3C1 SIGNALS AND SYSTEMS LABORATORY.

Stephen Komolafe (21336975)

1 MATLAB review

□ To see one of the basic Matlab commands in operation, create a matrix **A** as follows.

```
>> A = [1 5 ; 7 8]
```

The name to be given to the matrix is thus **A** and the numbers in the matrix are placed inside SQUARE brackets [ ]. Each row of the matrix is separated from the next one with a SEMI-COLON like this one – ; . Thus you have defined matrix **A** to have 2 rows (because there are 2 sets of numbers separated by a semi-colon) and 2 columns (because there are 2 numbers in each row). If you did not type in the same number of numbers in each row, Matlab would complain.

□ Now type in the command to create another matrix **B** such that

$$B = \begin{bmatrix} 2 & 3 \\ -3 & 2 \end{bmatrix} \tag{1}$$

```
>> B = [2 3 ; -3 2]
```

□ Check that Matlab holds B in memory by typing

```
>> B
```

and make sure it is as you expect.

```
>>B

B =

   2    3
  -3    2
```

□ Do the same for **A.**

```
>>A

A =

   1    5
   7    8
```

Matlab can manipulate these matrices, for example can multiply them.

□ Use Matlab to calculate **AB** by typing

```
>> A*B
```

□ Write the answer here and verify that it is as you expect.

```
>> A*B

ans =

  -13   13

  -10   37
```

Verify

$$\begin{Bmatrix} (1)(2)+(5)(-3) & (1)(3)+(5)(2) \\ (7)(2)+(8)(-3) & (7)(3)+(8)(2) \end{Bmatrix}$$

$$= \quad \begin{matrix} -13 & 13 \\ -10 & 37 \end{matrix}$$

□ Use Matlab to calculate the inverse of **A** by typing

```
>> inv(A)
```

□ Write the answer here and verify that it is as you expect.

```
>>inv(A)

ans =

  -0.2963   0.1852
   0.2593  -0.0370
```

Verify

$$\left(\frac{1}{(1)(8)-(5)(7)}\right)\begin{pmatrix}8 & -5\\ -7 & 1\end{pmatrix}$$

$$=\left(-\frac{1}{27}\right)\begin{pmatrix}8 & -5\\ -7 & 1\end{pmatrix}$$

$$=\begin{matrix}-0.2963 & 0.1852\\ 0.2593 & -0.0370\end{matrix}$$

Note: During this laboratory, you may find that you have typed several things erroneously and your results look confusing. If you are worried that you may have gotten Matlab into a funny state you can clear all the variables from memory by doing

```
>> clear
```

1.1 Plotting with Matlab

In Matlab, making graphical plots is done with the command plot which takes some arguments. Plotting takes place in the current figure window unless you specify otherwise. Matlab is a numerical package, therefore you cannot ask it to plot a function like $f(x) = x^2 + 6$ for instance without specifying a range and a number of points in x over which to evaluate the function $f(x)$. You will now do this for the function $x^2 + 6$.

□ Create a vector of equally spaced points in *x* from -5 to 5 with a spacing of 0.1 as follows.

```
>> x = (-5 : 0.1: 5);
creates 1x101 matrix x
x      1x101 double  1x101   double
```

The semi-colon after the command prevents Matlab from printing out the result of the command. To see what Matlab does otherwise you could repeat the command but without the semi-colon.

□ See what *x* is set to. (Type *x* by itself at the Matlab prompt).

```
x =

  Columns 1 through 6

   -5.0000   -4.9000   -4.8000   -4.7000   -4.6000   -4.5000

  Columns 7 through 12

   -4.4000   -4.3000   -4.2000   -4.1000   -4.0000   -3.9000

  Columns 13 through 18

   -3.8000   -3.7000   -3.6000   -3.5000   -3.4000   -3.3000
etc.
```

You should see that it is set to a row vector having numers from -5 to 5 incremented in steps of 0.1. You will observe many numbers and they may fill the screen.

□ Now calculate the corresponding points in $y = x^2 + 6$ by typing

```
   >> y = x.^2 + 6;
```

```
y =

  Columns 1 through 6

   31.0000   30.0100   29.0400   28.0900   27.1600   26.2500

  Columns 7 through 12

   25.3600   24.4900   23.6400   22.8100   22.0000   21.2100

  Columns 13 through 18

   20.4400   19.6900   18.9600   18.2500   17.5600   16.8900
etc.
```

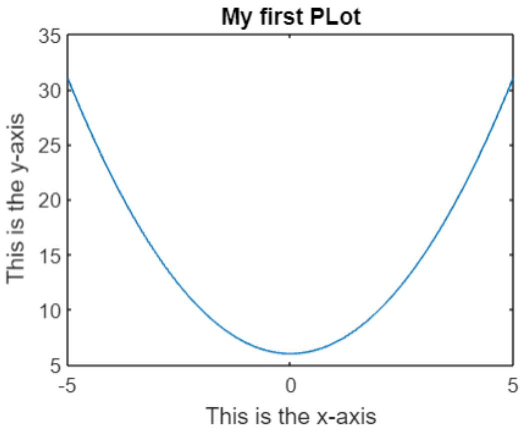
□ Examine the first 5 elements of *y* and verify that they are what you expect. Do this with the command

```
   >> y(1:5)
```

```
ans =

   31.0000   30.0100   29.0400   28.0900   27.1600
```

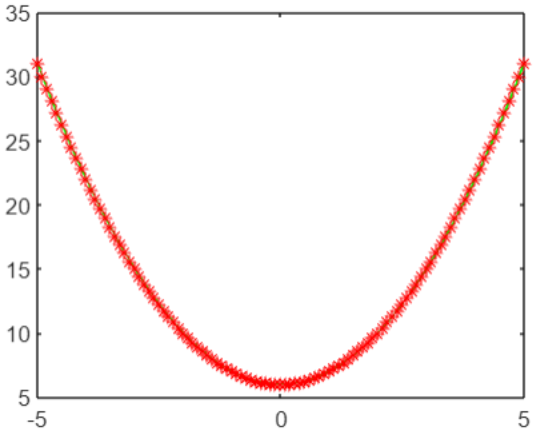
□ Use matlab to plot the graph of *y* vs *x* using:

```
   >> figure(1);plot(x,y);title('My first PLot');xlabel('This is the x-axis'); >> ylabel('This is the y-axis');
```



□ Use Matlab to plot a more informative graph of *y* vs *x* using:

```
>> figure(2);plot(x,y,’g-’,x,y,’*r’);
```

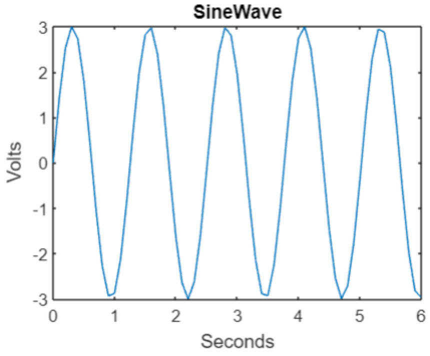


2 Signals

2.1 The Sine wave: a deterministic signal

1. □ Using Matlab, generate the deterministic signal $x_1 = 3\sin 5t$ and plot it over the range $t = 0 : 6$ seconds. This is a sine wave. Label the x-axis as *Seconds* and the y-axis as *Volts*. Let us assume this signal is the measurement of the voltage from an AC power supply. Thus the plot you have made is the plot which you would see if you hooked up an oscilloscope to the terminals of the power supply to measure voltage.

```
>>t = [0 : 0.1 : 6];
>>x_1 = 3*sin(5*t);
>>figure(1);plot(t,x_1);title('SineWave');xlabel('Seconds'); ylabel('Volts');
```



2. □ What is the maximum and minimum value of the sinusoid (in Volts), the frequency of the sinusoid in Hertz, and the period of the wave in seconds?

Maximum = 2.99248V

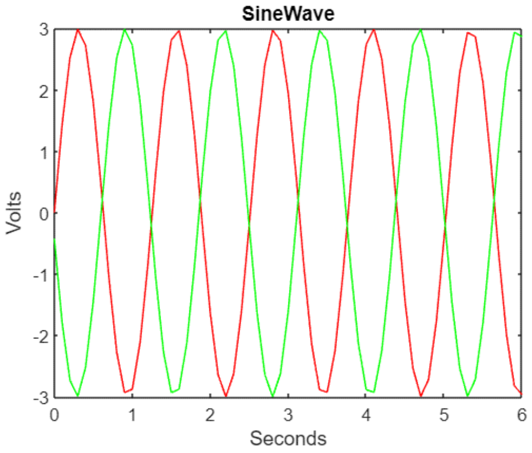
Minimum = -2.99997V

Frequency = 5/2π

Period = 2π/5s

3. □ Use Matlab to plot the graph of $x_2(t) = Asin(\omega_1 t + \phi)$ with $A = 3$, $\omega_1 = 5$, $\phi = -3$ (green colour), label the axes as previously and show $x_1(t)$ (in red) on the same plot. Two lines should now be plotted on the graph. Include this figure in your write up.

```
>>x_2= 3*sin((5*t)+(-3));
>>figure(2);plot(t,x_1,'r-',t,x_2,'g-');title('SineWave');xlabel('Seconds'); ylabel('Volts');
```



□ What is the frequency of x_2 in Hertz? What is the difference between the signals x_2 and x_1? Is there a phase lag? Is this a delay or an advance?

Frequency (x_2)= 5/2π Hz

x_2 is delayed by *0.6s* which causes a phase lag between both signals

4. □ By how much is x_2 delayed (in seconds) with respect to x_1? Use measurement or calculation. How does this value relate to the constant offset term, ϕ in the argument for the sin function in x_2? Show exactly how ϕ can be used to calculate the phase lag in *seconds*. (Hint: ϕ has units of radians.)

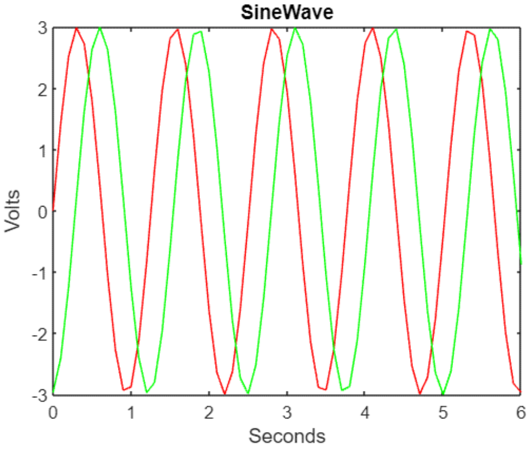
x_2 is delayed by 0.6 seconds with respect to x_1. The point measuring 0.3 seconds and 0.9 seconds were used to determine the delay.

The delay in seconds can be calculated using:

$$t = \frac{2\pi - \phi}{\omega_1}$$

5. □ Use Matlab to plot the graph of $x_3(t) = A\sin(\omega_1 t+\phi)$ with $A = 3$, $\omega_1 = 5$, $\phi = -3+2\pi$. Plot the function in Fig. 3 superimposed on the graph for x_1 (as in previous instructions). By how much is x_3 delayed (in seconds) with respect to x_1? Use measurement from the displayed graph only. How does this value relate to the constant offset term in the argument for the sin function in x_3? Explain any difference or similarity with x_2 in terms of the properties of the sine function.

```
>>x_3= 3*sin((5*t)+(-3+2\pi));
>>figure(2);plot(t,x_1,'r-',t,x_3,'g-');title('SineWave');xlabel('Seconds'); ylabel('Volts');
```

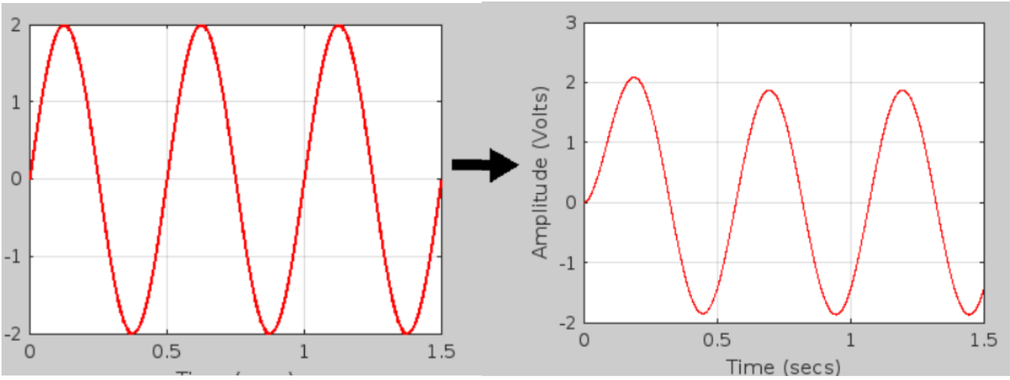


From observing the graph x3 is delayed by approximately 0.3 seconds, therefore: x3≈x1(t+0.3)

Since x3 and x2 are both delayed by a certain degree of ϕ w.r.t. x1, they are both phase-lagged.

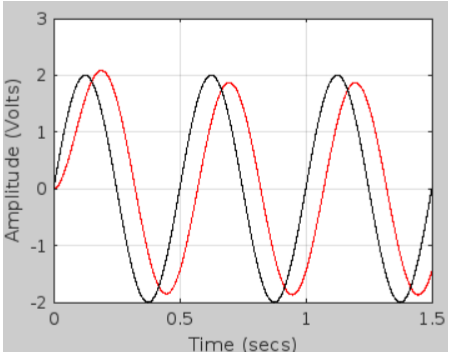
3 Linear Time Invariant Systems

1. □ Examine the input and and output signals corresponding to the default settings of LAB1. The input signal is a Sine wave. What is the output signal? (Is it a Sine wave or a Cosine wave or something else?)



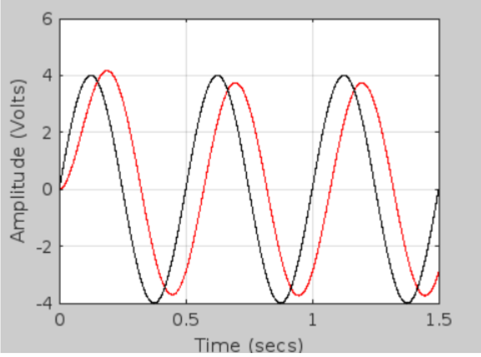
The output is a sine wave like the input as both waves intersect the y(amplitude) axis at 0

2. □ What is the difference between the input and output signals using System 1? What is the same? (Superimpose the input on the output to see this more clearly.)



Compared to the input signal, the output is delayed in reaching its minimum and maximum amplitude values. The first maximum point of the output is greater than the first maximum of the input. In terms of similarities both input and output signals cross the y axis at 0

3. □ Increase the amplitude of the input signal by a factor of 2. What is the corresponding increase in the amplitude of the output signal, after initial transients have decayed? How long does the system take to settle into a steady state response?

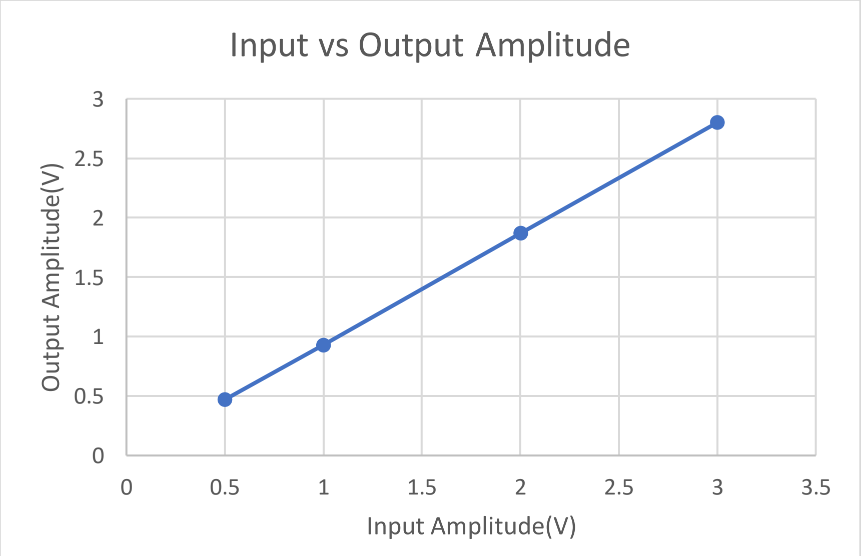


With each increase of the input amplitude by a factor of 2, the amplitude of the output also increases proportionally by that same factor

It takes the system 0.5715 seconds to settle into a steady state

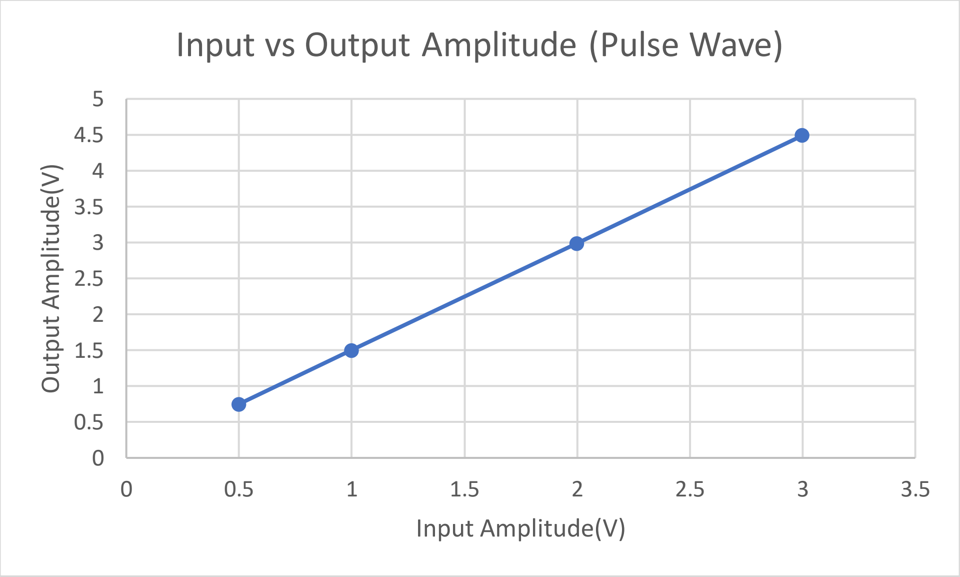
4. □ For input amplitudes of 0.5, 1.0, 2.0, 3.0, measure the corresponding output amplitudes, and plot a graph of input amplitude vs output amplitude below. Label your axes carefully and include the plot in your write up.

Input Amplitude(V)	Output Amplitude(V)
0.5	0.47
1	0.93
2	1.87
3	2.80



5. □ Switch the input signal to the Pulse Wave and set its frequency to 0.8 Hz and repeat the above experiment. Take the output amplitude to be the maximum value of the output pulse after initial transients have decayed.

Input Amplitude(V)	Output Amplitude(V)
0.5	0.75
1	1.50
2	2.99
3	4.49

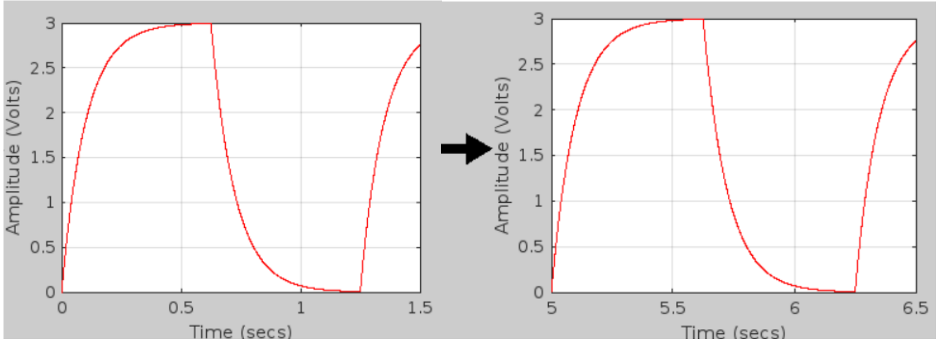


You have just observed a property of LTI systems, i.e., that scaling an input signal by a factor of *a*, causes a similar scaling in the output. Let $x_1(t)$, $y_1(t)$ denote the input and output signals respectively.

$$\begin{aligned}\text{IF } x_1(t) &\rightarrow y_1(t)\\ \text{THEN } ax_1(t) &\rightarrow \boxed{ay_1(t)}\end{aligned}$$

where *a* is *any* complex constant. (i.e. *a* can be either real or complex).

□ Change the time axis to observe some other window in time after 5 secs, say. Is there any difference in the output signal?

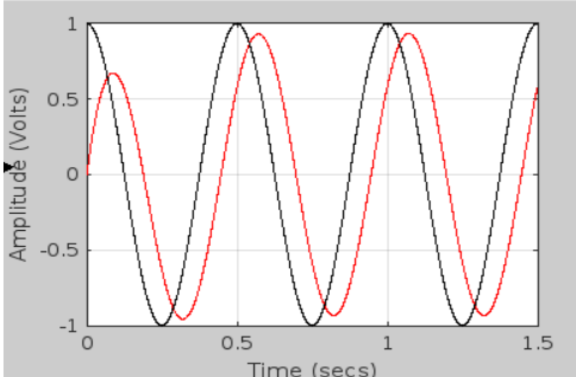


There is no difference between the output from 0 to 1.5 seconds and the output from 5 to 6.5 seconds

The final property of LTI systems is one which LAB1 is not set up to verify, but we anyway state it here. A Linear system possesses the important property of *superposition*. The response to a weighted sum (superposition) of several inputs is a weighted sum of the responses to each of the inputs.

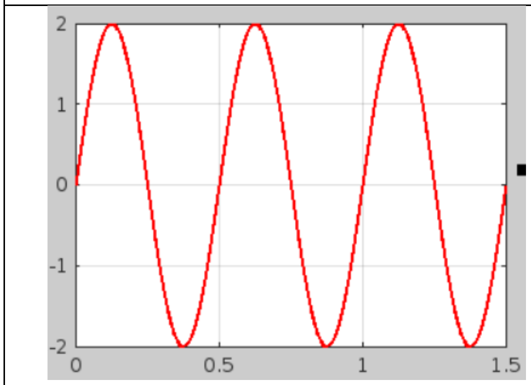
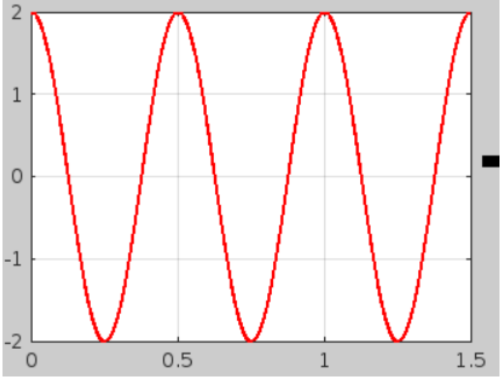
$$\begin{array}{rcl} \text{IF } x_1(t) & \rightarrow & y_1(t) \\ \text{AND } x_2(t) & \rightarrow & y_2(t) \\ \text{THEN } x_(t) + x_2(t) & \rightarrow & \boxed{y_1(t) + y_2(t)} \end{array}$$

6. □ Select the Cosine as the input signal and set the frequency to 2 Hz. What is the difference between the input and output signals using System 1? What is the same?



The output signal is a Sine wave. The outputs amplitude is also different compared to the amplitude of the input. The period of the input and output signals are the same.

7. □ For both the Sine and Cosine input signals (at frequency 2 Hz and amplitude 2) with System 1, write below mathematical expressions for the input and output signals using LAB1 to help you make measurements.

<table>
<tr><td colspan="2">$\omega = 2Hz$ $A = 2V$ $\lambda = 0.5s$ $x = t$</td></tr>
<tr><td>$Asin\left(\frac{2\pi}{\lambda}x\right)$
$= 2\sin\left(\frac{2\pi}{0.5}t\right)$
$\Rightarrow 2\sin(4\pi t)$</td><td>$Acos\left(\frac{2\pi}{\lambda}x\right)$
$= 2cos\left(\frac{2\pi}{0.5}t\right)$
$\Rightarrow 2cos(4\pi t)$</td></tr>
<tr><td> </td><td> </td></tr>
</table>

This final set of experiments has shown (albeit indirectly) another important property of LTI systems. Complex exponential functions are *eigenfunctions* of LTI systems. **The output signal has the same frequency as the input** although changed in phase and amplitude.

8. □ Using what you now know about LTI systems, classify System 2, System 3, and System 4 as LTI or non-LTI. Remember to check input amplitudes covering a wide range (e.g. 1 ↔ 6).

	System 1	System 2	System 3	System 4
LTI	✓	✓	⍰	⍰
Non-LTI	⍰	⍰	✓	✓

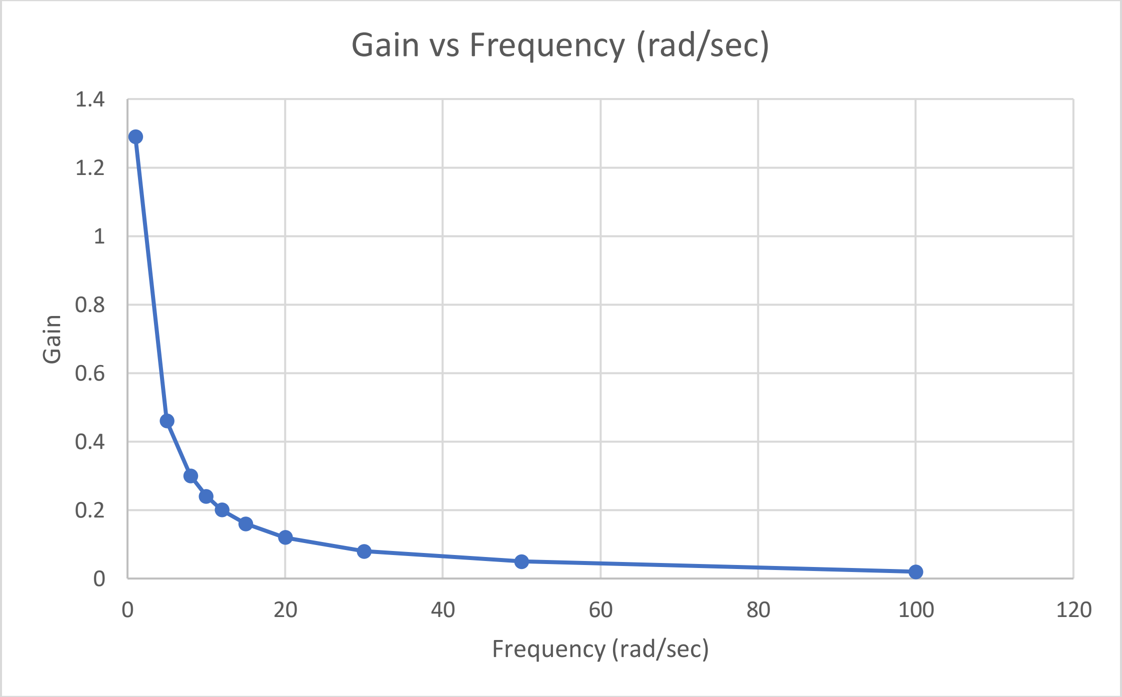
3.1 Gain and Phase as a function of frequency

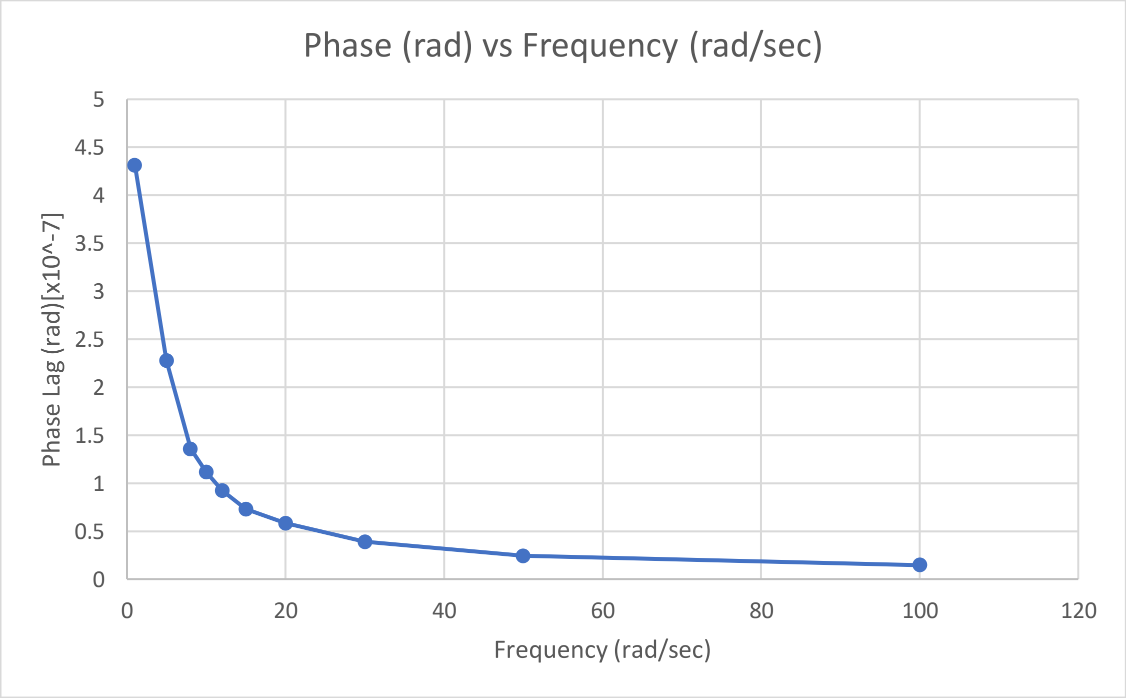
□ Complete the table below for the Sine input at various frequencies using System 1

INPUT		OUTPUT				
Freq. (rad/sec)	Amplitude (x)	Freq. (rad/sec)	Amplitude (y)	Phase Lag (sec)	Phase Lag (rad*($\times 10^{-7}$))	Gain y/x
1	1	1	1.29	0.089	4.314	1.29
5	1	5	0.46	0.047	2.278	0.46
8	1	8	0.3	0.028	1.357	0.3
10	1	10	0.24	0.023	1.115	0.24
12	1	12	0.2	0.019	0.921	0.2
15	1	15	0.16	0.015	0.727	0.16
20	1	20	0.12	0.012	0.581	0.12
30	1	30	0.08	0.008	0.387	0.08
50	1	50	0.05	0.005	0.242	0.05
100	1	100	0.02	0.003	0.145	0.02

$$(* \, Assuming \; that \; 1 \; second = \frac{\pi}{180 \times 60 \times 60} \; radians)$$

□ Plot a graph of Gain vs Frequency (rad/sec) and Phase (rad) vs Frequency (rad/sec) for System 1 using the information above.





□ Is the effect of the system the same at all frequencies? How does the system behaviour change with frequency?

The effect is the same at all frequencies.

As the frequency increases the gain and phase lag decreases.

□ Discuss the significance of this plot with respect to the effect of System 1 on the Pulse Wave and the Speech signal.

In terms of System 1:

- The amplitude (and hence the gain) of the Pulse Wave output increases as the frequency increases.
- The changes in the Speech signal are difficult to discern by eye

3.2 Characterising the Transient Responses of a System

In the last section you saw how the output of a system can be broken down into a transient and steady state response (If you are not sure what this means ask a demonstrator). You also learned how to characterise the steady state response to a sinusoidal input. In this section, the basic concepts in the characterisation of the transient response are introduced. A focus is placed on the characterisation of the transient response to a step input signal.

Consider the system in Figure 2.

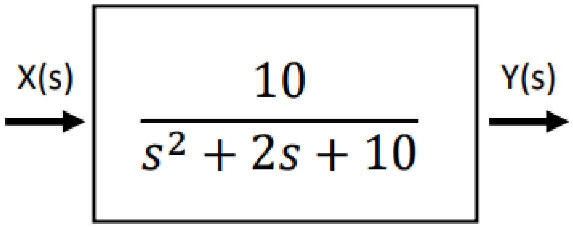


Figure 2: A 2^{nd} order LTI System.

This system can be created in Matlab using the tf() function (in fact the LTI systems in the lab1 GUI were created in this way). It takes two arrays as input parameters. The first is an array that contains the coefficients of the numerator polynomial in order of descending power and the second contains the coefficients of the denominator polynomial.

For example, a system with a transfer function 1/(s + 3) can be created by the line

```
>> the_sys = tf([1], [1 3]);
```

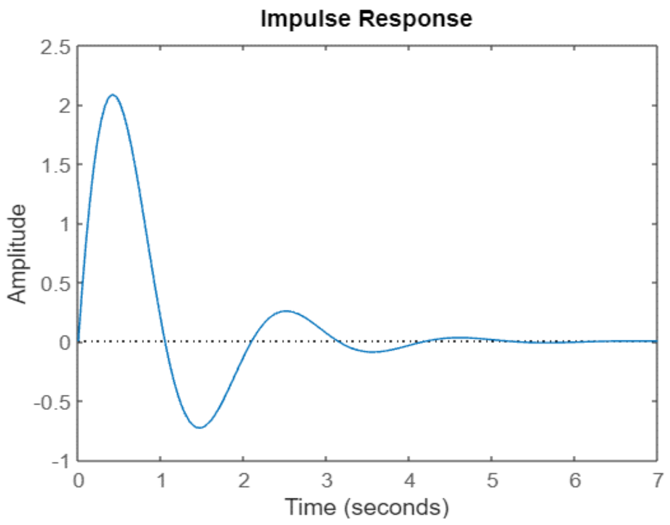
□ Using the tf() function, create a transfer function object that represents the system shown in Figure 2. Verify your answer by omitting the semi-colon at the end of the line used to create the system (this will cause Matlab to print the transfer function on screen).

```
>> sys = tf([10],[1 2 10])

sys =

        10
  --------------
  s^2 + 2 s + 10

Continuous-time transfer function.
```

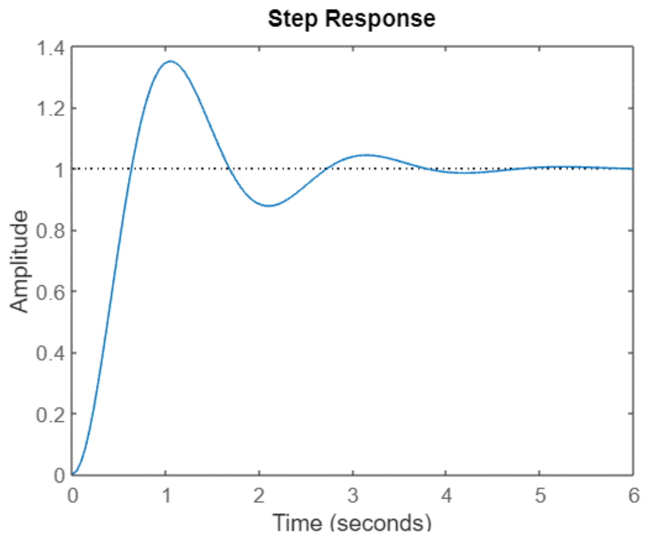
the_sys is not an array or a numeric type but is instead an object of the transfer function class. The object can be used to simulate the step and impulse responses of the system using the step and impulse functions respectively.

1. □ Create two new figures. In the first generate the impulse response and in the second plot the step response. Sketch the plots in your report and label your axes carefully.

```
>>impulse(tf([10],[1 2 10]))
```



```
>> step(tf([10],[1 2 10]))
```



The response of a system to a given input signal consists of the transient response and the steady-state response. The transient response describes the initial response of the system to the input signal, while the steady state response describes the behaviour of the output signal as time approaches infinity.

Higher Order Systems or systems whose transfer functions are unknown can be compared by measuring parameters from their step and impulse responses. Some of the important parameters are:

Impulse Response

- *Peak time*: the time at which the peak value of the output signal occurs.
- *Peak value*: the maximum absolute value.
- *Settling time*: the time after which the impulse response is entirely bounded by a userdefined threshold (e.g., ±0.1). It is effectively the time at which the signal transitions from the transient to steady state behaviours.

Step Response

- *Steady state value*: the value of the output signal as $t \to \infty$.
- *Rise time*: the time taken for the output to go from 10% to 90% of the steady state value.
- *% overshoot*: peak value__________________steady state value− steady state value × 100.

- *Settling time*: the time taken for the signal to be entirely bounded to within a tolerance of the steady state value.

2. □ Using the plots generated previously, record values for all of the parameters listed above. Often systems are specified in terms of the poles and zeros of the transfer function of the system, $H(s)$. The poles are the roots of the denominator polynomial of the transfer function. The zeros are the roots of the numerator polynomial of the transfer function. The location of the poles and zeros governs the behaviour of a linear time-invariant (LTI) system.

Impulse Response	
Peak Time:	0.414 s
Peak Value:	2.09
Settling Time:	3.95 s

Step Response	
Steady State Value:	1
Rise Time:	$0.575 - 0.149 = 0.426\ s$
Overshoot:	$\frac{1.35 - 1}{1} \times 100 = 35\%$
Settling Time:	$4.082 - 0.632 = 3.45\ s$

3. □ Determine the poles and zeros of the transfer function shown in Figure 2.

Poles:

$$\frac{-(2) \pm \sqrt{(2)^2 - 4(1)(10)}}{2(1)} = -\frac{2}{2} \pm \left(\frac{\sqrt{36}}{2}\right)\left(\sqrt{-1}\right) = -1 \pm 3\, j$$

$$\therefore Poles = (-1+3j)\ \&(-1-3j)$$

Zeros:

No $(s+a)$ on numerator

$$\therefore No\ zeros$$

Matlab can specify systems in terms of poles and zeros using the zpk() function. The first two inputs are arrays specifying the zeros and poles (in any order). The final parameter specifies a gain value for the system. For example if a system has the transfer function

$$H(s) = \frac{10(s+3)}{(s+1)(s+2)} \tag{2}$$

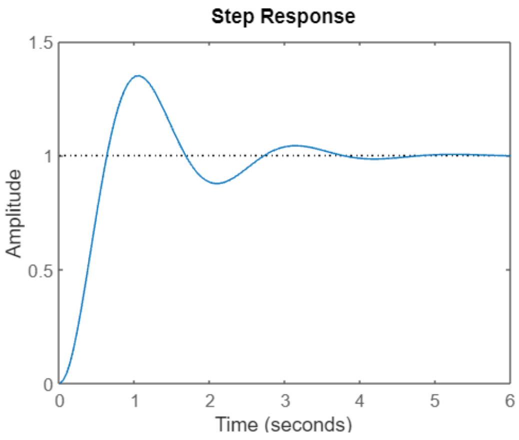
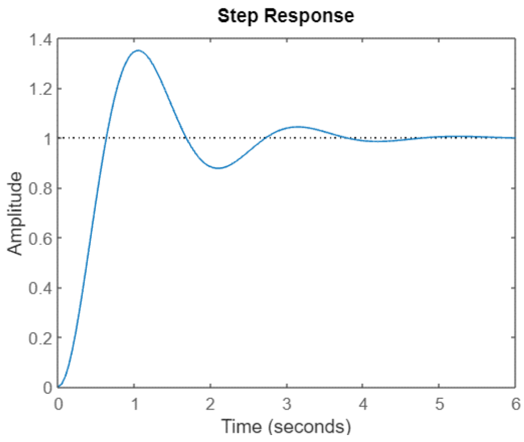
then the gain is 10 and there are poles at $s = -1, -2$ and a zero at $s = -3$. The correct call is then

```
>> the_sys = zpk([-3], [-1 -2], 10);
```

□ Use the zpk() function to specify a transfer system for the system shown in Figure 2 and verify your result by comparing the step or impulse response with the ones you generated earlier. Note if there are no zeros an empty array must be passed to zpk(). An empty array can be created using the following syntax:

```
>> A = [];
```

```
>>A = [];
>>sys = zpk(A, [-1+3j -1-3j], 10)

sys =

        10
  ---------------
  (s^2 + 2s + 10)

Continuous-time zero/pole/gain model.
```

zpk(...)	tf(...)
	

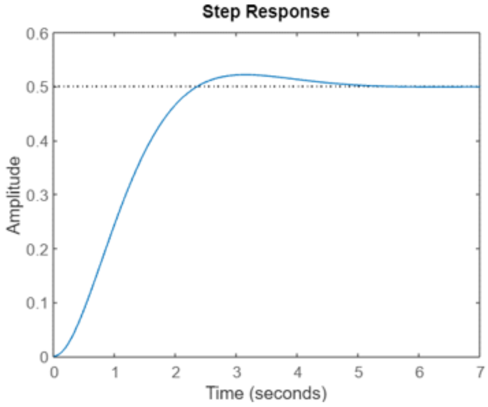
4. □ Create four 2nd order systems, each having a gain of 1, no zeros in the transfer function, and with the poles specified below:

- System A: poles at $s = -1 \pm j$.
- System B: poles at $s = -0.1 \pm j$.
- System C: poles at $s = -1, -2$.
- System D: poles at $s = -1, 2$.

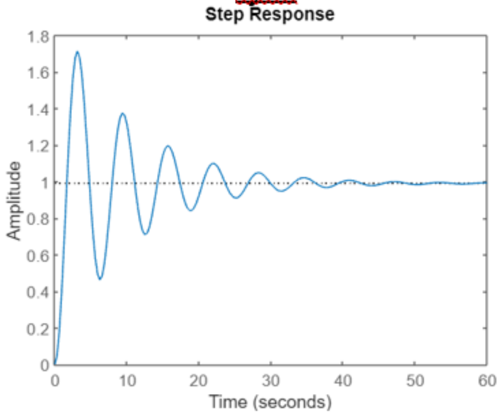
```
>>A = [];
>>sysA = zpk(A, [-1+j -1-j], 1)

sysA =

        1
  --------------
  (s^2 + 2s + 2)

Continuous-time zero/pole/gain model.
>>sysB = zpk(A, [-0.1+j -0.1-j], 1)

sysB =

           1
  -------------------
  (s^2 + 0.2s + 1.01)

Continuous-time zero/pole/gain model.
```

```
>>sysC = zpk(A, [-1 -2], 1)

sysC =

       1
  -----------
  (s+1) (s+2)

Continuous-time zero/pole/gain model.
>>sysD = zpk(A, [-1 2], 1)

sysD =

       1
  -----------
  (s+1) (s-2)

Continuous-time zero/pole/gain model.
```

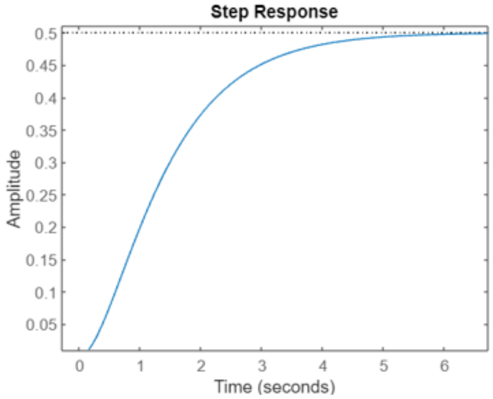
sysA



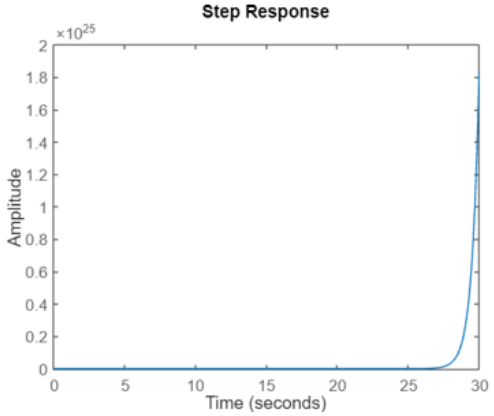
sysB



sysC



sysD



□ Investigate the effect the position of the poles has on the transient response of a system to a unit-step input $u(t)$. What are the general differences between response of the systems that have only real poles compared to the response of the other systems which have pairs of complex conjugate poles? What happens when one of the poles is situated on the right half of the s-plane?

Systems with complex conjugate poles

- The systems with complex conjugate poles (System A & B) display oscillatory behaviour in their responses which decays over a period of time.

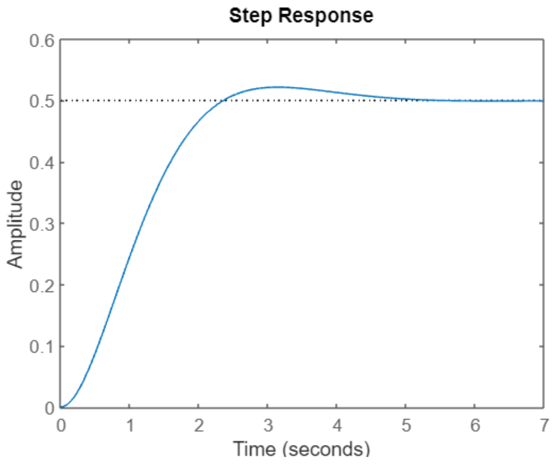
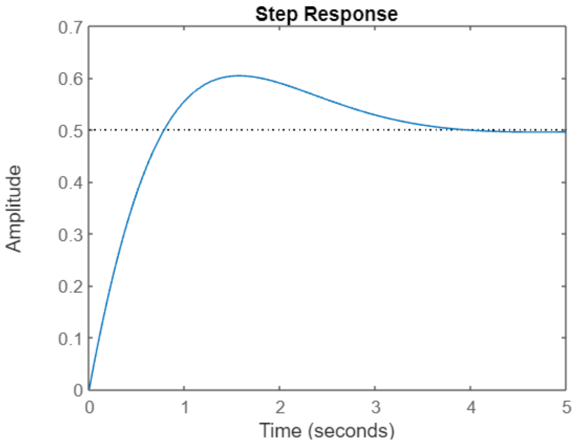
Systems with only real poles

- The systems with real poles only (System C & D) show exponential decay or growth behaviours in their response. The speed of which their transient response decays is proportional to how far away they are from the imaginary axis

Poles on right half of *s*-plane

- The system becomes instable and its response grows exponentially to infinity

5. □ Using System A above investigate what happens to the step response of a system when a zero is added at $s = -1$.

Before zero is added	After zero is added
	

- Peak value increased from 0.52V to 0.604V.
- Time taken to settle into steady state decreased from 5.2s to 3.93s.