Task1: Signal Processing I

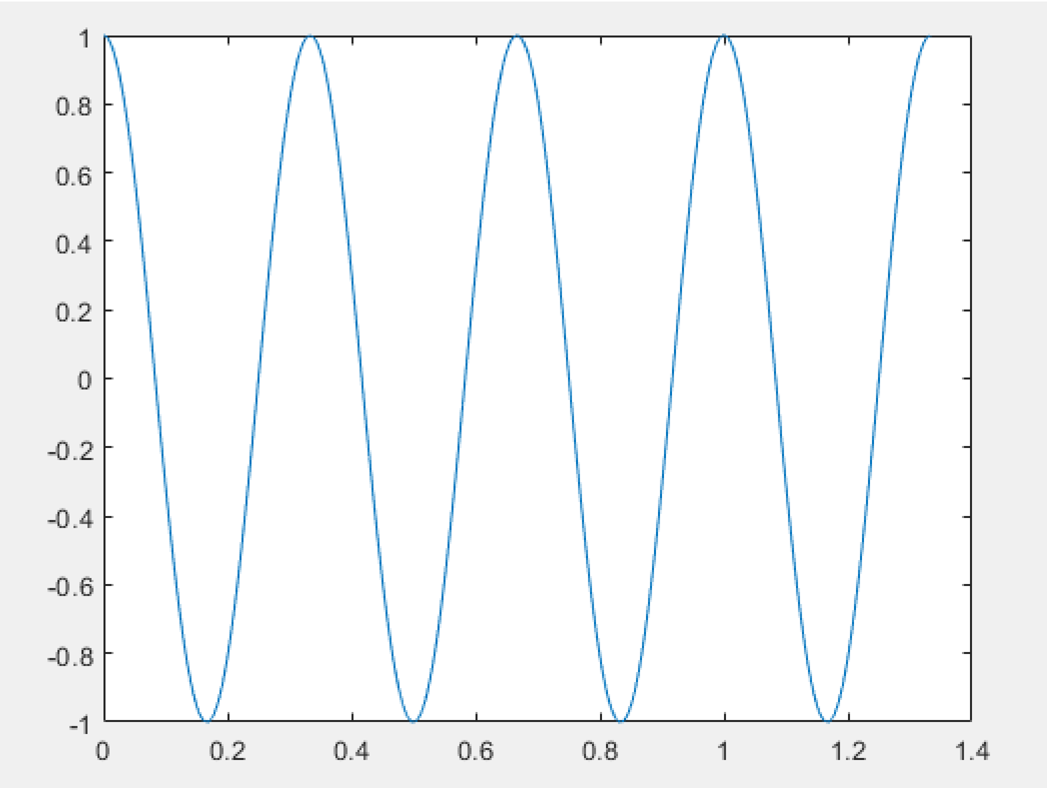
Name: Laurent Gérin **Degree:** MASc **ID:** 20505439

Problem 1

(a) What is the difference between a continuous (or analogue) and discrete (or digital) signals?

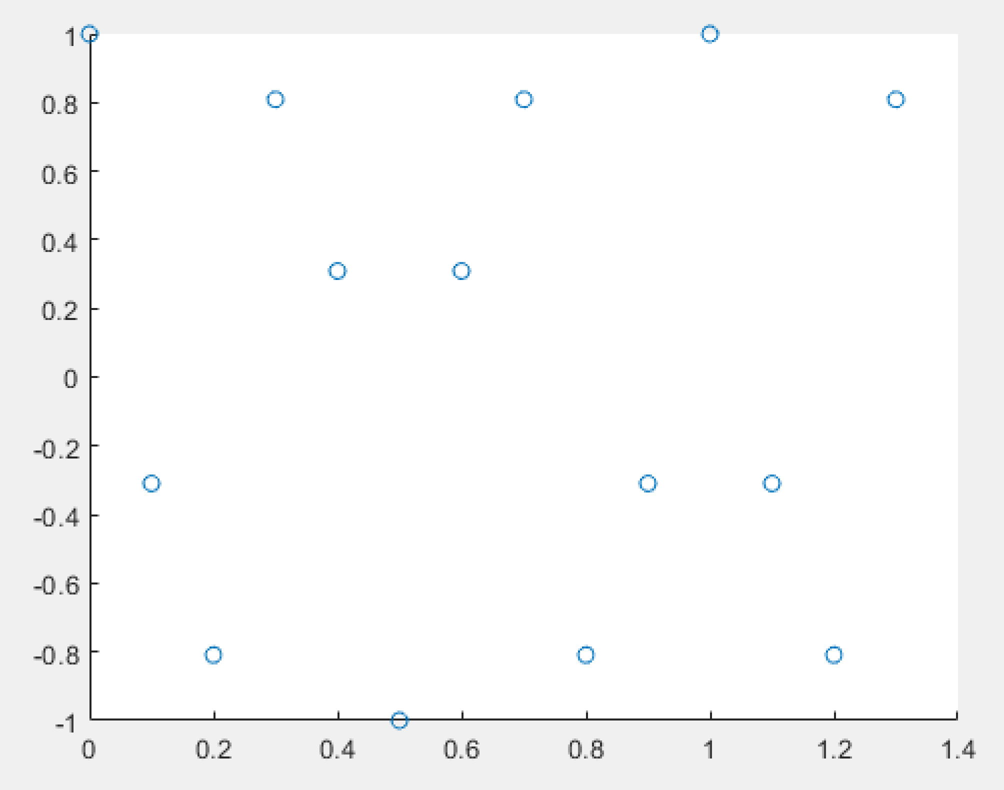
The difference is rather simple - a continuous / analogue signal is, as its name implies, continuous, whereas a discrete or digital signal is composed of discrete points, or "snapshots" of the analogue signal.

(b) Plot a 3 Hz cosine wave with high sampling rate (nearly analog signal). Please connect sampled points and plot only four cycles of the wave.




```
samplingRate = 10000;    %Hz, rate of sampling
waveFreq = 3;            %Hz, frequency of cosine wave
numCycles = 4;           %Number of cycles to plot

t = 0:1/samplingRate:numCycles/waveFreq;
y = cos(2*pi*waveFreq*t);
plot(t,y)
```

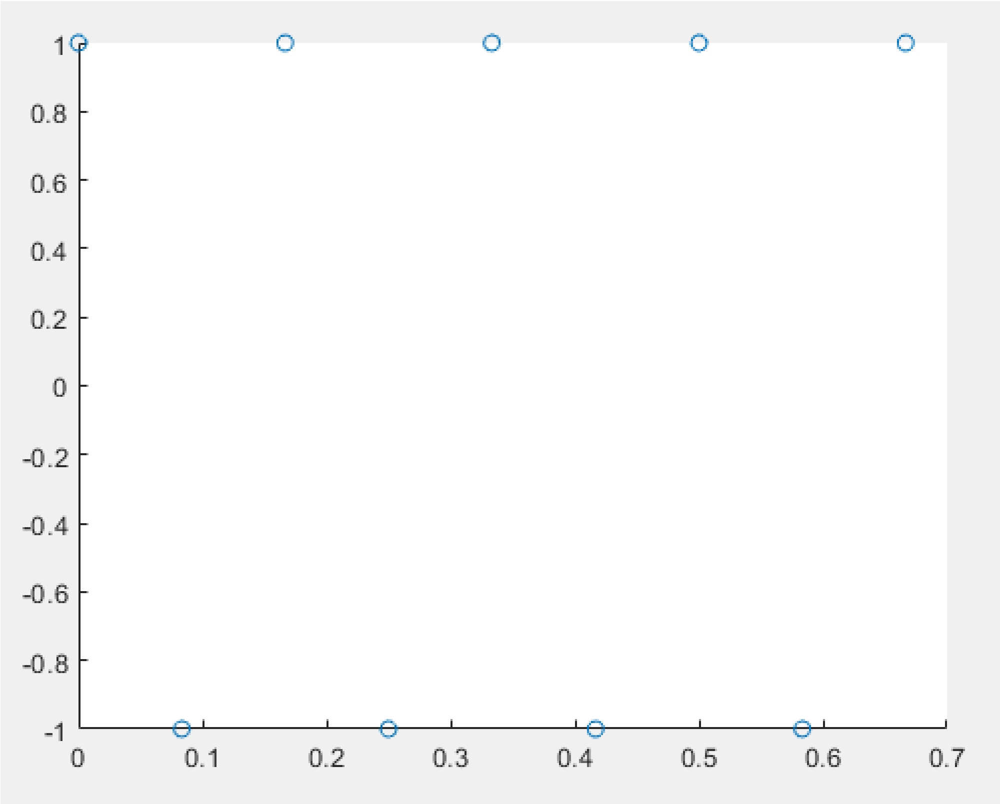
(c) Plot the 3 Hz cosine wave after sampling with 10 Hz. Please do not connect sampled points and plot the sampled data for four cycles of the wave.




```
figure
samplingRate = 10;
t = 0:1/samplingRate:numCycles/waveFreq;
y = cos(2*pi*waveFreq*t);
scatter(t,y)
```

(d) Plot the 6 Hz sine wave after sampling with 12 Hz. Please do not connect sampled points and plot only four cycles of the wave. Do you think that you can measure this wave if you add a phase angle (φ) on this sine wave? for example, the wave is sin(2πft + φ).

As we can see from the picture, because the sampling rate is harmonious with the original wave, we see the peaks and bottoms of the wave. If there were a phase shift, it would be easy to see where the new peaks are, and thus this wave could be measured.



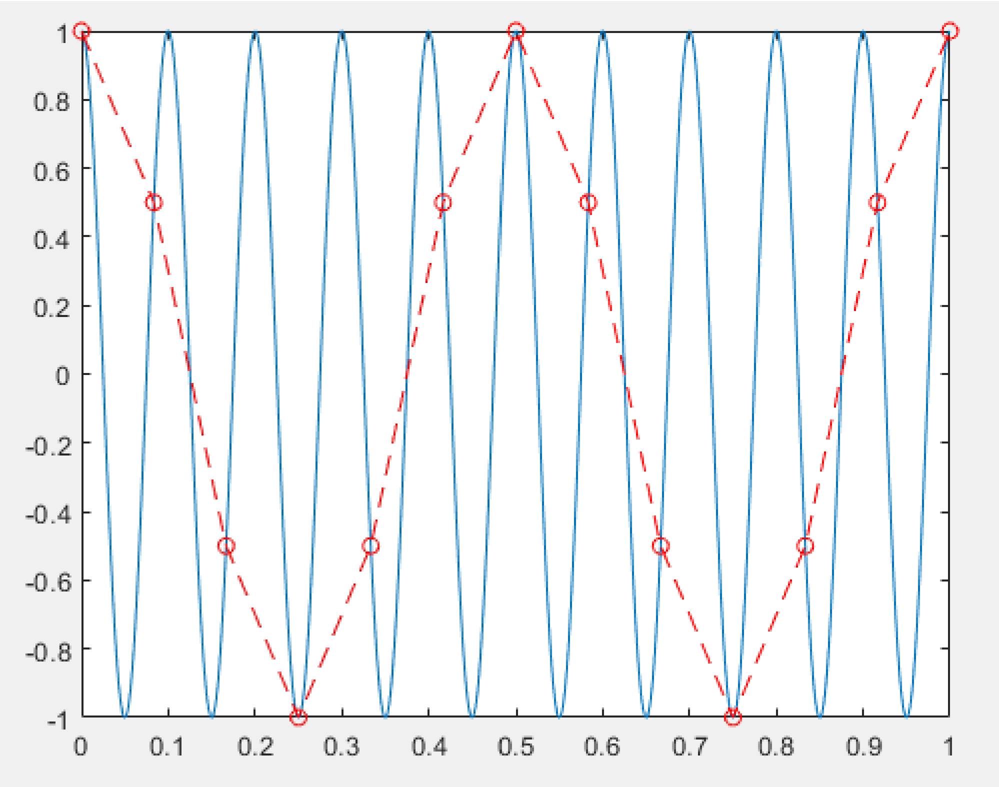

```
figure
samplingRate = 12;
waveFreq = 6;           %Hz, frequency of cosine wave
t = 0:1/samplingRate:numCycles/waveFreq;
y = cos(2*pi*waveFreq*t);
scatter(t,y)
```

Problem 2

(a) A 10 Hz sine wave is sampled at 12 Hz. Compute the alias frequency that can be represented in the resulting sampled signal. Plot the wave and sampled points.

The alias frequency is found to be 2 Hz:

$$f_s = |10 - 12| = 2$$

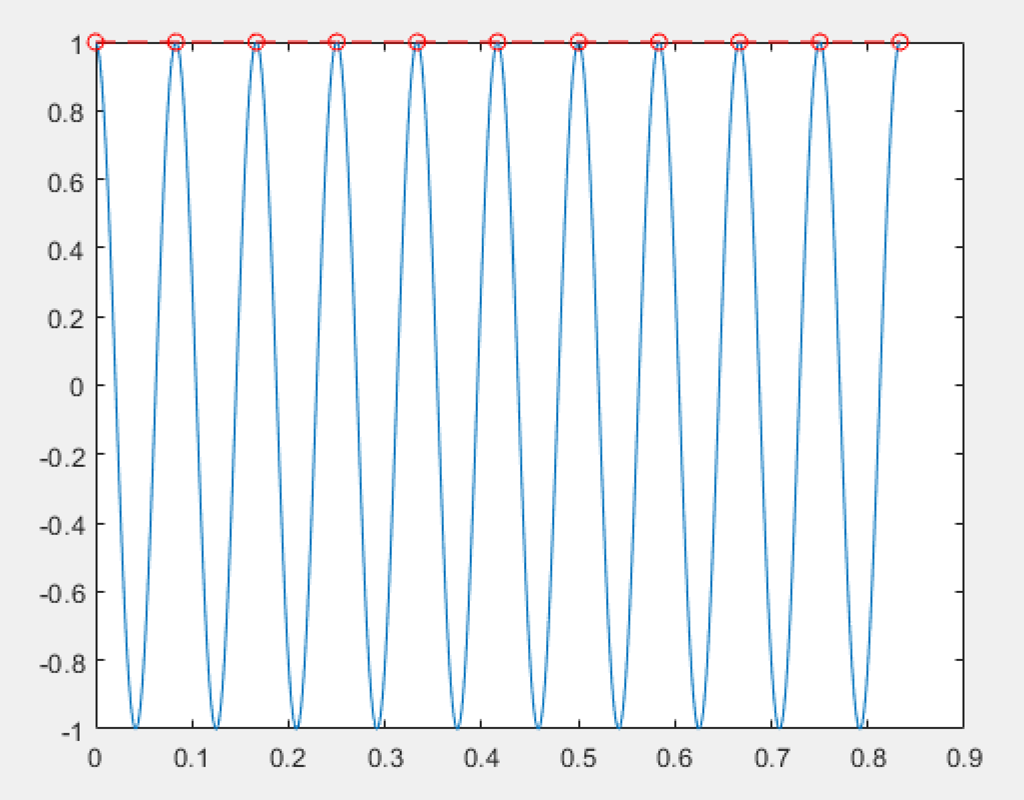



```
samplingRate = 10000;     %Hz, rate of sampling
waveFreq = 10;            %Hz, frequency of cosine wave
numCycles = 10;           %Number of cycles to plot

t = 0:1/samplingRate:numCycles/waveFreq;
y = cos(2*pi*waveFreq*t);
plot(t,y)
hold on

samplingRate = 12;
t = 0:1/samplingRate:numCycles/waveFreq;
y = cos(2*pi*waveFreq*t);
plot(t,y,'r--o')
```

(b) A 12 Hz cosine wave is sampled at 12 Hz. Compute the alias frequency that can be represented in the resulting sampled signal. Plot the wave and sampled points.

In this case we get an effective frequency of 0 Hz, since we are always hitting the same point in the cycle.

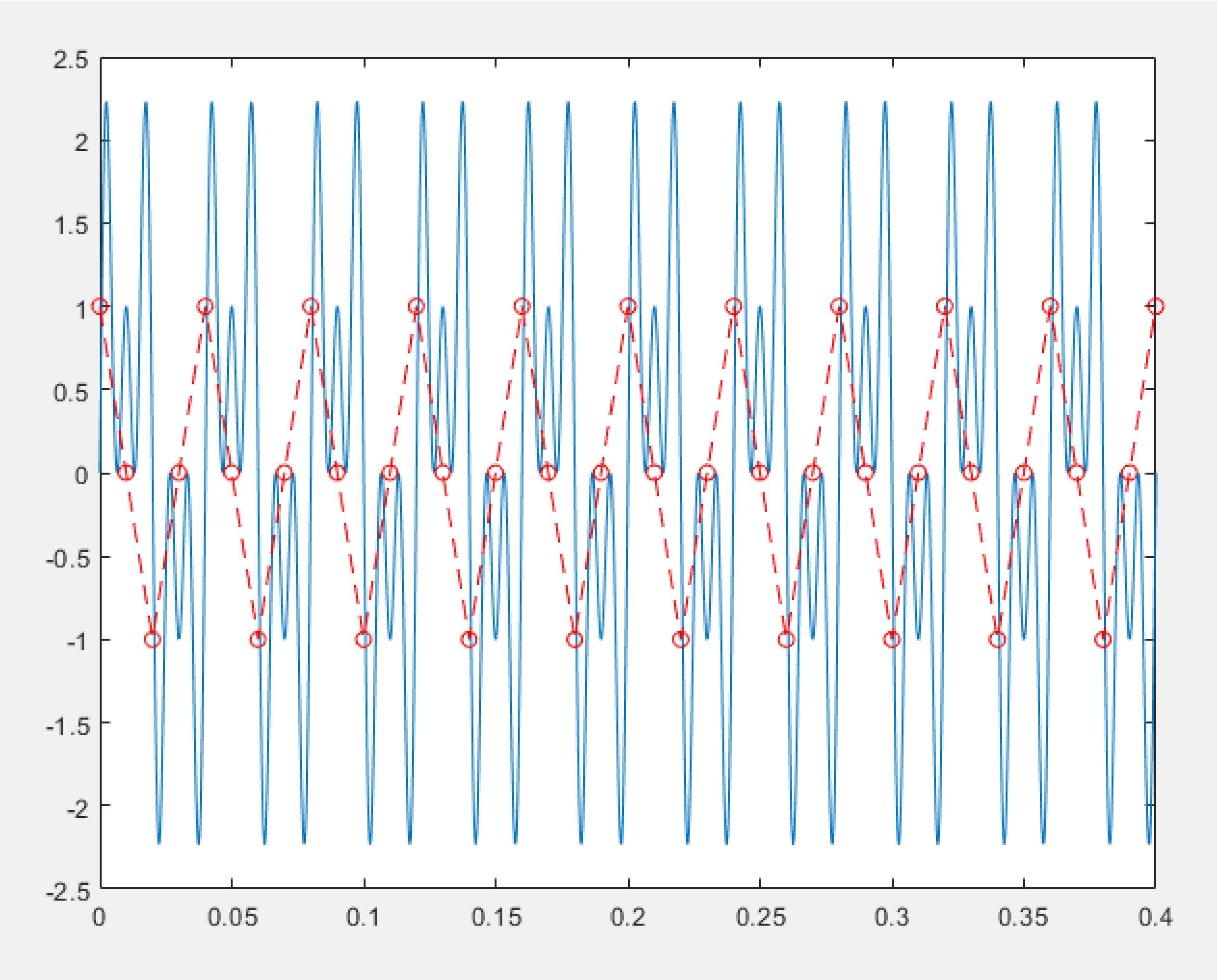
$$f_s = |12 - 12| = 0$$




```
samplingRate = 10000;     %Hz, rate of sampling
waveFreq = 12;            %Hz, frequency of cosine wave
numCycles = 10;           %Number of cycles to plot

t = 0:1/samplingRate:numCycles/waveFreq;
y = cos(2*pi*waveFreq*t);
plot(t,y)
hold on

samplingRate = 12;
t = 0:1/samplingRate:numCycles/waveFreq;
y = cos(2*pi*waveFreq*t);
plot(t,y,'r--o')
```

(c) Assume that the measured signal has a complex periodic signal of the form: $y(t) = A1sin2\pi(25)t + A2sin2\pi(75)t + A3sin2\pi(125)t$. If the signal is sampled at 100 Hz, determine the frequency content of the resulting discrete response signal.

The resulting signal will appear to only have frequencies at 25 Hz, since the sampling occurs higher than the Nyquist frequency for A1, and under for A2 and A3. Since |100-75| and |100-125| are both equal to 25, we get only 25 in the end.



Problem 3

(a) Please explain a quantization error. When do they occur? How to avoid?

Quantization error is an error stemming from the conversion of a continuous signal to a discrete signal. There is only a finite range of values which can be represented in a discrete signal, and so information is lost about what happens "between" the discrete points taken from the analog signal. This occurs in all digital sampling cases - one would need infinite memory to fully suppress this error. In practice, the error can be effectively minimized by choosing a high enough resolution for the discrete signal to ensure that any data of interest is not missed between captured points.

(b) Please explain a clipping error. When do they occur? How to avoid?

Clipping occurs when the amplitude of a signal is higher than what the sensor can handle. For instance, in a photo, clipping occurs when too much light hits the sensor (overexposure) and everything simply appears white, with loss of information. This can be avoided by choosing a sensor appropriate for the data being captured, with a working range large enough for the analog signal.

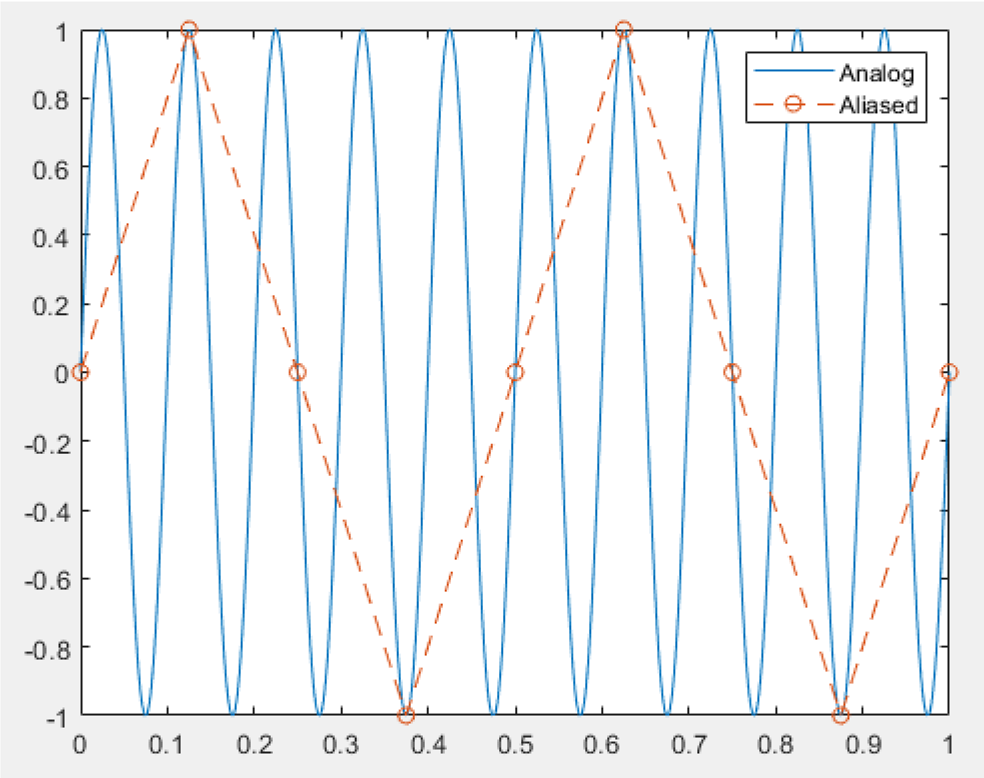
(c) Please explain an oversampling issue. When do they occur? What are the consequence of the oversampling?

Oversampling is less of an issue since it should not affect the quality of data. However, when sampling rate is higher than the Nyquist frequency, we have oversampling since in theory there is no more information to be obtained from a higher sampling rate. This means that more storage space is used with little return.

(d) Assume that a building vibrates with a 10 Hz sine wave, and you measured this vibration using your accelerometer and DAQ system. Please write a code to create three different sampled signals that are damaged by aliasing, quantization error and clipping error, respectively. Also, generate a signal having an oversampling issue. You should understand the topics of Aliasing, Quantization, Clipping, and Oversampling in `data_cquisition_V3.mlx` to solve this problem. You need to explain why your sampled signals contain each of these errors/issue. You can assume any sampling rate, output range, or ADC resolution to generate these signals. Your code should plot these three signals.

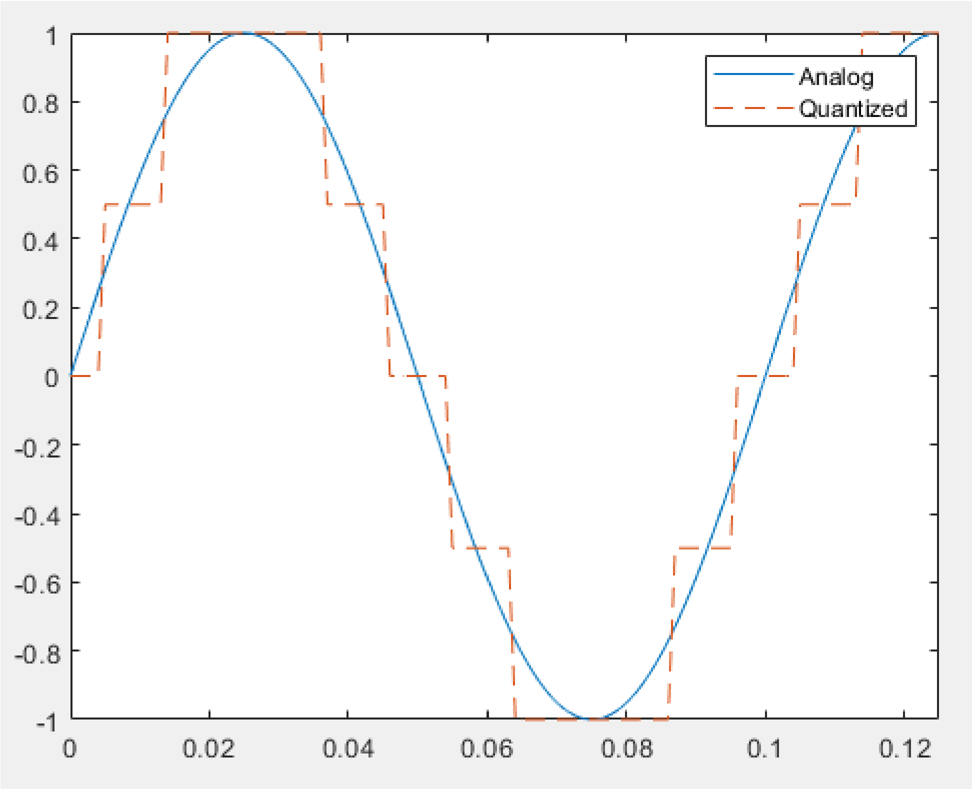
Aliasing:

The sampling rate is below the Nyquist frequency (8 Hz vs 10 Hz) and so we see that the sampled signal looks to have a very different frequency (2 Hz).



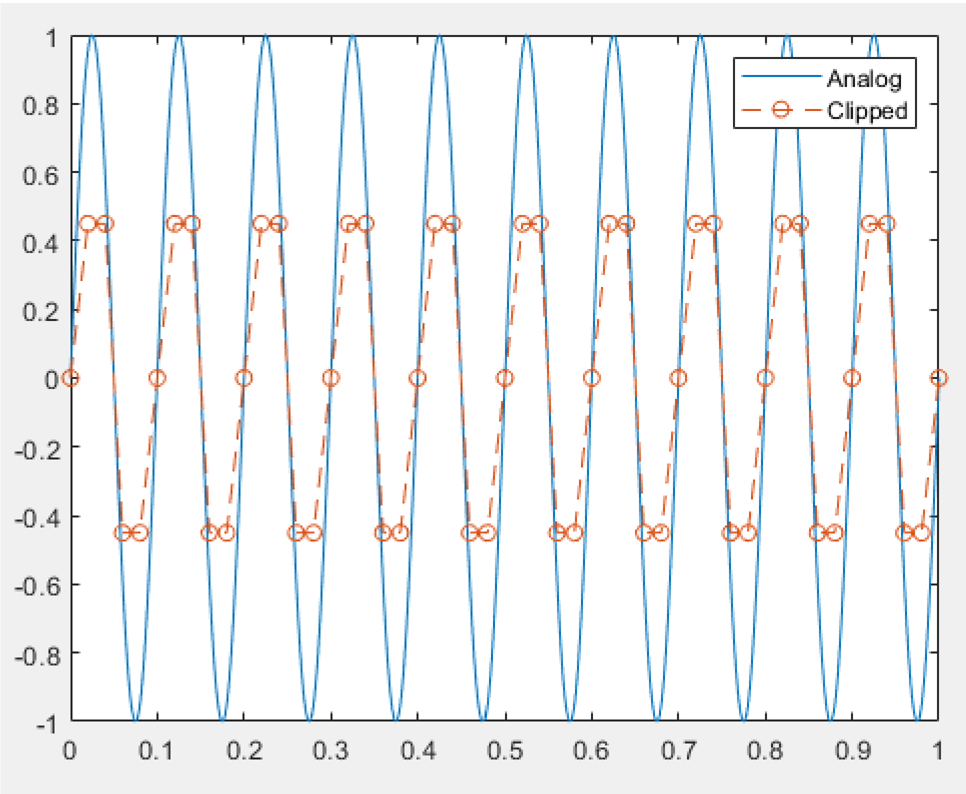
Quantization:

We simulated quantization by rounding each sample point to the nearest 0.5. We can clearly see the effect of low resolution sampling - we see these steps which hide the smooth nature of the signal.



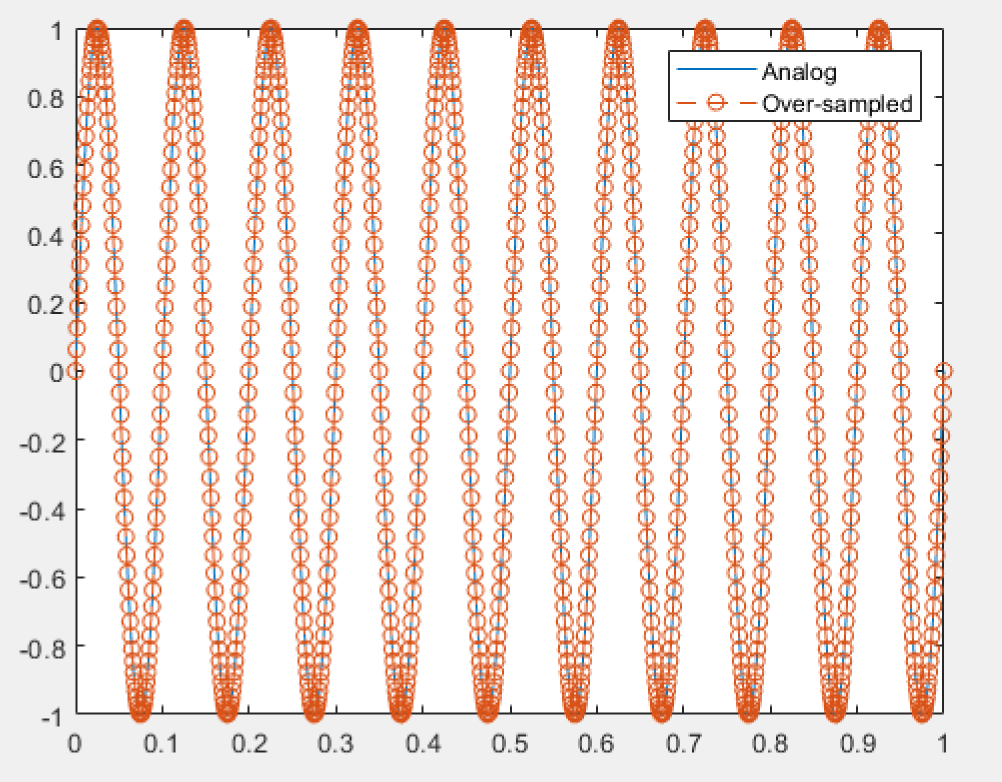
Clipping

We clip the data at +/- 0.45, which cuts off the peaks and throughs of the sine wave, and removing any information about what happens past +/- 0.45.



Over-sampling

We use a sampling frequency of 1000 Hz, which is 50 times larger than the Nyquist frequency (20 Hz). Thus the curve we obtain is very nice and smooth, but there is no more information about the nature of the signal than when we sampled at 25 Hz for instance.



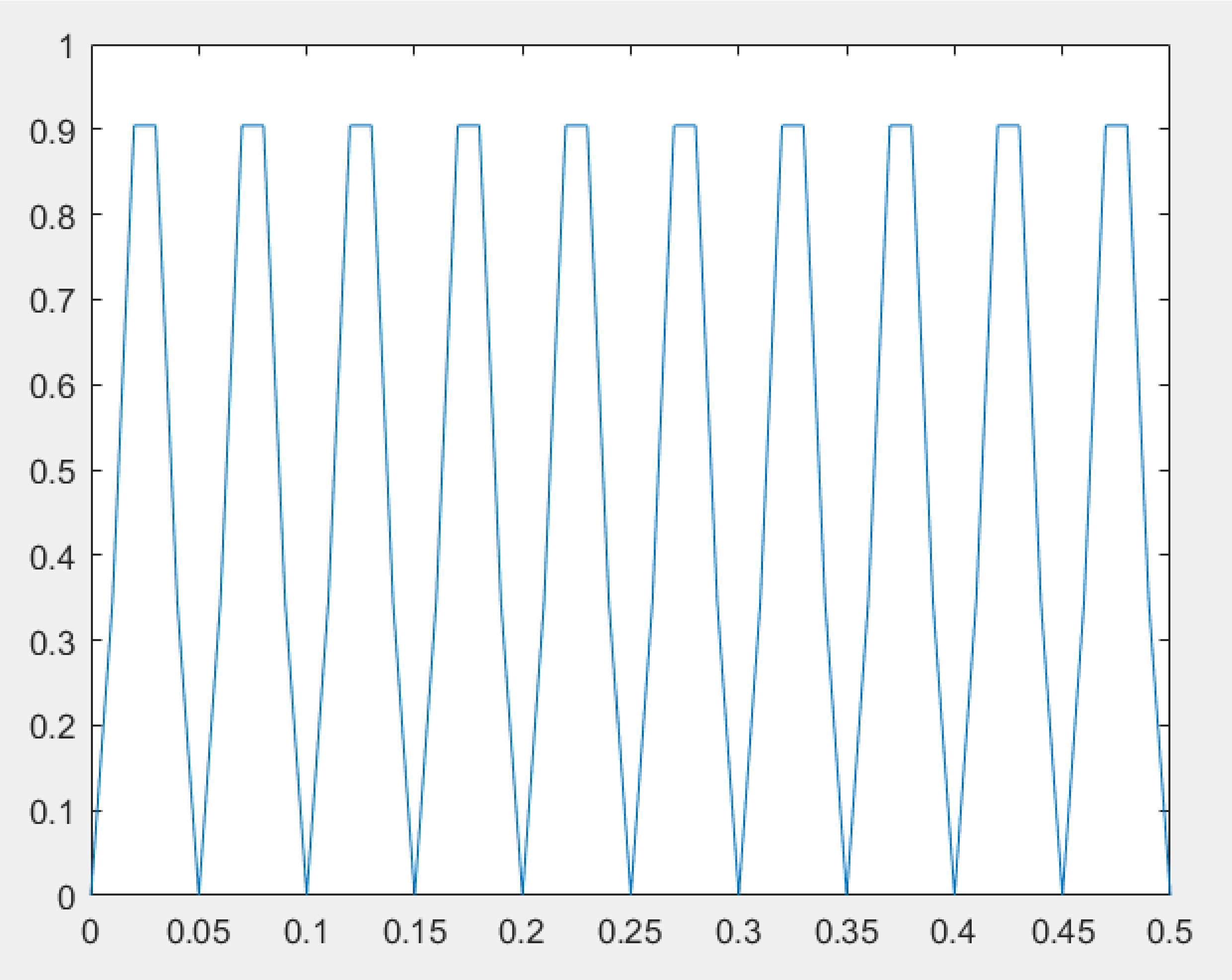

```
samplingRate = 10000;     %Hz, rate of sampling
waveFreq = 12;            %Hz, frequency of cosine wave
numCycles = 10;           %Number of cycles to plot

t = 0:1/samplingRate:numCycles/waveFreq;
y = cos(2*pi*waveFreq*t);
plot(t,y)
hold on

samplingRate = 12;
t = 0:1/samplingRate:numCycles/waveFreq;
y = cos(2*pi*waveFreq*t);
plot(t,y,'r--o')

figure
%Problem 2(c)
samplingRate = 10000;     %Hz, rate of sampling
waveFreq = 25;            %Hz, frequency of cosine wave
numCycles = 10;           %Number of cycles to plot

t = 0:1/samplingRate:numCycles/waveFreq;
y = sin(2*pi*waveFreq*t)+sin(2*pi*75*t)+sin(2*pi*125*t);
plot(t,y)
hold on

samplingRate = 100;
t = 0:1/samplingRate:numCycles/waveFreq;
y = cos(2*pi*waveFreq*t);
plot(t,y,'r--o')
```

```
%Problem 3(d)
clc, clear, close all

samplingRate = 10000;    %Hz, rate of sampling
waveFreq = 10;           %Hz, frequency of cosine wave
numCycles = 10;          %Number of cycles to plot

t0 = 0:1/samplingRate:numCycles/waveFreq;
y0 = sin(2*pi*waveFreq*t0);

%Aliasing
plot(t0,y0,'DisplayName','Analog')
hold on
samplingRate = 8;    %Hz, rate of sampling
t = 0:1/samplingRate:numCycles/waveFreq;
y = sin(2*pi*waveFreq*t);
plot(t,y,'--o','DisplayName','Aliased')
legend

%Quantization
figure
plot(t0,y0,'DisplayName','Analog')
hold on
samplingRate = 1000;    %Hz, rate of sampling
t = 0:1/samplingRate:numCycles/waveFreq;
y = sin(2*pi*waveFreq*t);
y = round(y/5, 1)*5;
plot(t,y,'--','DisplayName','Quantized')
axis([0 0.125 -inf inf])
legend


%Clipping
figure
plot(t0,y0,'DisplayName','Analog')
hold on
samplingRate = 50;    %Hz, rate of sampling
t = 0:1/samplingRate:numCycles/waveFreq;
y = sin(2*pi*waveFreq*t);
y(y > 0.45) = 0.45;
y(y < -0.45) = -0.45;
plot(t,y,'--o','DisplayName','Clipped')
legend

%Over-Sampling
figure
plot(t0,y0,'DisplayName','Analog')
hold on
samplingRate = 1000;    %Hz, rate of sampling
t = 0:1/samplingRate:numCycles/waveFreq;
y = sin(2*pi*waveFreq*t);
plot(t,y,'--o','DisplayName','Over-sampled')
legend
```

Problem 4

(a) Plot a wave1 sampled with a 100 Hz sampling rate. The wave1 is $y = sin(2\pi f_0 t) \cdot sin(2\pi f_0 t)$ where f0 = 10. Please connect sampled points and plot only ten cycles of the wave.

Note that due to the sampling rate, we miss the peaks of the wave.




```
samplingRate = 100;    %Hz, rate of sampling
waveFreq = 10;           %Hz, frequency of cosine wave
numCycles = 5;          %Number of cycles to plot.  Halved since we have square of
sines

t = 0:1/samplingRate:numCycles/waveFreq;
wave1 = (sin(2*pi*waveFreq*t)).^2;
plot(t,wave1)
```

(b) Derive a Fourier series (general form) of analytic wave1. You should find analytic equations for coefficients of a_0, a_n, and b_n.

This is easily done using the following identity:

$$\sin^2(\theta) = \frac{1-\cos(2\theta)}{2}$$

which reduces our function to a first order trigonometric function. This is essentially already a "Fourier Series", where

$a_0 = 1$,

$a_n = \begin{cases} -1/2, & n = 2 \\ 0, & \text{otherwise} \end{cases}$ and

$b_n = 0$

The Fourier series can thus be expressed as such:

$$\frac{1}{2} - \frac{1}{2}\cos(4\pi f_0 t)$$

(c) Derive a Fourier series (complex form) of analytic wave1. You should find an analytic equations for a coefficient of c_n.

Using the identity:

$$\cos(t) = \frac{1}{2}(e^{-it} + e^{it})$$

we can rewrite our wave equation as such:

$$\frac{1}{2} - \frac{1}{4}e^{-4i\pi f_0 t} - \frac{1}{4}e^{4i\pi f_0 t}$$

and thus we find our coefficients by inspection:

$$c_n = \begin{cases} 1/2, & n = 0 \\ -1/4, & n = \pm 2 \\ 0, & otherwise \end{cases}$$

(d) Derive a Fourier series (general form) of analytic wave2: $y = sin(2\pi f_0 t) \cdot sin(2\pi f_0 t) + 5$ You should find analytic equations for coefficients of a_0, a_n, and b_n.

By inspection - this will be the same wave as in the previous questions, only translated upwards by a value of 5. By inspection, this translation movement can be completed by increasing only a_0 by 10. Thus:

$a_0 = 11$,

$a_n = \begin{cases} -1/2, & n = 4 \\ 0, & \text{otherwise} \end{cases}$ and

$b_n = 0$

The Fourier series can thus be expressed as such:

$$\frac{11}{2} - \frac{1}{2}\cos(4\pi f_0 t)$$

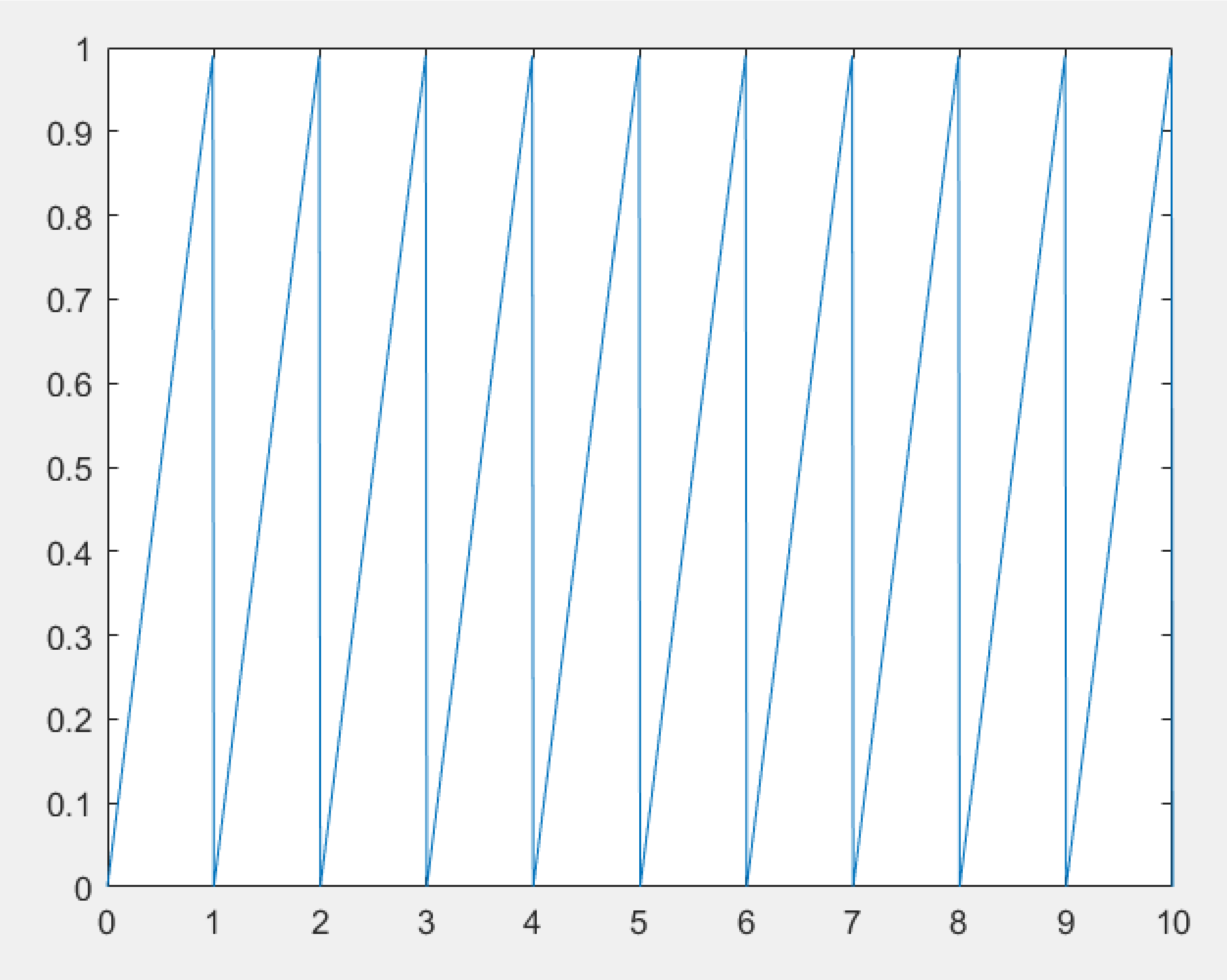
(e) Please compare the results of (b) and (d) and explain their difference.

The only difference between the two is in the value of a_0, which is 5 units larger in (d). This makes sense as the "zeroth" term in a Fourier series only accounts for overall translation of the function.

Problem 5

(a) Plot only ten cycles of a sawtooth wave:

$$x(t) = t - floor(t)$$




```
samplingRate = 100;    %Hz, rate of sampling
numCycles = 10;           %Number of cycles to plot

t = 0:1/samplingRate:numCycles;
y = t - floor(t);
plot(t,y)
```

(b) Derive a Fourier series (general form) for a sawtooth wave

First we find a_0:

$$\frac{a_0}{2} = \int_0^1 t dt = \frac{1}{2}$$

Then a_n, which is found to be zero (the sine terms are zero for all n, and the cosine terms are both equal to one and cancel each other):

$$a_n = 2\int_0^1 t\cos(2\pi nt)dt = 2\left[\frac{t}{2\pi n}\sin(2\pi nt) + \frac{1}{(2\pi n)^2}\cos(2\pi nt)\right]_0^1 = 0$$

Finally, b_n:

$$b_n = 2\int_0^1 t\sin(2\pi nt)dt = 2\left[\frac{1}{(2\pi n)^2}\sin(2\pi nt) + \frac{t}{(2\pi n)}\cos(2\pi nt)\right]_0^1 = 2\left[0 - \frac{1}{(2\pi n)}\right] = -\frac{1}{\pi n}$$

Thus we can express our wave as the function:

$$f(t) = \frac{1}{2} + \sum_{n=1}^{\infty}\left(\frac{-1}{\pi n}\right)\sin(2\pi nt)$$

(c) Derive a Fourier series (complex form) for the same sawtooth wave. You should find an analytic equation for a coefficient of c_n.

We find the value for c_n using the integral definition of the complex fourier series:

$$c_n = \int_0^1 te^{-2\pi int}dt = \left[\frac{(-2\pi int - 1)e^{-2\pi int}}{(2\pi in)^2}\right]_0^1 = \frac{-1}{2\pi in}$$

Note that for n = 0, c_n is undefined, but we know that:

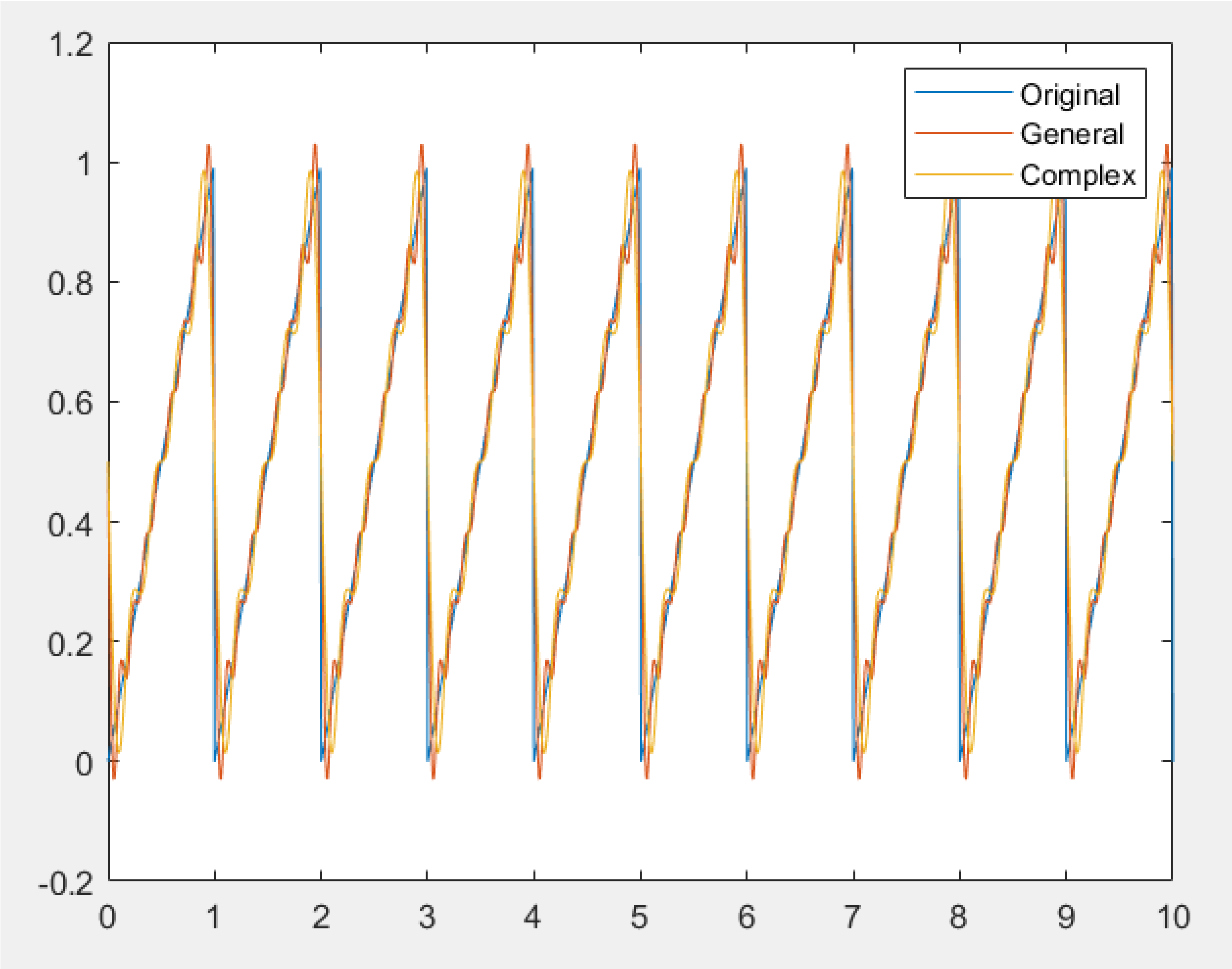
$$c_0 = a_0 = \frac{1}{2}$$

Thus we can write our complex form series as such:

$$f(t) = \sum_{n=-\infty}^{\infty} c_n e^{2\pi int},\ c_n = \begin{cases} 1/2, & n = 0 \\ \frac{-1}{2\pi in}, & otherwise \end{cases}$$

(d) Write a code to create and plot approximated sawtooth waves (# of coefficients (n) = 8) using the derived Fourier series in the general and complex forms. You should compare the waves from the general and complex forms.

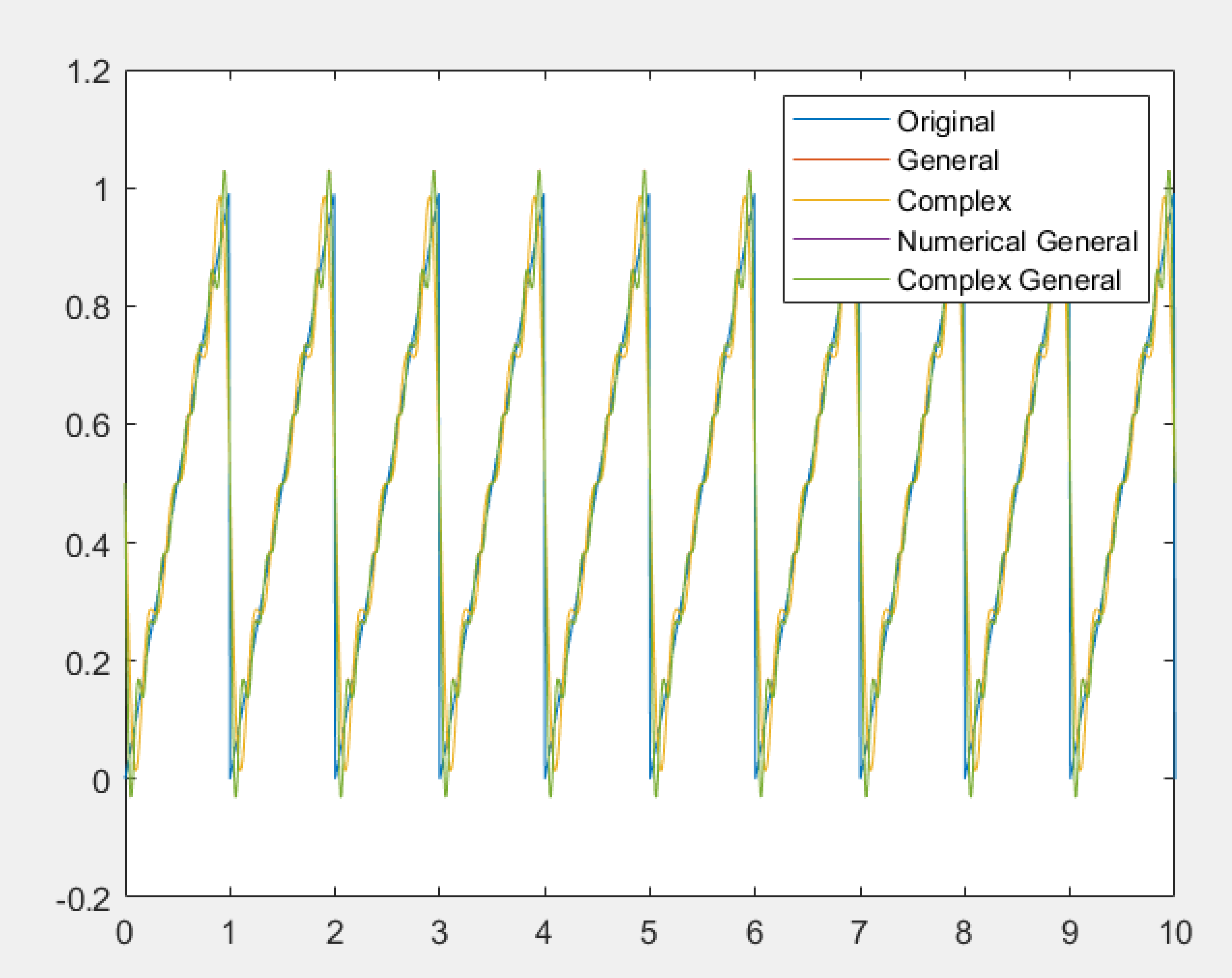
Overall the two forms are almost exactly identical. A very close look shows slight differences between the two on the plot. This is because the complex form of the series actually requires twice as many terms to achieve the same precision: each negative/positive pair of *n* is equal to one term of the general Fourier form.

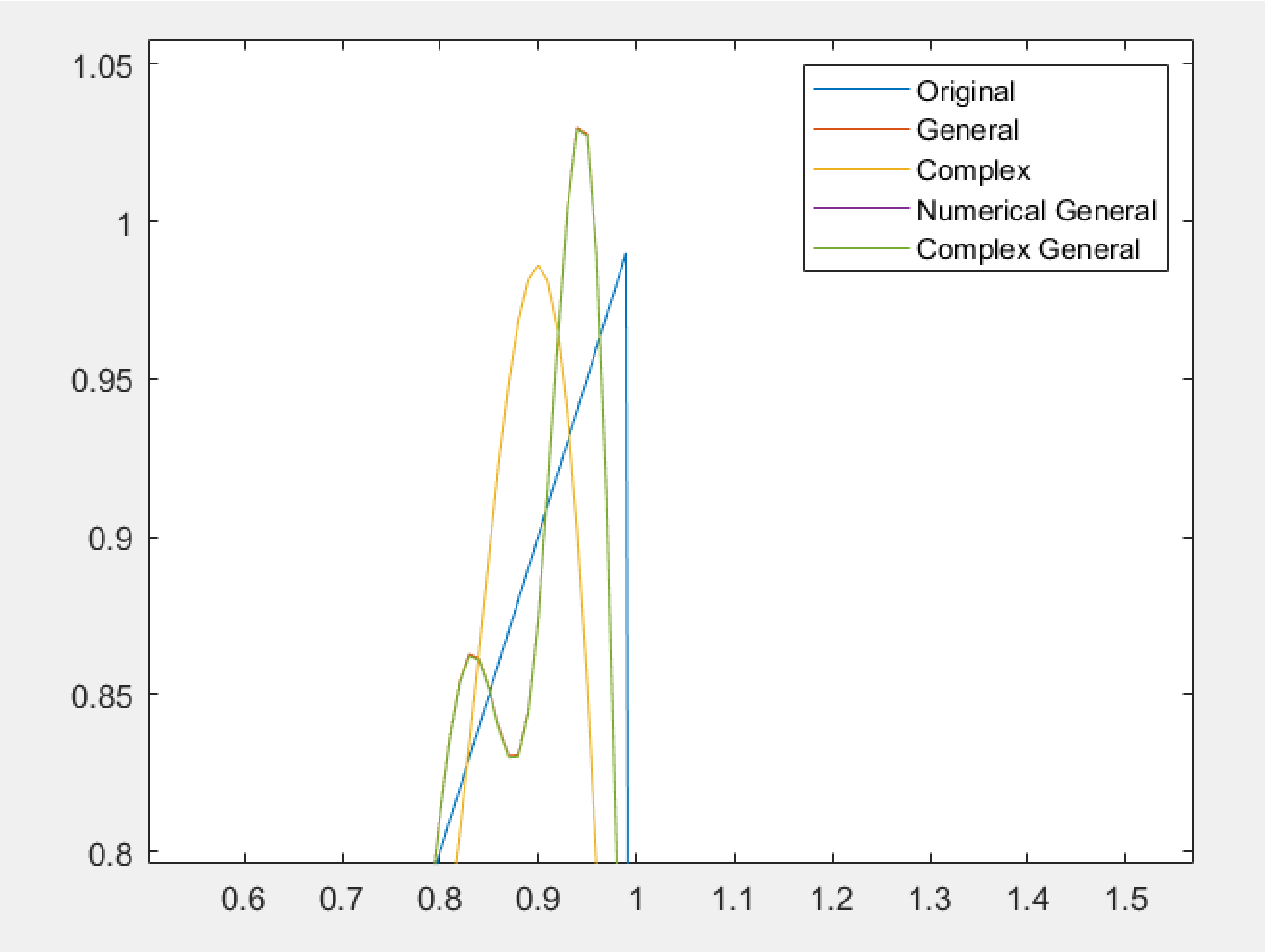



```
n = 8;  %number of terms

y = 0.5 * ones(1,length(t));

for i = 1:n
    y = y + (-1/(pi * i))*sin(2*pi*i*t);
end
clear i
hold on
plot(t,y,'DisplayName','General')

y = zeros * ones(1,length(t));
for j = round(-n/2):round(n/2)
    if j == 0
        y = y + 0.5;
    else
        y = y - 1/ (2 * pi * i * j) * exp(2 * pi * i * j * t);
    end
end
plot(t,y,'DisplayName','Complex')
legend
```

(e) Write a code to find numerical Fourier coefficients in the general and complex forms and compare them with the analytic Fourier coefficients found in (b) and (c).

We get essentially the exact same result even if we numerically integrate using the `trapz` function of MATLAB, which is expected.






```
t_range = 0:1/1000:1;
y_range = t_range - floor(t_range);
%general form
a_0 = 2 * trapz(t_range,y_range);
for i = 1:n
    a_n(i) = 2 * trapz(t_range,t_range.*cos(2 * pi * i * t_range));
    b_n(i) = 2 * trapz(t_range,t_range.*sin(2 * pi * i * t_range));
end
y_general = a_0/2 * ones(1,length(t));
for i = 1:n
    y_general = y_general + a_n(i) * cos(2*pi*i*t) + b_n(i) * sin(2*pi*i*t);
end
plot(t,y_general,'DisplayName','Numerical General')

%complex form
c_0 = a_0;
clear i;
for j = 1:round(n/2)
    c_n(j) = 2 * trapz(t_range,t_range.*exp(-2 * pi * i * j * t_range));
end
y_complex = c_0 * ones(1,length(t));
for j = 1:round(n/2)
    y_complex = y_complex + c_n(j) * exp(2*pi*i*j*t)- c_n(j) * exp(-2*pi*i*j*t);
end
```

```
plot(t,y_general,'DisplayName','Complex General')
```

Problem 6

Compute the Fourier transformation (integral) of the following functions and show the derivation process in detail.

(a) cosine wave: $y = cos(2\pi p_0 t)$

Using the formula for a Fourier transform, we get the following integral:

$$Y(f) = \int_{-\infty}^{\infty} \cos(2\pi p_0 t) e^{-i2\pi ft} dt$$

We express the cosine function in its exponential form:

$$Y(f) = \frac{1}{2}\int_{-\infty}^{\infty} e^{-i2\pi(p_0+f)t} dt + \frac{1}{2}\int_{-\infty}^{\infty} e^{-i2\pi(p_0-f)t} dt$$

Which is the integral definition for the Dirac delta function, and so we obtain the Fourier Transform of our function:

$$Y(f) = \frac{1}{2}\left(\delta(f+p_0) + \delta(f-p_0)\right)$$

(b) cosine wave + dc (direct current) wave: $y = cos(2\pi p_0 t) + d$

Since we are only adding a constant, we know from properties of integrals that the cosine component will have the same solution as in the previous problem. Thus, the solution only requires calculating the Fourier transform of a constant and adding it to the previous solution:

$$Y_b(f) = Y_a(f) + \int_{-\infty}^{\infty} de^{-i2\pi ft} dt$$

This is easily identified as another Dirac delta function, and so our final solution becomes:

$$Y(f) = d\delta(f) + \frac{1}{2}\left(\delta(f+p_0) + \delta(f-p_0)\right)$$

(c) Gaussian function: $y = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x)^2/2\sigma^2}$

Given the Fourier Transform definition, we must evaluate the following integral:

$$Y(f) = \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{\infty} e^{\frac{-t^2}{2\sigma^2}} e^{-i2\pi ft} dt$$

Using Euler's formula, we convert the right exponential term to its trigonometric form:

$$Y(f) = \frac{1}{\sigma\sqrt{2\pi}}\left[\int_{-\infty}^{\infty} e^{\frac{-t^2}{2\sigma^2}}\cos(2\pi ft)dt + i\int_{-\infty}^{\infty} e^{\frac{-t^2}{2\sigma^2}}\sin(2\pi ft)dt\right]$$

Now we can observe that the right integral is the multiplication of an even function (exponential of square) and an odd function (sine), which yields an odd function. Thus, the integral evaluates to zero. The left integral is the multiplication of two even functions, thus we can simplify somewhat our interval.

$$Y(f) = \frac{2}{\sigma\sqrt{2\pi}}\int_{0}^{\infty} e^{\frac{-t^2}{2\sigma^2}}\cos(2\pi ft)dt$$

Using an integral table, we integrate:

$$Y(f) = \frac{2}{\sigma\sqrt{2\pi}}\left[\frac{1}{2}\sqrt{\frac{\pi}{2\sigma^2}}e^{\frac{-\pi^2 f^2}{2\sigma^2}}\right]$$

Simplifying, we get the final result:

$$Y(f) = \frac{1}{2\sigma^2}e^{\frac{-\pi^2 f^2}{2\sigma^2}}$$

This is also a Gaussian function.

Problem 7

(a) Compute the Fourier transformation (integral) of the following function: $y = e^{-a|t|}(b * cos2\pi f_1 t + c * cos2\pi f_2 t)$

First we apply the integral definition of the Fourier transform:

$$Y = b\int_{-\infty}^{\infty} e^{-a|t|}\cos(2\pi f_1 t)e^{-i2\pi ft}dt + c\int_{-\infty}^{\infty} e^{-a|t|}\cos(2\pi f_2 t)e^{-i2\pi ft}dt$$

We then convert the cosine function to its complex form:

$$Y = \frac{b}{2}\int_{-\infty}^{\infty} e^{-a|t|}(e^{i2\pi f_1 t}+e^{-i2\pi f_1 t})e^{-i2\pi ft}dt + \frac{c}{2}\int_{-\infty}^{\infty} e^{-a|t|}(e^{i2\pi f_2 t}+e^{-i2\pi f_2 t})e^{-i2\pi ft}dt$$

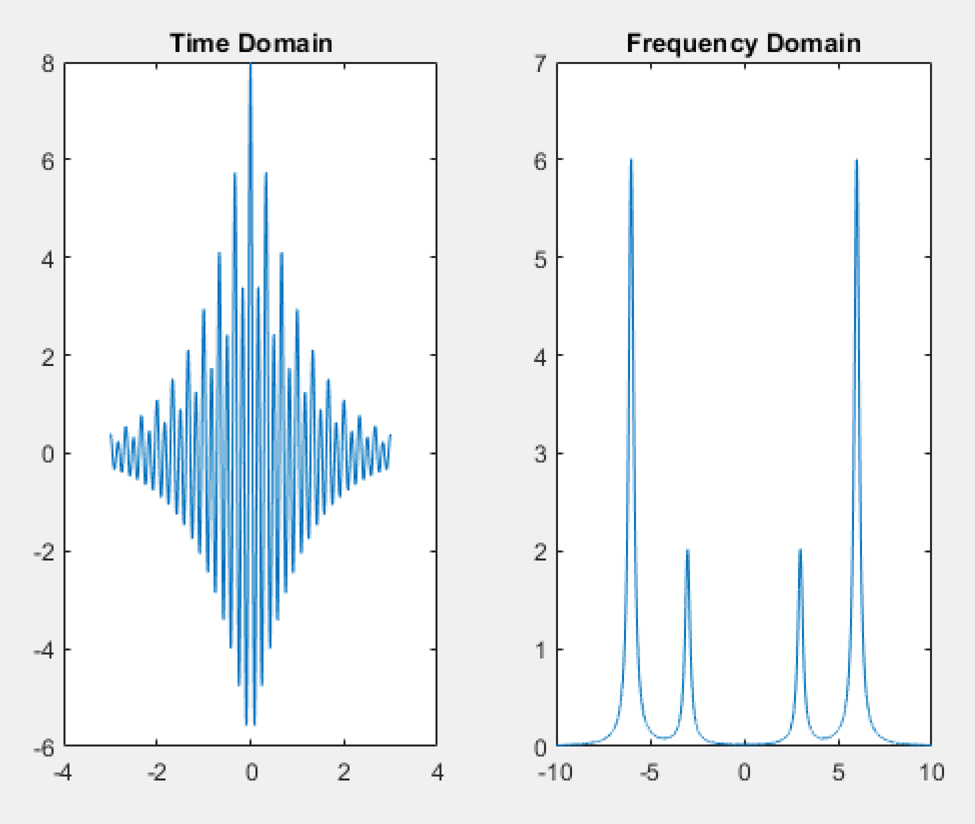
We reorganize per the laws of exponents:

$$Y = \frac{b}{2}\int_{-\infty}^{\infty} e^{-a|t|}(e^{i2\pi(f_1-f)t} + e^{-i2\pi(f_1+f)t})dt + \ldots$$

This is a tabulated integral, which gives the solution:

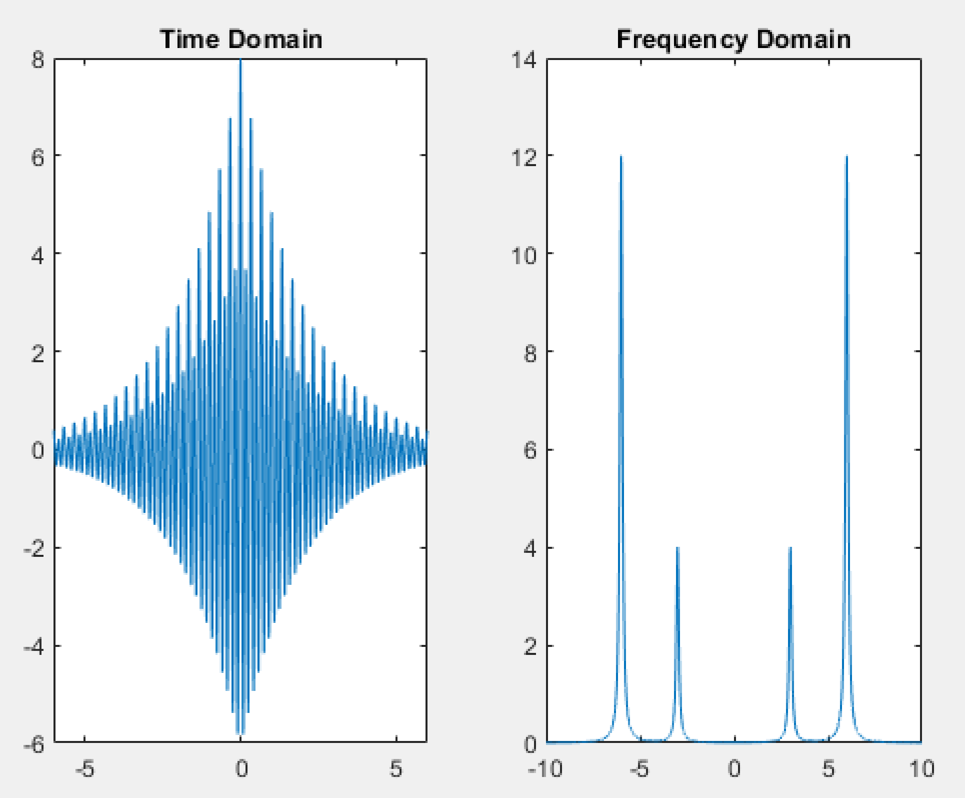
$$Y = b\left[\frac{a}{a^2+(2\pi(f-f_1))^2} + \frac{a}{a^2+(2\pi(f+f_1))^2}\right] + c\left[\frac{a}{a^2+(2\pi(f-f_2))^2} + \frac{a}{a^2+(2\pi(f+f_2))^2}\right]$$

(b) Plot y in time domain and frequency domain, where a = 1, b = 2, c= 6, f_1 = 3, and f_2 = 6




```
a = 1; b = 2; c = 6; f_1 = 3; f_2 = 6;
t = -3:0.001:3;
f = -10:0.001:10;
y1 = exp(-a*abs(t)).*(b * cos(2*pi*f_1*t) + c * cos(2*pi*f_2*t));
Y1 = b*a *(1./(a^2 + (2 * pi * (f - f_1)).^2) + 1./(a^2 + (2 * pi * (f +
f_1)).^2)) + ...
    c*a *(1./(a^2 + (2 * pi * (f - f_2)).^2) + 1./(a^2 + (2 * pi * (f +
f_2)).^2));
subplot(121)
plot(t,y1)
title("Time Domain")
subplot(122)
plot(f,Y1)
title("Frequency Domain")
```

(c) Plot y in time domain and frequency domain, where a = 0.5, b = 2, c= 6, f_1 = 3, and f_2 = 6




```
a = 0.5; b = 2; c = 6; f_1 = 3; f_2 = 6;
t = -6:0.001:6;
f = -10:0.001:10;
y1 = exp(-a*abs(t)).*(b * cos(2*pi*f_1*t) + c * cos(2*pi*f_2*t));
Y1 = b*a *(1./(a^2 + (2 * pi * (f - f_1)).^2) + 1./(a^2 + (2 * pi * (f +
f_1)).^2)) + ...
    c*a *(1./(a^2 + (2 * pi * (f - f_2)).^2) + 1./(a^2 + (2 * pi * (f +
f_2)).^2));
subplot(121)
plot(t,y1)
title("Time Domain")
subplot(122)
plot(f,Y1)
title("Frequency Domain")
```