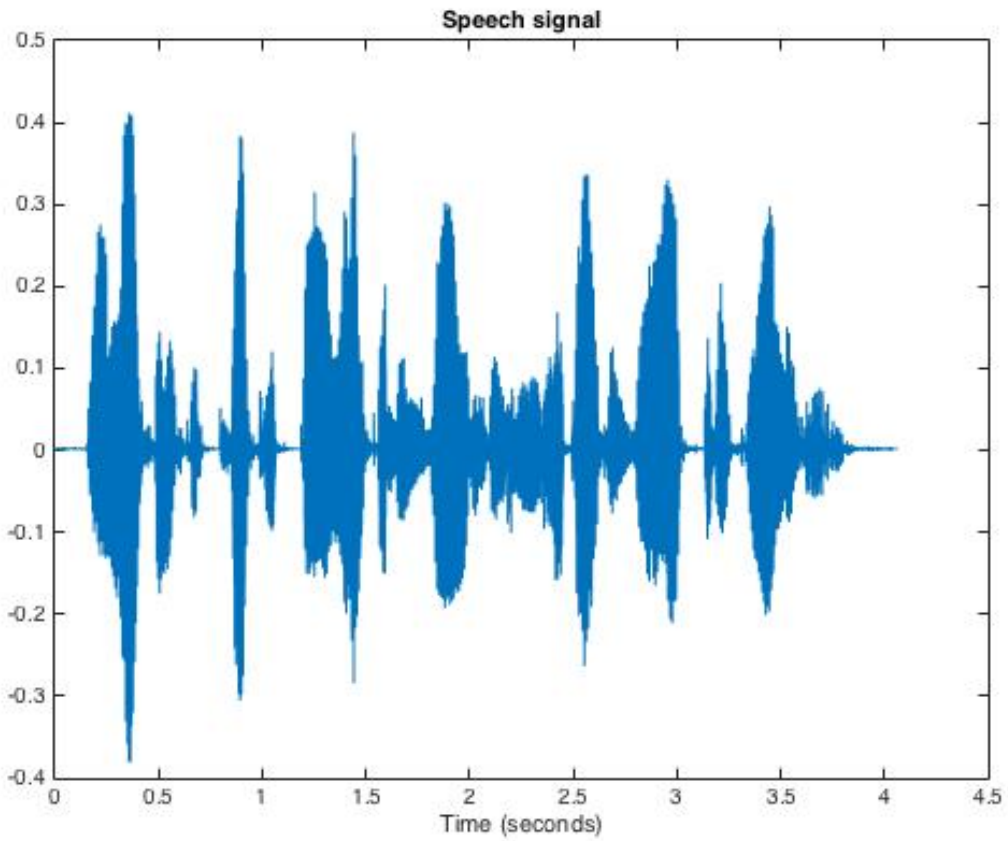
Find another window satisfying the perfect reconstruction condition for 50% overlapping, use it with your STFT program, and verify that it reconstructs a test signal.

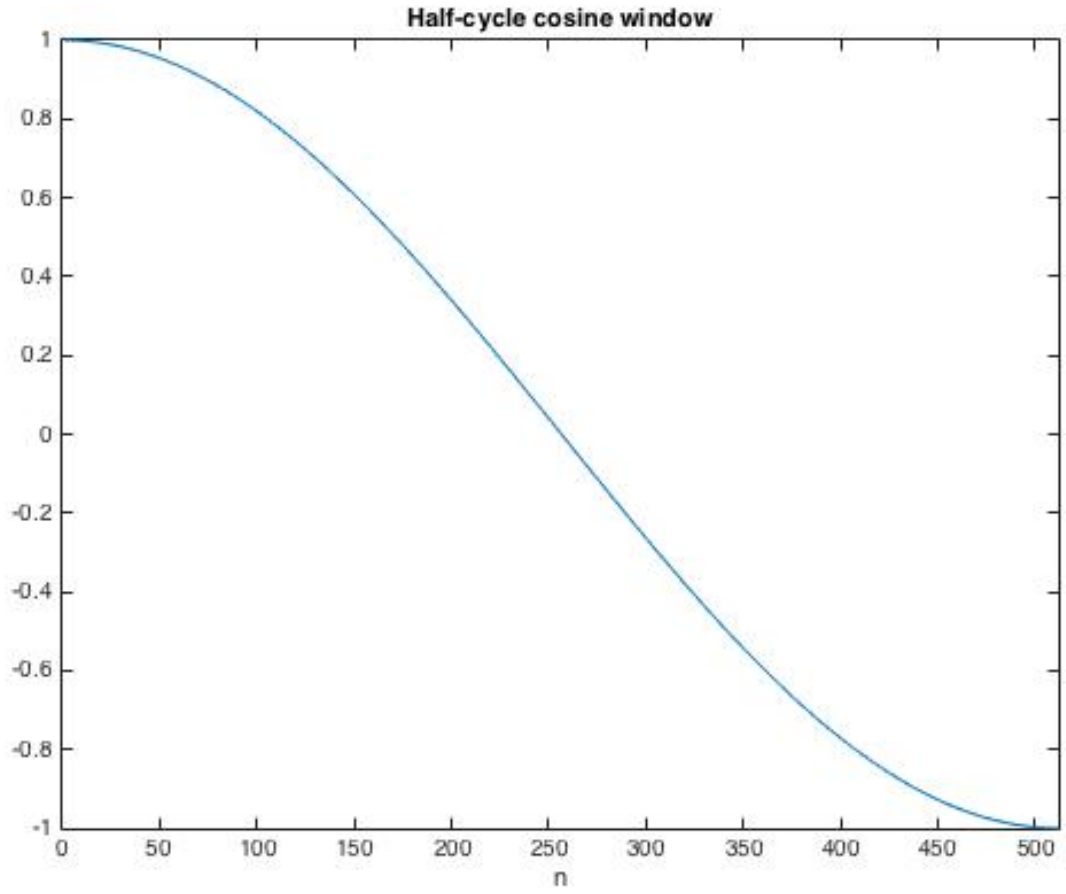
Matlab Program

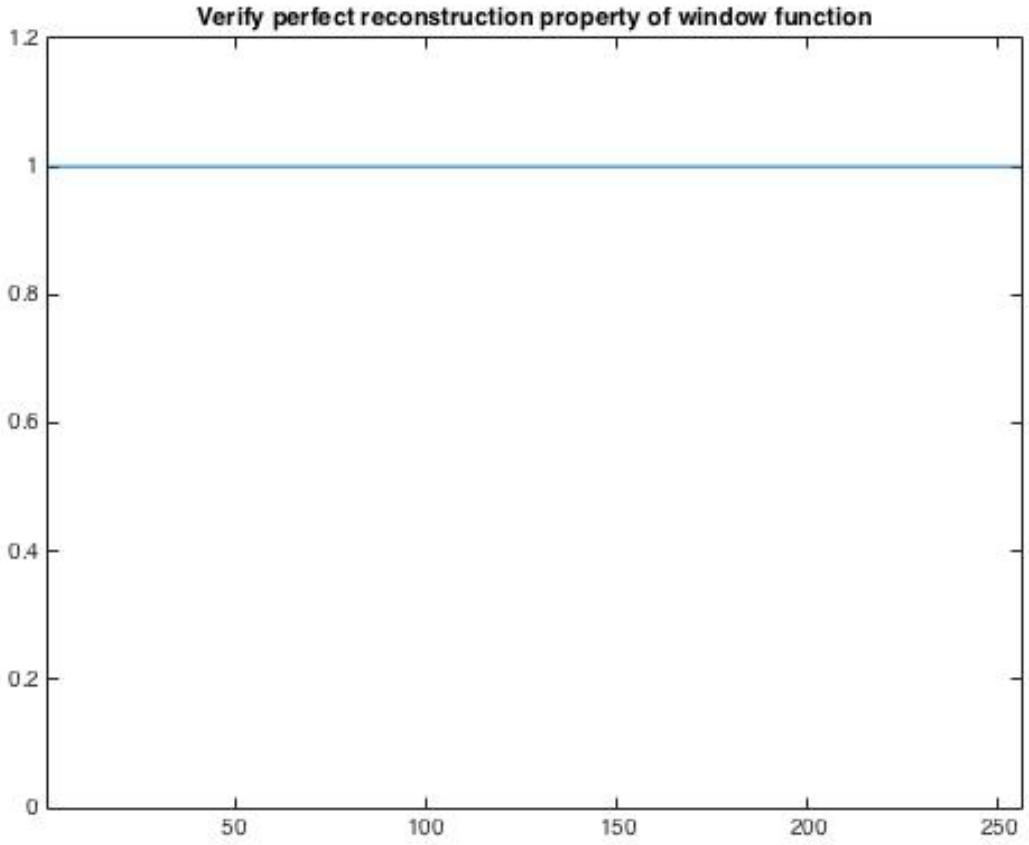
```
clc;clear all; close all

% Read a signal or audio file
[s,fs]=audioread('sp1.wav');
```

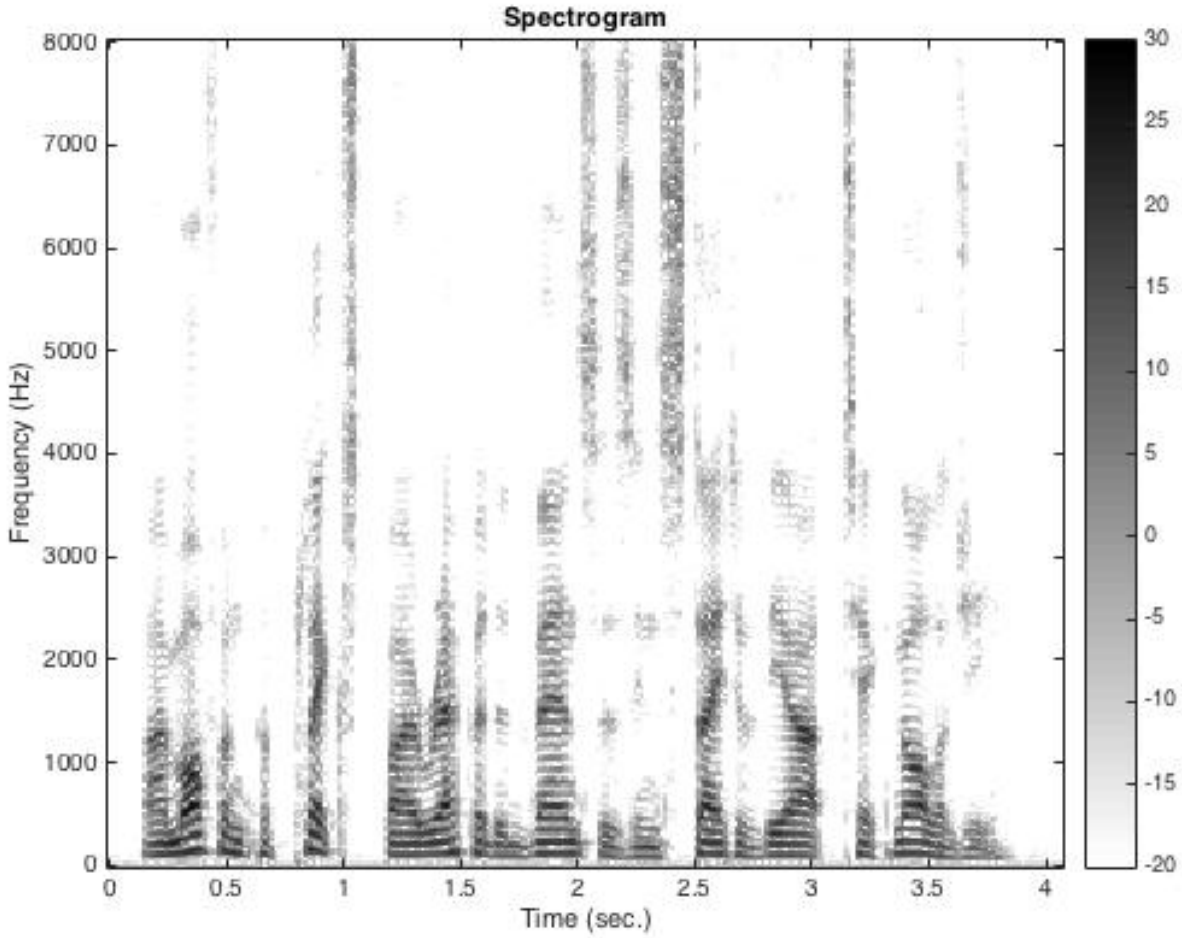
```
% Length of audio signal
N=length(s);

% Block length
R=512;

% Overlapping is 50% , Number of block calculation
Nb=floor(N/(R/2))-1;

% Plotting the speech signal
x=s(1:(Nb+1)*(R/2));
N = length(x);
figure,plot((1:N)/fs, x)
title('Speech signal')
xlabel('Time (seconds)')

soundsc(x, fs);    % Play the audio file

% half-cycle cosine window
n = (1:R) - 0.5;
w  = cos(pi*n/R);
figure,plot(w)
xlim([0 R])
title('Half-cycle sine window')
xlabel('n')

% perfect reconstruction (PR) property:
% w(n)^2 + w(n+R/2)^2 = 1.
pr = w(1:R/2).^2 + w(R/2+(1:R/2)).^2;
plot(pr)
title('Verify perfect reconstruction property of window function')
ylim([0 1.2])
xlim([1 R/2])

%  FORWARD STFT
X = zeros(R, Nb);
i = 0;
for k = 1:Nb
   X(:,k) = w'.*x(i + (1:R));
   i = i + R/2;
end
X = fft(X);              % Compute the FFT of each block
Tr = R/fs;               % Duration of each block (in seconds)

% plot spectrogram
figure,imagesc([0 N/fs], [0 fs/2], 20*log10(abs(X(1:R/2, :))));
cmap = flipud(gray);
colormap(cmap);
colorbar
axis xy
xlabel('Time (sec.)')
```

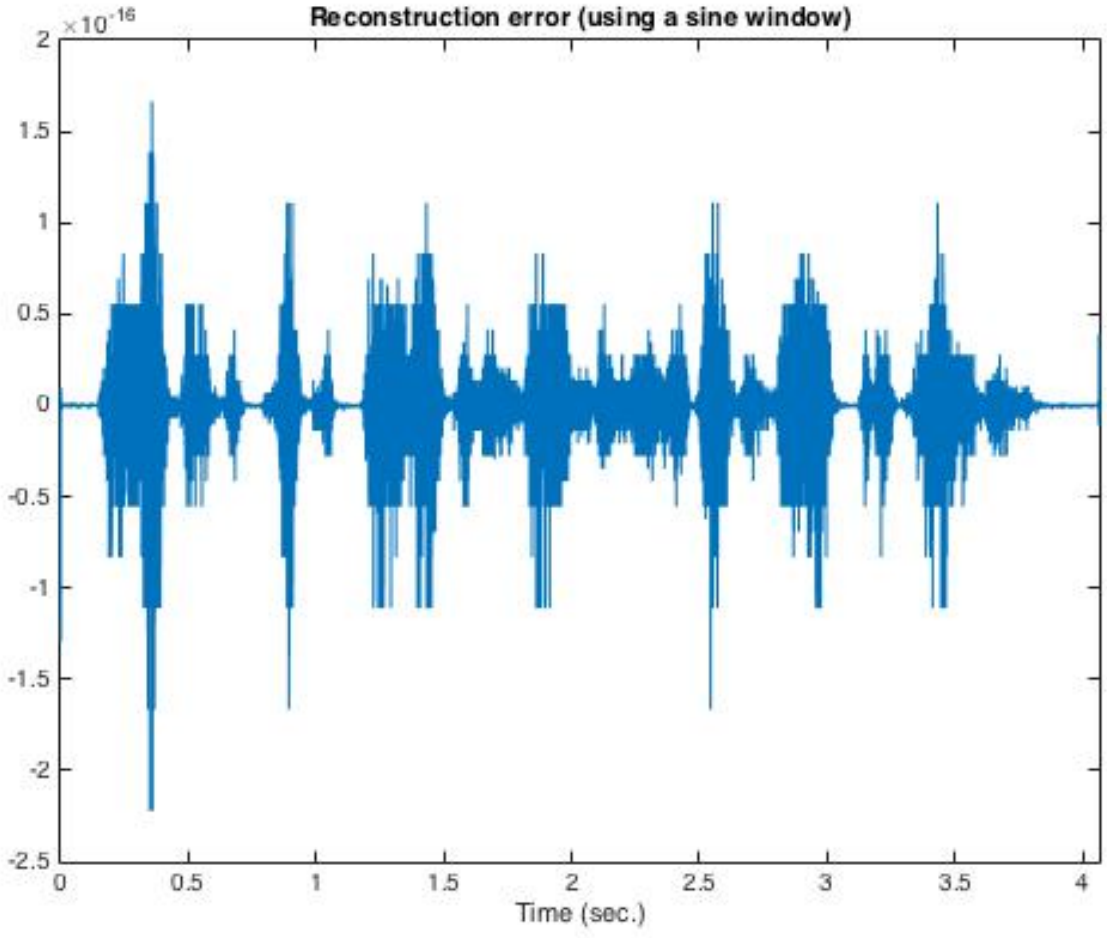
```
ylabel('Frequency (Hz)')
title('Spectrogram')
caxis([-20 30])

% INVERSE Stft

Y = ifft(X);       % inverse FFT of each column of X
y = zeros(size(x));
i = 0;
for k = 1:Nb
   y(i + (1:R)) = y(i + (1:R)) + w' .* Y(:,k);
   i = i + R/2;
end
% Take care of first half-block
y(1:R/2) = y(1:R/2) ./ (w(1:R/2).^2)';
% Take care of last half-block
y(end-R/2+(1:R/2)) = y(end-R/2+(1:R/2)) ./ (w(R/2+1:R).^2)';
% Display difference signal to verify perfect reconstruction
figure,plot((1:N)/fs,x-y)
title('Reconstruction error (using a sine window)')
xlabel('Time (sec.)')
xlim([0 N/fs])
% ylim([-1 1])
```




Half-cycle cosine window
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
50
100
150
200
250
300
350
400
450
500
n


Verify perfect reconstruction property of window function
1.2
1
0.8
0.6
0.4
0.2
0
50
100
150
200
250


Spectrogram
Frequency (Hz)
Time (sec.)
8000
7000
6000
5000
4000
3000
2000
1000
0
0.5
1
1.5
2
2.5
3
3.5
4
30
25
20
15
10
5
0
-5
-10
-15
-20


Reconstruction error (using a sine window)
×10^-16
Time (sec.)
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2.5
0
0.5
1
1.5
2
2.5
3
3.5
4

Conclusion

Here, I used half cosine window that satisfies the perfect reconstruction condition for 50% overlapping. I implemented STFT with 50 % overlapping and plotted the spectrogram. I also implemented the inverse STFT. Further, I took the difference of original signal and reconstructed signal. We get error that tends to zero. So there is a perfect reconstruction.

Question 3

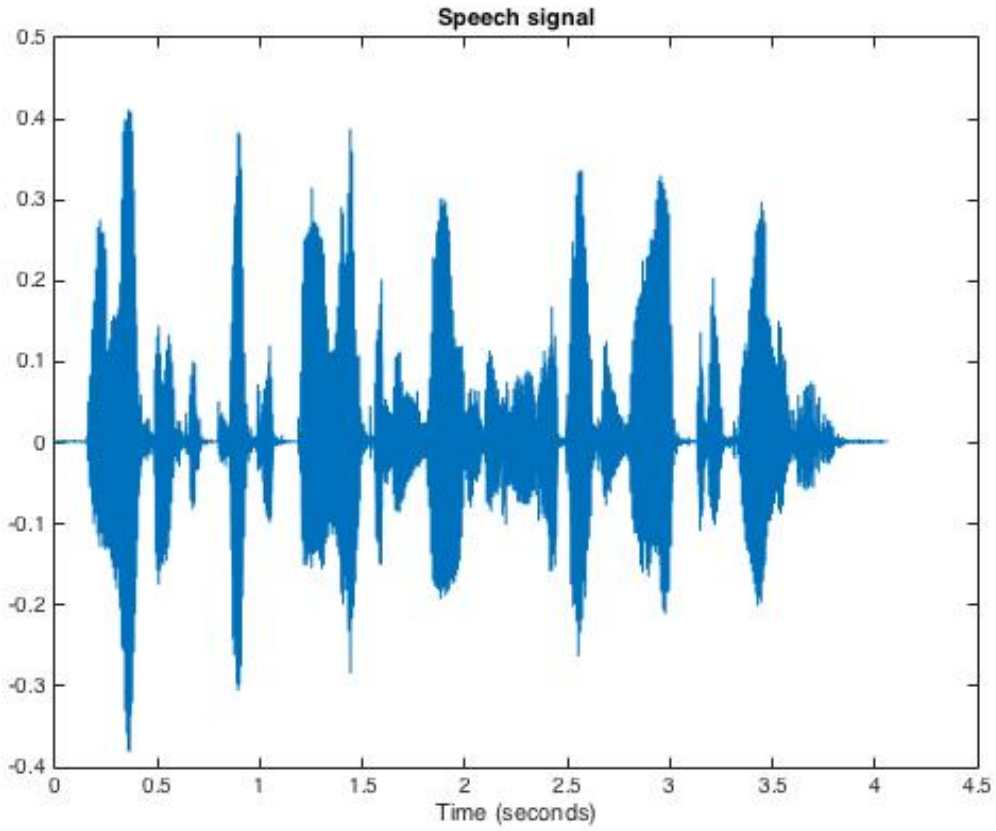
Find the perfect reconstruction (PR) condition for an STFT with 2/3 overlapping. Find a window that satisfies your PR condition. Write a MATLAB program to implement the STFT and its inverse with 2/3 overlapping and verify the reconstruction of a test signal.

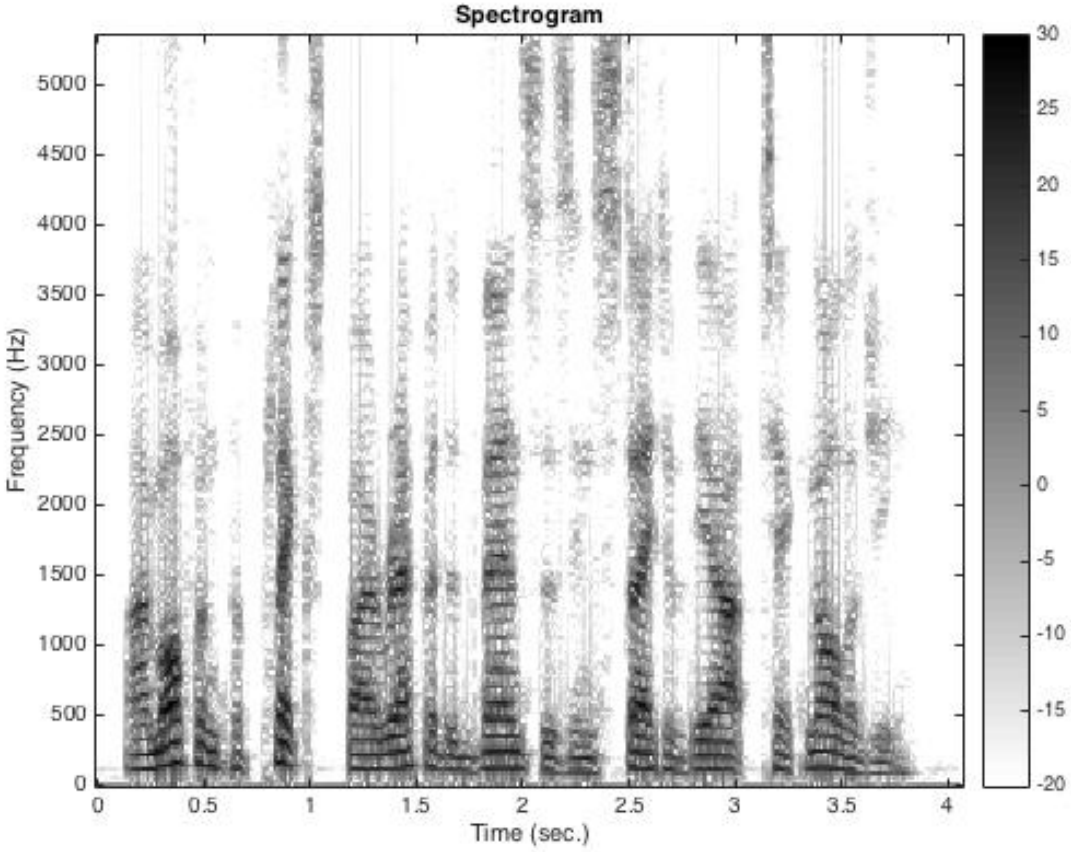
Matlab Code

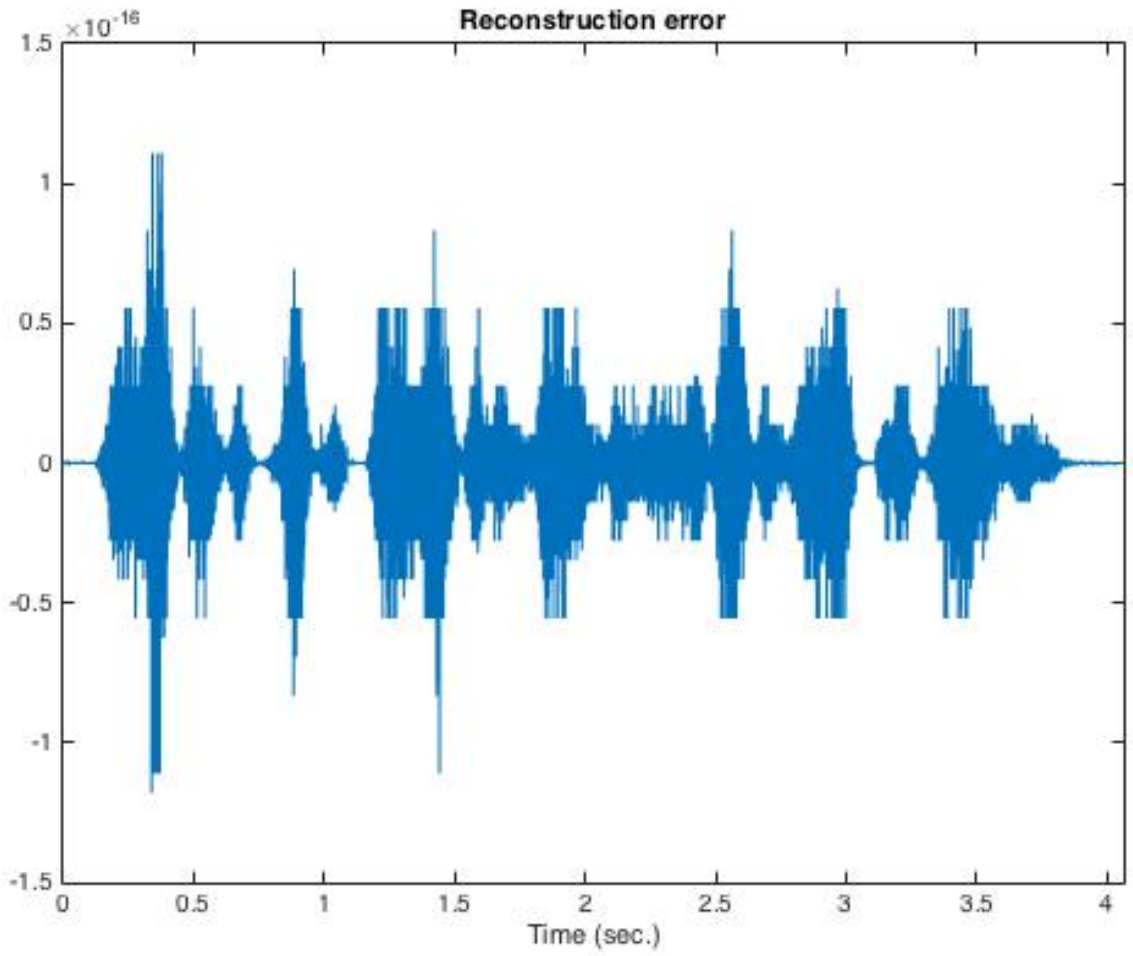
Without using a window

```
clc;clear all; close all

% Read a signal or audio file
[s,fs]=audioread('/Users/amitesh/Desktop/DSP 2/Assignment 5/Question 1/sp1.wav');
% [s,fs]=audioread('sp1.wav');

% Length of audio signal
N=length(s);
% Block length
R=600;

% Overlapping is 2/3 , Number of block calculation
Nb=floor(N/(R/3))-1;

% Plotting the speech signal
x=s(1:(Nb+1)*(R/3));
N = length(x);
figure,plot((1:N)/fs, x)
title('Speech signal')
xlabel('Time (seconds)')

soundsc(x, fs);    % Play the audio file


%  FORWARD STFT
X1 = zeros(R, Nb);
i = 0;
for k = 1:Nb-1
%    X1(:,k) = w'.*x(i + (1:R));
X1(:,k) = x(i + (1:R));
```

```
   i = i + R/3;
end
X = fft(X1);              % Compute the FFT of each block
Tr = R/fs;                % Duration of each block (in seconds)

% % plot spectrogram
figure,imagesc([0 N/fs], [0 fs/3], 20*log10(abs(X(1:R/3, :))));
cmap = flipud(gray);
colormap(cmap);
colorbar
axis xy
xlabel('Time (sec.)')
ylabel('Frequency (Hz)')
title('Spectrogram')
caxis([-20 30])

% INVERSE Stft

Y = ifft(X);       % inverse FFT of each column of X
y = zeros(size(x));
i = 0;
for k = 1:Nb-1
%    y(i + (1:R)) = y(i + (1:R)) + w' .* Y(:,k);
 y(i + (1:R)) = y(i + (1:R)) + Y(:,k)/3;
   i = i + R/3;
end
% Take care of first half-block
% y(1:R/3) = y(1:R/3) ./ (w(1:R/3).^2)';
y(1:R/3) = 3*y(1:R/3);
y(R/3+1:2*R/3) = 3*y(R/3+1:2*R/3)/2;

% Take care of last half-block
% y(end-2*R/3+(1:2*R/3)) = y(end-2*R/3+(1:2*R/3)) ./ (w(R/3+1:R).^2)';
y(end-2*R/3+(1:R/3)) = 3*y(end-2*R/3+(1:R/3))/2;
y(end-R/3+(1:R/3)) = 3*y(end-R/3+(1:R/3));

% Display difference signal to verify perfect reconstruction
figure,plot((1:N)/fs,x-y)
title('Reconstruction error')
xlabel('Time (sec.)')
xlim([0 N/fs])
% ylim([-1 1])
```


Speech signal
0.5
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
-0.4
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
Time (seconds)


Spectrogram
Frequency (Hz)
5000
4500
4000
3500
3000
2500
2000
1500
1000
500
0
0
0.5
1
1.5
2
2.5
3
3.5
4
Time (sec.)
30
25
20
15
10
5
0
-5
-10
-15
-20

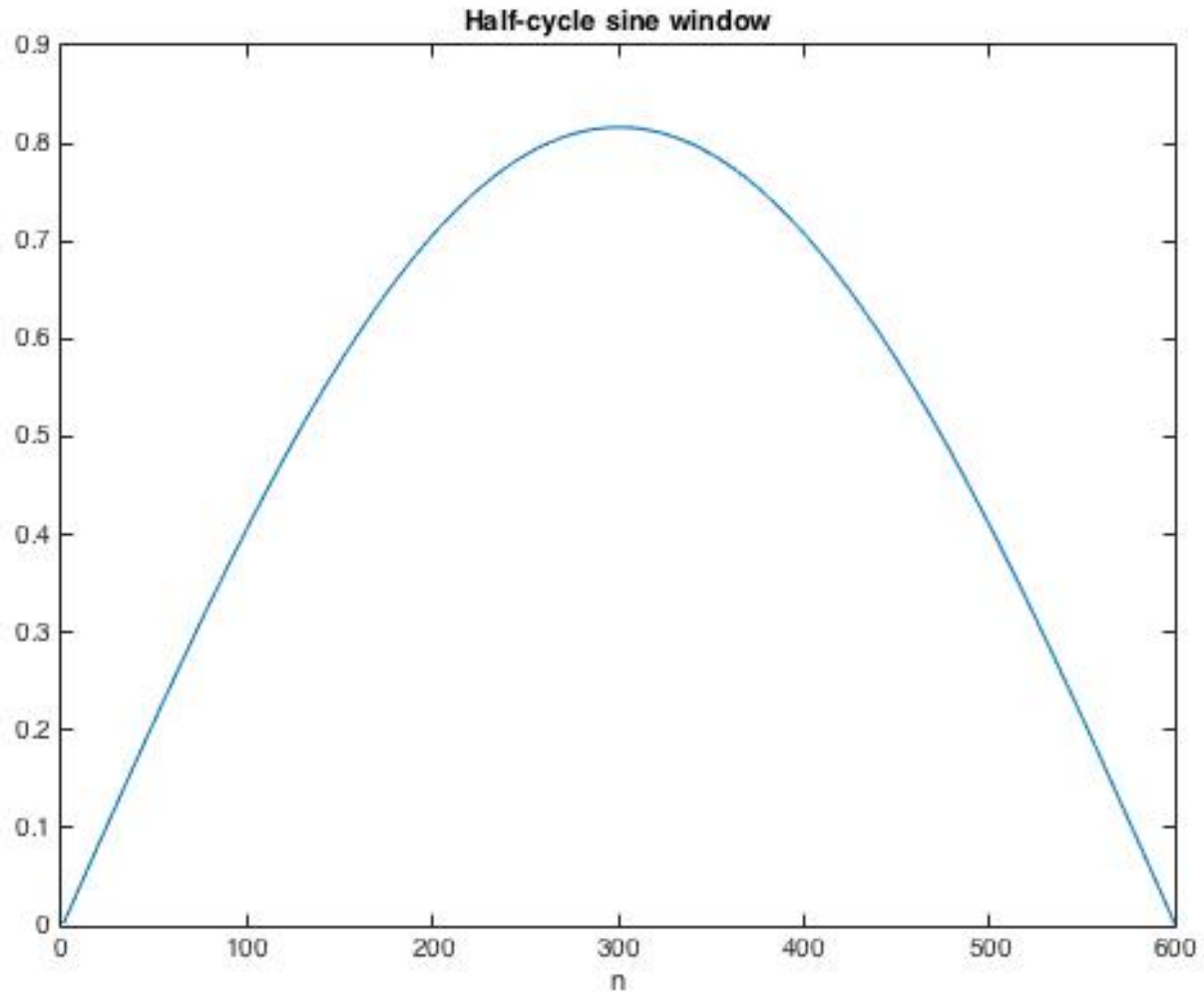


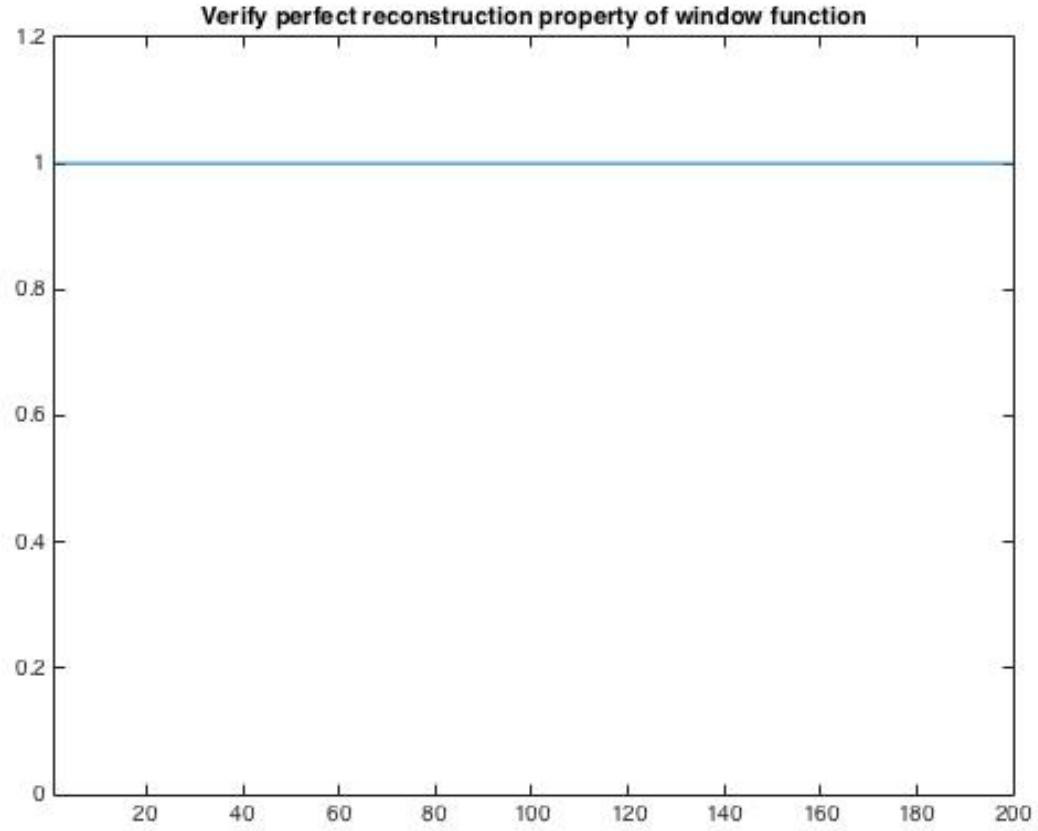
With using half sine window

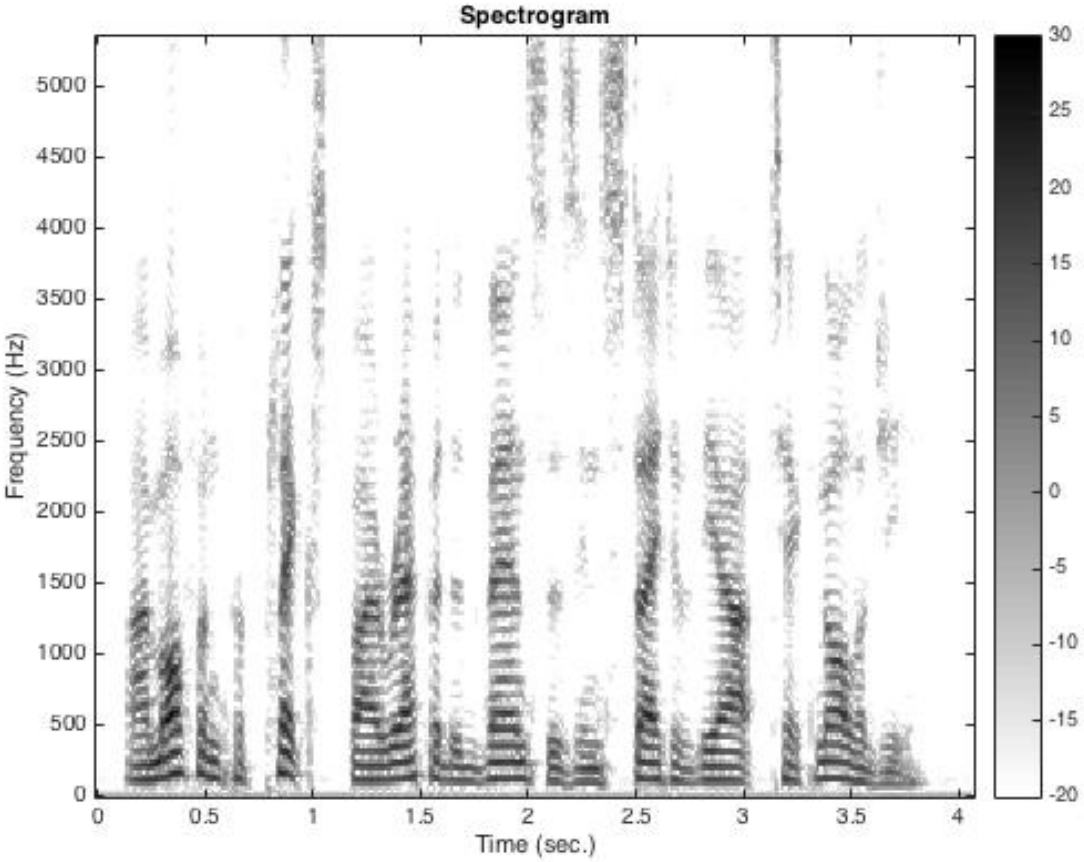
```
clc;clear all; close all

% Read a signal or audio file
[s,fs]=audioread('/Users/amitesh/Desktop/DSP 2/Assignment 5/Question 1/sp1.wav');
% [s,fs]=audioread('sp1.wav');

% Length of audio signal
N=length(s);
% Block length
R=600;

% Overlapping is 2/3 , Number of block calculation
Nb=floor(N/(R/3))-1;

% Plotting the speech signal
x=s(1:(Nb+1)*(R/3));
N = length(x);
figure,plot((1:N)/fs, x)
title('Speech signal')
xlabel('Time (seconds)')
```

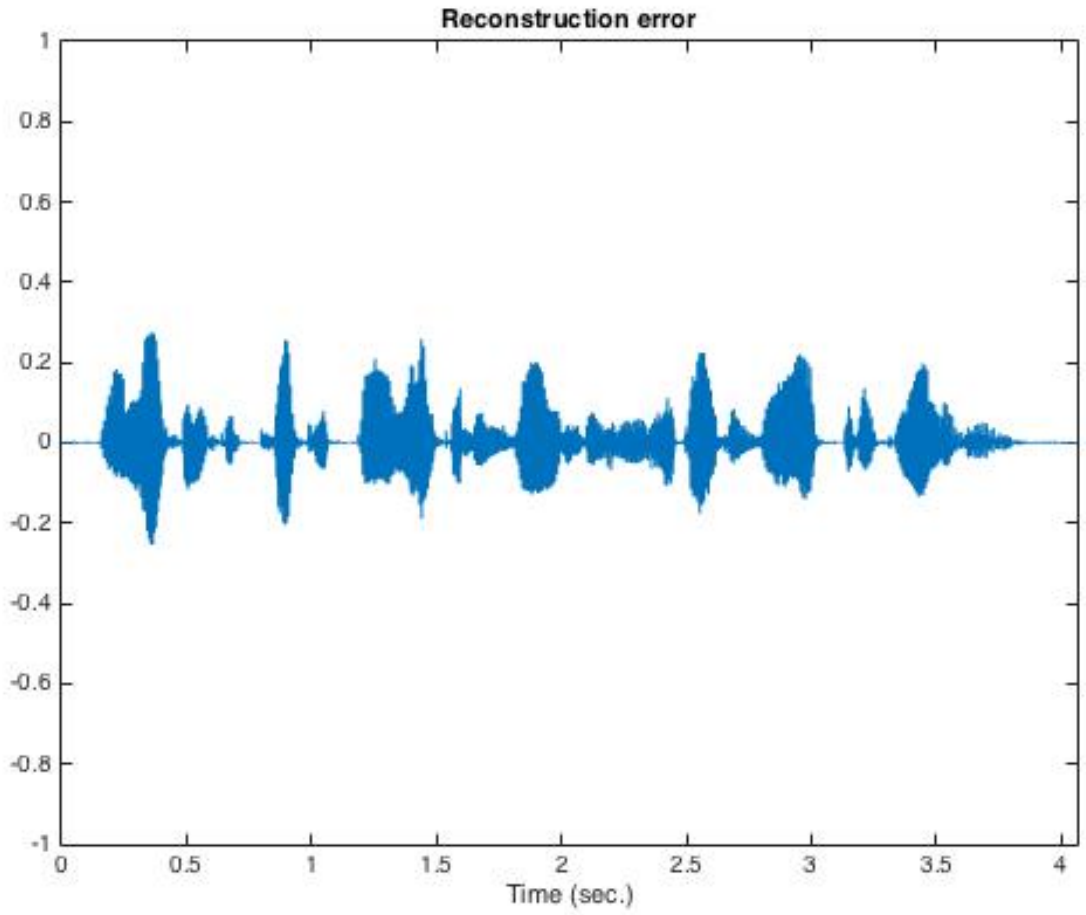
```
soundsc(x, fs);    % Play the audio file

% half-cycle cosine window
n = (1:R) - 0.5;
w  = 2*sin(pi*n/R)/sqrt(6);
figure,plot(w)
xlim([0 R])
title('Half-cycle sine window')
xlabel('n')

% perfect reconstruction (PR) property:
% w(n)^2 + w(n+R/2)^2 = 1.
pr = w(1:R/3).^2 + w(R/3+(1:R/3)).^2+w(2*R/3+(1:R/3)).^2;
figure,plot(pr)
title('Verify perfect reconstruction property of window function')
ylim([0 1.2])
xlim([1 R/3])

%  FORWARD STFT
X1 = zeros(R, Nb);
i = 0;
for k = 1:Nb-1
%    X1(:,k) = w'.*x(i + (1:R));
X1(:,k) = w'.*x(i + (1:R));
   i = i + R/3;
end
X = fft(X1);              % Compute the FFT of each block
Tr = R/fs;                % Duration of each block (in seconds)

% % plot spectrogram
figure,imagesc([0 N/fs], [0 fs/3], 20*log10(abs(X(1:R/3, :))));
cmap = flipud(gray);
colormap(cmap);
colorbar
axis xy
xlabel('Time (sec.)')
ylabel('Frequency (Hz)')
title('Spectrogram')
caxis([-20 30])

% INVERSE Stft

Y = ifft(X);       % inverse FFT of each column of X
y = zeros(size(x));
i = 0;
for k = 1:Nb-1
%    y(i + (1:R)) = y(i + (1:R)) + w' .* Y(:,k);
 y(i + (1:R)) = y(i + (1:R)) + w'.*Y(:,k)./3;
   i = i + R/3;
end
% Take care of first half-block
```

```
y(1:R/3) = 3*y(1:R/3)./ (w(1:R/3).^2)';
y(R/3+1:2*R/3) = (3*y(R/3+1:2*R/3)/2)./(w(R/3+1:2*R/3).^2)';

% Take care of last half-block

y(end-2*R/3+(1:R/3)) = (3*y(end-2*R/3+(1:R/3))/2)./(w(R/3+(1:R/3)).^2)';
y(end-R/3+(1:R/3)) = (3*y(end-R/3+(1:R/3)))./(w(2*R/3+(1:R/3)).^2)';


% Display difference signal to verify perfect reconstruction
figure,plot((1:N),x-y)
title('Reconstruction error')
xlabel('Time (sec.)')
xlim([0 N])
ylim([-1 1])
```




Verify perfect reconstruction property of window function
1.2
1
0.8
0.6
0.4
0.2
0
20
40
60
80
100
120
140
160
180
200


Spectrogram
Frequency (Hz)
5000
4500
4000
3500
3000
2500
2000
1500
1000
500
0
0
0.5
1
1.5
2
2.5
3
3.5
4
Time (sec.)
30
25
20
15
10
5
0
-5
-10
-15
-20



Conclusion

Here I implemented STFT and Inverse STFT using 2/3 overlapping. I verified the result obtained is correct as the difference in the original signal and the reconstructed signal tends to zero. The 1st part was done without using window.

In the second part I used a half sine window and I am getting a perfect reconstruction . However, I am getting a lot of error after reconstruction of the signal.

Question 4

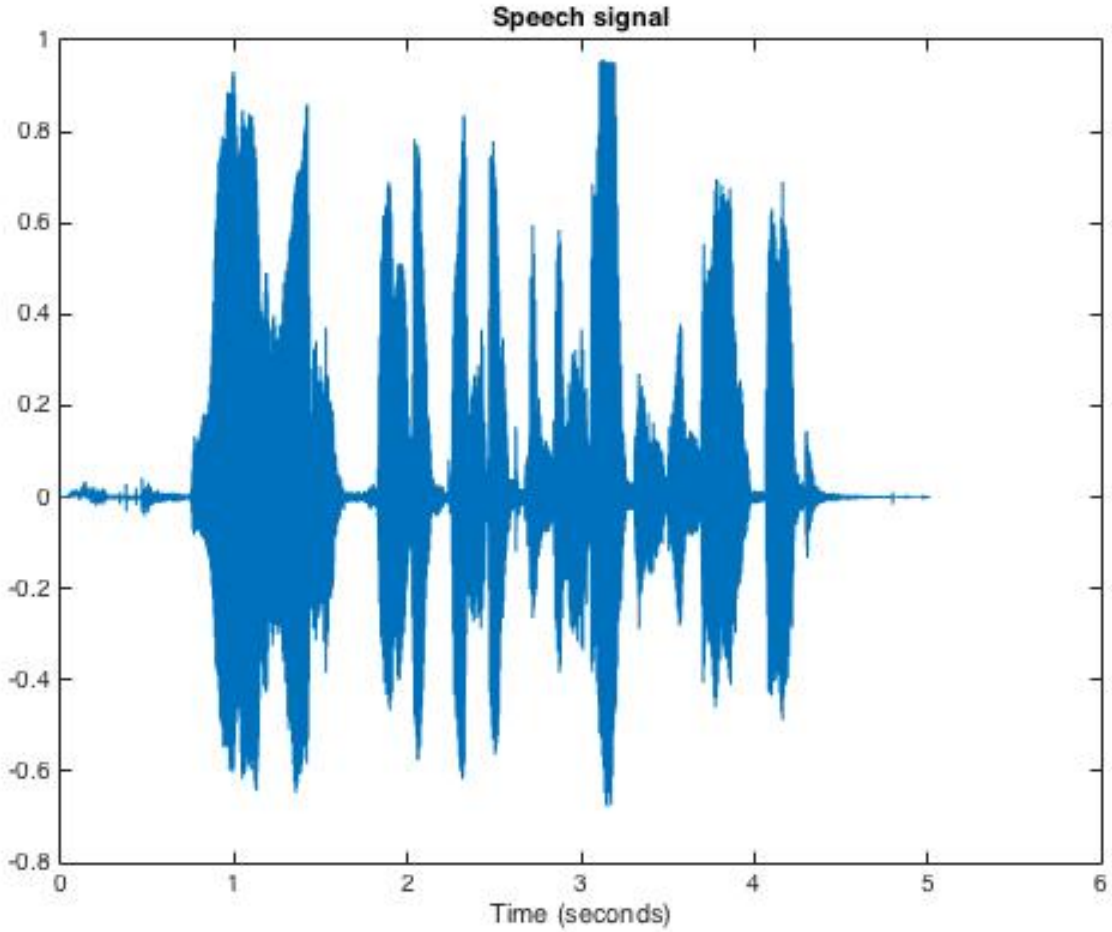
Use the STFT and thresholding to reduce the noise level in a noisy speech signal. Try to use your own speech signal recorded using a computer (most audio recordings with consumer equipment contain some hiss; try to reduce the sound of the hiss). In case your STFT program does not work, you may use the provided function my_stft provided as a .p file. A .p file is a type of MATLAB file that can be executed in MATLAB but whose contents are not viewable (see the pcode function).

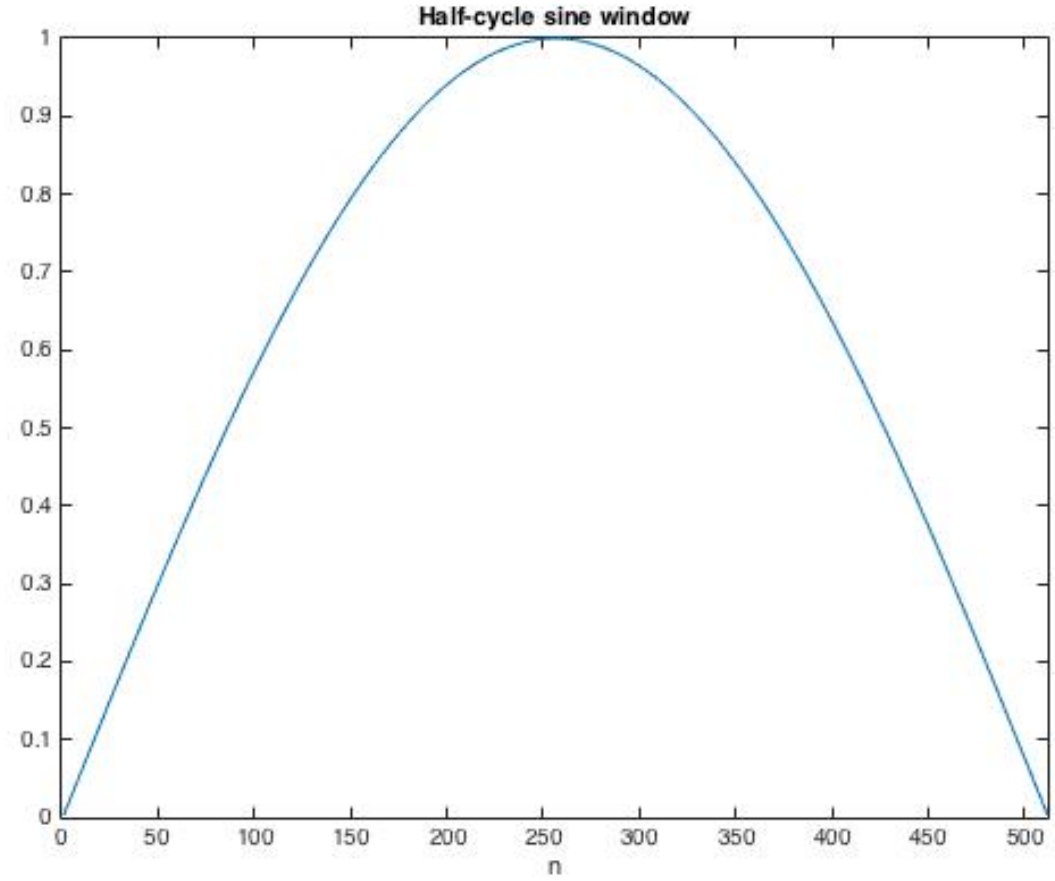
Matlab Code

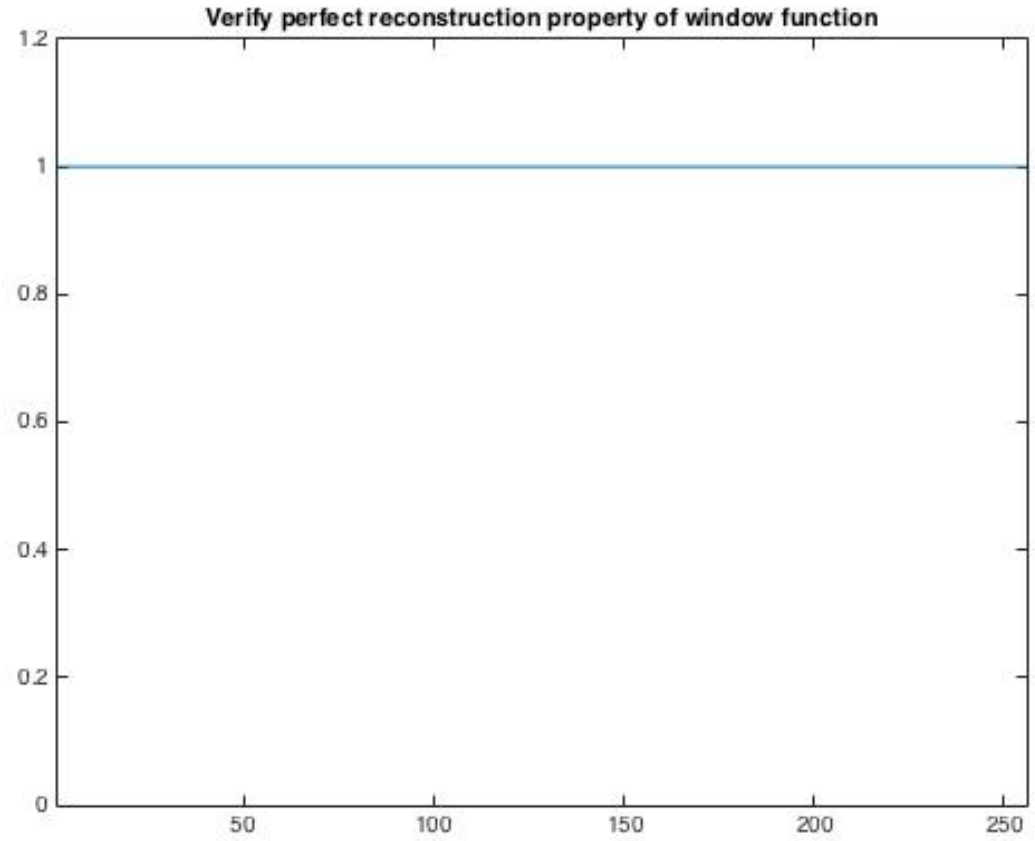
```
clc;clear all;close all
[s, fs] = audioread('noise_audio.mp3');
% Length of audio signal
N=length(s);
% Block length
R=512;
% Overlapping is 50% , Number of block calculation
Nb=floor(N/(R/2))-1;

% Plotting the speech signal
x=s(1:(Nb+1)*(R/2));
N = length(x);
figure,plot((1:N)/fs, x)
title('Speech signal')
xlabel('Time (seconds)')

% STFT of noisy speech signal

 soundsc(x, fs);    % Play the audio file

% half-cycle sine window
n = (1:R) - 0.5;
w  = sin(pi*n/R);
figure,plot(w)
xlim([0 R])
title('Half-cycle sine window')
xlabel('n')

pr = w(1:R/2).^2 + w(R/2+(1:R/2)).^2;
figure,plot(pr)
title('Verify perfect reconstruction property of window function')
ylim([0 1.2])
xlim([1 R/2])

%  FORWARD STFT
X = zeros(R, Nb);
i = 0;

for k = 1:Nb
   X(:,k) = x(i+(1:R)).*w';
   i = i + R/2;
end
X = fft(X)./sqrt(R);              % Compute the FFT of each block
Tr = R/fs;               % Duration of each block (in seconds)

% plot spectrogram
figure,imagesc([0 N/fs], [0 fs/2], 20*log10(abs(X(1:R/2, :))));
cmap = flipud(gray);
```

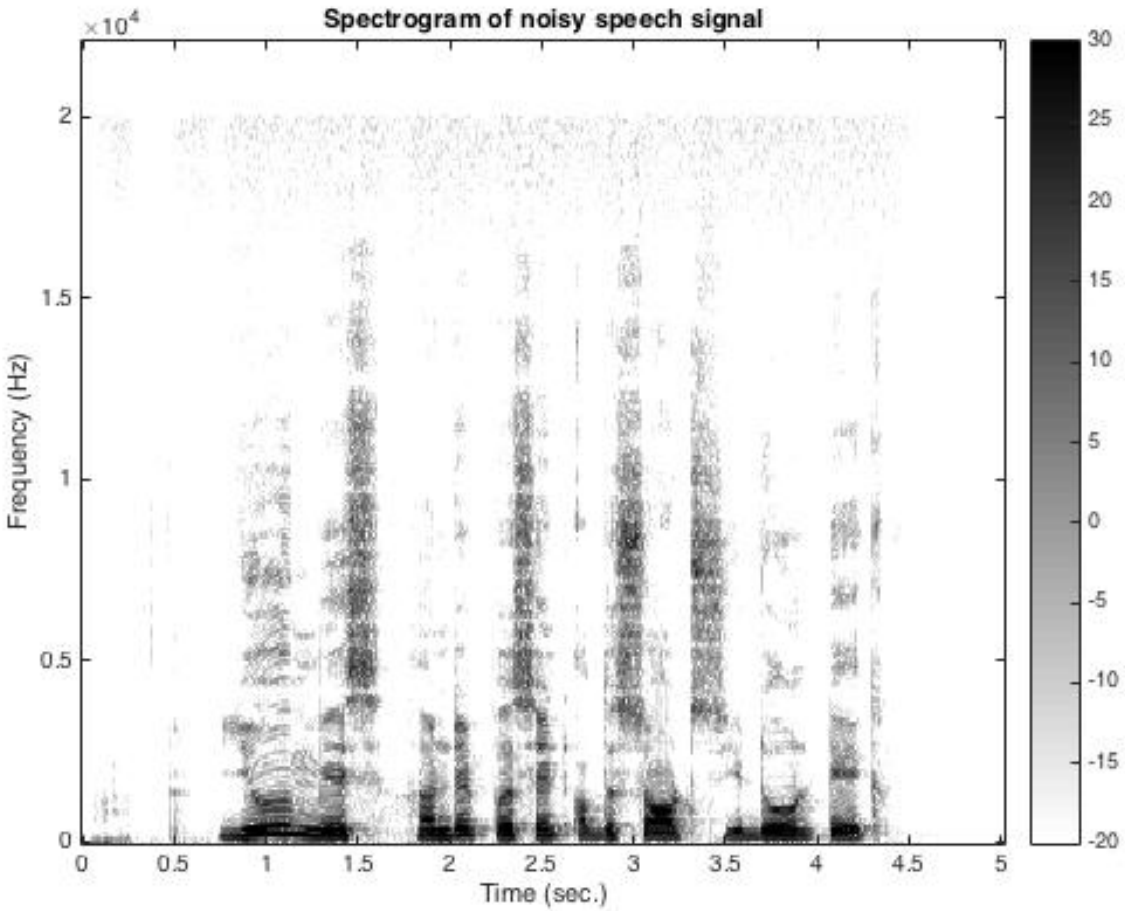
```
colormap(cmap);
colorbar
axis xy
xlabel('Time (sec.)')
ylabel('Frequency (Hz)')
caxis([-20 30])
title('Spectrogram of noisy speech signal')


% Apply threshold in STFT domain

Y = X;
Threshold = 0.35;
k = abs(Y) < Threshold;
Y(k) = 0;
Y(:,1) = 0;   % remove start transients
Y(:,end) = 0;   % remove end transients

% display STFT
figure,imagesc([0 N/fs], [0 fs/2], 20*log10(abs(Y(1:R/2, :))));
colormap(cmap);
caxis([-20 30])
colorbar
axis xy
xlabel('Time (sec.)')
ylabel('Frequency (Hz)')
title('Spectrogram after thresholding')

%% Denoised speech signal

% y = my_istft(Y);
Y1 = ifft(Y).*sqrt(R);       % inverse FFT of each column of X
y = zeros(size(x));
i = 0;
for k = 1:Nb
   y(i + (1:R)) = y(i + (1:R)) + w' .* Y1(:,k);
   i = i + R/2;
end
% Take care of first half-block
y(1:R/2) = y(1:R/2) ./ (w(1:R/2).^2)';
% Take care of last half-block
y(end-R/2+(1:R/2)) = y(end-R/2+(1:R/2)) ./ (w(R/2+1:R).^2)';
% Display difference signal to verify perfect reconstruction
figure,plot((1:N)/fs,x-y)
title('Reconstruction error')
xlabel('Time (sec.)')
xlim([0 N/fs])
% ylim([-1 1])


y = y(1:N);
```

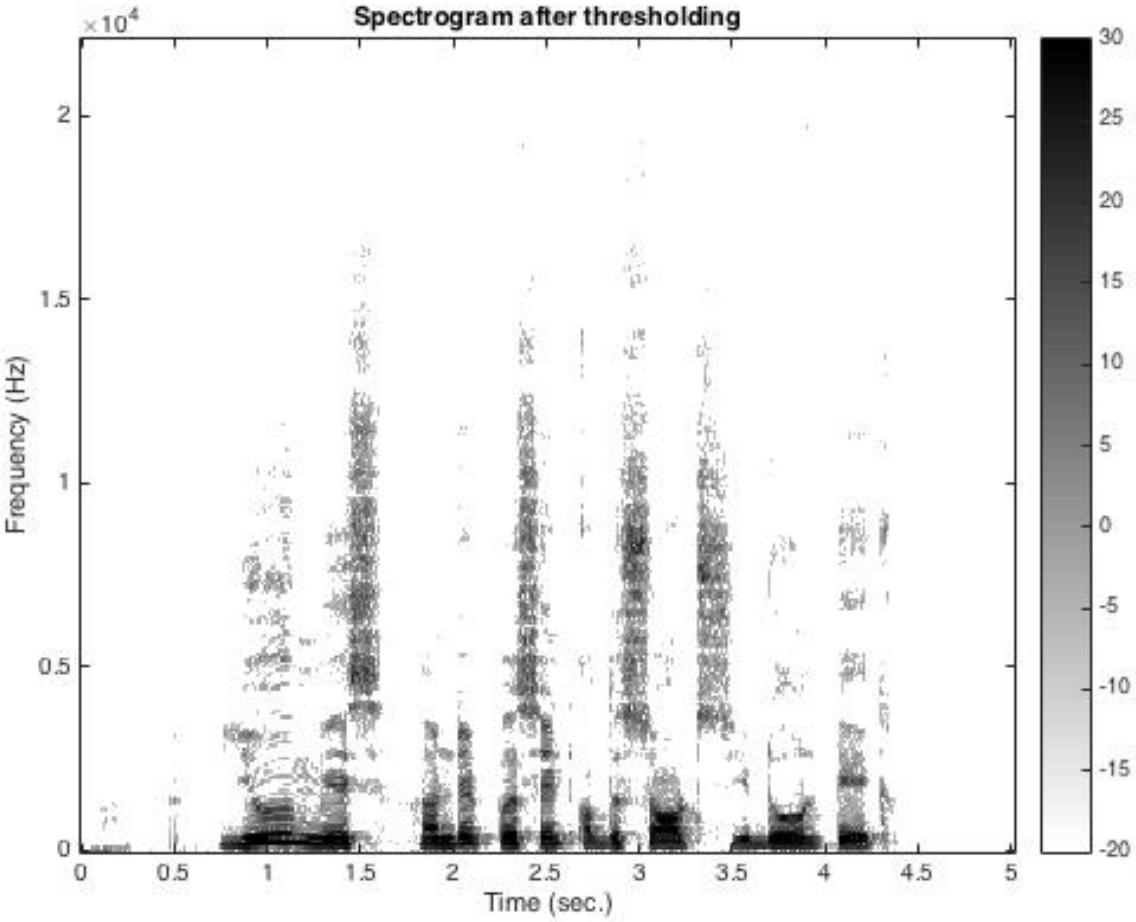
```
audiowrite('processed_speech.wav', y, fs);

figure,plot((1:N)/fs, y)
xlabel('Time (sec.)')
title('Processed signal')


soundsc(y, fs)

%% Compare noisy and processed speech fragments

%

n = 8401:9800;
figure,subplot(2, 1, 1)
plot(n/fs, x(n))
title('Noisy speech - One segment')
axis tight

subplot(2, 1, 2)
plot(n/fs, y(n))
title('Processed speech - One segment')
axis tight

%% Proving Parseval Identity
signal_sum=sum(x.^2)
stftcoeff_sum=abs(sum(sum(abs(X).^2)))
invstft_sum=sum(sum(y.^2))
```

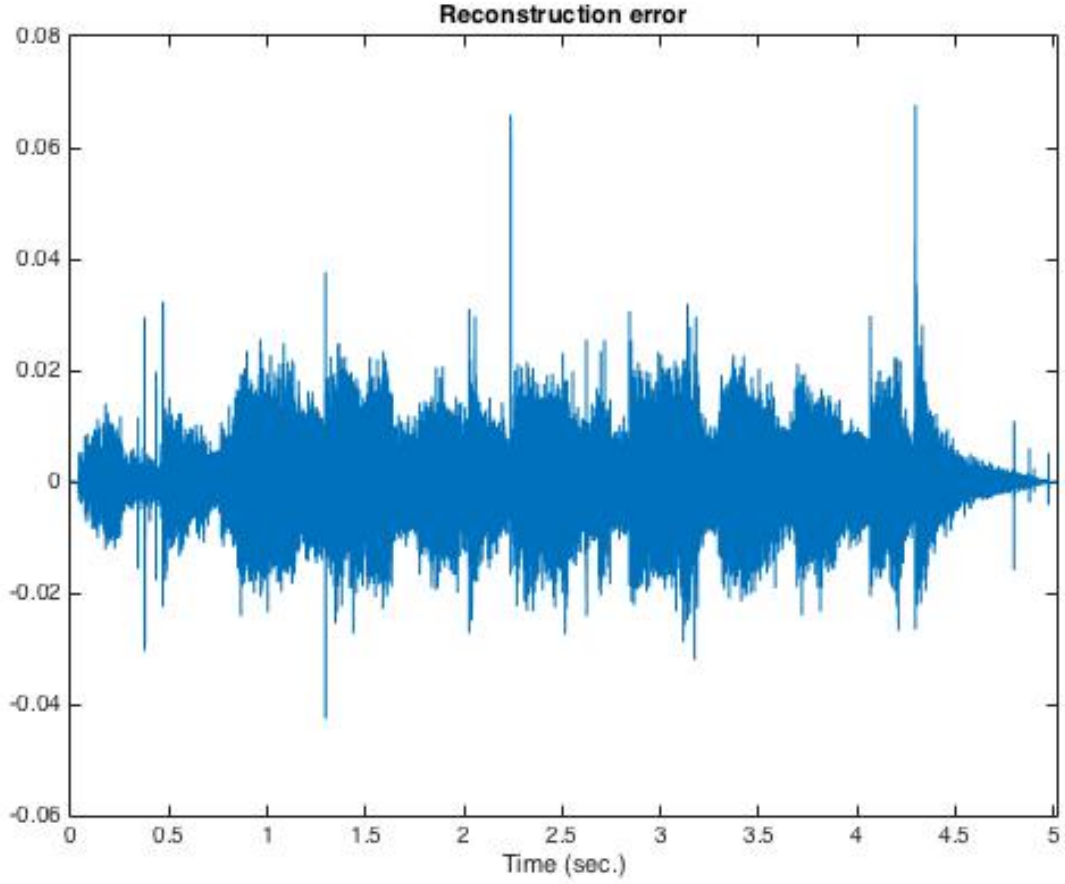

Speech signal
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
0
1
2
3
4
5
6
Time (seconds)

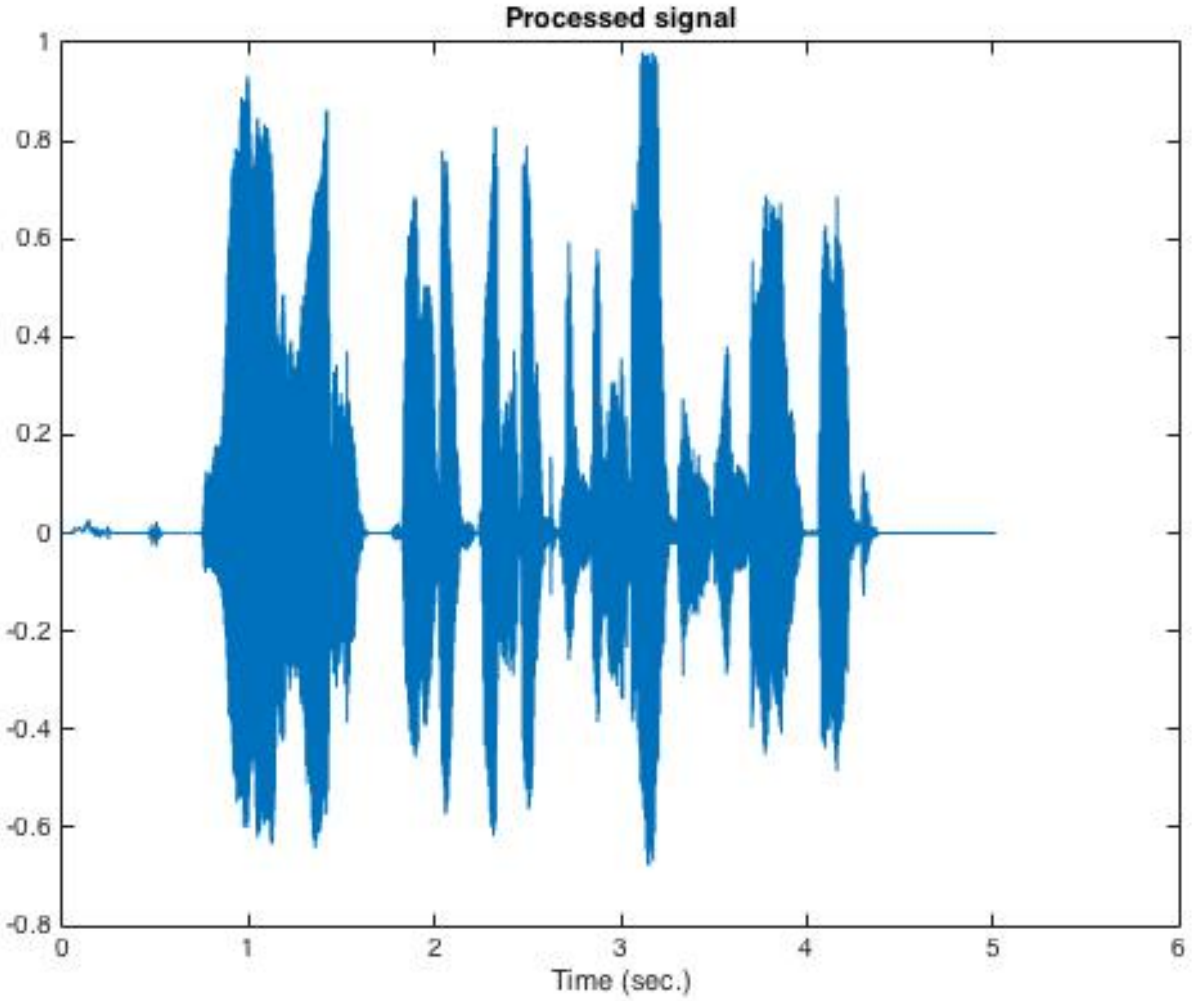

Half-cycle sine window
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
50
100
150
200
250
300
350
400
450
500
n

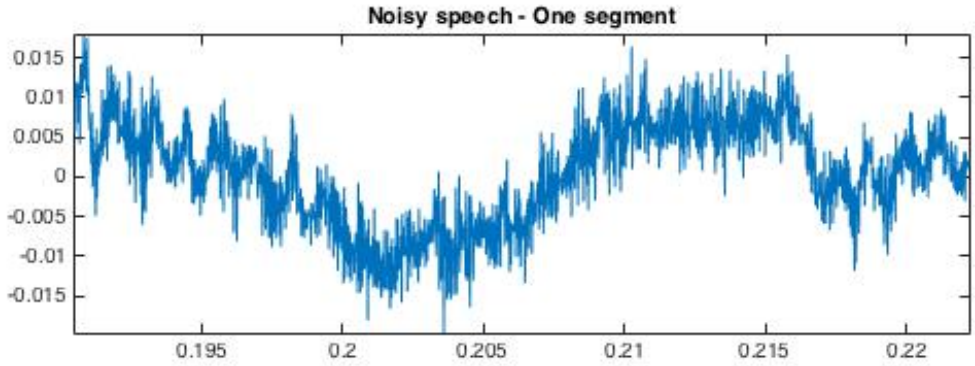

Verify perfect reconstruction property of window function
1.2
1
0.8
0.6
0.4
0.2
0
50
100
150
200
250

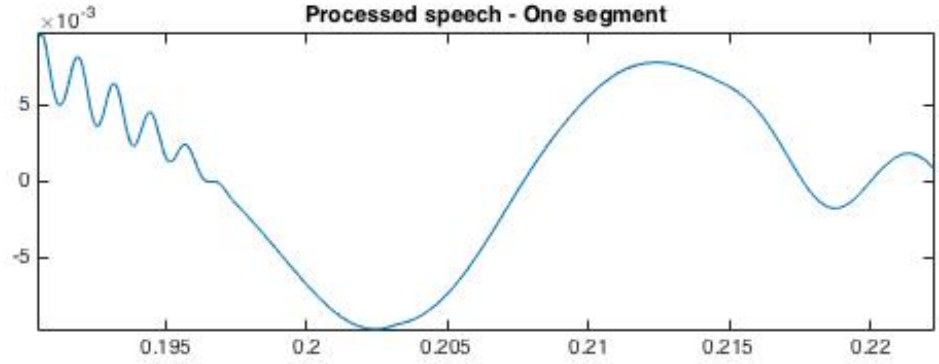

Spectrogram of noisy speech signal
×10^4
2
1.5
1
0.5
0
Frequency (Hz)
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
Time (sec.)
30
25
20
15
10
5
0
-5
-10
-15
-20


Spectrogram after thresholding
×10^4
Frequency (Hz)
Time (sec.)
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5
30
25
20
15
10
5
0
-5
-10
-15
-20


Reconstruction error
Time (sec.)
0.08
0.06
0.04
0.02
0
-0.02
-0.04
-0.06
0
0.5
1
1.5
2
2.5
3
3.5
4
4.5
5


Processed signal
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
0
1
2
3
4
5
6
Time (sec.)


Noisy speech - One segment
0.015
0.01
0.005
0
-0.005
-0.01
-0.015
0.195
0.2
0.205
0.21
0.215
0.22


Processed speech - One segment
×10-3
5
0
-5
0.195
0.2
0.205
0.21
0.215
0.22

Conclusion

I took my own speech signal as input. This signal contains a hiss sound in the background. I used 50 % overlapping with sine half function as a window function. I plotted the spectrogram. Lot of noise signal is spread across the spectrogram. I used a threshold of 0.35 to remove the noise signal after I did the STFT transform. Again I looked at the spectrogram after removing the noise and we see that spectrogram is quite clear. Later, I performed the inverse STFT on the denoised STFT and reconstructed the signal. I could hear that the hiss sound in the background is removed. I took a particular segment from both the noise signal and the processed signal and displayed the output. The processed signal is smooth and noise is removed.

signal_sum = 6.3859e+03

stftcoeff_sum = 6.3859e+03

invstft_sum = 6.0060e+03

Here, I also showed that signal energy in time domain is equivalent to sum of the square of magnitude of all stft coefficient in time frequency domain. It is also conserved in the inverse Stft.