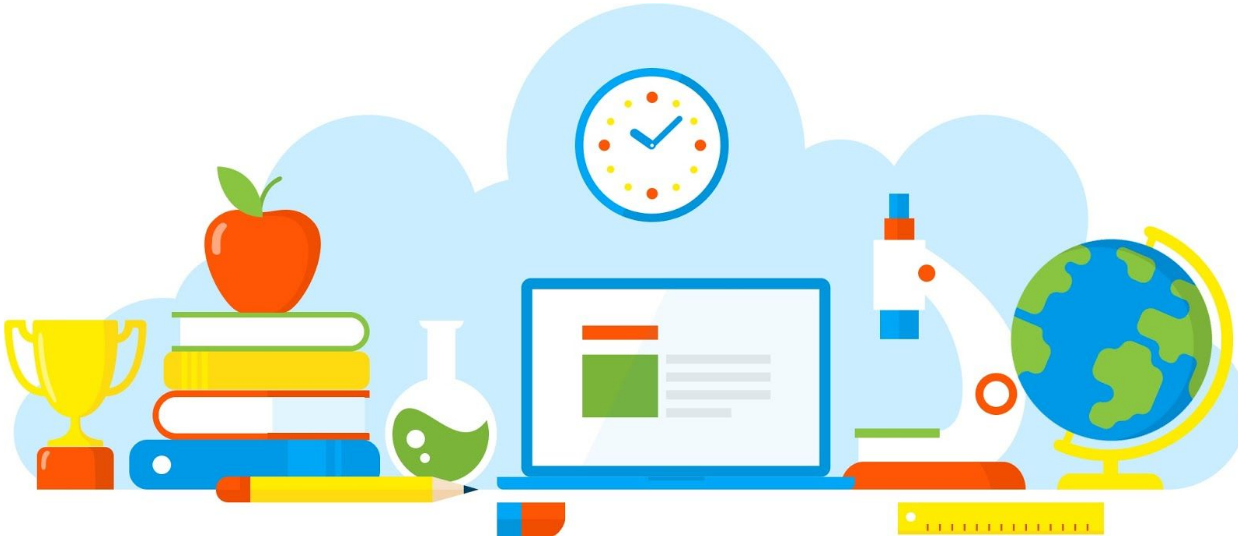
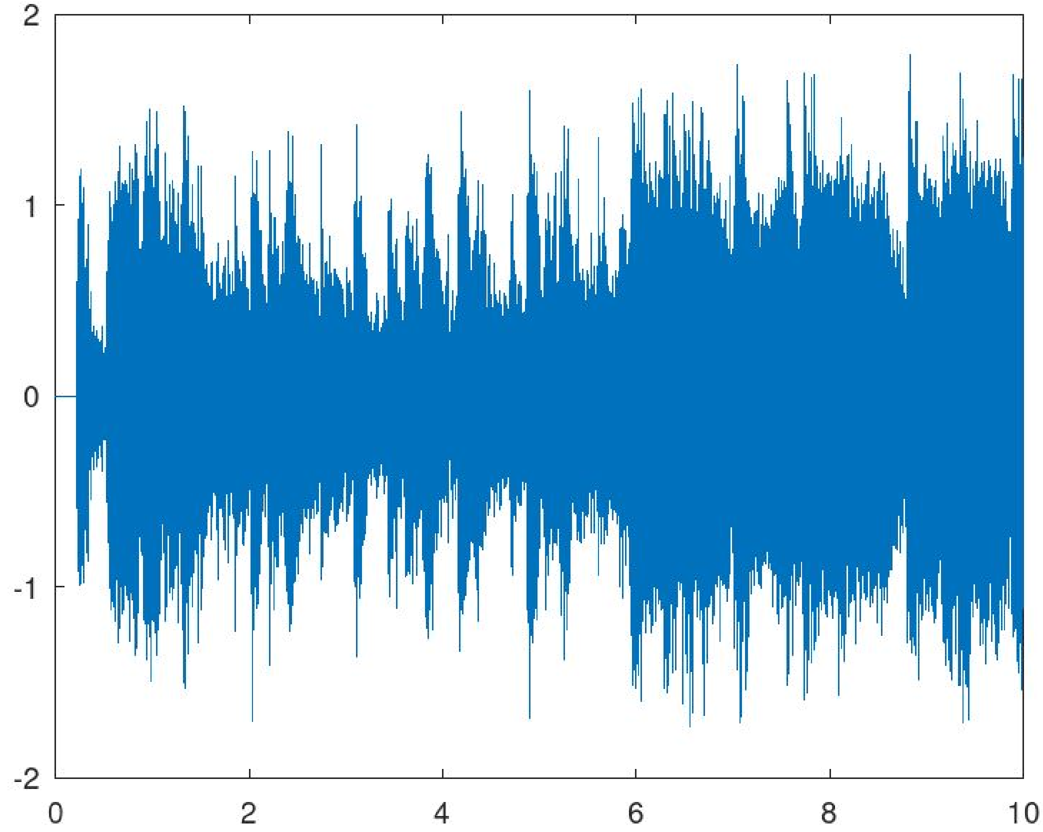


Analog Communication

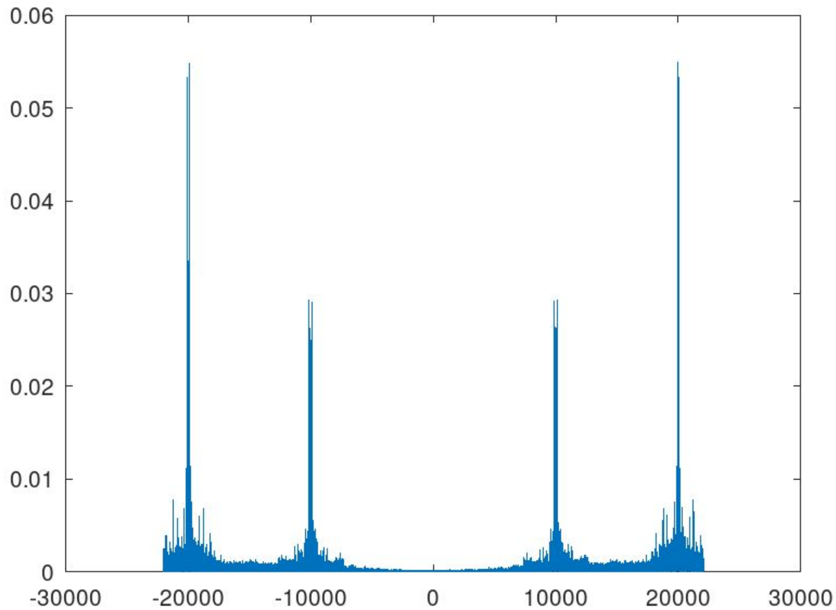
Ahmed Ashraf	SEC: 1	BN: 2
Mohamed Abo-Bakr	SEC: 2	BN: 13

$$s(t) = x_1(t)\cos\omega_1 t + x_2(t)\cos\omega_2 t + x_3(t)\sin\omega_2 t$$

1. The modulated signal:
   a. Time Domain

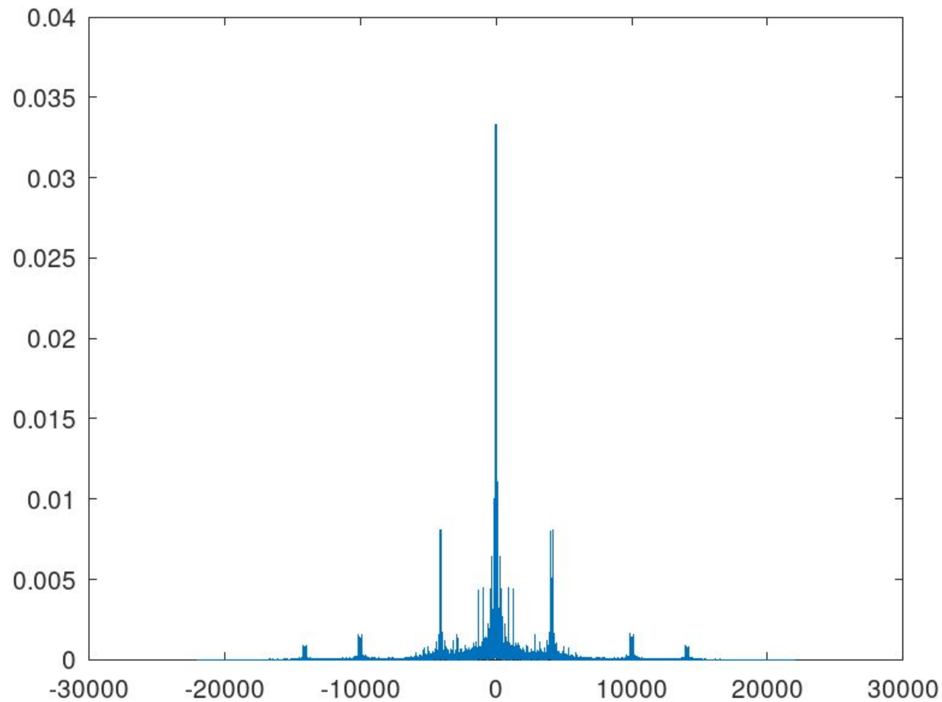


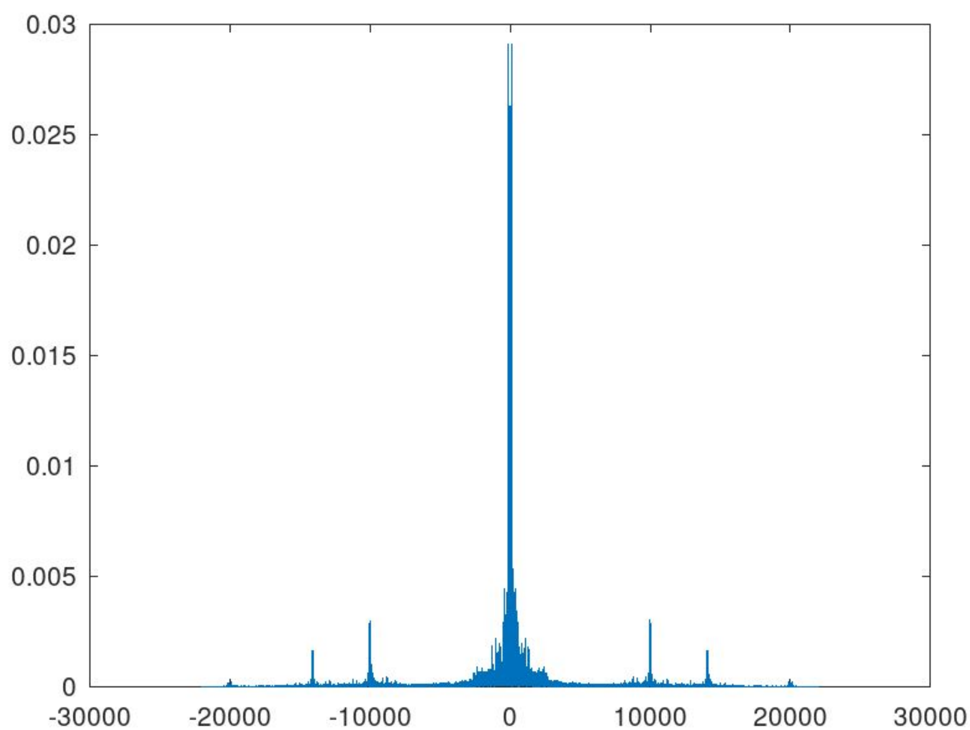
b. Magnitude Spectrum

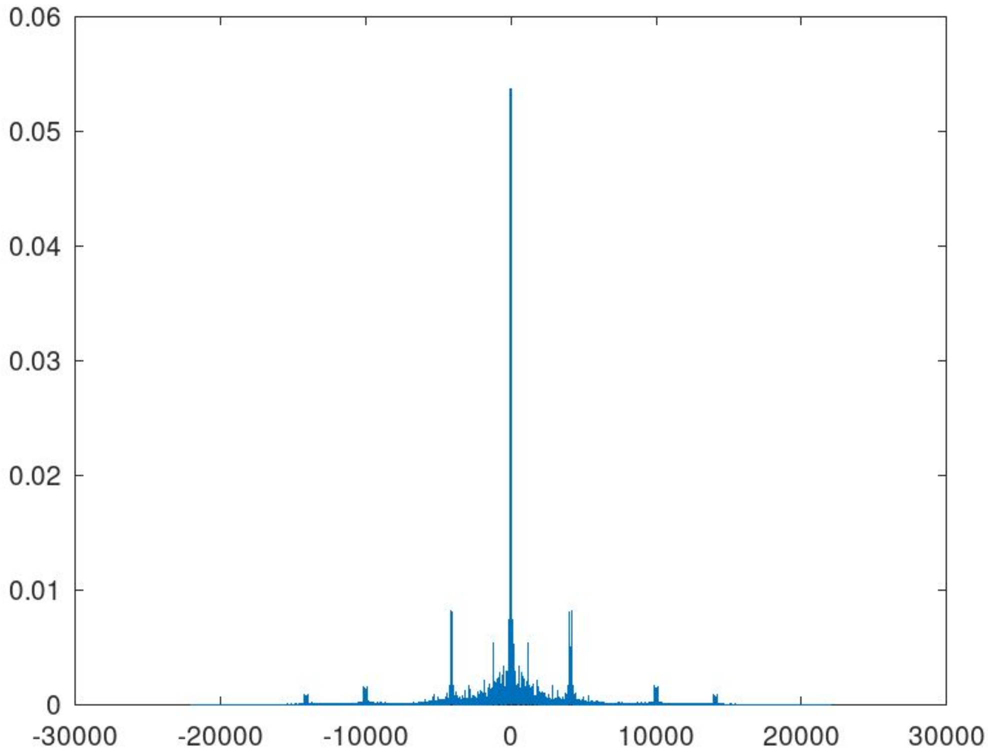


2. Synchronous Demodulation

The first signal was restored with high clarity in sound and almost there is no interference between the first signal and the others because there is a wide frequency range between w1 and w2. On the other hand, the interference between the second and third signal is very clear.


0.04
0.035
0.03
0.025
0.02
0.015
0.01
0.005
0
-30000
-20000
-10000
0
10000
20000
30000


0.03
0.025
0.02
0.015
0.01
0.005
0
-30000
-20000
-10000
0
10000
20000
30000



3. Phase Shifts

   The interference increases gradually directly proportional to the shift phase increasing especially the second and the third signals

   a. 10 Degrees: Almost the same as the sync demodulation.
   b. 30 Degrees: Started to distinguish the increase in interference between the second and third signal with increase in noise in the first signal.
   c. 90 Degrees: The interference and the noise is very clear in both second and third signals with high noise in the first signal.

Code

```
function de_y = demodulate(thresh, mod_sig, carrier)
   de_y = mod_sig .* carrier;

   % If you use Matlab
   % de_y = lowpass(de_y, thresh);

   % If you use Octave
   [b,a] = butter(1, thresh);
   de_y = filter(b,a,de_y);
end
function [mag, f] = frequency(sig,L, fs)

x = fft(sig);

x = fftshift(x);
mag2 = abs(x/L);
mag = mag2 * 2;
f = fs*(-L/2+1 : L/2) / L;
end


w1 = 2*pi*10000;     % omega's
w2 = 2*pi*20000;

lowPassThresh = 0.06;

% read signals
[y1, fs] = audioread("sig1.wav");
[y2, ] = audioread("sig2.wav");
[y3, ] = audioread("sig3.wav");

totalTime = 10;  %sec
T = 1/ fs;
L = totalTime * fs;  %length = time  *number of samples in one second
t = T: T : totalTime;
```

```
y1 = transpose(y1(:,2));     % one channel only
y2 = transpose(y2(:,2));
y3 = transpose(y3(:, 2));

% modulated signal
mod_sig = y1 .* cos(t*w1) + y2 .* cos(t*w2) + y3 .* sin(t*w2);

% demodulate signals
de_y1 = demodulate(lowPassThresh, mod_sig, cos(t*w1));
de_y2 = demodulate(lowPassThresh, mod_sig, cos(t*w2));
de_y3 = demodulate(lowPassThresh, mod_sig, sin(t*w2));

% sound(2*de_y1, fs);
% sound(2*de_y2, fs);
% sound(2*de_y3, fs);

% Phase Shift
de_y1_10 = demodulate(lowPassThresh, mod_sig, cos(t*w1+10));
de_y2_10 = demodulate(lowPassThresh, mod_sig, cos(t*w2+10));
de_y3_10 = demodulate(lowPassThresh, mod_sig, sin(t*w2+10));

sound(2*de_y1_10, fs);
sound(2*de_y2_10, fs);
sound(2*de_y3_10, fs);

de_y1_30 = demodulate(lowPassThresh, mod_sig, cos(t*w1+30));
de_y2_30 = demodulate(lowPassThresh, mod_sig, cos(t*w2+30));
de_y3_30 = demodulate(lowPassThresh, mod_sig, sin(t*w2+30));

sound(2*de_y1_30, fs);
sound(2*de_y2_30, fs);
sound(2*de_y3_30, fs);

de_y1_90 = demodulate(lowPassThresh, mod_sig, cos(t*w1+90));
de_y2_90 = demodulate(lowPassThresh, mod_sig, cos(t*w2+90));
de_y3_90 = demodulate(lowPassThresh, mod_sig, sin(t*w2+90));

sound(2*de_y1_90, fs);
sound(2*de_y2_90, fs);
```

```
sound(2*de_y3_90, fs);

% plot modulated signal
figure
plot(t, mod_sig);

% plot modulated signal magnitude spectrum
figure
[mag, f] = frequency(mod_sig, L, fs);
plot(f, mag);


% plot demodulated signal 1 spectrum
figure
[mag, f] = frequency(de_y1, L, fs);
plot(f, mag);

% plot demodulated signal 2 spectrum
figure
[mag, f] = frequency(de_y2, L, fs);
plot(f, mag);

% plot demodulated signal 3 spectrum
figure
[mag, f] = frequency(de_y3, L, fs);
plot(f, mag);

% play demodulated signal
% sound(2*de_y1, fs);
```