EE 391 Matlab MP1

Section: 2
Utku Kurtulmuş - 21903025

Part 1: Periodic and Aperiodic Audio Signals

FULL MATLAB CODE FOR PART-1

```
%Part1 : Periodic and Aperiodic Audio Signals
id = [2 1 9 0 3 0 2 5];
%first column represents the frequencies in HZ, second column represents
%the real part of the complex amplitude and the third column represents the
%imaginary part of the complex amplitude
signal1 = 100*[id(1), id(5), id(6); id(2)*2, -id(7), id(8);
               id(3), id(1), id(2); id(4) * 4, id(3), id(4);
               id(1), id(5), -id(6); id(2)*2, -id(7), -id(8);
               id(3), id(1), -id(2); id(4) * 4, id(3), -id(4);
              -id(1), id(5), id(6); -id(2)*2, -id(7), id(8);
              -id(3), id(1), id(2); -id(4) * 4, id(3), id(4);
              -id(1), id(5), -id(6); -id(2)*2, -id(7), -id(8);
              -id(3), id(1), -id(2); -id(4) * 4, id(3), -id(4)
               ];

signal2 = 100*[id(1), id(5), id(6); id(2)*sqrt(5), -id(7), id(8);
               id(3), id(1), id(2); id(4) * sqrt(15), id(3), id(4);
               id(1), id(5), -id(6); id(2)*sqrt(5), -id(7), -id(8);
               id(3), id(1), -id(2); id(4) * sqrt(15), id(3), -id(4);
              -id(1), id(5), id(6); -id(2)*sqrt(5), -id(7), id(8);
              -id(3), id(1), id(2); -id(4) * sqrt(15), id(3), id(4);
              -id(1), id(5), -id(6); -id(2)*sqrt(5), -id(7), -id(8);
              -id(3), id(1), -id(2); -id(4) * sqrt(15), id(3), -id(4)
             ];
%Part1 - (a)
%express given complex amplitudes in polar form (r,Θ)
amplitude_1 = zeros(16,2);
for i = 1:size(amplitude_1, 1)
   for j = 1:size(amplitude_1, 2)
       if j == 1
           amplitude_1(i,j) = sqrt(signal1(i,2)^2 + signal1(i,3)^2);
       else
           amplitude_1(i,j) = atan2(signal1(i,3),signal1(i,2));
       end
   end
end
amplitude_2 = zeros(16,2);
for i = 1:size(amplitude_2, 1)
   for j = 1:size(amplitude_2, 2)
       if j == 1
           amplitude_2(i,j) = sqrt(signal2(i,2)^2 + signal2(i,3)^2);
       else
           amplitude_2(i,j) = atan2(signal2(i,3),signal2(i,2));
       end
   end
end
```

```
% Express signals as a sum of complex exponential signals, z = re^(jϴ)
exp_signal1 = 0;
for i = 1:size(amplitude_1, 1)
   exp_signal1 = exp_signal1 + amplitude_1(i, 1) * exp(j*amplitude_1(i,
2));
end
exp_signal2 = 0;
for i = 1:size(amplitude_2, 1)
   exp_signal2 = exp_signal2 + amplitude_2(i, 1) * exp(j*amplitude_2(i,
2));
end
% Express signals as a sum of sinusoids,
% inverse euler's formula e^(jϴ) = cos(ϴ) + j*sin(ϴ)
% z = rcos(ϴ) + jr*sin(ϴ)
sin_signal1 = 0;
for i = 1:size(amplitude_1, 1)
   sin_signal1 = sin_signal1 + amplitude_1(i, 1) * cos(amplitude_1(i, 2)) +
j*amplitude_1(i, 1) * sin(amplitude_1(i, 2));
end
sin_signal2 = 0;
for i = 1:size(amplitude_2, 1)
   sin_signal2 = sin_signal2 + amplitude_2(i, 1) * cos(amplitude_2(i, 2)) +
j*amplitude_2(i, 1) * sin(amplitude_2(i, 2));
end
% End Of Part1 - (a)
% Part1 - (b)
% Define the sampling frequency, fs, and the duration of the signals, tend
fs = 100*max(max(signal1(:,1)), max(signal2(:,1)));
tend = 0.1; % seconds
% Define the time vector, ts, and the sample index vector, n
%creates a vector of evenly spaced time samples from 0 to tend with a
spacing of 1/fs.
ts = 0:(1/fs):tend;
n = 0:length(ts)-1; %index of each sample in the time vector,
% Obtain the samples of the signals
x1 = zeros(size(ts));
x2 = zeros(size(ts));
for i = 1:length(ts)
   % Evaluate signal 1 at time ts(i)
   %  A * cos(2πf t + ϴ) real part of the signal
   for j = 1:size(signal1, 1)
       x1(i) = x1(i) + amplitude_1(j, 1)*cos(2*pi*signal1(j, 1)*ts(i) +
amplitude_1(j, 2));
   end

   % Evaluate signal 2 at time ts(i)
   for j = 1:size(signal2, 1)
       x2(i) = x2(i) + amplitude_2(j, 1)*cos(2*pi*signal2(j, 1)*ts(i) +
amplitude_2(j, 2));
   end
end
% End Of Part1 - (b)
```

```matlab
% Part1 - (c)
soundsc(x1,fs);
soundsc(x2,fs);
% End of Part1 - (c)
% Part1 - (d)
% study plot command
% End Of Part1 - (d)
% Part1 - (e)
figure('Position', [100, 100, 800, 600]);
subplot(1,2,1);
plot(ts,x1);
ylabel('x1[n] - Amplitude', 'FontWeight','bold', FontSize=12);
xlabel('time (s)', 'FontWeight','bold',FontSize=12);
grid("on");
title('Signal 1 (Periodic)','FontSize',12);
subplot(1,2,2);
plot(ts,x2);
ylabel('x2[n] - Amplitude','FontWeight','bold','FontSize',12);
xlabel('time (s)','FontWeight','bold','FontSize',12);
grid("on");
title('Signal 2 (Not Periodic)','FontSize',12);
% End of Part1 - (e)
% Part1 - (f)
f_1 = signal1(:,1)'; %frequencies of signal1
f_2 = signal2(:,1)'; %frequencies of signal2
y_1 = zeros(1,16);
%ak = (1/2) * A * e^jΘ
for i = 1:size(amplitude_1,1)
   y_1(1,i) = 0.5*amplitude_1(i,1)*exp(1i*amplitude_1(i,2));
end
y_2 = zeros(1,16);
%ak = (1/2) * A * e^jΘ
for i = 1:size(amplitude_2,1)
   y_2(1,i) = 0.5*amplitude_2(i,1)*exp(1i*amplitude_2(i,2));
end
figure('Position', [100, 100, 800, 600]);
subplot(1,2,1);
stem(f_1, abs(y_1));
ylabel('Amplitude', 'FontWeight','bold', FontSize=12);
xlabel('Frequency (Hz)', 'FontWeight','bold',FontSize=12);
grid("on");
title('Signal 1 (Periodic)','FontSize',12);
subplot(1,2,2);
stem(f_2, abs(y_2));
ylabel('Amplitude','FontWeight','bold','FontSize',12);
xlabel('Frequency (Hz)','FontWeight','bold','FontSize',12);
grid("on");
title('Signal 2 (Not Periodic)','FontSize',12);
% End of Part1 - (f)
```

Plots for PART-1

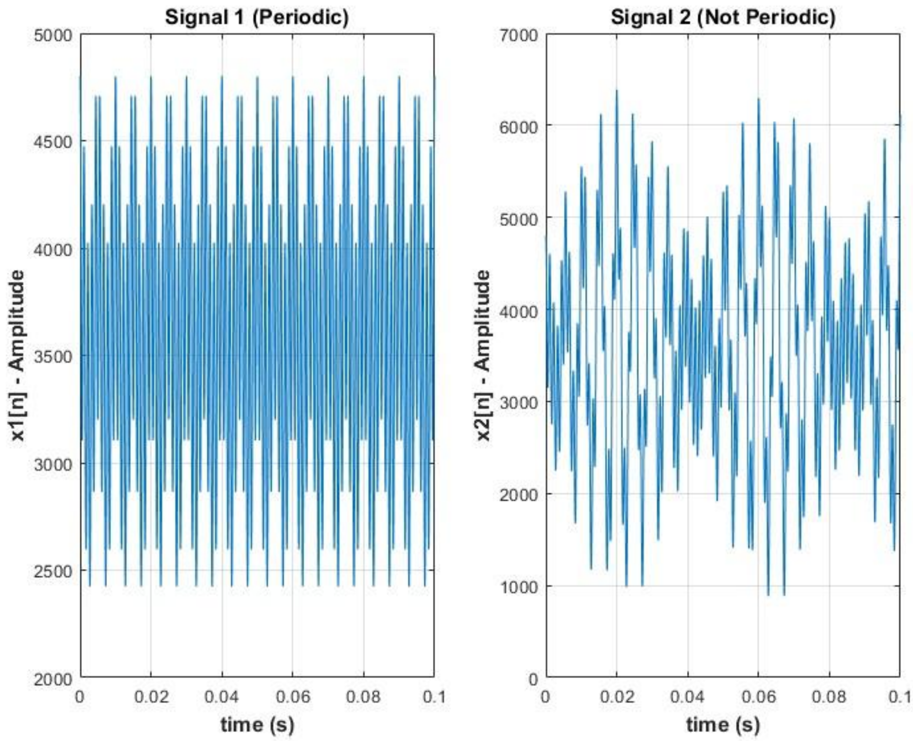


figure for part - 1b

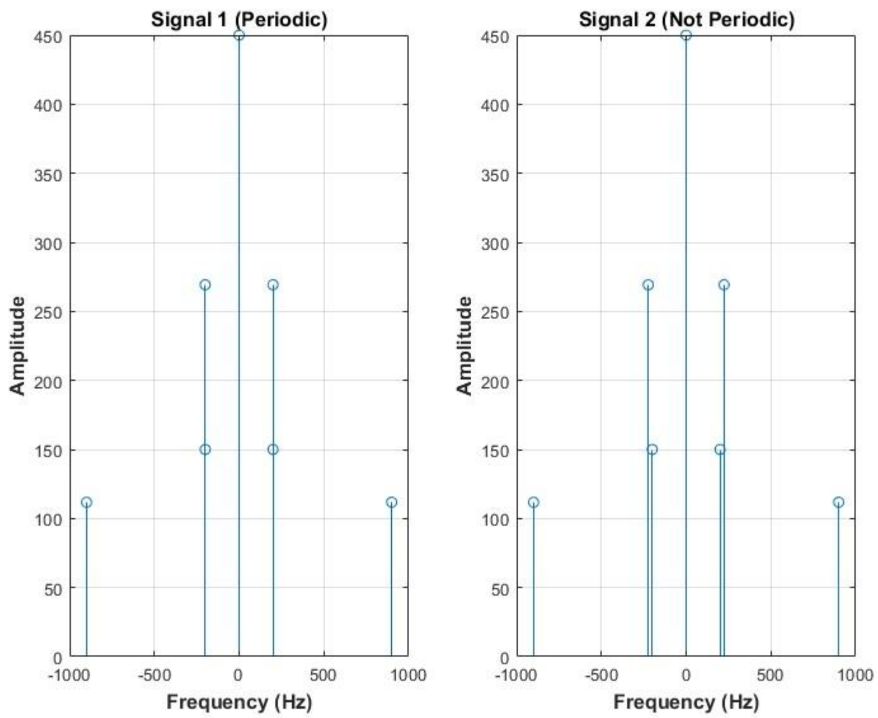


figure for part - 1f

Explanations for Part 1 - (e) and (f)

From the frequency spectrum of signal 1 and signal 2, we can see that their frequency spectrum is similar to each other. However, from the graphs of these two audio signals, we can see that their behavior is very different, and signal 1 repeats itself periodically but signal 2 does not. The main reason for that is what we call harmonic relations. The signals that forms signal 1 have harmonic relation, their frequencies are harmonically related. In other words, all frequencies are integer multiples of the fundamental frequencies. In signal 2 although the frequencies are very similar to signal 1, they have irrational numbers and this breaks the harmonic relations. Moreover, we observe complex conjugate symmetry on both of the signals even though signal 2 is not periodic because both of the audio signals are real-valued signals.

Part 2: Beat Frequency

FULL MATLAB CODE FOR PART 2

```
% Part 2: Beat Frequency
% Part 2 - (a)
% Create sinusoidal signal
f = 300; % Cyclic frequency
fs = f*100; % Sampling frequency
tend = 0.1; % in seconds
t = 0:1/fs:tend; % Time vector for sampling
x1 = sin(2*pi*f*t); % discritized signal
%hear the signal
%soundsc(x1,fs);
% Plot signal in time domain
plot(t, x1);
xlabel('Time (s)');
ylabel('Amplitude');
grid("on");
title('Sinusoidal Signal (x1)');
% End of Part 2 - (a)
% Part 2 - (b)
%Create second sinusoid x2(t) and compare with x1(t)
% Define new frequency for x2(t), increase the frequency of x1 by 10HZ
f2 = f + 10;
% Create new sinusoidal signal x2(t) by using the same time vector as x1(t)
x2 = sin(2*pi*f2*t); %discretized signal
% Listen the signal x2 to find the differences
%soundsc(x2, fs);
% Plot signals x1(t) and x2(t) in time domain
figure;
plot(t, x1, 'b', t, x2, 'r');
xlabel('Time (s)');
ylabel('Amplitude');
grid on;
title('Sinusoidal Signals x1(t) vs x2(t)');
legend('x1(t)', 'x2(t)');
% End of Part 2 - (b)
```

```
% Part 2 - (c)
%Create the signal x3
x3 = x1 + x2;
%Plot signals x1(t), x2(t), and x3(t) in time domain
figure;
plot(t, x1, 'b', t, x2, 'r', t, x3, 'g');
xlabel('Time (s)');
ylabel('Amplitude');
grid on;
title('Sinusoidal Signals x1(t), x2(t), and x3(t)');
legend('x1(t)', 'x2(t)', 'x3(t)');
%plot them also as subplots
figure;
subplot(3,1,1);
plot(t, x1);
xlabel('Time (s)');
ylabel('Amplitude');
grid on;
title('Sinusoidal Signal x1(t)');
subplot(3,1,2);
plot(t, x2);
xlabel('Time (s)');
ylabel('Amplitude');
grid on;
title('Sinusoidal Signal x2(t)');
subplot(3,1,3);
plot(t, x3);
xlabel('Time (s)');
ylabel('Amplitude');
grid on;
title('Sum of Sinusoidal Signals x3(t)');
% Obtain the Fourier transform of the signals
X1 = fft(x1);
X2 = fft(x2);
X3 = fft(x3);
% Obtain the frequency vector
N = length(x1);
fvec = (-N/2:N/2-1)*fs/N;

% Plot the frequency spectra of the signals
figure;
subplot(3,1,1);
plot(fvec, fftshift(abs(X1)), 'b');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
grid on;
title('Frequency Spectrum of Sinusoidal Signal x1(t)');
subplot(3,1,2);
plot(fvec, fftshift(abs(X2)), 'r');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
grid on;
title('Frequency Spectrum of Sinusoidal Signal x2(t)');
```

```
subplot(3,1,3);
plot(fvec, fftshift(abs(X3)), 'g');
xlabel('Frequency (Hz)');
ylabel('Magnitude');
grid on;
title('Frequency Spectrum of Sum of Sinusoidal Signals x3(t)');
% Listen the signal x3
soundsc(x3, fs);
% End of Part 2 - (c)
% Part 2 - (d)
%Beat frequency is equal to the difference between the frequencies of the
%two signal
beat_frequency = abs(f - f1);
% End of Part 2 - (d)
```

Plots for PART - 2

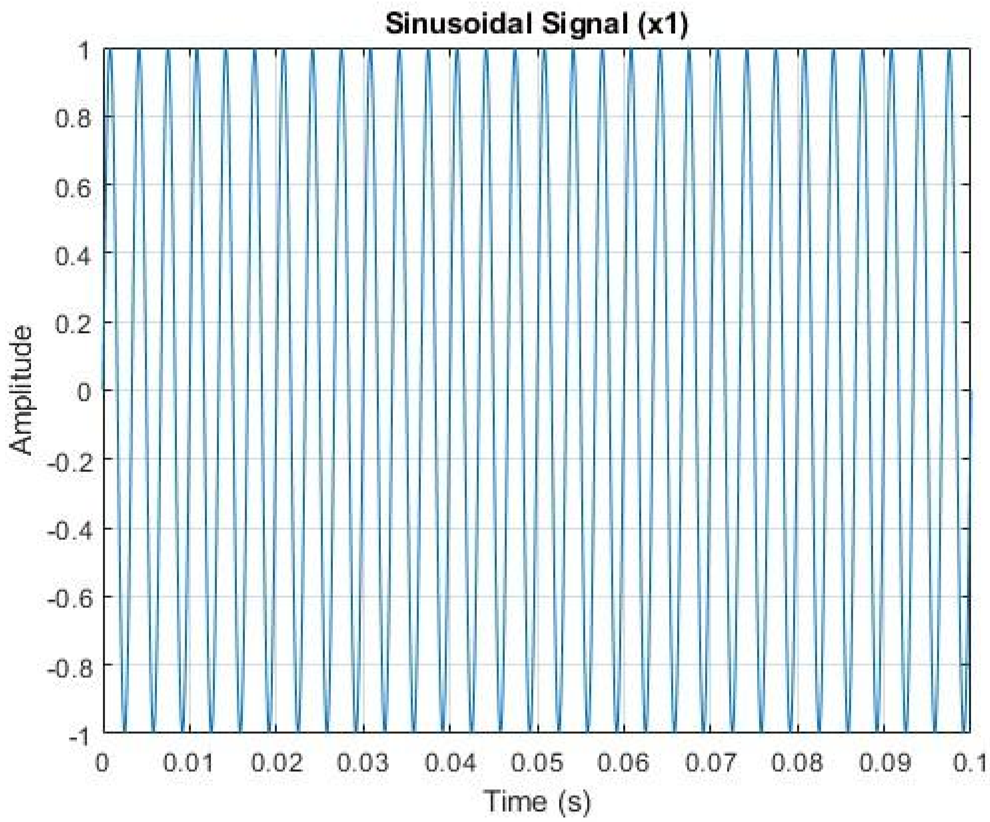


figure for part - 2a

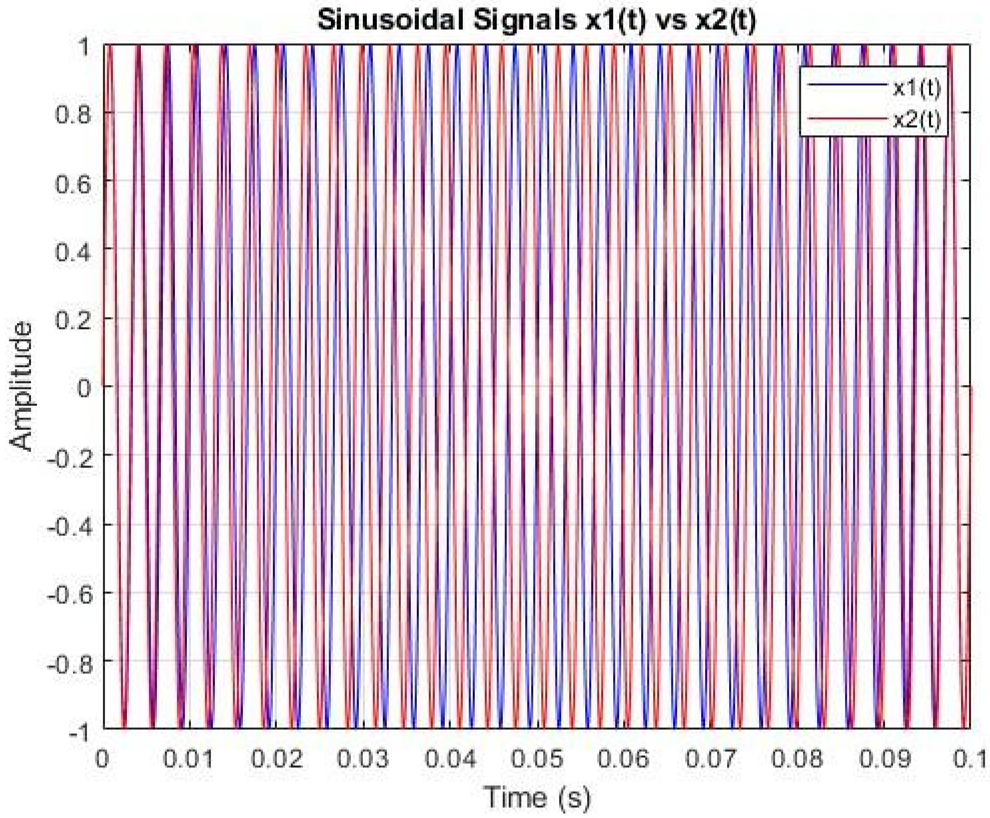
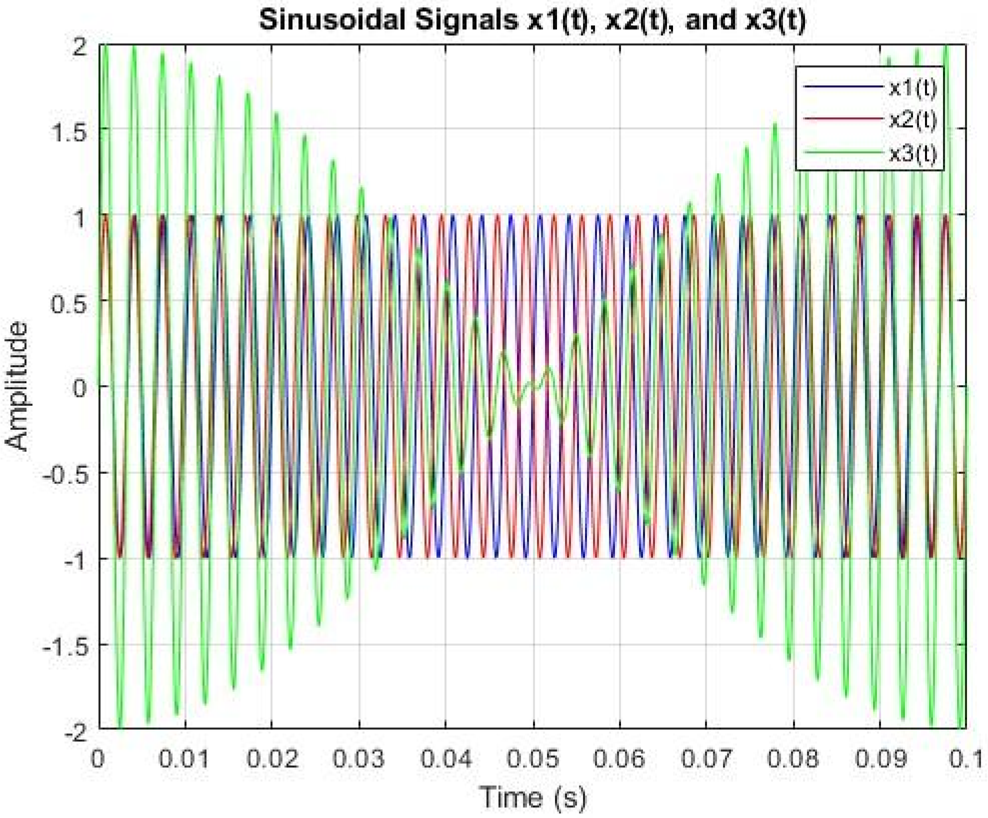
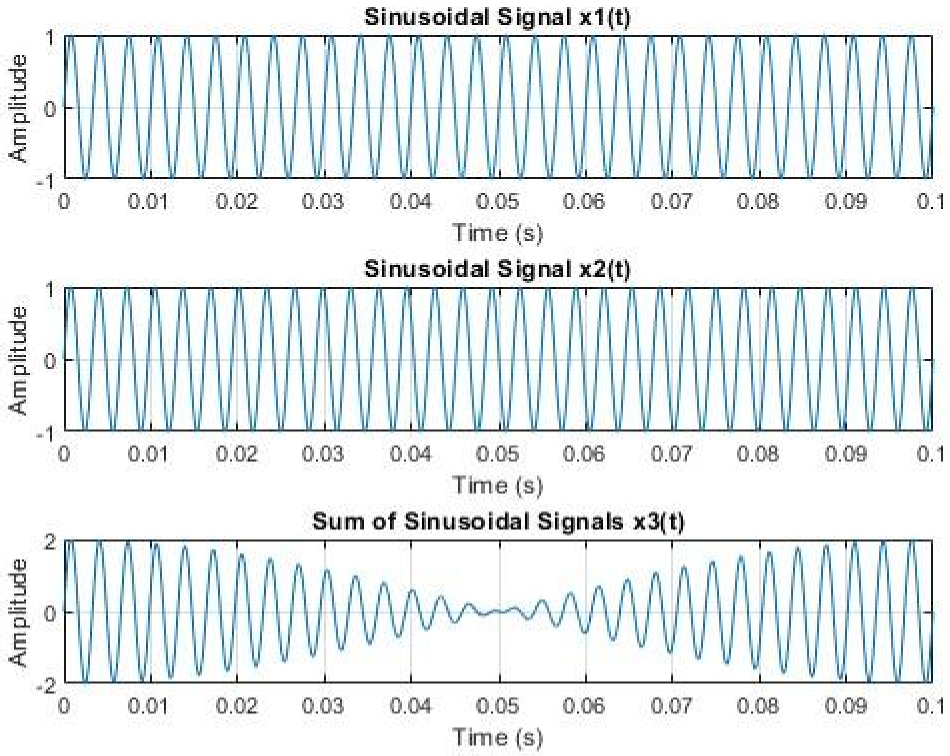
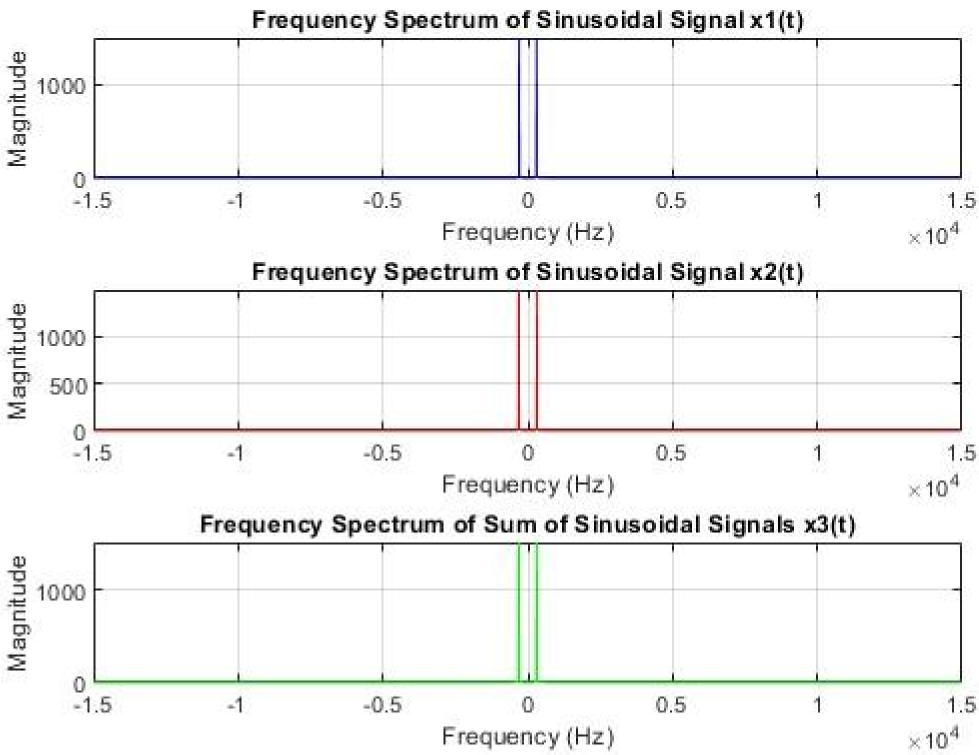


figure for part - 2b


Sinusoidal Signals x1(t), x2(t), and x3(t)
x1(t)
x2(t)
x3(t)
Amplitude
Time (s)


Sinusoidal Signal x1(t)
Amplitude
Time (s)
Sinusoidal Signal x2(t)
Amplitude
Time (s)
Sum of Sinusoidal Signals x3(t)
Amplitude
Time (s)



figures for part 2-c

Explanations for Part2

(b) → I can't hear a noticeable difference between x1 and x2, they are very similar. There is a constant high noise that repeats itself.

(c) → The signal x3, sounds different x1 and x2. The signal x3 warbles with a sequence. Its amplitude goes high and low and repeats itself. This is due to the phenomenon called the beating of tones in music.

(d) → We can see that the x3 repeats itself with a period, hence has a frequency called beating frequency. We can also observe that the beating frequency equal to the absolute differences between the cyclic frequencies of x1 and x2.