ECE-320 Lab 1: 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. This assignment is to be done with Matlab's ***sisotool****.(siso comes from* ***S****ingle* ***I****nput* ***S****ingle* ***O****utput)*

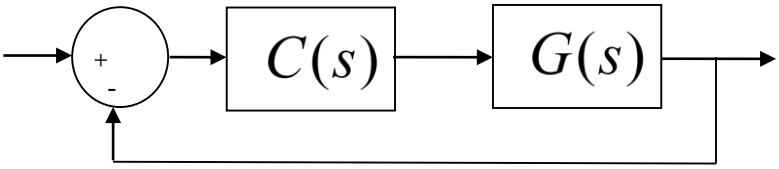
Some things to keep in mind about the root locus:

- *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.*
- ***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 just be a statement that you did all of the required work.*

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:

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

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.*

Note that if the closed loop system is stable, the I, PI, and PID controllers will produce a steady state error of zero for a step input unless the plant being controlled has a zero at the origin.

A sisotool summary is at the end of this lab.

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.

Getting Started

- Enter the transfer function for the plant, *G(s)*, in your workspace.
- Type **sisotool** in the Matlab 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)
- It is easiest if you use the zero/pole/gain format for the compensators. To do this, in the **SISO Design** window click on **Edit → SISO Tool Preferences → Options** and click on **zero/pole/gain**.

Loading the Transfer Function

- In the **SISO Design** window, 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 (for a proportional controller) is displayed. Make sure the poles and zeros of your plant are where you think they should be.

Generating the Step Response

- In the **SISO Design** window, click on **Analysis → Response to Step Command** (the system may take a while to respond)
- You will probably have two curves on your step response plot. To just get the output, in the **SISO Design** window select **Analysis → Other Loop Responses**. If you only want the output, then only *r* to *y* should checked. However, sometimes you will also want the *r* to *u* output, since it shows the control effort for P, I, and PI controllers.
- In the **SISO Design** window 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.
- Note that the assumed input for *sisotool* is always 1, so the **steady state error** is one minus the steady state value of the output.

Adding Constraints in the s-plane

- In the **SISO Desig**n window, 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. Enter the settling time constraint (less than 1 second), and then enter the percent overshoot constraint (less than 10%). You will not be able to enter the steady state error constraint, because that is not really controlled by pole locations in the s-plane.
- To meet the constraints, you want the poles to stay within the white area and avoid the shaded (yellow) area.
- ***Remember these constraints are only exact for ideal second order systems!!!***

Adding Constraints in the Step Response Graph

- Right click in the **LTI Viewer** window
- There are a number of ways to add constraints to the output. Usually the best is to select **Characteristics**, and then choose that you want. Each of the characteristics will cause a blue dot to appear on the output screen. If you hover over the blue dot you can find the corresponding values.
- Note that the assumed input for *sisotool* is always 1, so the steady state error is one minus the steady state value of the output.

Proportional (P) Control (This is the default for sisotool)

We will now look at the root locus plot for this system with a proportional controller. This is the default for *sisotool* and is what we are currently looking at. If you want to know the value of the proportional gain (k_p) you can do this in one of two ways:

- In the **SISO Design** window, click on **Designs → Edit Compensators**
- In the **Control and Estimation Tools Manager,** click on **Compensator Editor**

In this window, the controller is written as $C(s) = k_p$ **,** and you should be able to read off the value of the proportional gain.

Next**,** look at the step response as the gain varies (sliding the pink box on the root locus plot is the easiest way to do this). 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
- there is no value of *k* for which the system is unstable

Integral (I) Control

- In the **Control and Estimation** window, right click in the **Dynamics** window to enter and remove poles and zeros. You will be able to change these values very easily later. At this point just *add an integrator*.
- In this window, the controller is written as $C(s) = k_i \times \frac{1}{s}$ and you should be able to read off the value of the integral gain.
- 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).

Next look at the step response as the value of the integral gain varies. You should notice a few things

- There are some values of k_i ($k_i > 11$ or so) for which the system is unstable. To find this value of the gain, you may have to enter the values directly into the gain box (this is how you can enter precise values)
- The system is much slower now than with the proportional control (the settling time is generally larger)
- The steady state error for a step input is now zero.

Proportional+Integral (PI) Control

The form of a PI controller is $C(s) = \frac{k(s+z)}{s}$. Since we already have the $\frac{1}{s}$ part we need to add a real zero. *For a PI controller, one pole is always fixed at the origin.* The zero of the compensator can move, but the pole cannot. Sisotool will, in fact, let you move the pole, but you will no longer have a PI controller.

Right click in the **Dynamics** window and *add a real zero*. Note that the location of the zero will probably default to -1 (or somewhere else). We can move it later so do not worry about it.

In the **Dynamics** window, left click on the **Real Zero** box and a new **Location** box will pop up. You can enter the value you want for the real zero in this box and then hit **Enter**. The new location should show up in the Dynamics box.

In the **SISO Design** window, you can now grab the real zero and move it as well as grabbing the pick squares and move them (corresponding to changing the gain). You should play around with this a bit while looking at how the step response is changing. See if you can get a settling time of less than one second and a percent overshoot of less than 10% while keeping $z < 5$. **Note that you should be looking at the step response to determine this, *not* the root locus plot!**

Since we have a type 1 system, the steady state error is zero.

Proportional+Derivative (PD) Control

The form of a PD controller is $C(s) = k(s+z)$. To get this form from our integral controller, we need to remove the pole at the origin. In the **Dynamics** window, right click on the **Integrator** line, and select **Delete Pole/Zero**. Look at the form of the controller and it should look like a PD controller. We no longer have a type 1 system, so the steady state error is no longer zero.

Set the zero at -1 and the gain to 4. The **SISO Design** window shows the root locus plot that is pretty hard to see much, since it is also showing a pole near -130. To zoom in near zero, in the *white space* in the root locus plot right click the mouse and select **Properties**. Then select **Limits,** uncheck the **Auto-Scale** boxes**,** and set the **Real Axis** limits from -10 to 0, and the **Imaginary Axis** limits from -1.0 to 1.0, and then select **Close**. Try to adjust the gain and the zero so the steady state error is less than 0.1 and the percent overshoot is less than 10% while keeping the zero between 0 and -5.

Next, wet the zero to something less than -6 and you will see a very different root locus plot. (You may have to right click in the white area, select **Properties**, then **Limits**, and then choose **Auto-Scale**.) It should be much easier to meet all of the design requirements now (steady state error less than 0.1, PO less than 10%, and a settling time less than 1 second.) However, this type of controller generally requires a large control effort.

Proportional+Integral+Derivative (PID) Control

There are two general forms for a PID controller, depending on whether the two zeros are real or complex conjugate pairs, $C(s) = \frac{k(s+z_1)(s+z_2)}{s} = \frac{k(s+z)(s+z^*)}{s}$

Note that for a PID controller you are free to move the zeros, but the pole must remain at the origin. *Sisotool* will allow you to move the pole, *but you will no longer have a PID controller and you will not have a type 1 system.*

PID control with real zeros

In the **Dynamics** box, add an integrator and another zero.

Modify your controller so you meet all of the constraints as well as keeping $k_p \le 5, k_i \le 5, k_d \le 0.2$

PID control with complex conjugate zeros

In the **Dynamics** box, remove the real zeros and add complex conjugate zeros.

Modify your controller so you meet all of the constraints as well as keeping $k_p \le 5, k_i \le 5, k_d \le 0.2$

Printing/Saving the Figures:

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

Part B

Let's assume the plant has the transfer function

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

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

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

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 $|u(t)| < 0.45$. *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

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

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 hard* to meet settling time, probably need $T_s \approx 10\,\text{sec}$)
- PID controller with real zeros
- PID controller with complex conjugate zeros

sisotool summary

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**