EEEE2045: Control Coursework 2022/23

1.0 Introduction

This coursework is the only assessment outside of the progress tests which will be undertaken for the Control part of EEEE2045. Please follow the instructions carefully, especially those for the report. The coursework considers the use of the software MATLAB to undertake control system analysis.

2.0 Control Systems Analysis and Design

This control section of this module considers the transient (time domain) response of "systems" to changes in demand or changes to the environment in which they operate. By transient response we mean the response before we enter steady state.

u(t)
Dynamic System
c(t)

Figure 1: A Dynamic system with a time varying input and output

Examples:

1. u(t) the voltage to a piece of power electronics for feeding a motor, c(t) the voltage signal proportional to the motor speed (or position)
2. u(t) the voltage to a heating element, c(t) the voltage of a temperature sensor
3. u(t) the input to an n-stage audio amplifier, c(t) the output voltage.

System transient response is determined by a set of differential equations which relate the output of the system to changes in input (for example). In this module, we only consider systems that can be represented by linear differential equations.

Let us consider an example. Below is an RL circuit. If we wanted to know how the current in the inductor [c(t) in this case] responds to changes in the input voltage [u(t) in this case] we start by applying Kirchhoff's voltage law around the circuit loop.

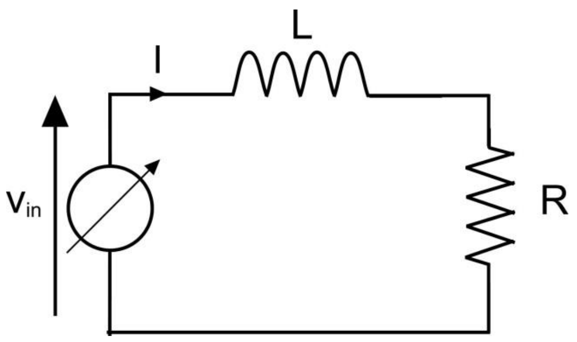


Figure 2: Example of controlled current in a simple LR circuit

$$V_{in} = L\frac{di}{dt} + iR$$

The response of the current, to changes in the input voltage can be found by solving this differential equation. However, this method is tedious (especially for complex systems with much higher order differential components) and so other approaches have been developed. By re-writing the equation in terms of the Laplace operator, it is possible to simplify the approach- this is known as solving differential equations using Laplace Transforms and is used extensively in this module. Re-written, using the Laplace operator (s), this equation becomes:

$$V_{IN}(s) = I(s)Ls + I(s)R = I(s)[Ls + R]$$

Since we are considering the change of the current (output of the system) to changes in voltage (input to the system), we can re-write this in the form of a transfer function:

$$G_p(s) = \frac{I(s)}{V_{in}(s)} = \frac{1}{Ls + R}$$

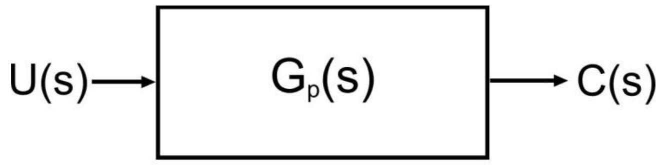


Figure 3: LR circuit, transformed into the Laplace domain for transient analysis

This is known as the Plant transfer function of the system. It shows the relation between the output and the input of a system. The plant shown above is actually very common in certain areas of electrical engineering, especially in power electronics (Buck Converters, DC motors etc.). In general, an nth order plant transfer function can be written as shown below:

$$Gp(s) = \frac{Z_p(s)}{P_p(s)}$$

Where $Z_p(s)$ is an mth order polynomial and $P_p(s)$ is an nth order polynomial in terms of s.

The solutions to the equation $Z_p(s)=0$ are called the ZEROS of the plant.

The solutions to the equation $P_p(s)=0$ are called the POLES of the plant.

The poles of a transfer function are particularly important because they determine the nature of the transient response of the system. On the s-plane, poles are marked with an 'X' and zeros with a '0'. Poles (or zeros) on the real axis are called real poles (or zeros). Otherwise, they are complex and occur in conjugate pairs.

From our previous example, there are no zeros in the plant, but there is a pole at –R/L.

Another example is shown below:

$$G_p(s) = \frac{12(s+3)(s-6)}{s(s+2)(s^2+4s+13)}$$

Here we have poles at: 0, -2 and 2±j3, and zeros at: -3 and +6

2.1 Useful terms:

System Transfer Function- Representation in the Laplace domain which relates the output of a dynamic system to its input
Open Loop Control System- A system without feedback or sensors.
Closed Loop Control System- A control system which uses feedback (e.g. from sensors) to ensure that the output operates at the desired level.
Plant Transfer Function- The transfer function of the system we are aiming to control. This is derived by taking Laplace Transforms of the differential equation that governs the output response to changes in input.
Controller Transfer Function- A transfer function added to an existing system to improve the response of the system to changes in input.
Feedback Transfer Function- The transfer function of the feedback path in a closed loop system. This is related to the transient response and gain of the sensor used to detect the output.
Steady State Error- The difference between the system output in steady state and the target value.
S-Plane- A plot of the poles and zeros in the complex domain, using 'X' to denote a pole and '0' to denote a zero.
Root Locus- A plot showing the trajectories of the closed loop poles of a closed loop system as a function of controller gain
Dominant Roots- Roots of a system which dominate the response to changes in input. This can be used to simplify higher order systems and approximate them with a transfer function of lower order (neglecting the non-dominant poles). If we have two sets of poles, and one set has a real component greater than approximately five times the other, then the poles with the lowest real component are said to be dominant. This means that the poles with the highest real component relate to transients which die out much sooner. For example:

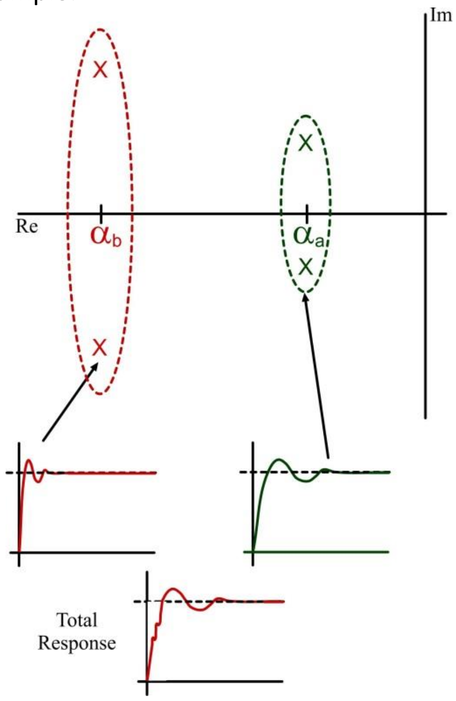


Figure 3: showing effect of dominant poles in a system

If $\frac{\alpha_b}{\alpha_a} > 5$ then the roots at α_a are said to be dominant because they take significantly longer to decay and therefore dominate the time response. The response can therefore be approximated with these poles (and the same overall DC gain) alone

It is important to note that when neglecting a root, its contribution to the steady state gain of the original transfer function must be considered in the "reduced" transfer function. i.e. if a pole (s+25) is neglected, its contribution to steady state gain (s tends to zero- see exercise 1) of 1/25 must still be in the transfer function when the control system is designed. i.e.

$$Gp(s) = \frac{10}{(s+25)(s+1)}$$

Would become

$$Gp(s) = \frac{(10/25)}{(s+1)}$$

If the pole at (s+25) was neglected to derive a simpler transfer function.

Step Response- Response of the system to instantaneous changes in the input.

2.2 Using Matlab for Control Systems Analysis and Design

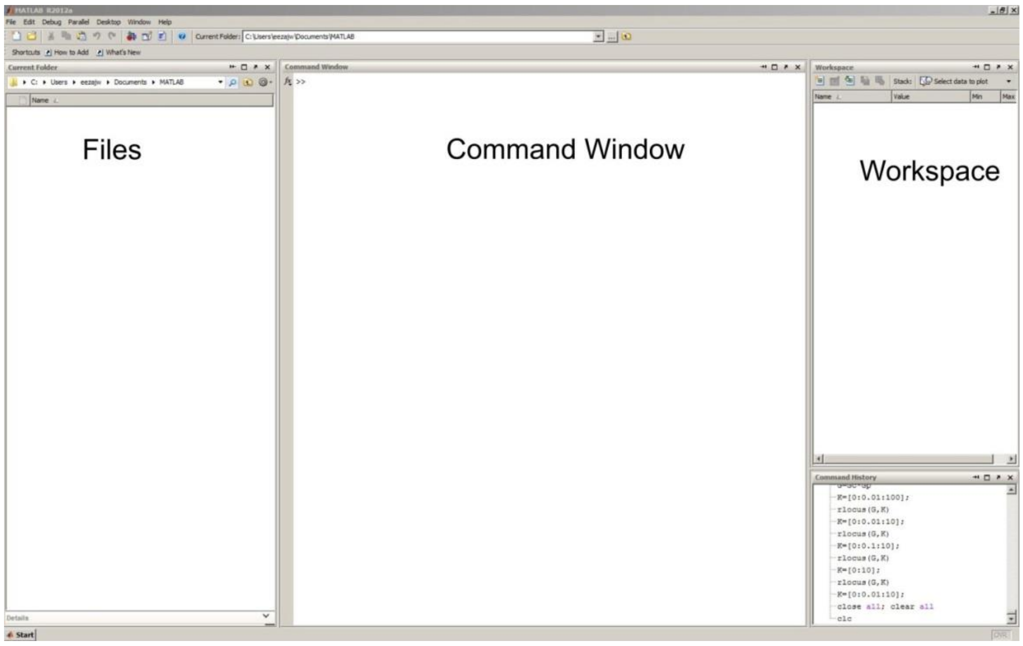
Matlab is a software tool used extensively in engineering for solving mathematical problems relating to various disciplines. The software has an extensive command library for working around transfer functions and displaying control engineering related diagrams such as the root locus, Bode Plots etc.

This coursework aims to give students the opportunity to work in the Matlab environment and build on experience with the software that they may already have. If students have not used it before, the coursework sheet has enough information to enable them to complete the exercises.

Control systems analysis can be carried out using various tools in Matlab. We will focus on the use of the command line interface. This approach will allow users in the future to write scripts for solving complex control problems, because the scripts use the same commands that you will learn to use in this work.

2.3 Inputting transfer functions in Matlab:

When opening Matlab, you will be presented with the following application window:



```
K=[0:0.01:100];
rlocus(G,K)
K=[0:0.01:10];
rlocus(G,K)
K=[0:0.1:10];
rlocus(G,K)
K=[0:10];
rlocus(G,K)
K=[0:0.01:10];
close all; clear all
clc
```

Figure 4: Matlab Command Line Interface window. Note: Some of the windows may be in different areas as this is reconfigurable.

Transfer functions can be inputted directly into Matlab using the 'tf' (transfer function) command, an array which represents the numerator coefficients, and one for the denominator coefficients. For example:

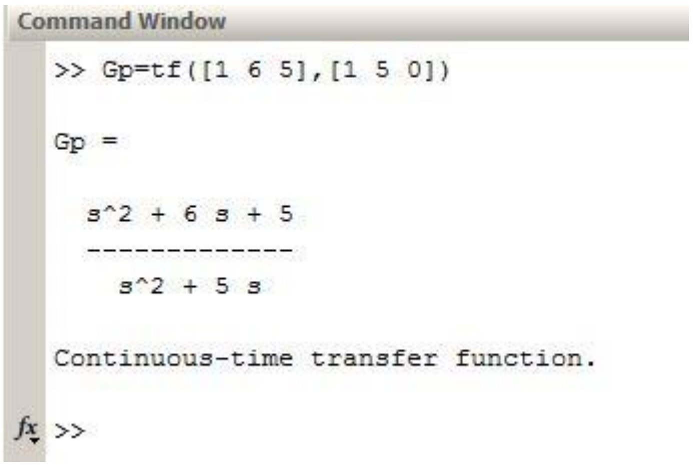
$$Gp(s) = \frac{(s^2 + 6s + 5)}{s^2 + 5s}$$

The array for the numerator, (s^2+6s+5) is [1 6 5]
The array for the denominator (s^2+5s) is [1 5 0].

This can be inputted into Matlab in the command window by typing:

Gp= tf([1 6 5], [1 5 0])

To achieve the following result in the command window:



```
>> Gp=tf([1 6 5],[1 5 0])

Gp =

  s^2 + 6 s + 5
  -------------
    s^2 + 5 s

Continuous-time transfer function.

>>
```

Figure 5: Results of using the "tf" command to define a transfer function in Matlab

Other commands may also be useful. Some elements of command line work in Matlab is similar and has similar functionality to those in C programming. A good example is the "for" loop. It is important to note that some syntax differences occur:

```
for(a=1:1:10)
        Code
end
```

Result: a= 1, 2, 3, 4 …….8, 9, 10

```
for(a=0: 0.1:10)
        Code
end
```

Result a= 0, 0.1, 0.2, 0.3, 0.4 ……. 9.8, 9.9, 10

2.4 Saving figures for this coursework

The easiest way to get good figures for a report, without having to change lots of settings is to do the following:

1. Open a Word Document- Matlab is pre-set to be able to copy figures into Word with a reasonable quality.
2. In the word document, type the exercise that the result is from so you can remember when referring to it later
3. On the figure, click edit, then Copy Figure
4. In the word document, under your current exercise, click paste (Or CTRL-V).
5. MAKE SURE YOU SAVE THE WORD FILE AFTER EACH UPDATE

3.0 Coursework Exercises

Throughout this document you will be asked to save figures from Matlab into a Word Document. To do this effectively, use the method described above.

3.1 Exercise A: Familiarisation with Matlab and First Order System Reponses

Exercise 1:

First, consider the simple plant:

$$G_p(s) = \frac{10}{s+20}$$

Enter this into the command window using the tf command: Gp=tf([10],[1 20])

To see how this system would respond to a unity step input we use the command "step". Since we have already defined the plant as Gp, we can simply type: step(Gp).

The step response shows how the system responds to instantaneous changes in input- this is quite a common response used in control engineering. It is important to note that the size of the step is not important- the shape of the response will always be the same, but it will be scaled by magnitude of the input.

What is the steady state value of the response? This can be determined by clicking anywhere on the curve and then dragging the cursor to the end of the transient. This will show a box with the time and the system output at this point.

The steady state gain of the transfer function is determined mathematically as:

$$\lim_{s\to 0}\left(G_p(s)\right)$$

Currently, this value is equal to 0.5 for the previous plant- as has been shown in the figure that has just been plotted. To get unity gain in steady state we need to edit the transfer function.

$$G_p(s) = \frac{a}{s+20}$$

What value of 'a' is required to achieve unity gain? Edit the transfer function and re-plot for a steady state gain equal to one. Remember that the time taken to reach 63.2% of the final value is equal to the "time constant" of a first order system. Using a cursor, find the time constant for this system.

For the report- Paste this figure into the word document. Comment on the steady state gain of the transfer function originally, and with the adjusted value.

Exercise 2:

When exercise 1 is complete, you can clear the last exercise by using the commands: Close All, Clear All, clc

This first command closes all figure windows (make sure you have pasted them into the word document!). The second command clears all the transfer functions we are currently working on. The final command “cleans” the workspace. This will start us with a clean workspace for the next exercise.

Consider the following transfer function:

$$G_p(s) = \frac{a}{s+a}$$

We want to see how the step response changes if we vary a for a= 1, 3 ,12 ,20

Open a figure by typing “figure” in the command window and press return. Then type “hold” and press return. This last command stops us from overwriting the previous waveform on the plot with the next one; allowing us to see all the responses on the same figure.

Start with a=1, calculate Gp(s), input it into the Matlab command window and then print the Step command on the figure. Then, continue to plot the responses for the other values of a on the same figure. Remember- if your previous plots disappear it is because you did not type “hold” when you opened the figure window. You only need to do this once! It is suggested that you use different colours for each response- this is easy.:

```
step(Gp,’b’)                    % Print step response in blue
step(Gp,’r’)                    % Print step response in red
step(Gp,’g’)                    %Print step response in green
step(Gp,’k’)                    %Print step response in black
```

Calculate the time constant of each waveform using the same method as before but remove the cursor labels before saving the figure for clarity. To do this, right click on the cursor label and click “delete”. Save the figure

For the report- Paste the figure into your Word document, comment on what it shows. Tabulate the time constants. Comment on how the time constant relates to the denominator in the plant.

One this figure has been saved. Type “close all, clear all” in the command window. Input the transfer function

$$G_p(s) = \frac{20}{s-20}$$

Until now, all the poles have had a positive value and would therefore sit in the left-hand side of the s-plane. The above plant has a “right-hand” (positive) pole. Plot the step response- save the figure.

For the report- Save the figure, comment on what it shows- what has happened here?

Exercise 3:

We will now consider dominant roots. Clear the workspace and close all figures that have been saved. Create a new figure, and “hold” the axes (see Exercise 2 if you have forgotten how to do this). Plot the response of the following function to a unity step input:

$$G_p(s) = \frac{10}{s+10}$$

On the same figure (remember that we have "held" the axes) plot the unity step response and calculate the time constant of:

$$G_p(s) = \frac{10a}{(s+10)(s+a)}$$

For a= 1, 10, 20, 100

Remember to change the colours as we did in Exercise 2. It might be easiest to take the original plant (10/(s+10) and multiply it by the one that you are adjusting… otherwise you will have to recalculate the polynomial for the denominator every time you change it. E.g.

For a=1:

```
Gp=tf([10],[1 10])          % this is the original plant
Temp=tf([1], [1 1])         % this is a/(s+a) for a=1
G=Gp*Temp                   % Multiply them both together
step(G, 'b')                % Plot the unity step response
```

For the report- Save figure and tabulate the time constant. What happens when 'a' is much smaller than 10? What about when it is much bigger than 10? How can this be explained? (Hint: consider the effects of dominant roots in a transfer function).

3.2 Exercise B: The 2nd Order Response

A lot of systems found in engineering are second order in nature. Electrical Engineering examples include circuits with inductive and capacitive components (resonant circuits), filters etc. A second order response has a transfer function with the following "standard" form:

$$G(s) = \frac{\omega_0^2}{s^2 + 2\omega_0\xi s + \omega_0^2}$$

ω_0 is the natural frequency. Note- ω_0^2 is used in the numerator to ensure unity dc gain.

ξ is the damping factor and determines the degree of overshoot:

- If $\xi>1$ the denominator will factorize into two real roots (or poles)
- If $\xi=1$; it will factorize into two equal real poles
- If $\xi<1$ it will have complex poles which can be represented in the s-plane as in Figure 6.

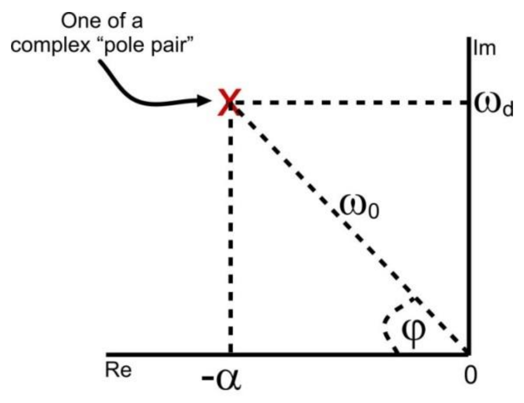


Figure 6: S-plane with Complex pole

We have roots (poles) at $s = -\alpha \pm j\omega_d$
Where ω_d is the damped natural frequency

From Trigonometry:

$$\xi = \cos\varphi$$

$$\alpha = \omega_0 \cos\varphi = \omega_0\xi$$

$$\omega_d = \omega_0 \sin\varphi = \omega_0\sqrt{(1-\xi^2)}$$

For example, consider the plant:

$$G_p(s) = \frac{200}{s^2 + 8s + 100}$$

If we compare this to the standard form:
$\omega_0 = 10$, dc gain=2, $2\xi\omega_0 = 8$; Therefore $\xi = 0.4$, $\alpha = 5$, $\omega_d = \sqrt{84}$

Exercise 4a:

Clear the workspace and close all figures previously saved. Open a new figure and hold the axes.

We will now consider the percentage peak overshoot- this is the level to which the response will overshoot the target value (unity in this case). In order to ensure that we get a reasonable accuracy in the plot so that we can read the data from the trace we need to define the time steps at which the step responses will be plotted. To do this we simply type:

```
time=0:0.01:6                %i.e. time axes from 0 to 6s in 10ms steps
step(Gp, time)               %Step response over defined time range
step(Gp, time, 'r')          %This can also be carried out with a colour
```

Plot the step response, using different colours, for:

$$G_p(s) = \frac{10}{s^2 + as + 10}$$

For a= 0.2, 0.6, 1.2, 1.6 and 3.0.

For each response, use the cursor to determine the overshoot, and the time at which it occurs. Tabulate these results- this should be done neatly in the final report but can written down for now or put into the Word file if you wish.

Add another column to the table with the damping factor, ξ. Calculate the damping factor from the data obtained. The following formula will be useful:

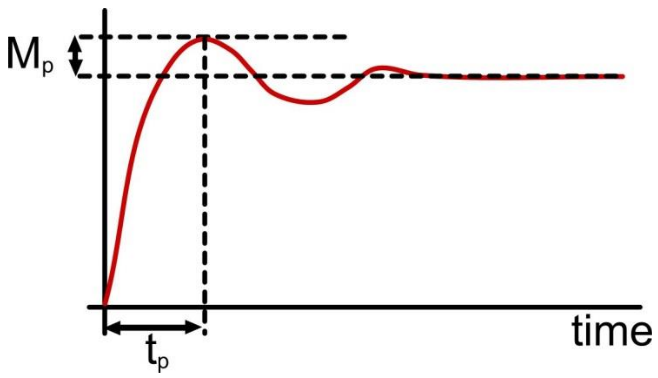


Figure 7: Peak overshoot and time to peak overshoot shown for a second order response

$$M_p = e^{\left[\frac{-\xi\pi}{\sqrt{1-\xi^2}}\right]}$$

$$t_p = \frac{\pi}{\omega_0\sqrt{1-\xi^2}}$$

Where M_p is the maximum overshoot (i.e. overshoot is the steady state value plus M_p), and t_p is the time to the maximum overshoot.

For the report- Save figure and discuss. Table with calculated damping factors- show working. Discussion about the effect of damping factor on the overshoot.

Exercise 4b:

Dominant roots can also be considered in higher order systems. Sometimes, they allow us to simplify the system we are controlling because the response is dominated by a simpler, first order system. When you have saved the previous figures, clear the workspace.

Open a new figure, hold the axes. Plot the unity step input responses of the following transfer function for a= 5, 7, 10, 25, 35.

$$G_p(s) = \frac{100a}{(s^2 + 12s + 100)(s + a)}$$

It is suggested that you used the technique in Exercise 3, to avoid having to expand the dominator brackets into a single polynomial for each value.

For the report- Save figure and discuss. Explain the effect of the dominant roots- this may be best explained with a diagram in the complex plane showing the poles for each value of ‘a’ and explaining why the shape of the response changes.

3.3 Exercise C: Closed loop control

We have currently been considering the response of a system to changes in input, without any knowledge of how the output changes. This is classed as open loop control. Open loop control looks like this:

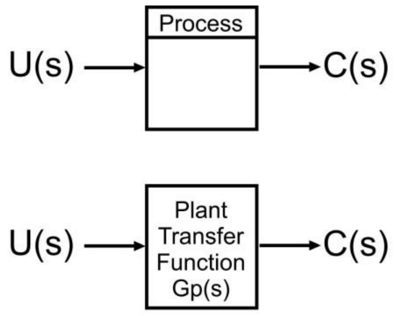


Figure 8: Open loop control system

And has the transfer function:

$$\frac{C(s)}{R(s)} = G_p(s)$$

This means that we apply an input and assume from knowledge of the system that the output will have a certain value. For example, if we have a 2 Ohm resistor we know that to get 1A of current to flow we need to apply 2V. This seems simple enough, but in the real world the resistance is a function of temperature (the environment), the component tolerance, and the age of the resistor. As a result, over time, this value of resistance may change. Alternatively, the tolerance of the resistor may only be 5% so it could have a value anywhere between 1.9 and 2.1 Ohm. If the current flow regulation is important, we need a better control method… closed loop control.

Closed loop control looks like this:

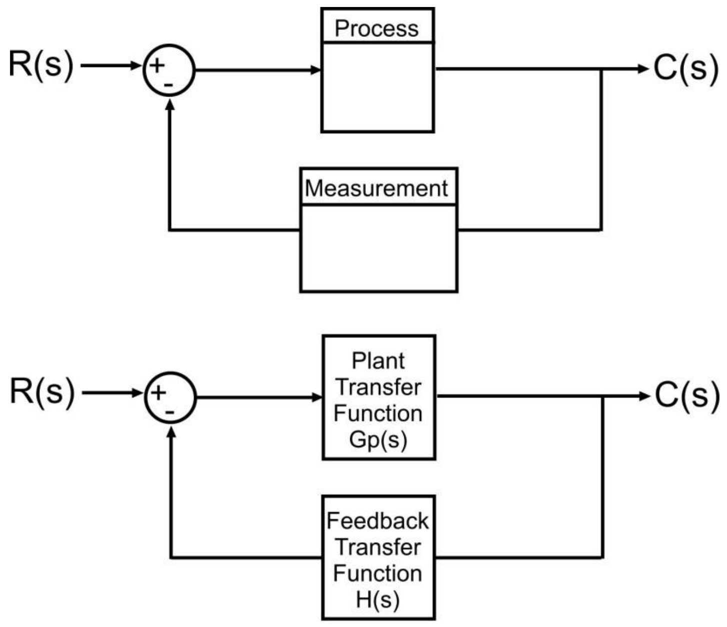

R(s)
+
-
Process
C(s)
Measurement
R(s)
+
-
Plant Transfer Function Gp(s)
C(s)
Feedback Transfer Function H(s)

Figure 9: Closed loop control system

And has the transfer function:

$$\frac{C(s)}{R(s)} = \frac{G_p(s)}{1 + H(s)G_p(s)}$$

Where H(s) is the feedback path transfer function and $G_p(s)$ is the forward path transfer function (the same as the open loop case, for example). Continuing with the resistor example, the current is now measured (using a sensor, for example), and is compared with a reference value (1A)- the error is then fed into the forward path, in order to attempt to regulate the output. H(s) is the dynamic response and gain of the sensor that we use to measure the current. In our examples, we will consider a unity gain, ideal sensor so H(s)=1. This is very rare in the real world as sensors need to attenuate the large signal being measured (current in a 1MW motor, for example) so that it can be fed into a digital controller. Unfortunately, "closing the loop" will not mean that the system is able to respond how we would like it to- we may not even get zero steady state error. How could we improve this? Control System Design.

What is control design?

Up until this point we have considered that we must live with the response of the system that we are working with. Control Design involves the addition of poles and zeros to the closed loop transfer function to get the system to respond to changes in a manner that has been defined or is required.

First, we need a design point. For instance, let's say that we want the output c(t) to follow our reference r(t) (note that these have been transferred into the time domain) and that we wish c(t) to settle to 2% of its final value in 500ms. We also want the response to be reasonably well damped ξ=0.7.

A useful formula for this is the time constant of a second order system, T.

$$T = \frac{1}{\xi\omega_0}$$

After 3T seconds the response should be within 5% of the final value. By 4T this is reduced to 2%. As a result, for our example:

$$4T = \frac{4}{\xi\omega_0} = 500ms, \qquad \therefore \omega_0 = 11.42$$

By putting ξ and ω_0 into the denominator of the standard second order response equation ($s^2 + 2\omega_0\xi s + \omega_0^2$) and setting this to zero, we can get the value of the roots required. In this case the desired closed loop roots (poles) are s=-8±j8.16.

For the system to respond as we want it to these poles MUST be dominant in the closed loop transfer function.

If the system is not able to meet the requirements on its own, then we need to add some poles and zeros. This is done by adding a Control Transfer Function to the system.

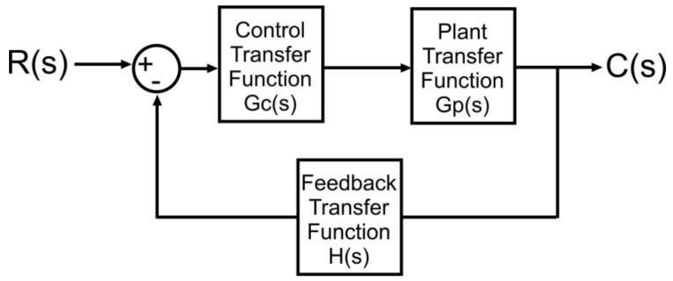

R(s)
+
-
Control Transfer Function Gc(s)
Plant Transfer Function Gp(s)
C(s)
Feedback Transfer Function H(s)

Figure 10: Closed loop control system, with controller for changing transient response

The poles and zeros of Gc(s) are used to change the response of the overall system that we are controlling. The transfer function of Gc(s) can be implemented using Operational Amplifier circuits or digitised and implemented using a micro-processor or micro-controller.

For the closed loop case with the controller, the closed loop transfer function becomes:

$$\frac{C(s)}{R(s)} = \frac{G_p(s)G_C(s)}{1 + G_p(s)G_C(s)H(s)}$$

Or, with H(s)=1 (unity feedback path or sensor gain):

$$\frac{C(s)}{R(s)} = \frac{G_p(s)G_C(s)}{1 + G_p(s)G_C(s)}$$

It should be clear from this that the roots of the denominator, i.e. the poles of our closed loop system, are now a function of both the original plant and the controller. We can therefore add poles, zeros and gain to Gc(s) to ensure that we have dominant poles at our design point. In some cases (for simple systems) we can compare our transfer function with the standard second order form to get our values for Gc(s). Alternatively, we can use a design method such as the Root Locus or Frequency Domain methods such as those using Bode Plots.

Assume we have a controller given by KGc(s), where Gc(s) only contains poles and zeros- K is effectively the gain of the controller. Given that we know where the poles and zeros of the controller are, the root locus is a plot in the complex domain of the locus of the poles of the CLTF as K is varied from 0 to a maximum value. As a result, we can derive the value of K which gives us the required design point. For now, we will simplify this- a full control methodology, including what the form of Gc(s) takes is covered the third-year module EEEE3083.

Exercise 5

Clear the workspace and close all figures previously saved. Assume that we have the following plant

$$G_p(s) = \frac{10}{s + 10}$$

And we want the transient response of the closed loop system to be within 2% of the target value in 20ms and have a damping factor equal to 0.707. From before:

$$4T = \frac{4}{\xi\omega_0} = 20ms, \qquad \omega_0 = 282.84\text{rads}^{-1}$$

From our standard second order equation this means that our dominant poles need to be at approximately -200±j200.

Let us start with a controller with the transfer function Gc(s)=K. This is called a proportional controller because the control action is proportional to the error (Reference- Output).

We can check for all values of K, whether we can meet the design point, by using the Matlab command "rlocus". To use this command, we type:

rlocus(G) where G is the forward path transfer function with the control gain set to 1.

This last point is important because by the definition of the root locus (closed loop pole locations as a function of control gain) this is controlled by Matlab. By setting it to unity, Matlab will simplify multiply by its own gain and calculate where the closed loop poles are for K from zero to infinity.

Try it- does the root locus pass through the design point? No- don't worry about zooming into the axes it's impossible to get a second order response from this combination of control and plant.

For the report- Save the root locus figure and discuss.

Now consider a more complex controller. This controller is very common and is called a PI controller because it has a Proportional component (P) and an integral component (I). Its transfer function is:

$$G_c(s) = \frac{K(s+a)}{s}$$

We now have two variables: K and a. How do we select the values for these?

There are three obvious methods:

1. "Tuning" methods (including trial and error)
2. Mathematical Approaches (Only work on simple systems)
3. Design methods such as the Root Locus or Bode Plot design method.

Point three is the best engineering approach for these types of systems- this is covered in the third-year module EEEE3083. Point one can be useful if we do not understand the plant well. Point 2 can only be used in simple systems- fortunately; this is the case for us!

Let's take a value of a=205. We will cover why shortly. We now need to plot the root locus again- this one is much more interesting than the last. This time the forward transfer function is:

$$G(s) = G_P(s)G_c(s) = \frac{10K(s+205)}{s(s+10)}$$

Remember- we set K=1 when using when using the rlocus command. Plot the root locus for this system. Use a cursor and move it until you are on the -200±j200 (our original design point). What is the gain at this point? This is the value of K which is needed for out controller. Save the figure.

Now we want to check the closed loop response in the time domain. Since we have the forward path transfer function, we can calculate the CLTF using the “feedback” command. Note that we now include our derived value of K in the transfer function. Since H(s)=1, this is achieved by typing:

```
CLTF=feedback(G,1)
```

Where G=Gp(s)*Gc(s)=G(s). In a new figure plot the response to a unity step input. Verify that the response is as expected. Are we within 2% of the settling point in 20ms?

For the report- Save Step response figure. Discuss the results.

So, how did I know that a=205 would work?

Simple maths (in this case):

$$G_c(s) = \frac{K(s+a)}{s} \qquad G_p(s) = \frac{10}{s+10}$$

$$CLTF = \frac{C(s)}{R(s)} = \frac{G_p(s)G_C(s)}{1 + G_p(s)G_C(s)}$$

The closed loop poles are found by setting the denominator to zero- $1 + G_p(s)G_C(s) = 0$.

$$1 + G_p(s)G_C(s) = s^2 + (10K + 10)s + 10Ka$$

This is a second order system, so, we can set it equal to our standard second order response denominator:

$$s^2 + 2\omega_0\xi s + \omega_0^2 = s^2 + (10K + 10)s + 10Ka$$

We had $\xi = 0.707$ and $\omega_0 = 282.84$rads^{-1} as out design point. By comparing coefficients:

K=39, a=205

The same ones we used on the previous exercise. Note that the effect of the closed loop zero is neglected in this case. This analytical method has limited use- more general methods are taught in the third-year module EEEE3083.

4.0 Design Questions:

The following two questions should now be attempted by using skills you have learned during the above exercises. The aim is to test that you have understood the work covered in lectures and this coursework.

1. Assuming we have a circuit whose input voltage to output current transfer function is:

$$\frac{I(s)}{V(s)} = \frac{6}{15s + 1}$$

Design a PI controller to yield a response with ω_0 = 15 rads^{-1} with ξ= 0.9. Using the command “bode” plot the bode plots for the closed loop system- what is the bandwidth of this system?

2. A position system is characterised by the following transfer function:

$$\frac{\theta(s)}{V_{motor}(s)} = \frac{15}{(s+150)(s+8)}$$

Where φ is the position of the system, and V_{motor} is the voltage applied to the motor. Design a closed loop transfer function, using a P+I controller, to achieve within 2% of the target value in 750ms and a damping factor of ξ= 0.6.

Hint: Trying coefficient comparison method we have used before will be tricky as we have a third order system- how might we get around this?

5.0 Report Contents

Each student on the module should write a report. This should be well laid out with quality, fully labelled and captioned figures. You may add a front cover. Although there is no page limit, you must think about what is reasonable for the work. Marks will be reduced for unnecessary over/under length reports.

A structure of the report should be:

Title Page

1.0 Aim of the lab (1/3 page max)

2.0 Approach (How have you used Matlab for this) (2/3 page max)

3.0 Results and Discussion (See "For the report" in all the above)
- 3.1 Exercise 1
- 3.2 Exercise 2
- 3.3 Exercise 3
- 3.4 Exercise 4a
- 3.5 Exercise 4b
- 3.6 Exercise 5

4.0 Design Questions Solutions

5.0 Summary/Conclusions

Note on the discussion: This should not be too long. It should also not contain a description of the figure- I already know what it looks like because it is included in the report. What I am looking for is your understanding of what you see and what it means- this is where the marks will be earned.

Al Watson, September 2023

Appendix A: Rubric for Coursework

	Exercises [60 Total- 10 Each]				
	No diagram or discussion [0]	Diagram present, poor discussion [1-3]	Diagram present, some control elements in discussion [4-6]	Diagram Present, very good control elements in discussion. [7-8]	Diagram Present, excellent discussion using control terminology and clearly understood. [9-10]
E1					
E2					
E3					
E4a					
E4b					
E5					
	Design Questions				
	Working should be clear, solution clearly marked, any assumptions clear [30 Total- 15 Each]				
D1					
D2					
	Presentation [10]				
	Not followed instructions [0]	Presentation generally poor, poor grammar and layout [1-3]	Presentation OK but still errors [4-6]	Presentation good with minimal grammar and language errors [7-9]	Excellent layout, sections, grammar, and language [10]
	Comments				