ECE-320 Lab 2: Root Locus For Controller Design

In this Lab you will explore the use of the root locus technique in designing controllers. The root locus indicates the possible location of the closed loop poles of a system as a parameter (usually the gain k) varies from small to large values. Often we implement controllers/compensators to put the dominant poles of a system in a location that will produce a more desirable response. This assignment is to be done with Matlab's ***sisotool****.*

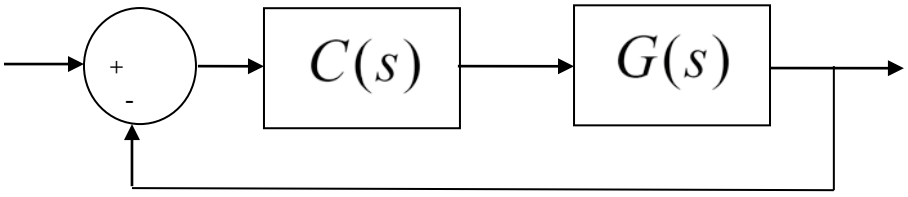
Some things to keep in mind about the root locus:

- *The only possible closed loop poles are on the root locus.*
- *The root locus starts (k = 0) on poles and ends ($k = \infty$) on zeros.*
- *If you click on the root locus plot, it gives you the value of gain k and the location of the closed loop poles on that branch for that value of k. It* ***does not*** *tell you all of the closed loop pole locations if there is more than one branch.*
- ***All of the poles and zeros you add to the system should be in the left half plane.***

Memo *Your (brief) memo for this lab should indicate that you did all of the required work, and that you understand that you are worthless scum and will be royally screwed if you did not. It will be very obvious to me on the next two labs if you did not do this lab (and I will change your grade accordingly). Be creative in your memo.*

Controller Configuration

In this lab we will be assuming a unity feedback controller of the following form, where *C(s)* is the controller and *G(s)* is the plant (this is the notation **sisotool** uses).



Common Controller Types

Proportional (P) Controller: $C(s) = k_p = k$

Integral (I) Controller: $C(s) = \frac{k_i}{s} = \frac{k}{s}$

Proportional+Integral (PI) Controller: $C(s) = k_p + \frac{k_i}{s} = \frac{k(s+z)}{s}$

Proportional+Derivative (PD) Controller: $C(s) = k_p + k_d s = k(s+z)$

Proportional+Integral+Derivative (PID) Controller:

$$C(s) = k_p + \frac{k_i}{s} + k_d s = \frac{k(s+z)(s+z^*)}{s} = \frac{k(s+z_1)(s+z_2)}{s}$$

For the PID controller, we can have either two complex conjugate zeros or two real zeros.

*In **sisotool**, you will see the transfer functions form of the controllers. To determine the parameters* k_p, k_i, *and* k_d *you need to equate powers of s of the transfer functions with the forms above. It is easiest if you use the zero/pole/gain format for the compensators. To do this click on **Edit → SISO Tool Preferences → Options** and click on **zero/pole/gain**.*

Note that if the system is stable, the I, PI, and PID controllers will produce a steady state error of zero unless the system being controlled has a zero at the origin which would cancel with the controller's pole at the origin.

*Introduction to **sisotool***

Getting Started

- Enter the transfer function for the plant, *G(s)*, in your workspace.
- Type **sisotool** in the command window
- Click **close** when the help window comes up
- Click on **Vew**, the **Design Plots Configuration**, and turn off all plots except the **Root Locus** plot (set the **Plot Type** to **Root Locus** for **Plot 1** and set the **Plot Type** to **None** for all other plots)

Loading the Transfer Function

- In the sisotool **Design Windo**w, Click on **File → Import.**
- We will usually be assigning *G(s)* to block *G* (the plant). Under the **System** heading, click on the line that indicates **G**, then click on **Browse**.
- Choose the available Model that you want assigned to G (click on the appropriate line) and then click on **Import** and then on **Close**.
- Click **OK** on the System Data (Import Model) window

- Once the transfer function has been entered, the root locus is displayed, make sure the poles and zeros of your plant are where you think they should be.

Generating the Step Response

- Click on **Analysis → Response to Step Command**
- You will probably have two curves on your step response plot. To just get the output, type **Analysis → Other Loop Responses**. If you only want the output, then only *r* to *y* is checked. However, sometimes you will also want the *r* to *u* output, since it shows the control effort for P, I, and PI controllers.
- You can move the location of the pole in the root locus plot by putting the cursor over the pink button and holding the left button down as you move the pole locations. You should note that the step response changes as the pole locations change.
- The bottom of the root locus window will show you the closed loop poles corresponding to the cursor location (you need to click the left button). However, if you need all of the closed loop poles you have to look at all of the branches.

Entering a Compensator (Controller)

- Click on **Designs →Edit Compensator**
- Right click in the **Dynamics** window to enter real poles and zeros. You will be able to change these values very easily later.
- You can either edit the pole/zero locations in this window, or by grabbing the poles and zeros in the root locus plot and moving them. However, sometimes you just have to go back to this window.
- Look at the form of *C* to be sure it's what you intended, and then look at the root locus with the compensator.
- You can again see how the step response changes with the compensator by moving the locations of the poles (grab the pink dot and slide it).
- You can also change the location of the pole and zeros of the compensator by grabbing them and sliding them. Be careful not to change the poles and zeros of the plant though!

Adding Constraints

- Right Click on the root locus plot, and choose **Design Requirements** then either **New** to add new constraints, or **Edit** to edit existing constraints.
- At this point you have a choice of various types of constraints.
- ***Remember these constraints are only exact for ideal second order systems!!!***

Printing/Saving the Figures:

To save a figure **sisotool** has created, click **File → Print to Figure**

Odds and Ends :

You may want to fix the axes. To do this,

- Right click on the root locus plot
- Choose **Properties**
- Choose **Limits**
- Set the limits and turn the **Auto Scale** off

You may also want to put on a grid, as another method of checking your answers. To do this, right click on the Root Locus plot, then choose **Grid**

It is easiest if you use the zero/pole/gain format for the compensators. To do this click on **Edit → SISO Tool Preferences → Options** and click on **zero/pole/gain**.

Part A

Let's assume the plant we are trying to control has the transfer function

$$G(s) = \frac{30}{s^2 + 11s + 30} = \frac{30}{(s+5)(s+6)}$$

This is second order system with two real poles, located at -5 and -6. Our general goal will be to speed up the response of the system and produce a system with a steady state error for a unit step input of 0.1 or less, a percent overshoot of 10% or less, and a settling time of less than 1 second. To keep things reasonable, assume we want $k \leq 10$ for all designs, *however the values of* k_p, k_i, and k_d *may individually be larger than 10.*

You need to use the zero/pole/gain format for the compensators. To do this click on ***Edit → SISO Tool Preferences → Options*** *and click on* ***zero/pole/gain****.*

Step 1: Entering the Constraints

Enter the percent overshoot and settling time constraints into **sisotool**. Remember, these are only *guidelines*.

Step 2: Proportional (P) Control

a) Determine the root locus of this system with proportional gain. (This is the default for **sisotool.**)
b) Look at the step response as the gain increases. You should notice a few things:

- as *k* increases, the imaginary part of the closed loop poles increases, thus the PO (Percent Overshoot) increases
- as *k* increases, the steady state error decreases

- since the real part of the pole does not change once *k* is greater than about 0.008, the settling time remains fairly constant at about 0.8 seconds.
- there is no value of *k* for which the system is unstable

Do as well as you can to meet both constraints.

Step 3: Integral (I) Control

a) Look at the root locus for the system with an integral controller. You will need to click on **Designs → Edit Compensators**. Then you need to right click in the left box and place the pole at zero. Note that once you have placed the compensator poles and zeros, you can click and drag them. However, *for an integral controller the pole is always at zero*.

b) For what value of *k* will the system become unstable?

c) Try and find a value of *k* that gives fast response with little overshoot, with a settling time less than or equal to 2 seconds.

d) Is steady state error zero? Is the system response very fast? Can you meet the (1 second) settling time requirement?

Step 4: Proportional+Derivative (PD) Control

a) Look at the root locus for the system with a PD controller. You will need to click on **Designs →Edit Compensators**. You need to **delete** the pole from the I controller and add a real zero. Note that for this type of design the zero can be moved.

b) Find a controller that produces a settling time of 0.1 seconds (or less) and a steady state error of 0.1 or less. (Remember $k \leq 10$).

Step 5: Proportional+Integral (PI) Control

a) Look at the root locus for the system with a PI controller. You will need to click on **Designs →Edit Compensators**. Then you need to **add real pole at the origin**. *For a PI controller, one pole is always fixed at the origin.* The zero of the compensator can move, but the pole cannot.

b) Look at the root locus and step response for the system with the zero of the PI controller between 0 and -5. Find a controller that produces a settling time of less than 0.8 seconds, a percent overshoot less than 2%, and a steady state error of 0. Are all of your poles in the ``acceptable'' regions? *Hint: you may want to scale the axes.*

Step 6: Proportional+Integral+Derivative (PID) Control

a) Place the zeros of the controller at $-7 \pm 7j$ and plot the root locus.

b) Find a value of *k* (on this root locus) so that the step response of the system has a PO less than 10% and a settling time less than 0.5 seconds. Are your poles and zero's within the ``acceptable'' region?

c) Keeping the real part of the zero at -7, reduce the imaginary part of the zero as much as possible while keeping the same basic shape of the root locus. At some point, the root locus will take on a very different shape. Find a value of *k* (on the root locus) so that the step response of the system has a PO less than 2%, a settling time less than 0.02 seconds, and a steady state error of less than 0.01. Save the step response, the root locus (zoom in near the origin so you can see something) and the values of k_p, k_i, and k_d that produced the step response for your memo (Remember $k \le 10$) Are your poles and zero's within the ``acceptable'' region?

d) Assume we want a PID controller with real zeros at -7 and -8. Determine the root locus of this system. Find a value of *k* so that the PO is less than 2% and the settling time is less than 0.02 seconds. Are your poles and zero's within the ``acceptable'' region?

Part B

Let's assume the plant has the transfer function

$$G(s) = \frac{5050}{s^2 + 2.647s + 174.8}$$

This is a model I obtained from one of the rectilinear systems in the controls lab.

You need to use the zero/pole/gain format for the compensators. To do this click on ***Edit → SISO Tool Preferences → Options*** *and click on* ***zero/pole/gain***.

You should try and meet the following constraints

$$\begin{aligned} e_{ss} &\le 0.1 \\ T_s &\le 0.5\,\text{sec} \\ P.O. &\le 25\% \end{aligned}$$

You should do your best to meet each one of these, but in any event you must have

$$\begin{aligned} k_p &\le 0.5 \\ k_i &\le 5 \\ k_d &\le 0.01 \end{aligned}$$

In addition, for the I and PI controllers you must have $|u(t)| < 0.45$. *Your step response graphs should also show the control effort for these two types of controllers!*

You need to try to meet these design constraints for a

- I controller (hard to meet settling time, probably need $T_s \approx 4\,\text{sec}$)
- PD controller
- PI controller (hard to meet settling time, probably need $T_s \approx 3.5\,\text{sec}$)
- PID controller with real zeros
- PID controller with complex conjugate zeros

Part C

Let's assume the plant has the transfer function

$$G(s) = \frac{938.4}{s^2 + 1.25s + 329.8}$$

This is a model I obtained from one of the torsional systems in the controls lab.

You need to use the zero/pole/gain format for the compensators. To do this click on ***Edit → SISO Tool Preferences → Options*** *and click on* ***zero/pole/gain***.

You should try and meet the following constraints

$$\begin{aligned} e_{ss} &\leq 0.1 \\ T_s &\leq 1.0\,\text{sec} \\ P.O. &\leq 25\% \end{aligned}$$

You should do your best to meet each one of these, but in any event you must have

$$\begin{aligned} k_p &\leq 0.5 \\ k_i &\leq 5 \\ k_d &\leq 0.01 \end{aligned}$$

In addition, for the I and PI controllers you must have $|u(t)| < 0.39$. *Your step response graphs should also show the control effort for these two types of controllers!*

You need to try to meet these design constraints for a

- I controller (hard to meet settling time, probably need $T_s \approx 4\,\text{sec}$)
- PD controller (probably won't work, do the best you can)
- PI controller (*really har*d to meet settling time, probably need $T_s \approx 10\,\text{sec}$)
- PID controller with real zeros
- PID controller with complex conjugate zeros