National University of Sciences and Technology (NUST)
School of Electrical Engineering and Computer Science

Department of Electrical Engineering

Faculty Member: Dr. Wajahat Hussain **Dated:** 21-2-2107

Course/Section: BEE6-B **Semester:** 6th Semester

EE-330 Digital Signal Processing

Lab2: Complex Exponentials and Sinusoids

Name	Reg. no.	Report Marks / 10	Lab Quiz-Viva Marks / 5	Total / 15
Abdullah Bin Asif	111596			
Saad Iqbal	111394			
Usman Iqbal	111393			

National University of Sciences and Technology (NUST)
School of Electrical Engineering and Computer Science

Lab2: Complex Exponentials and Sinusoids

Objectives

The goal of this Part is to gain familiarity with complex numbers and their use in representing sinusoidal signals such as $x(t) = Acos(\omega t + \emptyset)$ as complex exponentials $z(t) = Ae^{j\theta}e^{j\omega t}$. The key is to use the appropriate complex amplitude together with the real part operator as follows:

$$x(t) = Acos(\omega t + \emptyset) = Real\{Ae^{j\theta}e^{j\omega t}\}$$

- ✓ How to work with Complex Numbers in MATLAB
- ✓ Familiarization with MATLAB Function and commands for Complex Exponentials
- ✓ Sinusoid Addition Using Complex Exponentials
- ✓ Spectrogram of sinusoid

1. Introduction to Complex Exponentials

1.1. Introduction

Manipulating sinusoidal functions using complex exponentials turns trigonometric problems into simple arithmetic and algebra. In this lab, we first review the complex exponential signal and the phasor addition property needed for adding cosine waves. Then we will use MATLAB to make plots of phasor diagrams that show the vector addition needed when adding sinusoids.

1.1.1. Complex Numbers in MATLAB

Here are some of MATLAB's built-in complex number operators:

Conj	Complex conjugate
Abs	Magnitude
Angle	Angle (or phase) in radians
Real	Real part
Imag	Imaginary part
i,j	pre-defined as $\sqrt{-1}$
x = 3 + 4i , I	suffix defines imaginary constant (same for j suffix)
exp(j*theta)	Function for the complex exponential $e^{j\theta}$

Each of these functions takes a vector (or matrix) as its input argument and operates on each element of the vector. Notice that the function names $mag()$ and $phase()$ do not exist in MATLAB.

1.1.2. Sinusoid Addition Using Complex Exponentials

Recall that sinusoids may be expressed as the real part of a complex exponential:

$$x(t) = Acos(2\pi f_o t + \emptyset) = Real\{Ae^{j\theta}e^{2j\pi f_0 t}\} \quad (1)$$

The Phasor Addition Rule shows how to add several sinusoids:

$$x(t) = \sum_{k=1}^{N} A_k cos(2\pi f_o t + \emptyset) \quad (2)$$

Assuming that each sinusoid in the sum has the same frequency, f_0. This sum is difficult to simplify using trigonometric identities, but it reduces to an algebraic sum of complex numbers when solved using complex exponentials. If we represent each sinusoid with its complex amplitude

$$X_k = A_k e^{j\emptyset_k} \quad (3)$$

Then the complex amplitude of the sum is

$$X_s = \sum_{k=1}^{N} X_{k=} A_s e^{j\emptyset_s} \quad (4)$$

Based on this complex number manipulation, the Phasor Addition Rule implies that the amplitude and phase of $x(t)$ in equation (2) are as and ϕ_s, so

$$x(t) = A_s cos(2\pi f_o t + \emptyset_s) \quad (5)$$

We see that the sum signal $x(t)$ in (2) and (5) is a single sinusoid that still has the same frequency, f_o , and it is periodic with period $T_0 = 1/f_o$.

1.1.3. Harmonic Sinusoids

There is an important extension where $x(t)$ is the sum of N cosine waves whose frequencies (f_k) are different. If we concentrate on the case where the f_k are all multiples of one basic frequency f_o

$$f_k = kf_0 \ (HARMONIC\ FREQUENCIES),$$

Then the sum of N cosine waves given by (2) becomes:

$$x_h(t) = A_k cos(2\pi f_o t + \emptyset_k) = Real\{\sum_{k=1}^{N} X_k\, e^{j2\pi k f_0 t}\} \quad (6)$$

This particular signal $x_h(t)$ has the property that it is also periodic with period $T_o = 1/f_o$, because each of the cosines in the sum repeats with period T_o. The frequency f_o is called the ***fundamental frequency***, and T_o is called the ***fundamental period***.

1.1.4. Sinusoid in Matlab

Following is Matlab function create sinusoid if frequency ff and duration dur.

```
%The corrected function should look something like:
function [xx,tt] = goodcos(ff,dur)
tt = 0:1/(100*ff):dur; %-- gives 100 samples per period
xx = cos(2*pi*ff*tt);
```

1.2. Lab Tasks

1.2.1. Complex Exponentials

In the Pre-Lab part of this lab, you learned how to write M-files. In this section, you will write two functions that can generate sinusoids or sums of sinusoids.

1.2.2. Lab Task 1:

1.2.2.1. M-file to generate a Sinusoid

Write a function that will generate a single sinusoid, $x(t) = Acos(\omega t + \emptyset)$ by using four input arguments: amplitude (A), frequency (ω), phase (ϕ) and duration (dur). The function should return two outputs: the values of the sinusoidal signal (x) and corresponding times (t) at which the sinusoid values are known. Make sure that the function generates 20 values of the sinusoid per period. Call this function one_cos(). Hint: use goodcos() from par (a) as a starting point. Demonstrate that your one_cos() function works by plotting the output for the following parameters: A = 95, ω = 200 rad/sec, ϕ = π/5 radians, and dur=0.025 seconds. Be prepared to explain to the lab instructor features on the plot that indicates how the plot has the correct period and phase. What is the expected period in millisecond?

Matlab code:

```
% single sinosidal
function [xx tt]=one_cos(amplitude,omega,phase,dur)
tt=0:(2*pi)/(omega*20):dur;
xx=amplitude*cos(omega*tt+phase);
stem(tt,xx);
```

Matlab command window:

```
>> [xx,tt]=one_cos(95,200,pi/5,0.025)

xx =

  Columns 1 through 14

   76.8566   55.8396   29.3566    0.0000  -29.3566  -55.8396  -76.8566  -90.3504  -95.0000  -90.3504  -76.8566  -55.8396  -29.3566    0.0000

  Columns 15 through 16

   29.3566   55.8396


tt =

  Columns 1 through 14

         0    0.0016    0.0031    0.0047    0.0063    0.0079    0.0094    0.0110    0.0126    0.0141    0.0157    0.0173    0.0188    0.0204

  Columns 15 through 16

    0.0220    0.0236

fx >>
```

Figure No.1

Matlab graph:

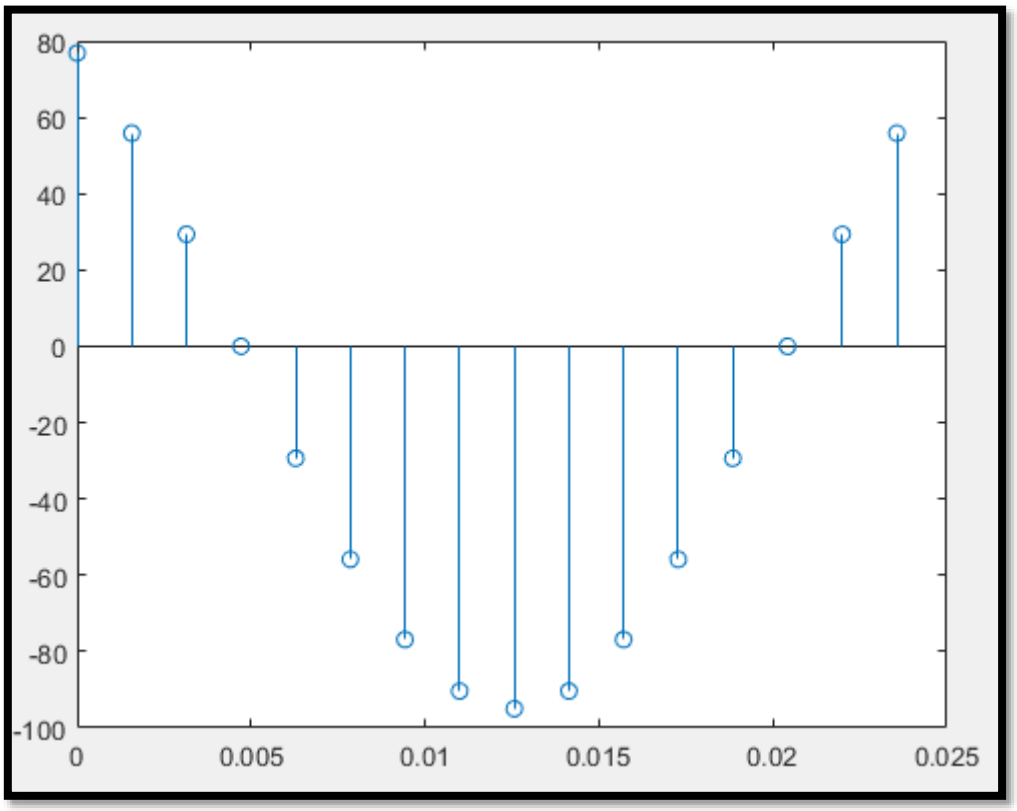

Figure No.2

Comments & Reasoning:

Using relation:

$$T = \frac{2\pi}{\omega}$$

$$T = 31.4159\ msec$$

1.2.2.2. Sinusoidal Synthesis with an M-file: Different Frequencies

Since we will generate many functions that are a "sum of sinusoids," it will be convenient to have a function for this operation. To be general, we will allow the frequency of each component (f_k) to be different. The following expressions are equivalent if we define the complex amplitudes $X_k = A_k e^{j\emptyset_k}$

$$x(t) = Real\{\sum_{k=1}^{N} X_k\, e^{j2\pi f_k t}\} \qquad (7)$$

$$x(t) = \sum_{k=1}^{N} A_k cos(2\pi f_k t + \emptyset_k) \qquad (8)$$

1.2.3. Lab task 2:

1.2.3.1. Write the Function M-file

Write an M-file called syn_sin.m that will synthesize a waveform in the form of (7).Although for loops are rather inefficient in MATLAB but you must write the function with one loop in this lab. The first few statements of the M-file are the comment lines—they should look like:

```
function [xx,tt] = syn_sin(fk, Xk, fs, dur, tstart)
%SYN_SIN Function to synthesize a sum of cosine waves
% usage:
% [xx,tt] = syn_sin(fk, Xk, fs, dur, tstart)
% fk = vector of frequencies
% (these could be negative or positive)
% Xk = vector of complex amplitudes: Amp*e^(j*phase)
% fs = the number of samples per second for the time axis
% dur = total time duration of the signal
% tstart = starting time (default is zero, if you make this input optional)
% xx = vector of sinusoidal values
% tt = vector of times, for the time axis
% Note: fk and Xk must be the same length.
% Xk(1) corresponds to frequency fk(1),
% Xk(2) corresponds to frequency fk(2), etc.
```

The MATLAB syntax length(f_k) returns the number of elements in the vector f_k, so we do not need a separate input argument for the number of frequencies. On the other hand, the programmer (that's you) should provide error checking to make sure that the lengths of f_k and X_k are the same. See help error. Finally, notice that the input fs define the number of samples per second for the cosine generation; in other words, we are no longer constrained to using 20 samples per period. Include a copy of the MATLAB code with your lab report.

Matlab code:

```
%SYN_SIN Function to synthesize a sum of cosine waves
function [xx,tt] = syn_sin(fk, Xk, fs, dur, tstart)

if(nargin==4)
    tstart= 0;
end
if(length(fk)==length(Xk))
tt=tstart:1/fs:tstart+dur;
xx=zeros(1,length(tt));
for i=1:length(fk)
temp=(real(Xk(i)*exp(j*2*pi*fk(i)*tt)));
xx=temp+xx;
end
stem(tt,xx);
else
    disp('Input are invalid');
end
```

1.2.3.2. Testing

In order to use this M-file to synthesize harmonic waveforms, you must choose the entries in the frequency vector to be integer multiples of some desired fundamental frequency. Try the following test and plot the result.

*[xx0,tt0] = syn_sin([0,100,250],[10,14*exp(-j*pi/3),8*j],10000,0.1,0);*
%-Period = ?

Matlab graph:

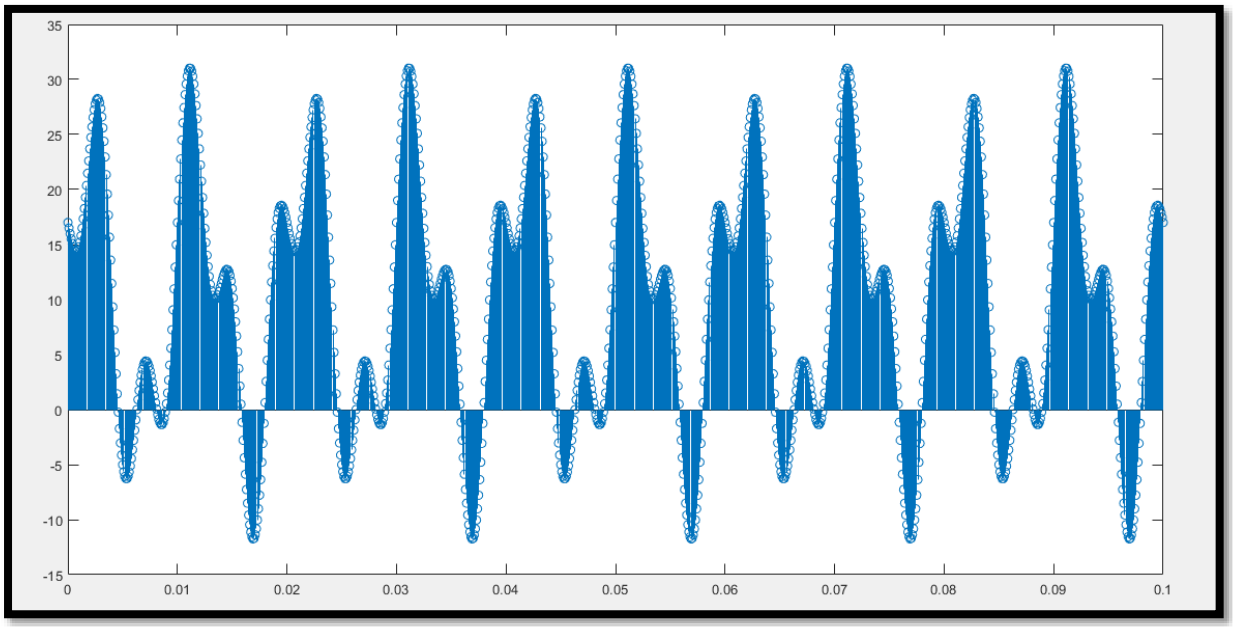

Figure No.3

Measure the period of xx0 by hand. Then compare the period of xx0 to the periods of the three sinusoids that make up xx0?

Comments & Reasoning:

Time period of xx0 is 20 msec. (Measured by hand)

$$f_1 = 0Hz$$
$$f_2 = 100Hz$$
$$f_3 = 250Hz$$

As, Fundamental frequency is H.C.F in harmonic frequencies, Therefore 50Hz is ***Fundamental Frequency***, which gives time period of 20msec. It is the same as calculated above.

1.2.4. Lab Task 3:

1.2.4.1. Representation of Sinusoids with Complex Exponentials

(a) Generate the signal and make a plot versus t.

$$x(t) = \Re e\{2e^{j\pi t} + 2e^{j\pi(t-1.25)} + (1-j)e^{j\pi t}\}$$

Use the syn_sin function and take a range for t that will cover three periods starting at t = −0.5 secs. Include the MATLAB code with your report.

Matlab command window:

```
>> [xx0,tt0] = syn_sin([0.5 0.5 0.5],[2 2*exp(-j*pi*1.25) 1-j],1000,6,-0.5);
>>
```

Figure No.4

Matlab graph:

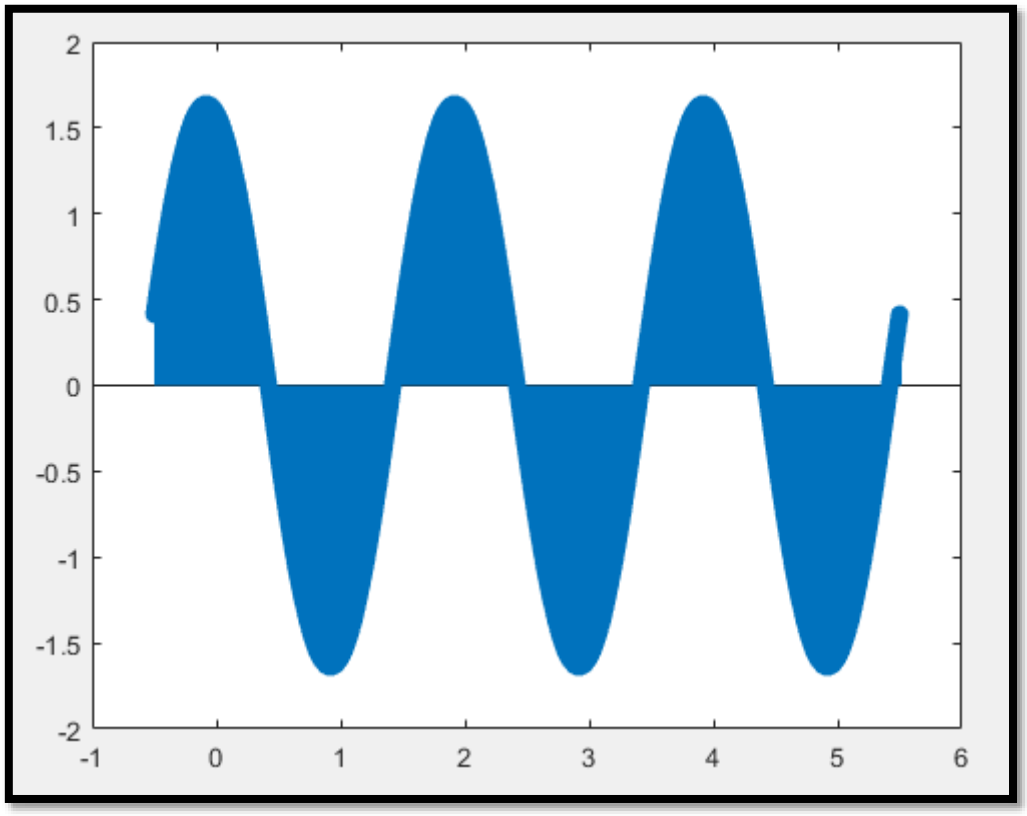

Figure No.5

(b) From the plot of $x(t)$ versus t, measure the frequency, phase and amplitude of the sinusoidal signal by hand. Show annotations on the plots to indicate how these measurements were made and what the values are. Compare to the calculation in part (c).

Matlab graph:

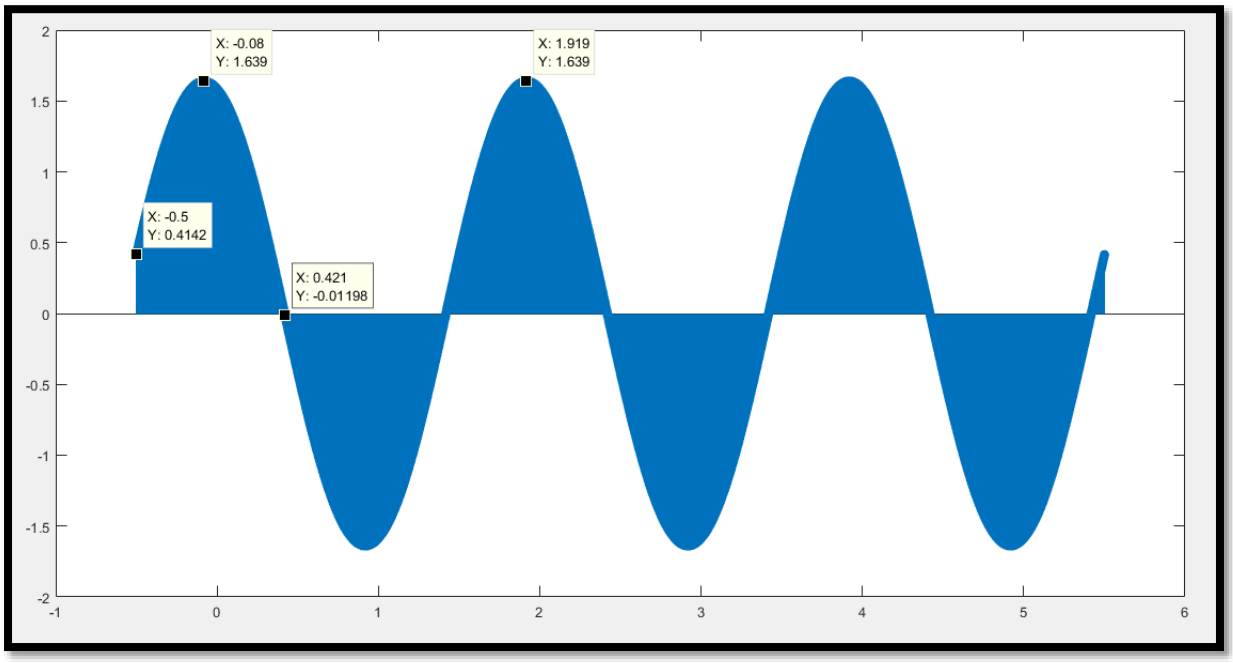

Figure No.6

Comments & Reasoning:

Attributes	Calculations using Graph	Calculation using phasor addition theorem
Frequency	0.5 Hz	0.5 Hz
Time period	2sec	2 sec
Amplitude	1.639 Units	1.6390 Units
Phasor	0.2482 rads	0.2555 rads

(c) Use the phasor addition theorem and MATLAB to determine the magnitude and phase of x(t).

```
>> [xx0,tt0] = syn_sin([0.5 0.5 0.5],[2 2*exp(-j*pi*1.25) 1-j],1000,6,-0.5);
>> info=(2)+(2*exp(-j*pi*1.25))+(1-j);
>> info

info =

   1.5858 + 0.4142i

>> abs(info)

ans =

    1.6390

>> angle(info)

ans =

    0.2555

>> freq=0.5;
>> timeperiod=1/freq

timeperiod =

     2
```

Figure No.7

1.2.5. Lab task 4:

(a) Plot the spectrogram of synthesized signal generated in lab task 3 and explain what you see in plot

Matlab command window:

```
>> [xx0,tt0] = syn_sin([0.5 0.5 0.5],[2 2*exp(-j*pi*1.25) 1-j],128,6,-0.5);
>> spectrogram(xx0,256,200,256,128)
>>
```

Figure No.8

Matlab graph:

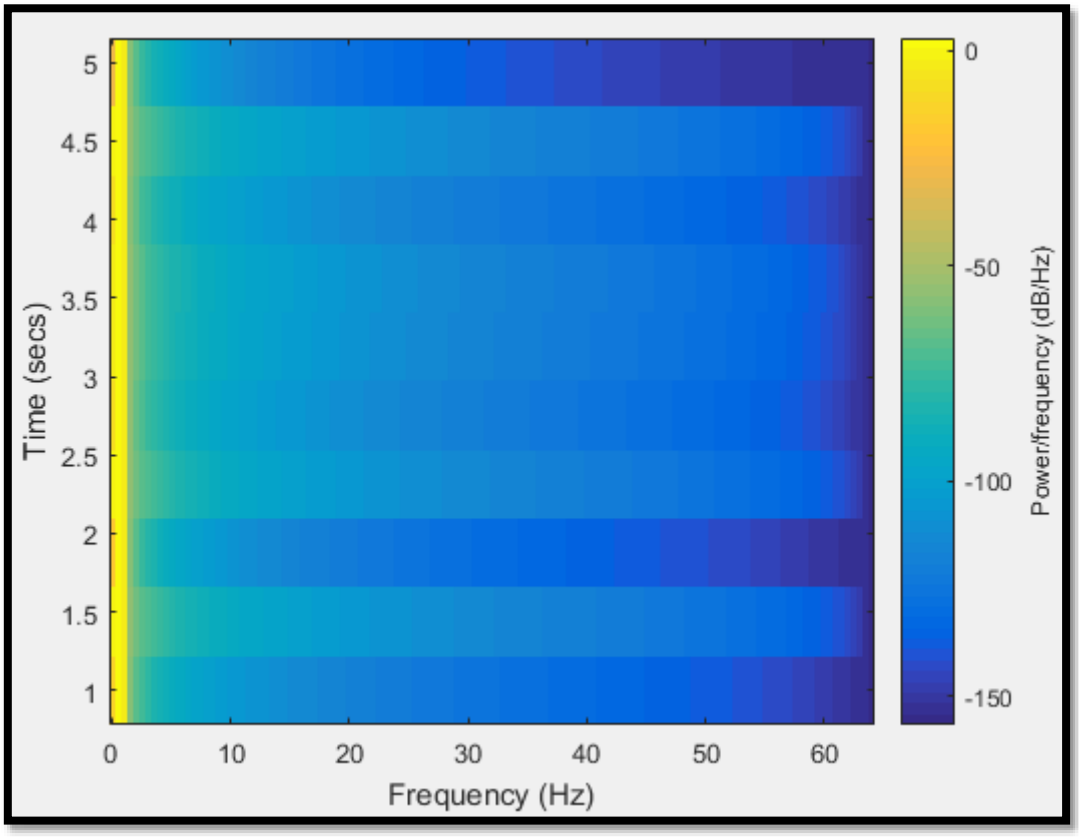

Figure No.9

(b) Create three sinusoids of 2 second and concatenate them after concatenation plot spectrogram of concatenated signal and explain the result

Matlab code:

```
tt_temp=0:1/2048:2;
%generating 3 sinosidals for dur 2sec
x1=real(10*exp(2*pi*100*j*tt_temp));
x2=real(180*exp(2*pi*250*j*tt_temp));
x3=real(100*exp(2*pi*175*j*tt_temp));
xx=[x1 x2 x3];
spectrogram(xx,128,100,128,2048);
```

Matlab command window:

```
>> mix_cos
>>
```

Figure No.10

Matlab graph:

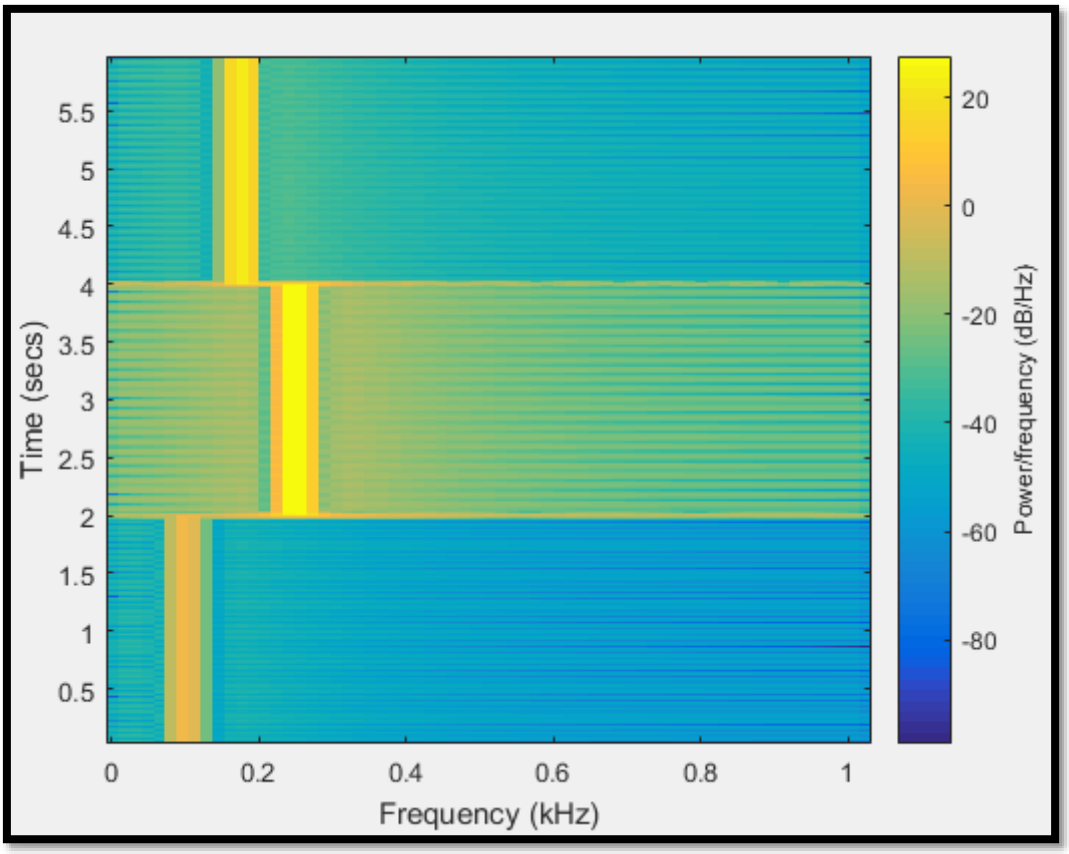

Figure No.11

Comments & Reasoning:

In first 2sec, 100 Hz frequency is present in sample.

In next 2 sec, 250 Hz frequency is present in sample.

In last 2 sec, 175 Hz frequency is present in sample.

(c) Explain the effect of window size on time and frequency resolution and also explain what advantage of spectrogram over Fourier transform is

Comments & Reasoning:

Assuming, we have aliasing-free signal sampled at f_s:

In order to have accurate FT, at least one complete period should present in window. So, low frequencies need window size greater than their Time period. Therefore, **as we increase window size, the frequency resolution enhanced**.

Whereas, **as we increased window size, then we can't predict frequencies present at instantaneous time**. So, **time resolution is decreased by increasing window size**.

There lies trade of between time and frequencies resolution in deciding window size.

1.3. Conclusion:

In this lab, we learnt to use Matlab tools to perform various operation regarding complex exponentials and sinusoids. We also developed our understanding regarding short Fourier transform (Spectrogram operation).
