|  |
| --- |
| EECE26 |
| Communication Project Super-Heterodyne Receiver |
| Cairo University Faculty of Engineering |

|  |
| --- |
| Youssef Samir Zidan  9220988 |

Contents

[1. Discussion 0](#_Toc183639931)

[a) Discuss how you designed each of the blocks of the system in Figure 1. 0](#_Toc183639932)

[b) Answer the following in your document: 1](#_Toc183639933)

[1- In two or three sentences, discuss the role of the RF, the IF and the baseband detector. Indicate why we need the IF stage? 1](#_Toc183639934)

[2- Suppose you want to demodulate the first station (i.e. at 𝜔0), plot the spectrum of the outputs of the RF, the IF and the baseband stages. Hint: use the ‘fft’ command in MATLAB. You may also find the ‘fftshift’ command useful. 1](#_Toc183639935)

[3- Use the command ‘sound’ on the demodulated signal and check whether you can successfully listen to the radio station. Please comment about this step in your report 2](#_Toc183639936)

[4- Add “noise” to your signal and then play “sound” the signal. What is the effect of the noise? 2](#_Toc183639937)

[5- Repeat parts 2 and 3 but after removing the RF BPF. That is, the RF stage does not exist, what would happen if you try to demodulate the station at 𝜔0? 2](#_Toc183639938)

[6- What happens (in terms of the spectrum and the sound quality) if the receiver oscillator has frequency offset by 0.2 KHz and 1.2 KHz? 3](#_Toc183639939)

[Samples (First to run) 4](#_Toc183639940)

[Modulation (second to run) 4](#_Toc183639941)

[Demodulation 4](#_Toc183639942)

1. Discussion

## a) Discuss how you designed each of the blocks of the system in Figure 1.



Baseband Detection

IF Stage

𝜔𝐼𝐹

Oscillator

𝜔𝐶 + 𝜔𝐼𝐹

Mixer

RF Stage

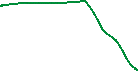
𝜔𝑐



AM

Modulator

* The first stage is that I extracted the data from the signals getting importants parameters like sampling frequecny and BWs
* Then it came time for modulating which was as straight forward as just multplying the signal by .\*cos(2pi\*Fc\*t) in an element wise multplication
* Then the another function taked the whole recevier part
* At RF stage the user selects the required signal then a BPF is placed at the intened frequency to minmize the image interferance
* Then a mixer simply multplies with .\*cos(2pi\*(Fc+IF)\*t) to get the band in intermidate frequency which could be an input directly to an ADC (but more BW overhead for ADC)
* Then we put another BPF filter in IF stage to get rid of any interferance and other signals
* For the last stage we modulate again with IF only to get the signal to the baseband then finally we put a LPF to get rid of all noise and interferance and hear the signal clearly



## b) Answer the following in your document:

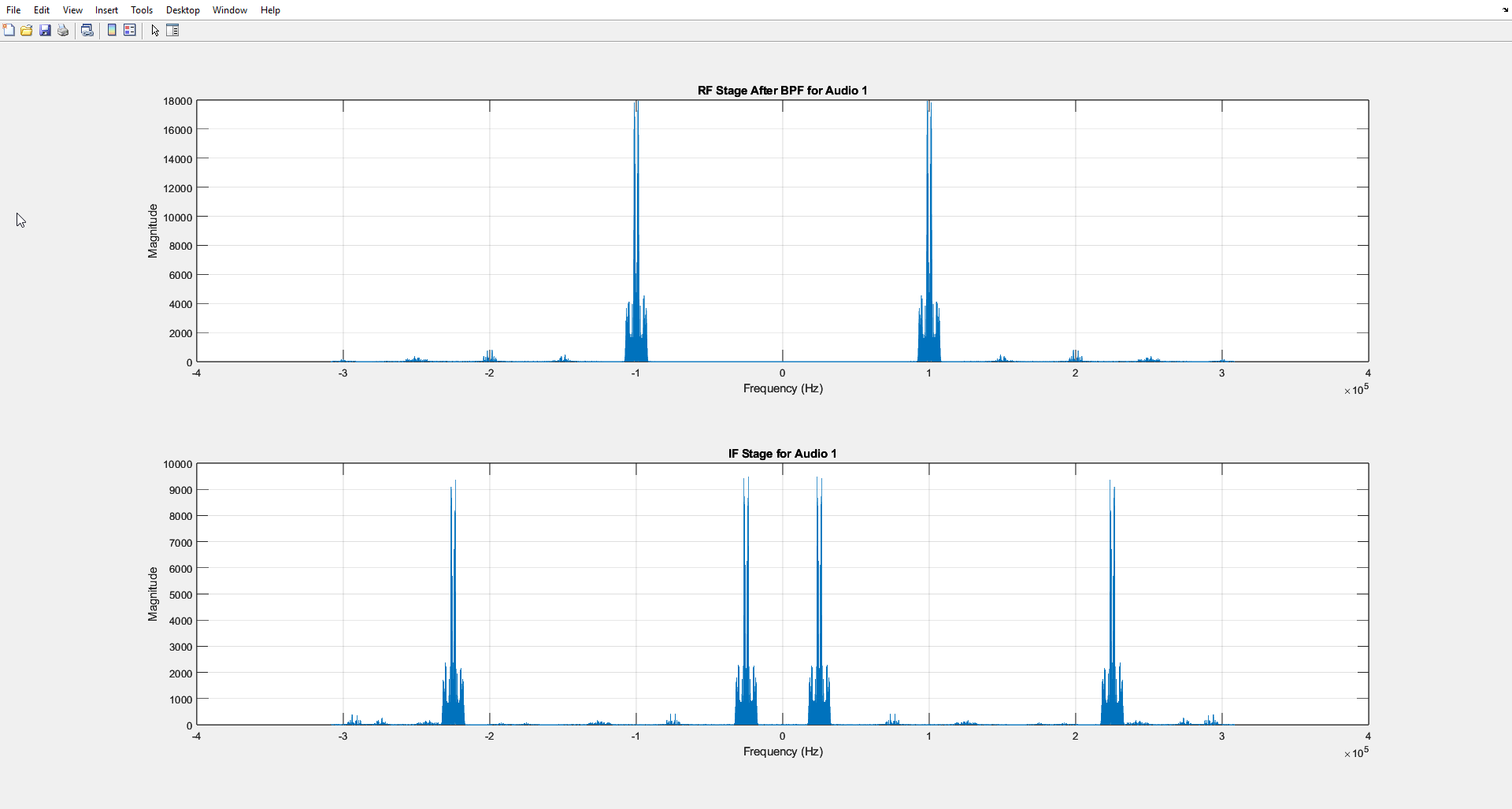
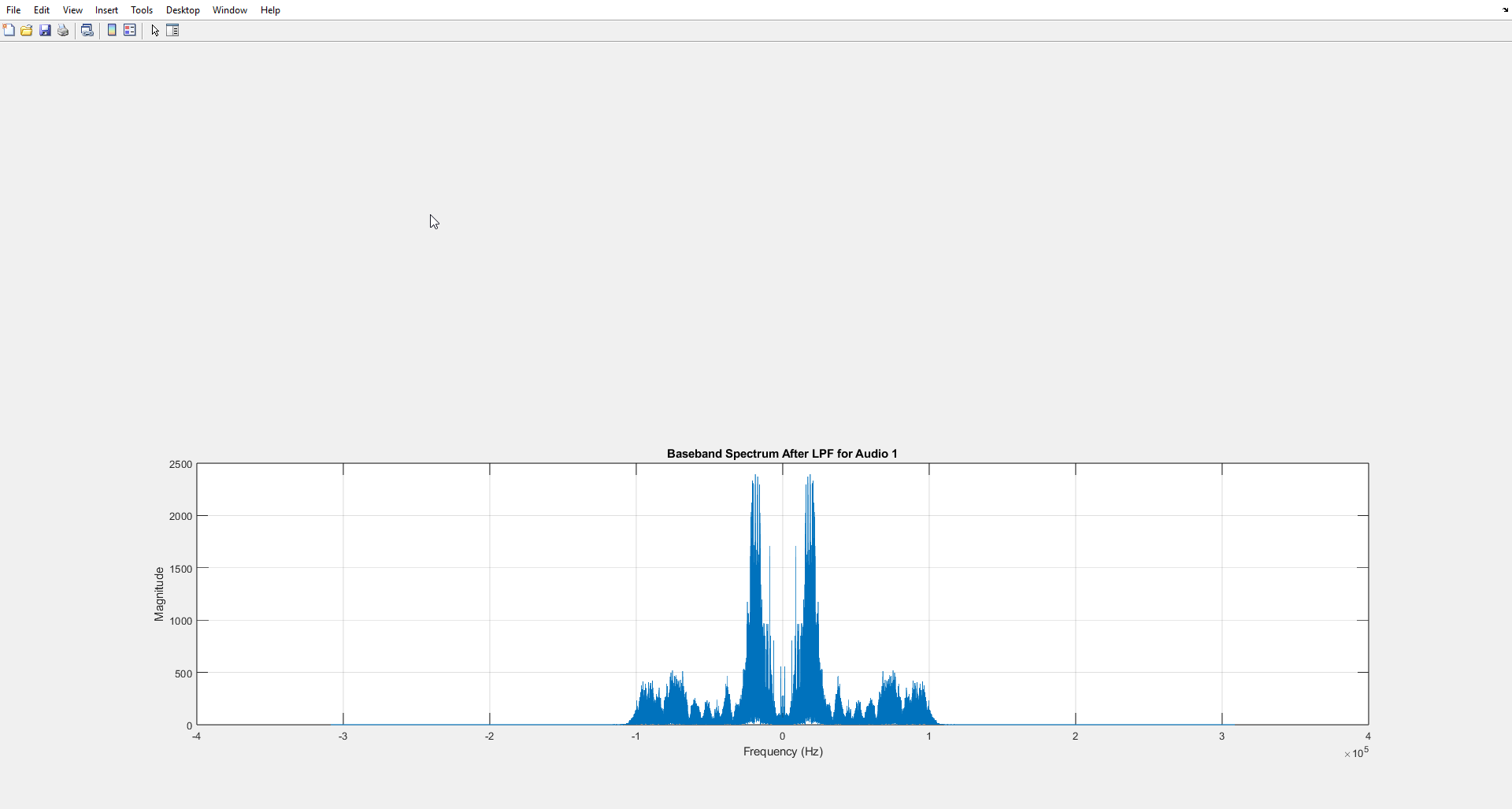
### In two or three sentences, discuss the role of the RF, the IF and the baseband detector. Indicate why we need the IF stage?

* RF detector selects the desired station from multiple signals when input signal is at Wc

IF detector converts any signal to a fixed intermediate frequency regardless of the original carrier frequency. which simplifies the hardware and improves filtering performance

* Baseband detector is used to demodulate the signal from IF to the original Baseband

### Suppose you want to demodulate the first station (i.e. at 𝜔0), plot the spectrum of the outputs of the RF, the IF and the baseband stages. Hint: use the ‘fft’ command in MATLAB. You may also find the ‘fftshift’ command useful.

* 
* 

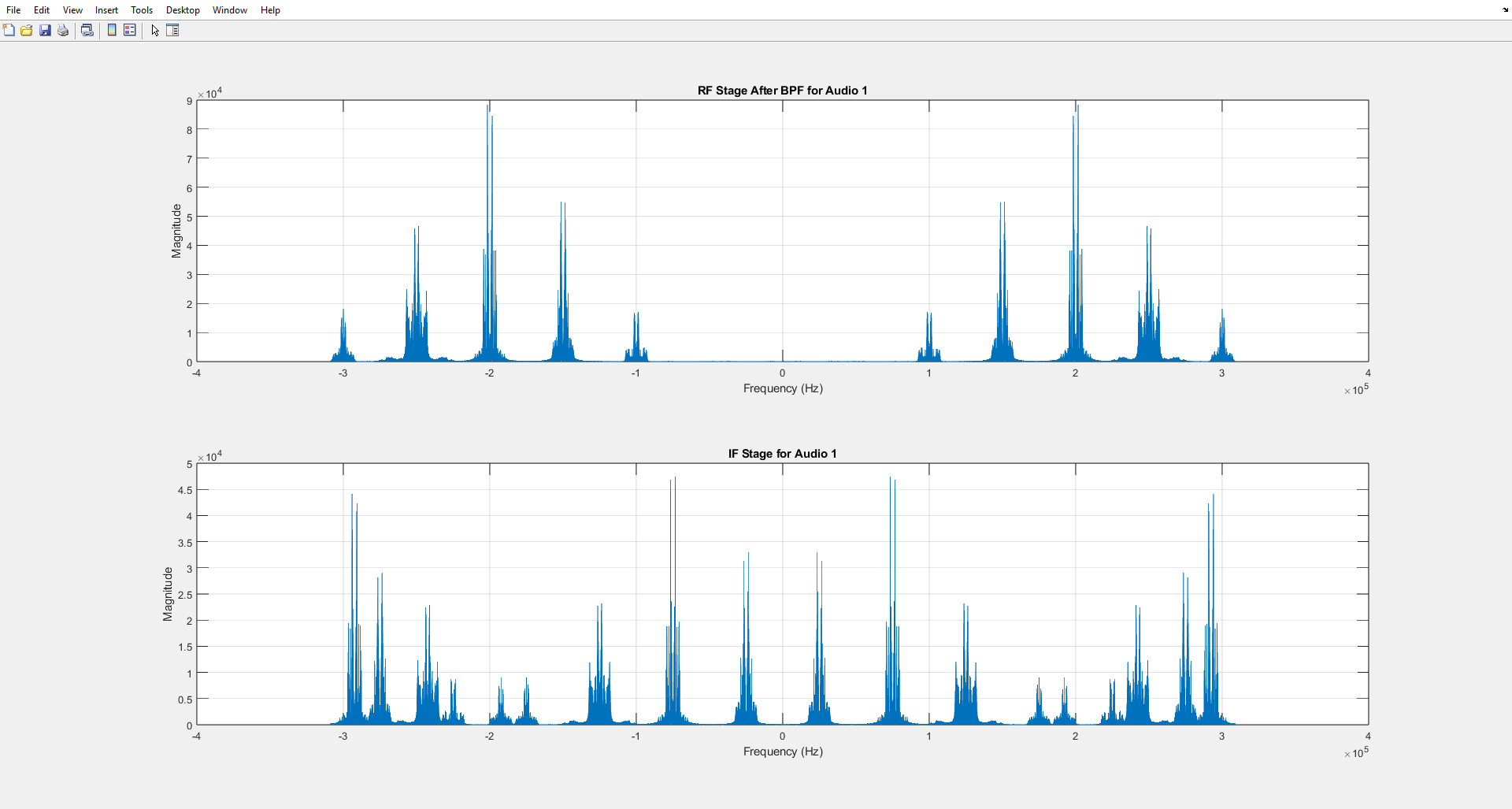
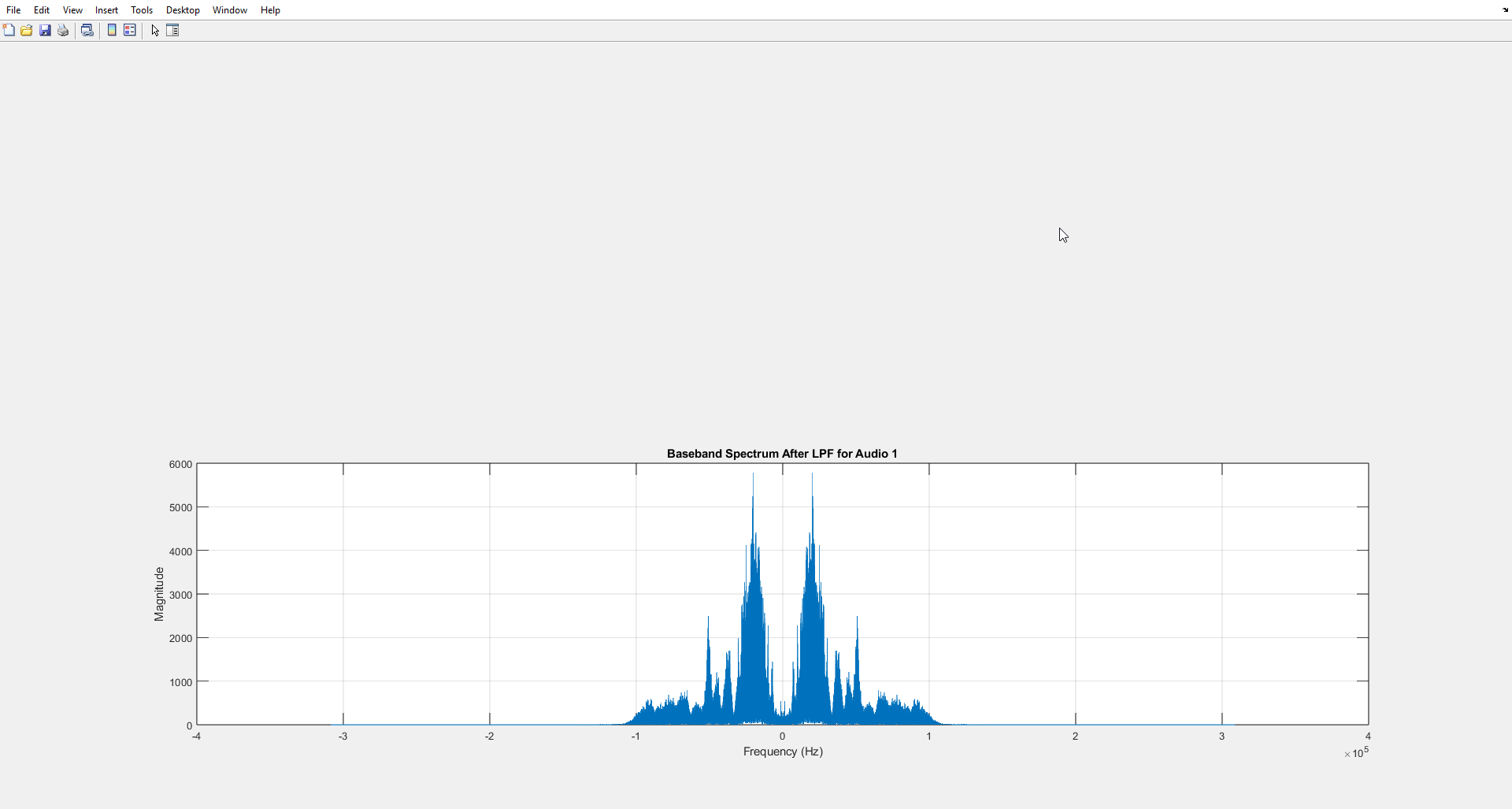
### Use the command ‘sound’ on the demodulated signal and check whether you can successfully listen to the radio station. Please comment about this step in your report

* Yes I can hear it clearly and I had to downsample again to hear it

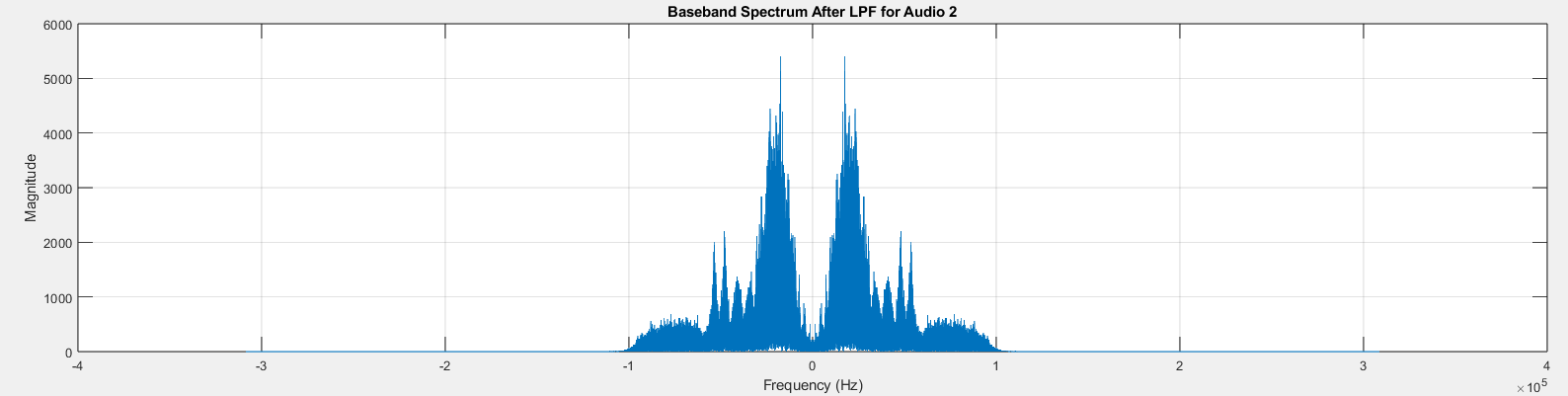
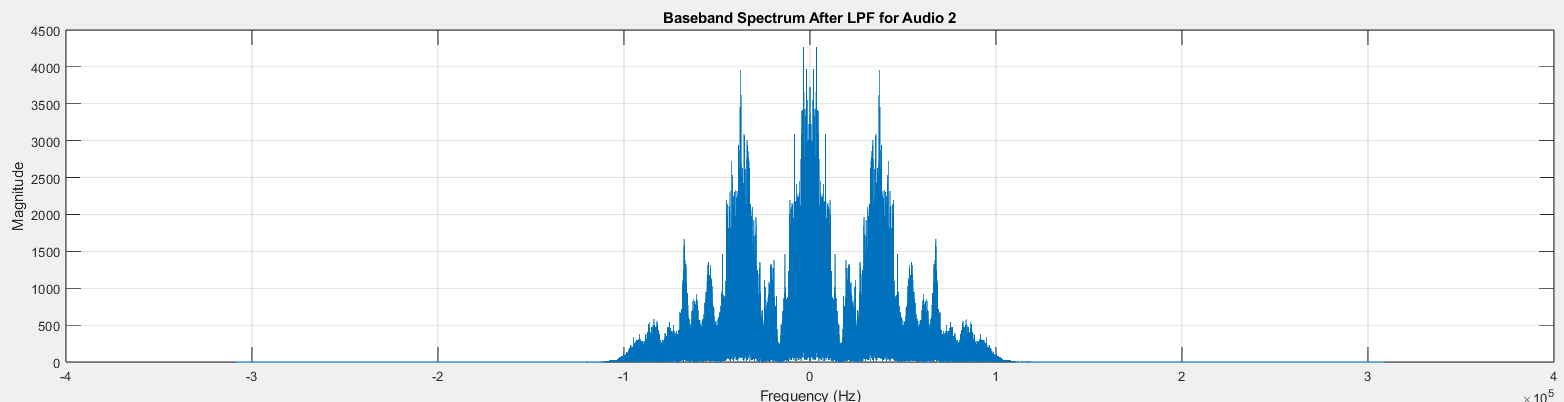
### Add “noise” to your signal and then play “sound” the signal. What is the effect of the noise?

* Using receivedSignal = awgn(sentsignal,10);
* I heared a white noise with the signal وشوشة

### Repeat parts 2 and 3 but after removing the RF BPF. That is, the RF stage does not exist, what would happen if you try to demodulate the station at 𝜔0?

* I only heard the image signal at 150khz completely overlapping and shadowing the intended signal
* 
* 

### What happens (in terms of the spectrum and the sound quality) if the receiver oscillator has frequency offset by 0.2 KHz and 1.2 KHz?

* carrier\_LO = cos(2 \* pi \* (fc\_mod + IF + offset) \* t); % Local oscillator signal
* at 0.2khz I could bearly hear the signal with very annoying sound effect
* 
* at 1.2khz I couldn’t hear the signal at all
* 

## Samples (First to run)

%%section1: load data

[c1,fs1]=audioread("Short\_BBCArabic2.wav");

[c2,fs2]=audioread("Short\_FM9090.wav");

[c3,fs3]=audioread("Short\_QuranPalestine.wav");

[c4,fs4]=audioread("Short\_RussianVoice.wav");

[c5,fs5]=audioread("Short\_SkyNewsArabia.wav");

audioSignals = {c1, c2, c3, c4, c5};

samplingRates = [fs1, fs2, fs3, fs4, fs5];

Left\_channel = cell(1, length(audioSignals));

Right\_channel = cell(1, length(audioSignals));

monoSignals = cell(1, 5);

paddedSignals = cell(1, 5);

maxLength = 0;

for i=1:length(audioSignals)

Left\_channel{i}=audioSignals{i}(:,1);

Right\_channel{i} = audioSignals{i}(:, 2);

monoSignals{i}=sum(audioSignals{i}, 2);

maxLength = max(maxLength, length( monoSignals{i}));

end

%sound(monoSignals{2},fs2)

% Pad signals to the maximum length

for i = 1:length(monoSignals)

x = monoSignals{i};

% Pad the signal with zeros to match the maximum length

paddedSignals{i} = [x; zeros(maxLength - length(x), 1)];

end

%% section 2 : plot and extract data

% plotting for the all original signal

BWs = zeros(1,5);

for i = 1:length(audioSignals)

x = audioSignals{i};

fs = samplingRates(i);

% FFT

N = length(x);

X = fft(x, N);

f = (-N/2:N/2-1) \* fs / N;

%[pxx, f\_pxx] = periodogram(x, [], length(x), fs);

% avg\_pxx = mean(pxx, 2);

% pxx=abs(x).^2;

% tot\_sum=cumsum(pxx);

% normlizedsum=tot\_sum/tot\_sum(end);

BWs(i)=obw(monoSignals{i},fs);

% Plot

grid on

figure;

plot(f, abs(fftshift(X)) / N);

title(['Frequency Spectrum of Signal ', num2str(i)]);

xlabel('Frequency (Hz)');

ylabel('Magnitude');

end

grid off

BWs\_Hz = BWs ;

BWs\_kHz = BWs\_Hz / 1000;

## Modulation (second to run)

fc\_mod = zeros(1,5); % Carrier frequencies

t = cell(1, 5); % Time vectors for each signal

modulated\_signal = cell(1, 5); % Modulated signals

sentsignal=[];

max\_length=0;

interp\_factor = 14;

fs=44100;

fs\_interpolated=interp\_factor\*fs;

for i = 1:length(audioSignals)

% Retrieve the current mono signal

x = paddedSignals{i};

% Take quarter of the length of the signal

half\_length = floor(length(x) / 1);

x = x(1:half\_length); % Use the first quarter of the signal

% Carrier frequency (in Hz)

fc\_mod(i) = 100e3 + (i - 1) \* 50e3; % Carrier frequency for the i-th signal

x\_inter=interp(x,interp\_factor);

t = (0:1/fs\_interpolated:((size(x\_inter, 1)-1)/fs\_interpolated))';

t = t(:, ones(1, size(x\_inter, 2)));

Mod\_carrier = cos(2 \* pi \* fc\_mod(i) \* t);

modulated\_signal{i} = x\_inter .\* Mod\_carrier;

%modulated\_signal{i}=ammod(x\_inter,fc\_mod(i),fs\*interp\_factor);

% t\_interpolated = (0:length(x\_inter)-1) / (interp\_factor \* fs); % New time vector

max\_length = max(max\_length, length(modulated\_signal{i}));

end

for i = 1:length(modulated\_signal)

% Combine into the FDM signal

if isempty(sentsignal)

sentsignal = modulated\_signal{i};

else

sentsignal = sentsignal + modulated\_signal{i};

end

end

% Plot the combined FDM signal in the time domain

figure;

subplot(2, 1, 1);

t\_total = (0:length(sentsignal)-1) / (10\*fs); % Time vector for the combined signal

plot(t\_total, sentsignal);

title('FDM Signal (Time Domain)');

xlabel('Time (s)');

ylabel('Amplitude');

% Plot the combined FDM signal in the frequency domain

subplot(2, 1, 2);

N = length(sentsignal); % Number of samples in the combined signal

f = (-N/2:N/2-1) \* 10\*fs / N; % Frequency axis

spectrum\_fdm = fftshift(fft(sentsignal, N)); % Compute and shift the FFT

plot(f, abs(spectrum\_fdm) / N);

title('FDM Signal (Frequency Domain)');

xlabel('Frequency (Hz)');

ylabel('Magnitude');

%sound(sentsignal, 10 \* fs);

## Demodulation

% Constants

offset = 0;

IF = 25e3 ; % Intermediate frequency (25 kHz)

fs\_interpolated = interp\_factor \* fs; % Sampling frequency after interpolation

%receivedSignal = awgn(sentsignal,10);

receivedSignal = sentsignal;

% Initialize arrays for demodulated signals

demodulated\_signals = cell(1, length(audioSignals));

% User input for channel selection

fprintf("Choose one of these channels:\n");

fprintf("1. Short\_BBCArabic2\n");

fprintf("2. Short\_FM9090\n");

fprintf("3. Short\_QuranPalestine\n");

fprintf("4. Short\_RussianVoice\n");

fprintf("5. Short\_SkyNewsArabia\n");

i = input("Choose: ");

% Carrier frequency for the given signal

fc\_mod = 100e3 + (i - 1) \* 50e3; % Carrier frequency for the selected channel

bw = BWs(i); % Bandwidth for the selected signal

if bw / 2 > 25000

bw = 24999; % Cap bandwidth to avoid exceeding Nyquist limit

end

% Filter specifications for RF stage

F\_stop1 = fc\_mod - bw - bw/4;

F\_pass1 = fc\_mod - bw;

F\_pass2 = fc\_mod + bw;

F\_stop2 = fc\_mod + bw + bw/4;

A\_pass = 1;

A\_stop1 = 40;

A\_stop2 = 40;

% Step 1: Bandpass filter to isolate the modulated signal

rf\_bandpass\_filter = fdesign.bandpass(F\_stop1, F\_pass1, F\_pass2, F\_stop2, A\_stop1, A\_pass, A\_stop2, fs\_interpolated);

rf\_bandpass\_filter = design(rf\_bandpass\_filter, 'equiripple');

filtered\_signal = filter(rf\_bandpass\_filter, receivedSignal);

%filtered\_signal=receivedSignal;

% Plot RF stage after BPF

figure('Name', 'RF Stage: Filtered Signal');

rf\_spectrum = fftshift(fft(filtered\_signal));

N = length(rf\_spectrum);

f = (-N/2:N/2-1) \* fs\_interpolated / N;

subplot(2, 1, 1);

plot(f, abs(rf\_spectrum));

title("RF Stage After BPF for Audio " + num2str(i));

xlabel("Frequency (Hz)");

ylabel("Magnitude");

grid on;

% Step 2: Mix with a local oscillator to shift the signal to IF

t = (1:length(filtered\_signal))' / fs\_interpolated; % Time vector

carrier\_LO = cos(2 \* pi \* (fc\_mod + IF + offset) \* t); % Local oscillator signal

mixed\_signal\_IF = filtered\_signal .\* carrier\_LO; % Mixer output

% Plot IF stage

IF\_spectrum = fftshift(fft(mixed\_signal\_IF));

subplot(2, 1, 2);

plot(f, abs(IF\_spectrum));

title("IF Stage for Audio " + num2str(i));

xlabel("Frequency (Hz)");

ylabel("Magnitude");

grid on;

% Step 3: Bandpass filter the mixed signal to isolate IF

F\_stop1\_IF = IF - bw - bw/4;

F\_pass1\_IF = IF - bw;

F\_pass2\_IF = IF + bw;

F\_stop2\_IF = IF + bw + bw/4;

if\_bandpass\_filter = fdesign.bandpass(F\_stop1\_IF, F\_pass1\_IF, F\_pass2\_IF, F\_stop2\_IF, A\_stop1, A\_pass, A\_stop2, fs\_interpolated);

if\_bandpass\_filter = design(if\_bandpass\_filter, 'equiripple');

filtered\_signal\_IF = filter(if\_bandpass\_filter, mixed\_signal\_IF);

% Step 4: Demodulate the IF signal by multiplying with an IF carrier

carrier\_IF = cos(2 \* pi \* IF \* t); % IF carrier signal

baseband\_signal = filtered\_signal\_IF .\* carrier\_IF; % Demodulation

% Step 5: Low-pass filter to recover the baseband signal

F\_pass = bw;

F\_stop = bw + 5000;

lowpass\_filter = fdesign.lowpass(F\_pass, F\_stop, A\_pass, A\_stop2, fs\_interpolated);

lowpass\_filter = design(lowpass\_filter, 'butter');

Base\_Band\_received\_signal\_LPF = filter(lowpass\_filter, baseband\_signal);

% Resample and playback the demodulated signal

Base\_Band\_received\_signal\_LPF = 4 \* resample(Base\_Band\_received\_signal\_LPF, 1, interp\_factor);

sound(Base\_Band\_received\_signal\_LPF, fs);

% Plot baseband spectrum after LPF

figure('Name', 'Baseband Signal After LPF');

baseband\_spectrum = fftshift(fft(Base\_Band\_received\_signal\_LPF));

N = length(baseband\_spectrum);

f = (-N/2:N/2-1) \* fs\_interpolated / N;

subplot(2, 1, 2);

plot(f, abs(baseband\_spectrum));

title("Baseband Spectrum After LPF for Audio " + num2str(i));

xlabel("Frequency (Hz)");

ylabel("Magnitude");

grid on;