Name: علي ماهر عبدالسلام	Sec:3	B.N:5

Super-heterodyne Receiver

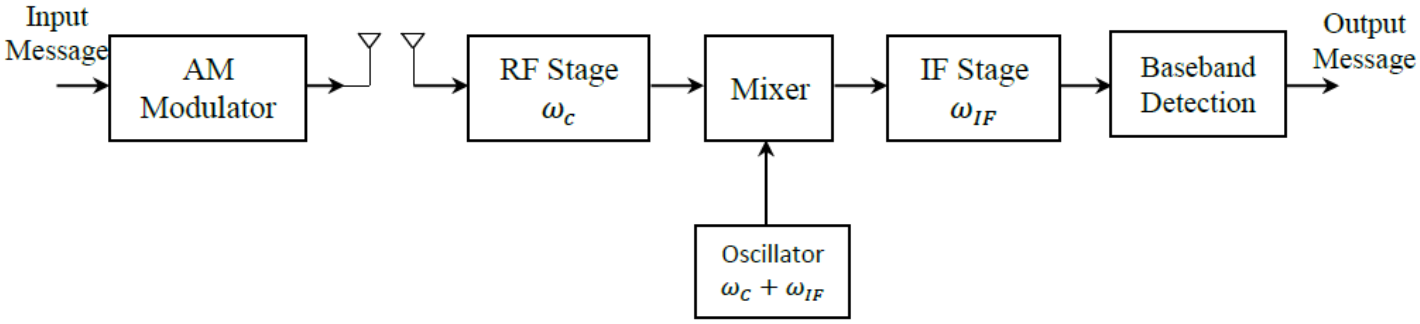


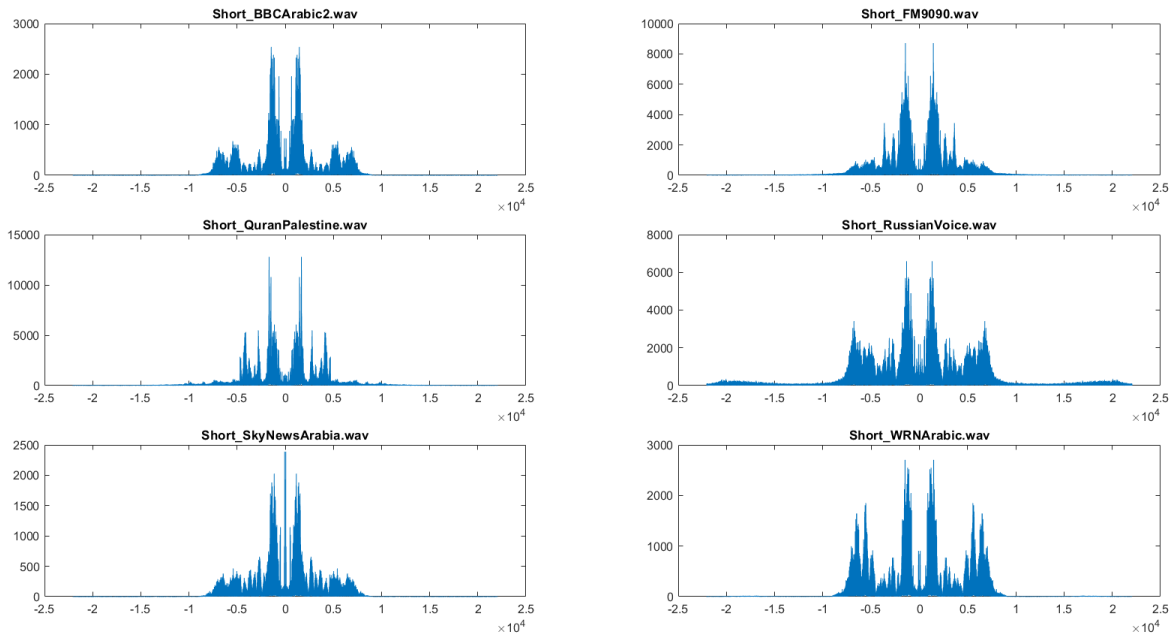
Figure 1: AM modulator and a super-heterodyne receiver

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

In the MATLAB code, I wrote 2 functions, main one for the block diagram which is divided into sections to make it easier to read the parts that represent each block. In addition, a small function for the purpose to draw the outputs in frequency domain. And 2 more functions for number 4 and 5 in the requirements in which only a small part is changed from main code.

1 – Input Messages

1. Storing the names of the 6 radio sound files into a vector
2. Entering a loop to save all the audio vectors inside a matrix that each row represent different message



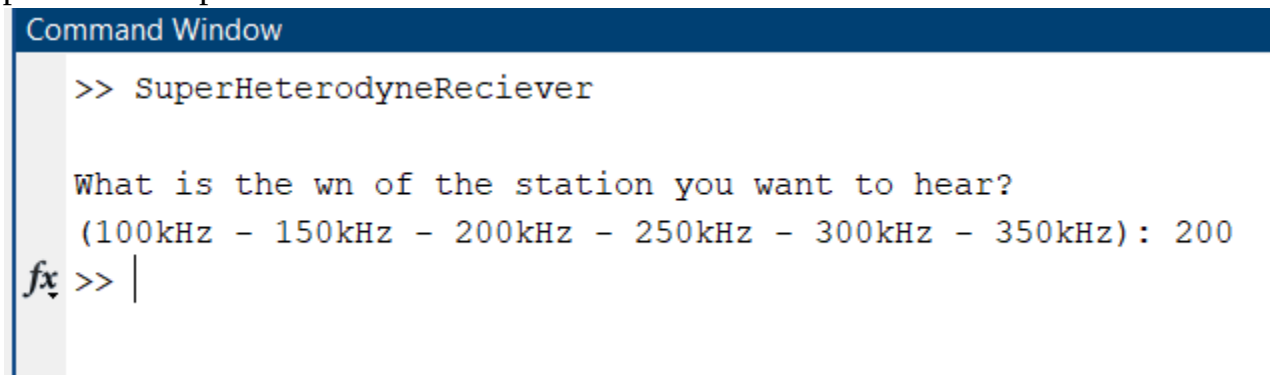
2 – AM Modulator

1. In this section, I decided to multiplex all 6 messages together and not only 2
2. Stored the 6 values of carrier frequency inside a vector
3. Now I must multiply the sampling frequency by a large factor so I don't clip any of the 6 messages at transmission
4. By doing tests on the simulation I found out that I should multiply it by 18
5. So to make the dimensions of the carrier same or higher than the messages so I can multiply them, I also did some tests on the time vector
6. Then finally to make both have same dimensions I added zeros at the end of the message signals, multiplied by carrier then summed them

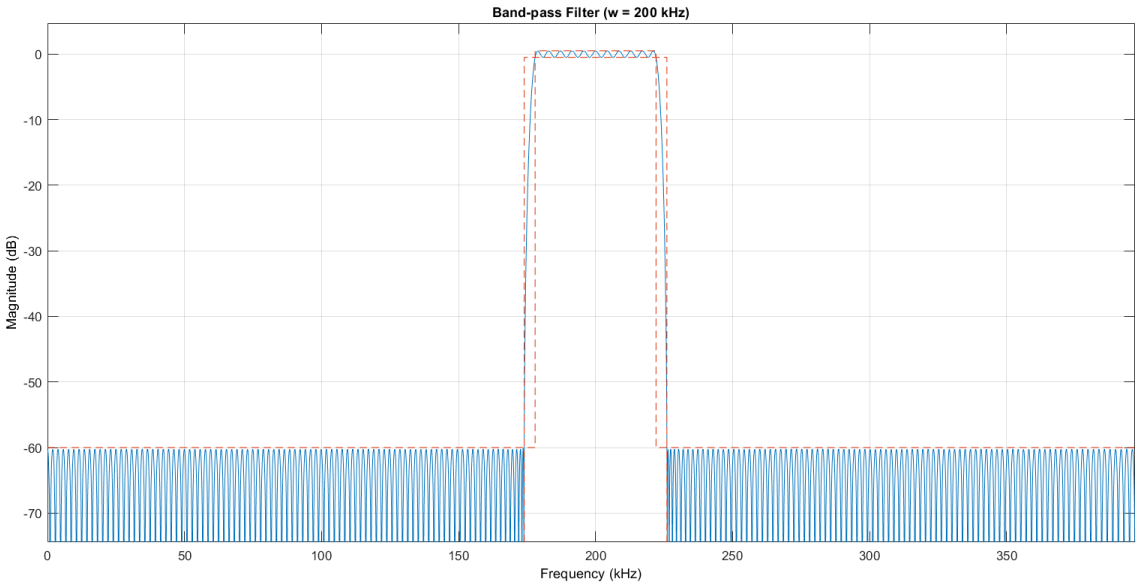
3 – RF Stage

1. Since the idea was to make the receiver's ω_c tunable. I ask the user to enter which ω_c they want.

Example of user input:



2. Then design the first BP filter based on it, and made its thickness 4kHz since in real and practical life there isn't a sharp filter

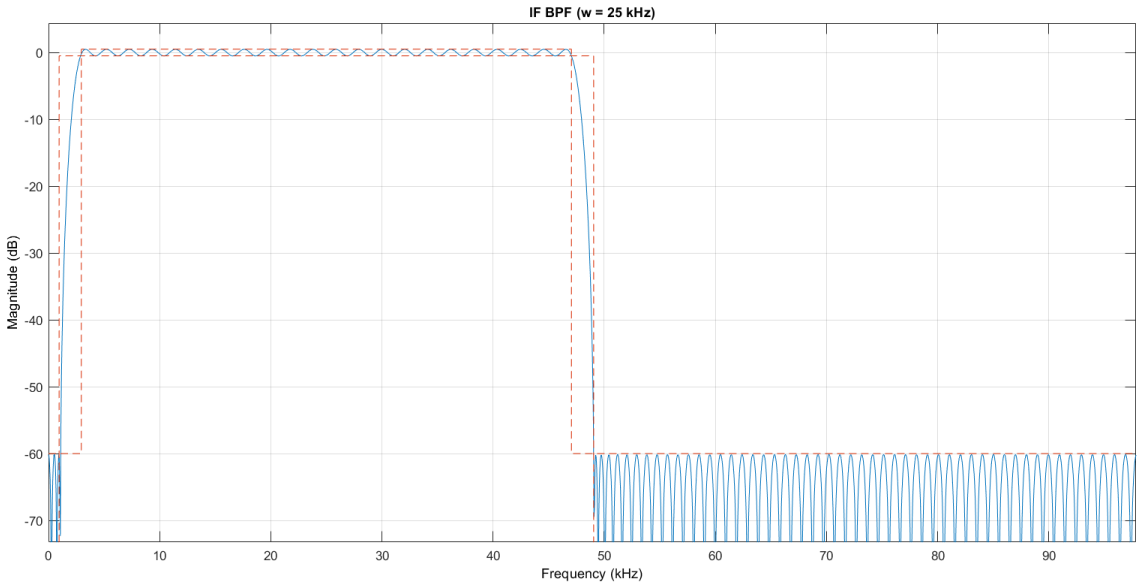


4 – Mixer

1. Simply making a variable for ω_{IF}
2. Making the $\cos\big((\omega_n + \omega_{IF})t\big)$ carrier and multiplying it by the signal

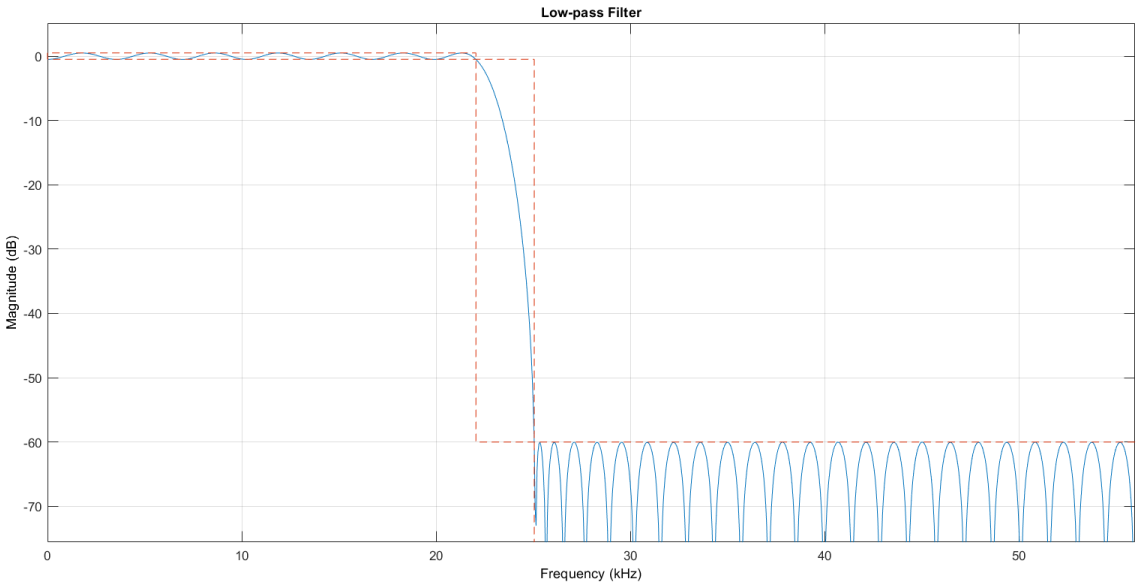
5 – IF Stage

1. Filter around ω_{IF} with only thickness 2kHz so it has more accuracy than the outer BP filter



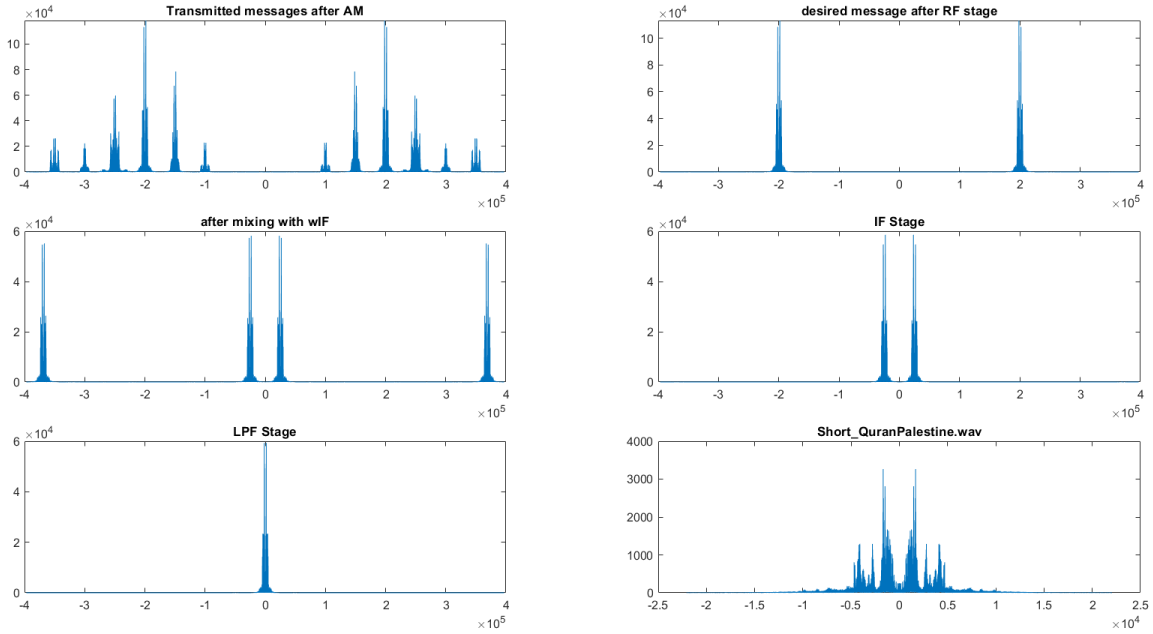
6 – Baseband Detection

1. Multiply by a carrier with a suitable frequency (ω_{IF}) that bring the signal around zero
2. Then filter the side bands using a LPF so only the main signal is left in the middle



7 – Output Message

1. Finally reversing the interpolation step of the signal then creating an audio file to hear it



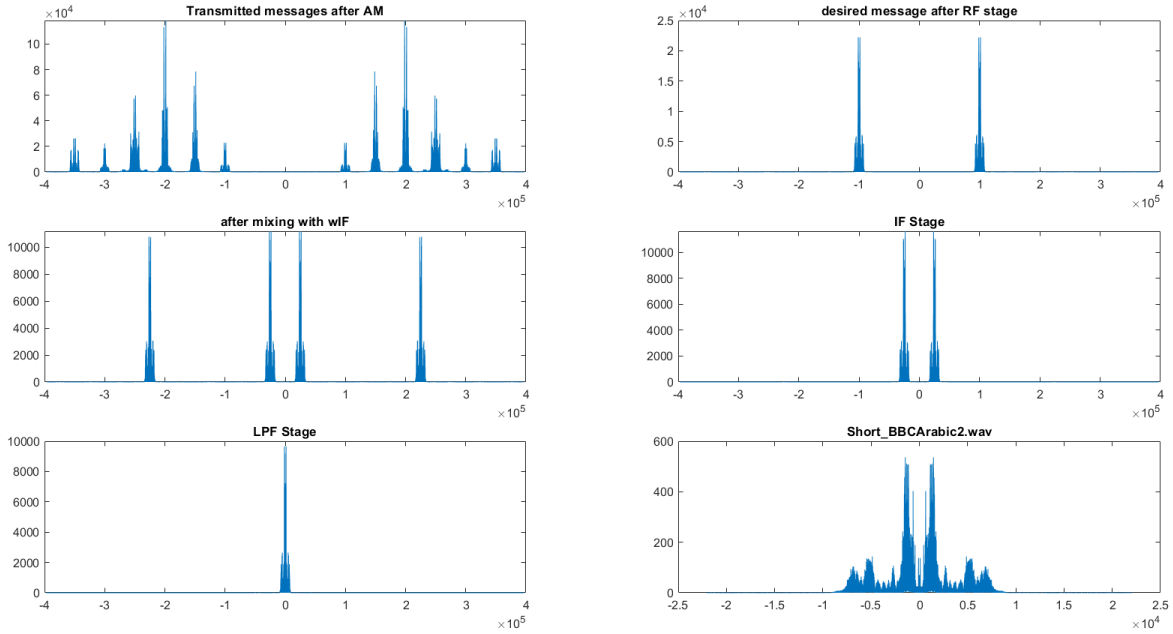
b) Answer the following in your document:

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?

- Role of RF: for image rejection of the signal which is at distance $2\omega_{IF}$ away from the main signal I want to transmit and filters around ω_c
- Role of IF: Removes any images other than the first two centered at ω_{IF}. We need it because only RF can't provide selectivity on ω_c since it is very high. So the small IF frequency that we control its value will be a lot easier
- Role of Baseband detector: Returns the final message into the baseband

2. Suppose you want to demodulate the first station (i.e. at ω_0), 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.

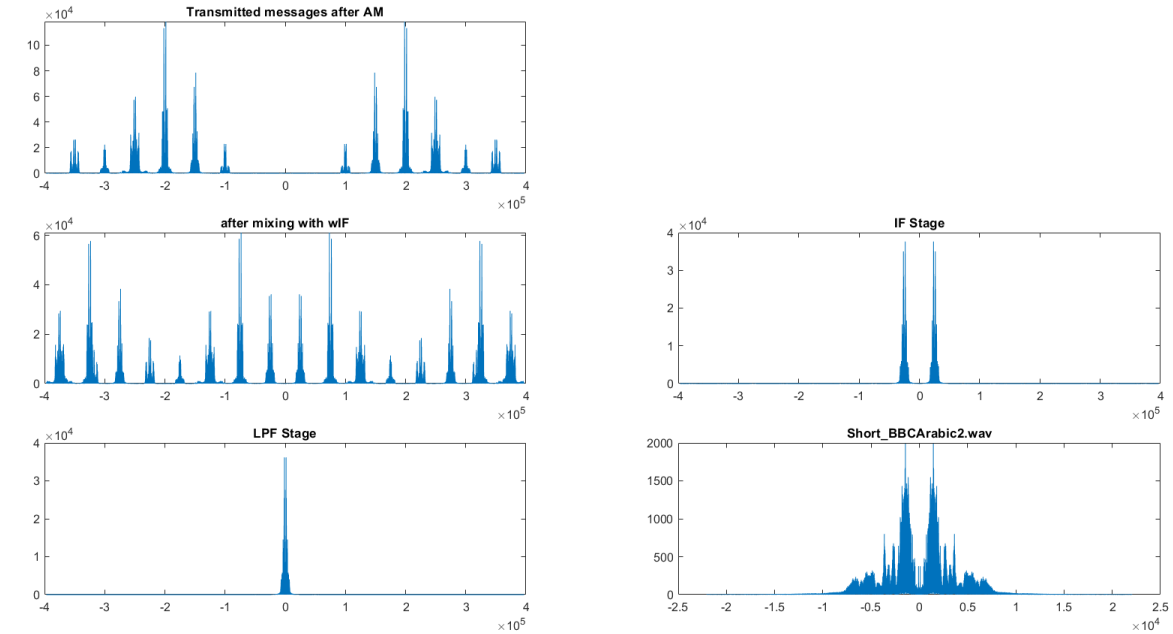
Using the plot function I wrote:



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

I do hear the sound and it doesn't seem to be any changes in the quality. But my observation is that the sound file size has decreased to nearly half of the original, which appears in the amplitude of the output plot. Also there are 2 or 3 more seconds added to the file but has no sound (zero).

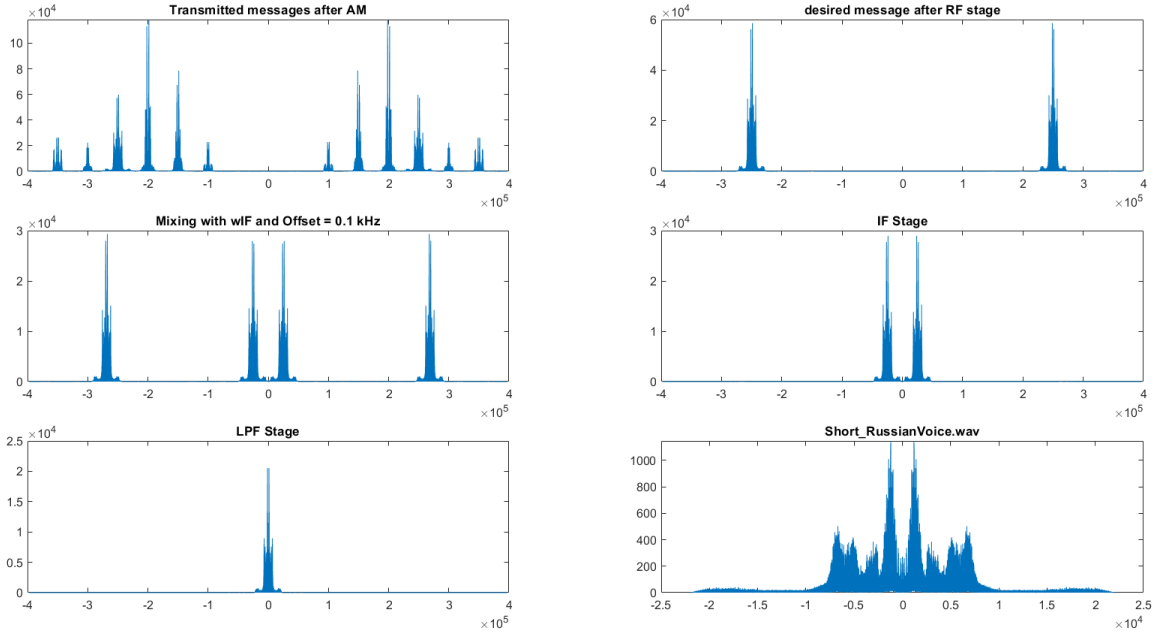
4. Repeat parts 2 and 3 but after removing the RF BPF. That is, the RF stage does not exist, what would happen if you try to demodulate the station at ω_0?



From the graph it seems that a lot of signals got mixed, then after demodulating and hearing the sound, it wasn't only the sound signal from first station but also 2nd station which is at distance 50 kHz far on top of it. So this confirms the idea of image at $2\omega_{IF}$ that RF filters out

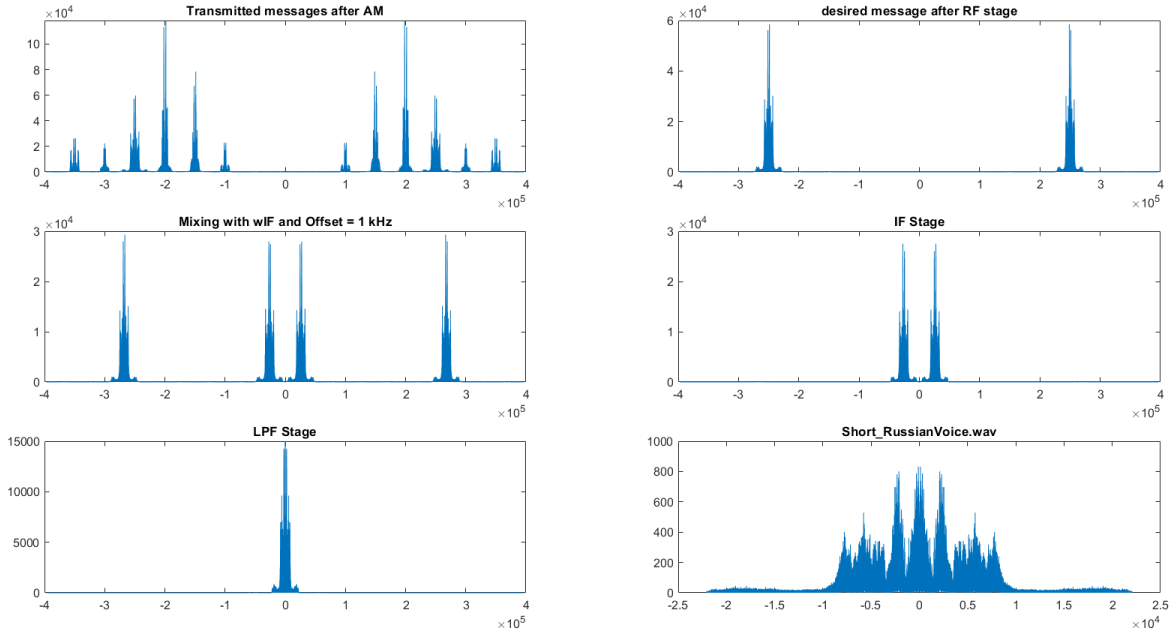
5. What happens (in terms of the spectrum and the sound quality) if the receiver oscillator has frequency offset by 0.1 KHz and 1 KHz?

At 0.1 KHz offset:



From the graphs the desired signal is delivered without any interference from others but it seems that the amplitude of the message is lowered because of that offset. After hearing the sound it still can be heard but there are some noise and it has digital sound and not clear human voice.

At 1 KHz offset:



Now the amplitude is very low that the words can't be heard at all and it only sounds like some digital noise.

To solve the offset problem we can either simply amplify the output signal or add something like Costas receiver at the oscillator to cancel the offset.

MATLAB Code of main function:

```
clear
%% Reading the audio files
file(1) = "Short_BBCArabic2.wav";    file(2) = "Short_FM9090.wav";
file(3) = "Short_QuranPalestine.wav";    file(4) = "Short_RussianVoice.wav";
file(5) = "Short_SkyNewsArabia.wav";    file(6) = "Short_WRNArabic.wav";
% sampling frequency is the same for all, fs = 44100
m = zeros(6,1);
for j = 1:6
    [temp, fs] = audioread(file(j));
    %combining the 2 channels
    temp = temp(:,1) + temp(:,2);
    temp = reshape(temp, 1, size(temp,1));
    if size(m,2) > size(temp,2)
        temp(size(m,2)) = 0;
    else
        m(:,size(temp,2)) = 0;
    end
    m(j,:) = temp;
end
% plot the 6 messages in frequency domain
figure(1)
row = 3; col = 2; n = 1;
for j = 1:6
    messageplot(m(j,:),fs,file(j), row, col, n);
    n = n + 1;
end
Bw = fs/2; % bandwidth of the messages
```

```
%% AM Modulator
fc = zeros(6,1);
for j = 0:5
    % carrier frequency
    fc(j+1) = 100000 + j * 50000; %max is w = 350 kHz
```

```
end
% making sure to achieve the Nyquist criteria
% least needed fs = 700 khz
factor = 18;
fs = fs*factor; %fs = 793.8 khz
m1 = interp( m(1,:), factor);     m2 = interp( m(2,:), factor);
m3 = interp( m(3,:), factor);     m4 = interp( m(4,:), factor);
m5 = interp( m(5,:), factor);     m6 = interp( m(6,:), factor);
m = [m1; m2; m3; m4; m5; m6];
clear m1 m2 m3 m4 m5 m6
% carrier formation
ts = 1/fs;
t = 0 : ts : factor;
length = size(t,2); %length = 14,288,401
% making the signals having the same dimensions with the carrier
m(:,length) = 0;
carrier = zeros(6,length);
for j = 1:6
    carrier(j,:) = cos(2*pi*fc(j)*t);
    % mutliplying the signals
    m(j,:) = m(j,:).*carrier(j,:);
end
% multiplexing the 6 signals
mt = m(1,:) + m(2,:) + m(3,:) + m(4,:) + m(5,:) + m(6,:);
% plot the final transmitted message in frequency domain
figure(2)
row = 3; col = 2;
messageplot(mt,fs, 'Transmitted messages after AM', row, col, 1);
```

```
%% RF Stage
% since the reciever is tunable, I ask the user to pick only one message
prompt1 = "\nWhat is the wn of the station you want to hear?\n";
prompt2 = "(100kHz - 150kHz - 200kHz - 250kHz - 300kHz - 350kHz): ";
k = input(prompt1 + prompt2);
for j = 1 : 6
    if k*power(10,3) == fc(j)
        i = j;
        message = file(j);
    end
end
% filter design for the wanted message
xRF = 4000; % thickness of the filter
F_stop1 = fc(i) - Bw - xRF;     % Edge of the stopband
F_pass1 = fc(i) - Bw;           % Edge of the passband
F_pass2 = fc(i) + Bw;           % Closing edge of the passband
F_stop2 = fc(i) + Bw + xRF;     % Edge of the second stopband
RF = fdesign.bandpass('Fst1,Fp1,Fp2,Fst2,Ast1,Ap,Ast2',F_stop1,F_pass1,F_pass2,F_stop2,60,1,60,fs);
RF = design(RF, 'equiripple');
mt = filter(RF, mt);
messageplot(mt,fs, 'desired message after RF stage', row, col, 2);
```

```
%% Mixer
wif = 25000; %wIF = 25kHZ
carrier = cos(2*pi*(fc(i) + wif)*t);
mt = mt.*carrier;
messageplot(mt,fs, 'after mixing with wIF', row, col, 3);
```

```
%% IF Stage
xIF = 2000;
%filter design for the wanted message
F_stop1 = wif - Bw - xIF;   % Edge of the stopband
F_pass1 = wif - Bw;         % Edge of the passband
F_pass2 = wif + Bw;         % Closing edge of the passband
F_stop2 = wif + Bw + xIF;   % Edge of the second stopband
IF = fdesign.bandpass('Fst1,Fp1,Fp2,Fst2,Ast1,Ap,Ast2',F_stop1,F_pass1,F_pass2,F_stop2,60,1,60,fs);
IF = design(IF, 'equiripple');
mt = filter(IF, mt);
messageplot(mt,fs, 'IF Stage', row, col, 4);
```

```
%% Baseband Detection
carrier = cos(2*pi*wif*t);
mt = mt.*carrier;
% design of low-pass filter
xLPF = 3000;
```

```
F_pass = Bw;         % Edge of the passband
F_stop = Bw + xLPF; % Edge of the stopband
LPF = fdesign.lowpass('Fp,Fst,Ap,Ast',F_pass,F_stop,1,60,fs);
LPF = design(LPF,'equiripple');
mt = filter(LPF, mt);
messageplot(mt,fs, 'LPF Stage', row, col, 5);
```

```
%% Output Message
% reversing the matrix of the message back to original
mt = downsample(mt,factor);
fs = fs/factor;
messageplot(mt, fs, message, row, col, 6);
audiowrite("Output "+ message, mt, fs);
```

```
%% Filter plots
% fvtool(RF)      % drawing the RF filter
% fvtool(IF)      % drawing the IF filter
% fvtool(LPF)     % drawing the LPF filter
```

MATLAB Code of the plotting function:

```
function messageplot(m, fs, txt, x, y, n)
subplot(x,y,n)
m = fft(m);
N = size(m,2);
k=-N/2:N/2-1;
plot(k*fs/N,fftshift(abs(m)))
title(txt, 'Interpreter', 'none');
```