TRIBHUVAN UNIVERSITY

Institute of Engineering

Central Campus, Pulchowk

Lab Report On:

Signal Analysis

Experiment No. 2

Submitted To:

Department of Electronics and Computer Engineering

Submitted By:

Rajil Bajracharya (073BEX430)

Submitted On:

24-06-019

BASIC SIGNALS

1. Objectives

- To understand the basic types of signals.
- To learn about discrete and continuous-time signals.
- To plot various signals and identify their periods and nature.
- To investigate the result of adding and/or multiplying two signals.

2. Theory

2.1 Basic Types of Signals

Following are the basic types of signals in signal analysis:

2.1.1 Unit Step Function:

The unit step function, u(t) is defined as follows:

$$u(t) = \begin{cases} 1 & for\ t \geq 0 \\ 0 & for\ t < 0 \end{cases}$$

2.1.2 Unit Delta Function:

The unit delta function is defined as follows:

$$\delta(t) = \begin{cases} 1 & for\ t = 0 \\ 0 & for\ t \neq 0 \end{cases}$$

2.1.3 Unit Ramp Function:

The unit ramp function is defined as follows:

$$r(t) = \begin{cases} \mathrm{k}t & for\ t = 0 \\ 0 & for\ t \neq 0 \end{cases} \quad where\ k = 1$$

2.1.4 Harmonic Functions:

The harmonic functions are the sine function and the cosine function, i.e.,

$$f(t) = A\sin(2\pi ft + \phi)$$
$$or$$
$$f(t) = A\cos(2\pi ft + \phi)$$

2.2 Other Types of Signals

2.2.1 Square Wave Signal:

The square wave signal consists of pulses (of square shape). It can be defined in many ways, one of them being as follows:

$$f(t) = sgn(\sin(2\pi ft))$$
$$or$$
$$f(t) = sgn(\cos(2\pi ft))$$

Where, sgn(x) is the sign function defined as follows:

$$sgn(x) = \begin{cases} 1 & for\ x \geq 0 \\ -1 & for\ x < 0 \end{cases}$$

2.2.2 Sawtooth Wave Signal:

The sawtooth wave signal gets its name from the fact that the signal itself resembles the shape of the sawtooth.

2.2.3 Complex Exponential Signal:

The complex exponential signal is given by:

$$f(t) = Ae^{j(\omega t + \phi)}$$

This signal can be thought as being composed of real and imaginary harmonic components according to Euler's identity as follows:

$$f(t) = A[\cos(\omega t + \phi) + j\sin(\omega t + \phi)]$$

2.3 Composite Signals:

A composite signal is obtained by adding and/or multiplying two or more of the aforementioned signals. For example, the composite signal obtained by multiplying the exponential and a harmonic signal is an exponentially changing harmonic signal, an example of which is shown in the following exercise.

$$f(t) = e^{\omega t} \times \sin(\omega t)$$

Here, f(t) represents a sinusoidal wave whose amplitude decreases exponentially with time.

2.4 MATLAB Plotting Tools/Functions:

Some of the functions available in MATLAB to plot various signals/functions are listed below:

Functions	**Uses**
Plot(x, y)	Plots y as a function of x, x and y are matrices/arrays of the same length.
Subplot(i, j, k)	Divides the available graphing space into $i \times j$ partitions, and selects the k^{th} one.
Hold ON	Holds the curve/function displayed on the graph, that is the present graph is not erased when the next function is plotted, on the same graph.
Legend ON Legend(string1, string2, string3..)	Sets legend ON. Then assigns a legend for each of the functions plotted on the graph.
Axis([xmin xmax ymin ymax])	Sets the limits for the axes on the graph.
Grid ON	Displays Gridlines for the preceding graph.
xlabel(string) ylabel(string)	Labels the x and y- axes as the *strings* passed as parameters to these functions.

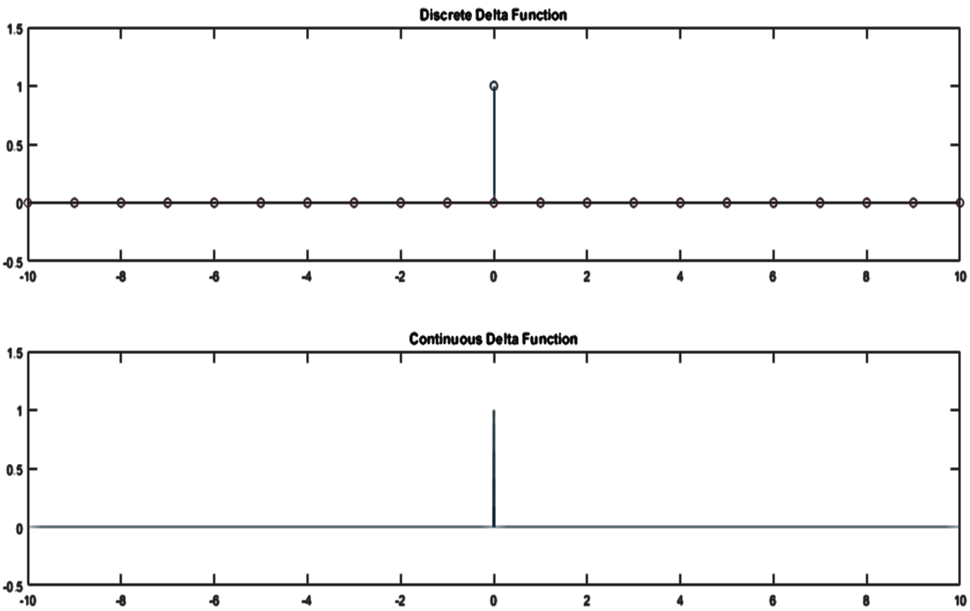
Stem(x, y)	Plots the points (x, y) and joins the points (x, 0) with (x, y) correspondingly, used to plot discrete-time signals.
Title(string1)	Sets the title of the preceding graph as *string1*.

3. Code and Outputs:

1.3.1

```
Fsd = 1;
Fsc = 100;
dtd = 1/Fsd;
dtc = 1/Fsc;
td = -10:dtd:10;
tc = -10:dtc:10;
nd = numel(td);
nc = numel(tc);
yd = zeros(nd);
yc = zeros(nc);
yd((nd + 1)/2) = 1;
yc((nc + 1)/2) = 1;
subplot(2,1,1)
stem(td, yd)
title('Discrete Delta Function')
axis([-10 10 -0.5 1.5])
subplot(2,1,2)
plot(tc, yc)
title('Continuous Delta Function')
axis([-10 10 -0.5 1.5])
```

OUTPUT:

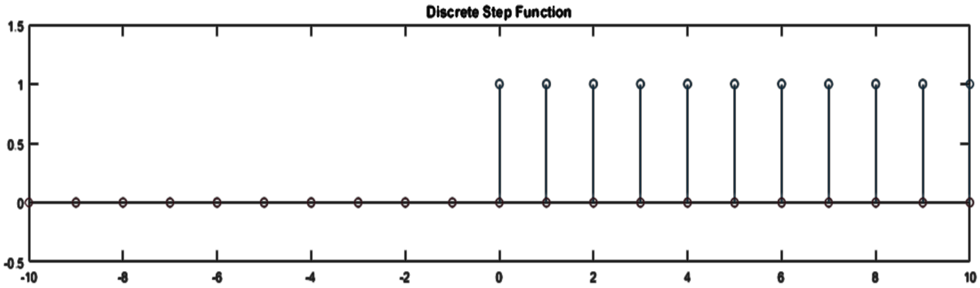


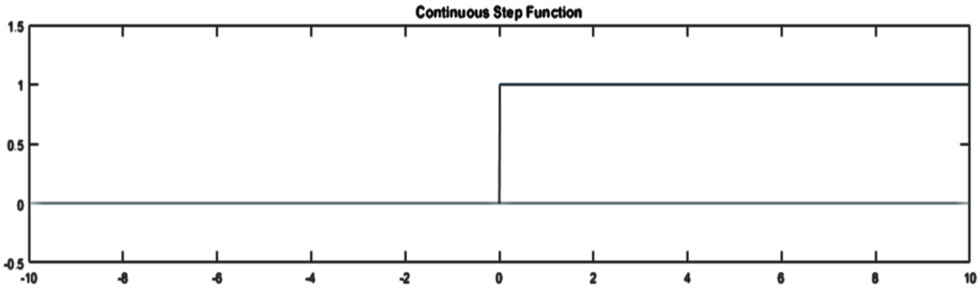
1.3.2

```
Fsd = 1;
Fsc = 100;
dtd = 1/Fsd;
```

```
dtc = 1/Fsc;
td = -10:dtd:10;
tc = -10:dtc:10;
nd = numel(td);
nc = numel(tc);
yd = zeros(nd);
yc = zeros(nc);
for i = 1:nd
    if(td(i) >= 0)
        yd(i) = 1;
    end
end
for j = 1:nc
    if(tc(j) > 0)
        yc(j) = 1;
    end
end
subplot(2,1,1)
stem(td, yd)
title('Discrete Step Function')
axis([-10 10 -0.5 1.5])
subplot(2,1,2)
plot(tc, yc)
title('Continuous Step Function')
axis([-10 10 -0.5 1.5])
```

OUTPUT:



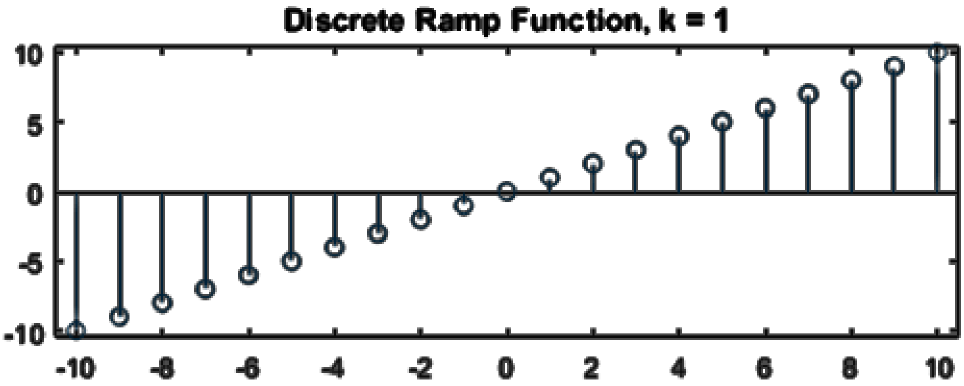


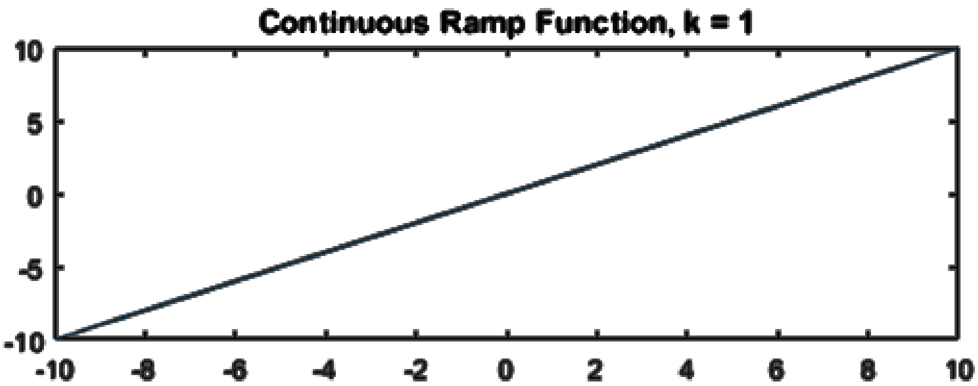
1.3.3

```
Fsd = 1;
Fsc = 100;
dtd = 1/Fsd;
dtc = 1/Fsc;
k = 1;
td = -10:dtd:10;
tc = -10:dtc:10;
nd = numel(td);
nc = numel(tc);
yd = k * td;
yc = k * tc;
subplot(2,1,1)
stem(td, yd)
title('Discrete Ramp Function, k = 1')
```

```
axis([-10.5 10.5 -10.5 10.5])
subplot(2,1,2)
plot(tc, yc)
title('Continuous Ramp Function, k = 1')
axis([-10 10 -10 10])
```

OUTPUT:



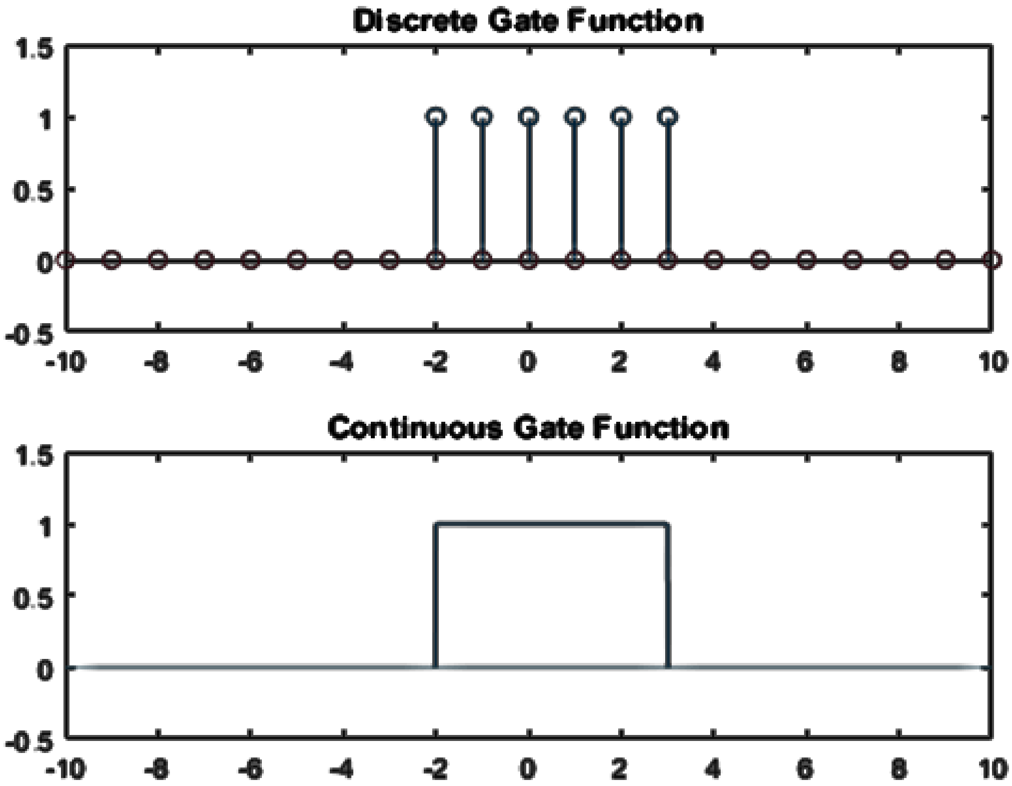


1.3.4

```
Fsd = 1;
Fsc = 100;
dtd = 1/Fsd;
dtc = 1/Fsc;
td = -10:dtd:10;
tc = -10:dtc:10;
nd = numel(td);
nc = numel(tc);
yd = zeros(nd);
yc = zeros(nc);
for i = 1:nd
    if(td(i) >= -2 && td(i) <= 3)
        yd(i) = 1;
    end
end
for j = 1:nc
    if(tc(j) >= -2 && tc(j) <= 3)
        yc(j) = 1;
    end
end
subplot(2,1,1)
stem(td, yd)
title('Discrete Gate Function')
axis([-10 10 -0.5 1.5])
subplot(2,1,2)
plot(tc, yc)
title('Continuous Gate Function')
```

```
axis([-10 10 -0.5 1.5])
```

OUTPUT:



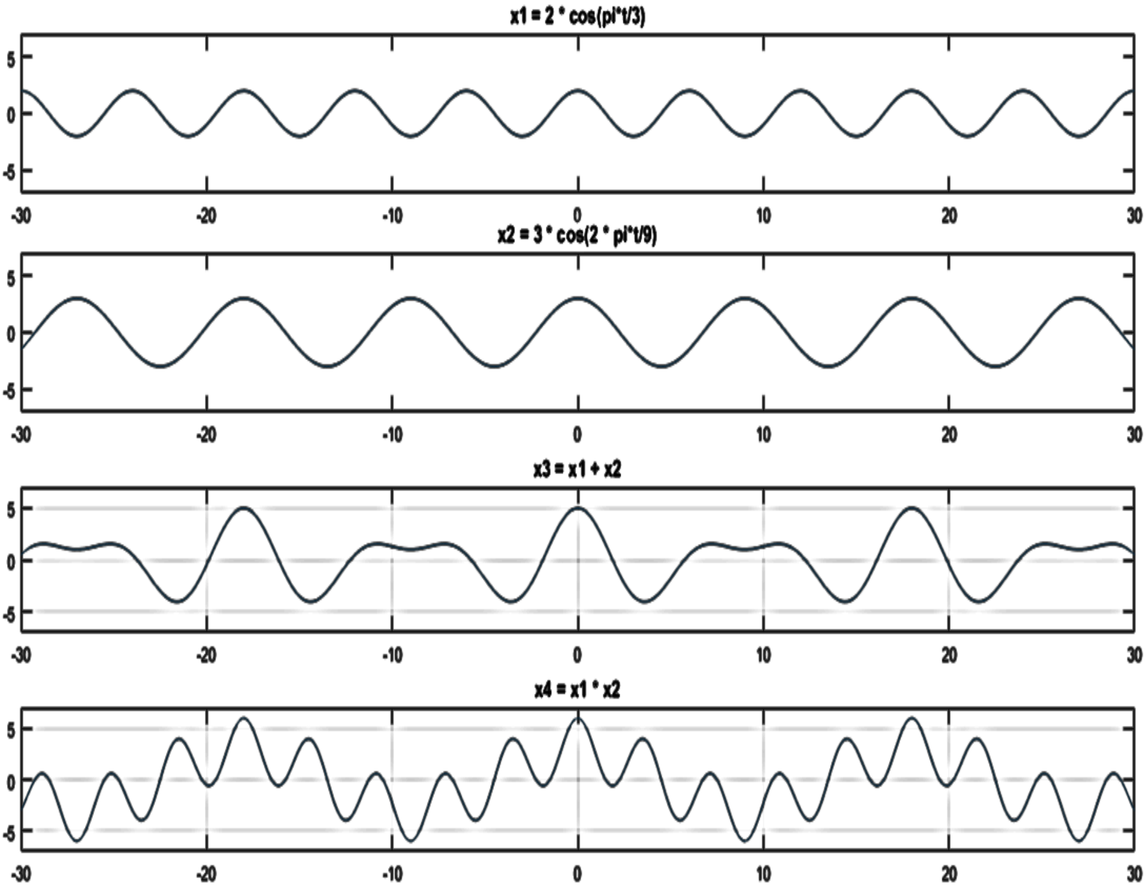
1.3.5

```
Fs = 100;
dt = 1/Fs;
t = -30:dt:30;
n = numel(t);
x1 = 2 * cos(pi * t/3); %period = 6
x2 = 3 * cos(2 * pi * t / 9); %period = 9
x3 = x1 + x2;   %period = -(-14.41) + 4.045
x4 = x1 .* x2; %period = 18
subplot(4,1,1)
plot(t, x1)
title('x1 = 2 * cos(pi*t/3)')
axis([-30 30 -7 7])

subplot(4,1,2)
plot(t, x2)
title('x2 = 3 * cos(2 * pi*t/9)')
axis([-30 30 -7 7])

subplot(4,1,3)
plot(t, x3)
title('x3 = x1 + x2')
axis([-30 30 -7 7])
grid on;
subplot(4,1,4)
plot(t, x4)
title('x4 = x1 * x2')
axis([-30 30 -7 7])
grid on;
```

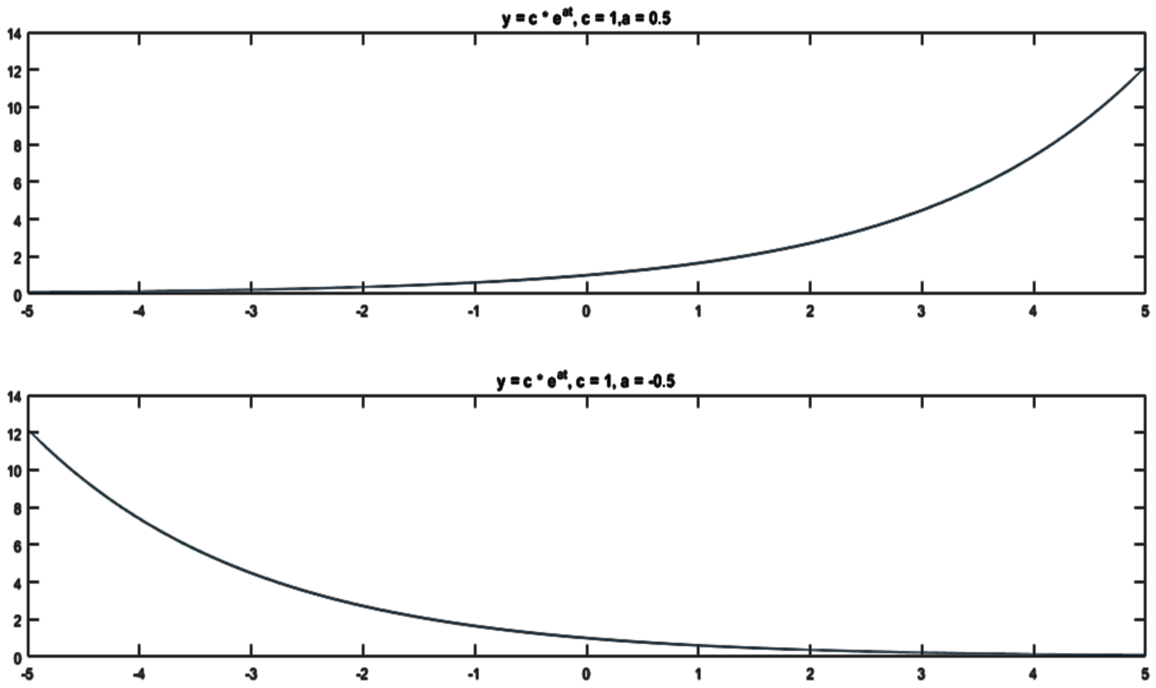
OUTPUT:



1.3.6

```
Fs = 100;
dt = 1/Fs;
t = -5:dt:5;
c = 1;
a = [0.5 -0.5];
x1 = c * exp(a(1) * t);
x2 = c * exp(a(2) * t);
subplot(2, 1, 1)
plot(t, real(x1));
title('y=c*e^{at},c=1,a=0.5')
subplot(2,1,2)
plot(t, real(x2));
title('y=c*e^{at},c=1,a=-0.5')
```

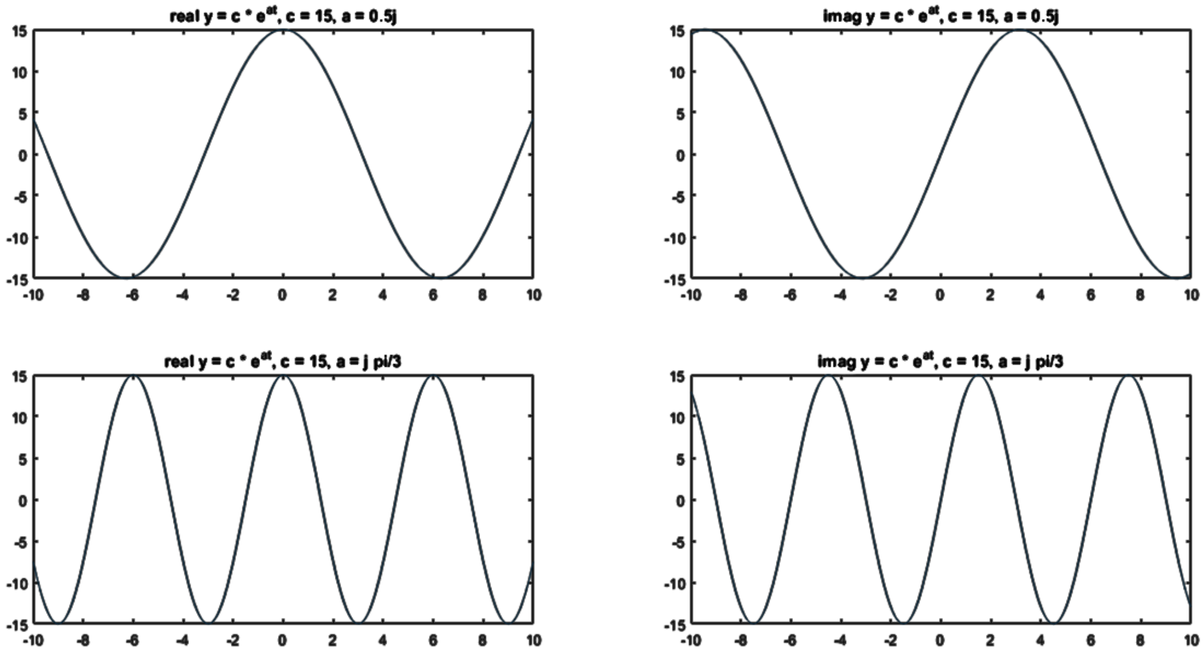
OUTPUT:



1.3.7

```
Fs = 100;
dt = 1/Fs;
t = -10:dt:10;
c = 15;
a = [complex(0, 0.5) complex(0, pi/3)];
x1 = c * exp(a(1) * t);
x2 = c * exp(a(2) * t);
subplot(2, 2, 1)
plot(t, real(x1));
title('realy=c*e^{at},c=15,a=0.5j')
subplot(2, 2, 2)
plot(t, imag(x1));
title('imagy=c*e^{at},c=15,a=0.5j')
subplot(2,2,3)
plot(t, real(x2));
title('realy=c*e^{at},c=15,a=j pi/3')
subplot(2,2,4);
plot(t, imag(x2));
title('imagy=c*e^{at},c=15,a=j pi/3')
```

OUTPUT:



4. Discussion and Conclusion

In this way, we used MATLAB to plot various types of basic signals such as Dirac-delta function, unit step function, ramp function, harmonic functions, complex exponential signal as well as the functions formed by the addition and multiplication of two out-of-phase harmonic (cosine) signals. The graphical nature of these signals was studied. The period of the composite signals was found through MATLAB itself and a phase difference of $\frac{\pi}{2}$ was seen between the real and imaginary components of the complex exponential signals. This is due to the fact that a complex exponential signal can be represented by a real cosine part and an imaginary sine part from the Euler's identity: $e^{j\theta} = cos\theta + j\ sin\theta$ and these harmonic components have a $\frac{\pi}{2}$ phase difference. It was also seen that while plotting these signals, the discrete and continuous form of the various signals can be obtained simply by varying the sampling frequency(`Fs`) to an appropriate value and using the `stem()` and `plot()` functions for discrete and continuous signals respectively.