Table of Contents

Challenge

In this notebook, your challenge is to expand your tool set for sketching root locus plots to answer two more important questions about the behavior of our system.

1. **HOW** does a root locus branch leave a complex pole of $G(s)H(s)$?
2. **HOW** does a root locus arrive at a complex zero of $G(s)H(s)$?

Answers to these questions used to be important for engineers who needed to sketch root locus plots by hand. Today, with MATLAB and other software packages that can produce accurate plots quickly, they're less important. *HOWEVER*, learning to compute "angles of departure and arrival" to and from complex zeros and poles of $G(s)H(s)$ is a way for us to explore a *graphical* representation of the angle criterion. Getting a good feel for the angle criterion in graphical form will be *intrinsic value* as we move on to learning to *design* controllers with the root locus. It will help us understand how we can place controller zeros to ensure that our system's root locus will pass through a desired point on the S-plane.

IMPORTANT: This notebook provides an interactive plot as a tool to help you build understanding. This plot is coded in the Python language, so in order to use Octave language in this notebook, we need a special `magic` command at the top of each code cell. To enable this, you MUST run the cell below before attempting today's assignments. In the assignments themselves, `%%octave` must be at the top of each code cell so Jupyter knows to interpret your code as Octave, rather than Python.

```
In [8]:  %load_ext oct2py.ipython
```

```
The oct2py.ipython extension is already loaded. To reload it, use:
  %reload_ext oct2py.ipython
```

Controller Design: Steps 7-8 of the root locus sketch

The last two steps of the root locus sketching procedure as outlined in the "Root Locus Overview" notebook, steps 7 and 8, read:

- (7) Calculate the angles of arrival to any complex zeros and the angles of departure from any complex poles
- (8) Confirm that you have the correct number of branches on your root locus plot

The last step, step 8, is fairly self-explanatory. Step 7 is not. To get a feel for how to calculate "angles of arrival" and "angles of departure" to/from complex zeros & poles, respectively, let's take a step back and think about what it really means for us to calculate the angle criterion for the root locus:

$$ \angle(G(s)H(s)) = \angle \left(\frac{-1}{K}\right) = \pm 180^\circ$$
.

A graphical representation of the angle criterion

What we've been doing so far is saying that if the "open loop transfer function" can be written as $G(s)H(s)=\frac{N(s)}{D(s)}$, where $N(s)$ and $D(s)$ are the numerator and denominator polynomials, respectively, we can write the total angle of the transfer function $G(s)H(s)$ as:

$$\angle(G(s)H(s)) = \angle(N(s)) - \angle(D(s))$$

When we have had a "trial" eigenvalue s_1 to evaluate, to determine if it is or is not on the root locus, we have been evaluating:

$$\left.\angle(G(s)H(s))\right|_{s=s_1} = \left.\angle(N(s))\right|_{s=s_1} - \left. \angle(D(s))\right|_{s=s_1}$$

Which is computed by:

$$\angle(N(s))-\angle(D(s)) = \tan^{-1}\left(\frac{\left.Imag(N(s)\right|_{s=s_1})}{\left.Real(N(s)\right|_{s=s_1})}\right)-\tan^{-1}\left(\frac{\left.Imag(D(s)\right|_{s=s_1})}{\left.Real(D(s)\right|_{s=s_1})}\right)$$

However, if we consider a factored version of our "open loop transfer function," it will take the form:

$$G(s)H(s) =\frac{N(s)}{D(s)}= A\frac{(s+z_1)(s+z_2)\ldots(s+z_m)}{(s+p_1)(s+p_2)\ldots(s+p_n)}$$

Which has m zeros at $s=-z_1,s=-z_2,\ldots,s=-z_m$, and n poles at $s=-p_1,s=-p_2,\ldots,s=-p_n$, any of which might be complex, we can break down this calculation even further. Based on the complex exponential form of a complex number:

$$ s = \left|s\right|e^{j\angle(s)}$$

And the fact that $e^{a}e^{b}=e^{a+b}$, we know that multiplying two complex numbers s_1 and s_2 can be written as:

$$s_1s_2 = \left|s_1\right|\left|s_2\right|e^{j\left(\angle s_1+\angle s_2\right)}$$

With this in mind, we can treat each factored zero in $N(s)$ and each factored pole in $D(s)$ as separate complex numbers, and evaluate each at our trial point $s=s_1$, we find that:

$$\left.G(s)H(s)\right|_{s=s_1} = \left|s_1+z_1\right|\left|s_1+z_2\right|\ldots\left|s_m+z_m\right|e^{j(\angle(s_1+z_1)+\angle(s_1+z_2)+\ldots+\angle(s_1+z_m))} - \left|s_1+p_1\right|\left|s_1+p_2\right|\ldots\left|s_n+p_n\right|e^{j(\angle(s_1+p_1)+\angle(s_1+p_2)+\ldots+\angle(s_1+p_n))}$$

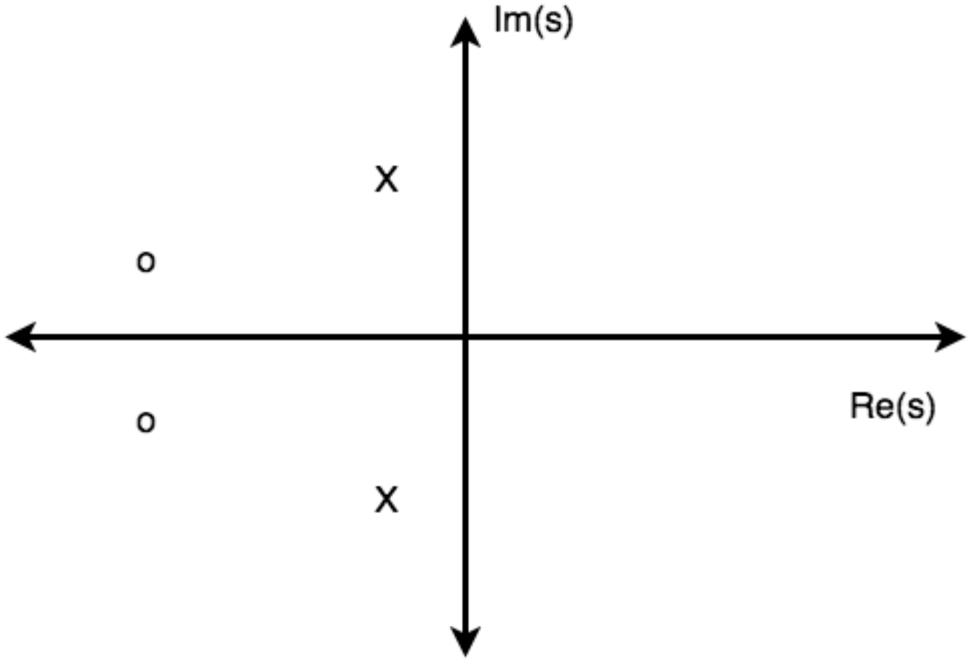
This may look like a bit of a mess, but it's actually very useful. It allows us to rewrite the angle criterion as:

$$\angle \left.G(s)H(s)\right|_{s=s_1} = (\angle(s_1+z_1)+\angle(s_1+z_2)+\ldots+\angle(s_1+z_m)) - (\angle(s_1+p_1)+\angle(s_1+p_2)+\ldots+\angle(s_1+p_n))=\pm 180 ^\circ$$

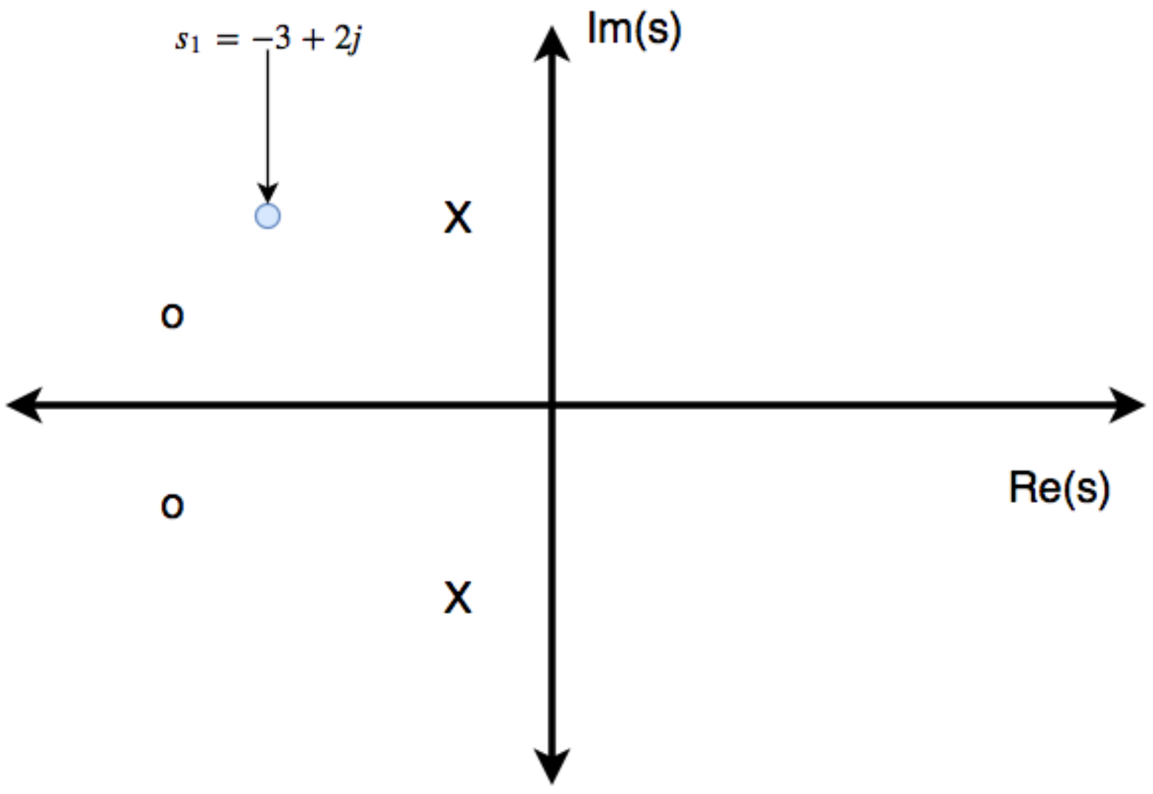
The usefulness of this form of the angle criterion comes from the fact that each of these individual "pieces" of the overall transfer function evaluated at some trial point $s=s_1$

represents something very special graphically. To see how, let's look at the following pole-zero plot for an "open loop transfer function,"

$$G(s)H(s) = \frac{(s+z_1)(s+z_2)}{(s+p_1)(s+p_2)}=\frac{(s+4-j)(s+4+j)}{(s+1-2j)(s+1+2j)}$$



If we'd like to check whether a point $s_1 = -3+2j$ is on the root locus for this system under feedback control, we'll need to apply the angle criterion, as usual. Let's plot our trial point on the s-plane with the poles and zeros of $G(s)H(s)$.

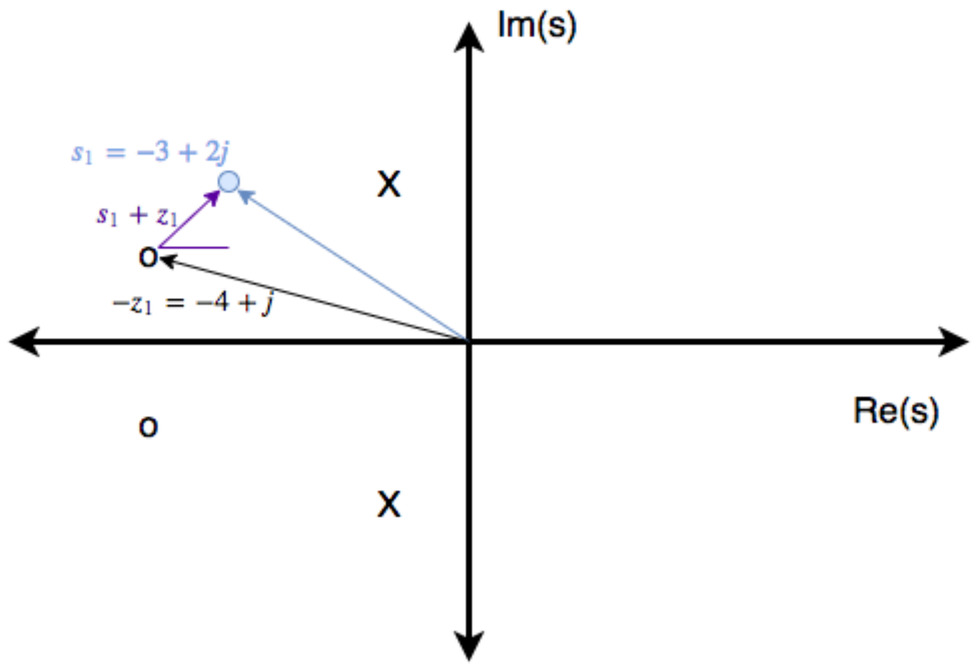


Next, let's look at the very first term in the numerator of the open loop transfer function $G(s)H(s)$,

$$(s+z_1)=(s+4-j)$$
and see what this represents, graphically, on the pole-zero plot. If we evaluate $(s+4-j)$ at our "trial point" $s=s_1=-3+2j$, this term turns into

$$(s_1+z_1) = (-3+2j+4-j) = (1+j)$$
.

Referring to the figure below, if we draw and arrow to point s_1 (blue vector) and an arrow to point $-z_1$ (black vector), it can be scene that $(s_1+z_1) = (-3+2j+4-j) = (1+j)$ is **the arrow pointing from the zero to our trial point!**. This is the result of simple vector addition as shown in the figure below!



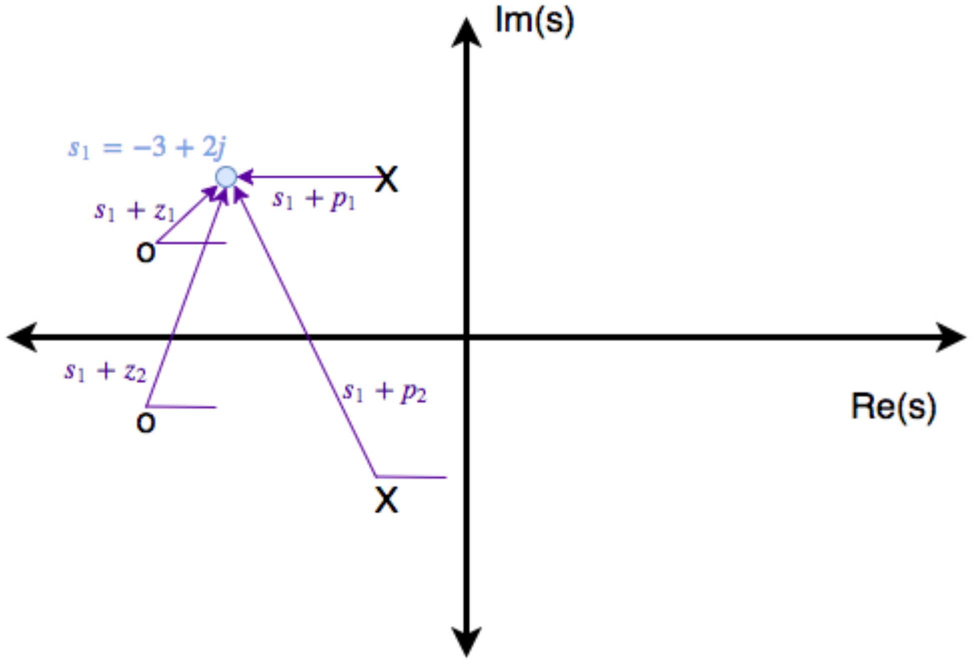
NOTE: Something that is initially confusing at this point in learning this technique is that on the pole-zero plot we *plot* the zero at $s=-z_1$. However, we *evaluate* $(s+z_1)$ in the transfer function.

You can think of the operation on the pole-zero plot to get the complex number s_1+z_1 is actually *subtracting* the black vector $s=-z_1$ from the blue vector s_1 using "tip to tip" vector subtraction. **If this does not make sense to you, seek some guidance! This is a very important concept to get nailed down**.

In this example, z_1 is the **negative** of the zero itself, so $z_1 = 4-j$. Therefore, the purple arrow, s_1+z_1, is the number:

$$s_1+z_1 = (-3+2j)+(4-j) = 1+j$$
And this number, when shown with its tail starting at our zero at $-z_1$, points directly at our trial point.

Carrying this concept forward, if we evaluate each of the components in $G\left.(s)H(s)\right|_{s=s_1}$, we can draw each of these points as arrows on the complex plane **from each of the poles and zeros of** $G(s)H(s)$ **to our trial point**. This looks more or less like this:



Then, checking the angle criterion of the root locus for this point $s_1 = -3+2j$ is as simple as saying:

$$\angle(\left.G(s)H(s)\right|_{s=s_1}) = \sum \angle(zeros(G(s)H(s))\rightarrow s_1) - \sum \angle(poles(G(s)H(s))\rightarrow s_1)=\pm 180^\circ$$
So let's check this trial point for our system using this "new" method (it's still just the angle criterion calculated a different way!). First we need to identify each vector from the poles and zeros of $G(s)H(s)$ to the trial point s_1.

\begin{equation} \left\{ \begin{matrix} (s_1+z_1)&=1+j\\ (s_1+z_2)&=1+3j\\ (s_1+p_1)&=-2\\ (s_1+p_2)&=-2+4j \end{matrix} \right. \end{equation}
Now, we need to calculate the angles of the vector from all of our poles and zeros of $G(s)H(s)$ to the trial point.

\begin{equation} \left\{ \begin{matrix} \angle(s_1+z_1)&=\tan^{-1}\frac{1}{1}&=\frac{\pi}{4}\\ \angle(s_1+z_2)&=\tan^{-1}\frac{3}{1}&=1.2490\\ \angle(s_1+p_1)&=\tan^{-1}\frac{0}{-2}&=\pi\\ \angle(s_1+p_2)&=\tan^{-1}\frac{4}{-2}&=2.0344 \end{matrix} \right. \end{equation}
Finally, to check to see whether s_1 is on the root locus for $G(s)H(s)$, we merely apply the formula we developed above. If our point s_1 is on the root locus,

$$\angle(\left.G(s)H(s)\right|_{s=s_1}) = \sum \angle(zeros(G(s)H(s))\rightarrow s_1) - \sum \angle(poles(G(s)H(s))\rightarrow s_1)=\pm 180^\circ$$
In words, we can write this form of the angle criterion as follows:

If a trial point s_1 **is on the root locus, the sum of all of the angles of the arrows from the zeros of** $G(s)H(s)$ **to the trial point minus the sum of all of the angles from the poles of** $G(s)H(s)$ **to the trial point will be** $\pm 180^\circ$.

For this example, this means that:

$$\angle(\left.G(s)H(s)\right|_{s=s_1}) = \frac{\pi}{4}+1.2490-\pi-2.0344 = -3.1416\approx -\pi$$
So our point $s_1=-3+2j$ **is** on the root locus! Could we have determined this without calculating all of these angles individually (in other words, by doing things the way we did them earlier this week)? Yes, **BUT** this new technique will help us with our next task, which is determining angles of departure from complex poles and angles of arrival to complex zeros on the root locus.

Example: Graphical representation of the angle criterion

Before we move on, take a look at the interactive plot below, which shows you the concept we've just been discussing for a different system. On the left, you can see the system's open loop poles with the root locus plot and a trial point that you can move around. On the right, you'll see the individual angles from each pole to the trial point, along with the "total angle" of the transfer function $G(s)H(s)$. For the demonstration, the transfer function is of the form:

$$G(s)H(s) = \frac{\omega_{n1}^2\omega_{n2}^2}{(s^2+2\zeta_1\omega_{n1}s+\omega_{n1}^2)(s^2+2\zeta_2\omega_{n2}s+\omega_{n2}^2)}$$
The sliders on the plot give you control over where each pole pair is, along with where your trial point is. Take particular note of how the "angles of departure" from the open loop poles are affected by where the poles of $G(s)H(s)$ are on the s-plane to prime you for the next part of this reading. Play around with these sliders so that you get a trial point that IS on the root locus, and watch how the individual angles change. In this example, there are no zeros, so the total angle of $G(s)H(s)$ is equal to $\angle G(s)H(s) = 0-\sum(\angle(poles\rightarrow trial))$.

```
from __future__ import print_function
from ipywidgets import interact, interactive, fixed, interact_manual
import ipywidgets as widgets

from matplotlib.pyplot import *
from matplotlib.patches import Arc,Arrow
from scipy.signal import lti,step
from numpy import *
from numpy.lib import scimath
from control import rlocus

#this defines a python function that plots our system poles as we move K, a
def updatePlot(trialpt_wn,trialpt_zeta,w1,z1,w2,z2):
    #two poles (fourth order system)
    # (s^2+2*z1*w1*s+w1^2)*(s^2+2*z2*w2*s+w2^2)
    #s^4 + 2*z1*s^3*w1 + 2*z2*s^3*w2 + s^2*w1^2 + 4*z1*z2*s^2*w1*w2 + s^2*w
    #set up our open loop system as a transfer function
    sys_openloop = lti([w1**2*w2**2],[1, (2*z2*w2+2*z1*w1), (w1**2+w2**2+4*

    #now we want to get the eigenvalues of the open loop system and plot the
    eigs_openloop = roots([1, (2*z2*w2+2*z1*w1), (w1**2+w2**2+4*z1*z2*w1*w2)
    realparts_ol = real(eigs_openloop)
    imagparts_ol = imag(eigs_openloop)

    #now we want to get the coordinates of the trial point so we can plot i
    eig1_trialpt  = 0j+-trialpt_wn*trialpt_zeta + scimath.sqrt((2*trialpt_ze
    realparts_trial = real(eig1_trialpt)
    imagparts_trial = imag(eig1_trialpt)

    #now calculate the angles of the vectors from each pole of GH to the tr
    angle_p1t = arctan2(imagparts_trial-imagparts_ol[0],realparts_trial-real
    angle_p2t = arctan2(imagparts_trial-imagparts_ol[1],realparts_trial-real
    angle_p3t = arctan2(imagparts_trial-imagparts_ol[2],realparts_trial-real
    angle_p4t = arctan2(imagparts_trial-imagparts_ol[3],realparts_trial-real

    #now calculate the total angle of GH = sum(angles from zeros to trial p
    angle_GH_total = 0 - angle_p1t -angle_p2t - angle_p3t - angle_p4t

    #now we plot the points
    figure(figsize=(15,5))
    subplot(1,2,1)
    title('Root Locus For System and Trial Point at s='+"{0:.2f}".format(rea
    [rlist,klist]= rlocus(sys_openloop,kvect = linspace(0,200,1000),Plot=Fal
    plot(real(rlist[:,0]),imag(rlist[:,0]),real(rlist[:,1]),imag(rlist[:,1])
    plot(realparts_ol,imagparts_ol,'kx',ms=15,mew=3)
    plot(realparts_trial,imagparts_trial,'k.',ms=15,mew=2)
    plot([realparts_ol[0], realparts_trial],[imagparts_ol[0],imagparts_trial
    plot([realparts_ol[1], realparts_trial],[imagparts_ol[1],imagparts_trial
    plot([realparts_ol[2], realparts_trial],[imagparts_ol[2],imagparts_trial
    plot([realparts_ol[3], realparts_trial],[imagparts_ol[3],imagparts_trial
    xlabel('Real Part')
    ylabel('Imaginary Part')

    axis('equal')
    xlabel('')
    grid('on','both')

    #now plot the angles and their sum
    ax2=subplot(1,2,2)
    ax2.plot([0,cos(angle_p1t)],[0,sin(angle_p1t)],color='orange',linestyle='
    ax2.plot([0,cos(angle_p2t)],[0,sin(angle_p2t)],color='blue',linestyle='-.
    ax2.plot([0,cos(angle_p3t)],[0,sin(angle_p3t)],color='red',linestyle='-.'
    ax2.plot([0,cos(angle_p4t)],[0,sin(angle_p4t)],color='green',linestyle='-
    ax2.plot([0,cos(angle_GH_total)],[0,sin(angle_GH_total)],color='black',li
    ax2.add_patch(Arrow(0,0,1,0,width=.05,edgecolor='black',facecolor='black'
    ax2.add_patch(Arc((0, 0), .25, .25, theta1=0.0, theta2=180/pi*angle_p1t,
    ax2.add_patch(Arc((0, 0), .35, .35, theta1=0.0, theta2=180/pi*angle_p2t,
    ax2.add_patch(Arc((0, 0), .45, .45, theta1=0.0, theta2=180/pi*angle_p3t,
    ax2.add_patch(Arc((0, 0), .55, .55, theta1=0.0, theta2=180/pi*angle_p4t,
    ax2.add_patch(Arc((0, 0), .65, .65, theta1=0.0, theta2=180/pi*angle_GH_to
    title('Angle Criterion Check')
    xlabel('Real axis')
    ylabel('imaginary axis')
    axis('equal')
    legend(['angle from pole 1 to trial: '+"{0:.2f}".format(angle_p1t*180/pi)


    show()

print("TO KEEP THE SIMULATION RUNNING SMOOTHLY, CLICK A DESIRED SLIDER POSITI
interact(updatePlot, trialpt_wn=widgets.FloatSlider(min=1,max=120,step=1,valu
TO KEEP THE SIMULATION RUNNING SMOOTHLY, CLICK A DESIRED SLIDER POSITION
RATHER THAN DRAGGING THE SLIDER
```

Step 7: Angles of Departure and Arrival

It turns out that if we plot the root locus for the example system we were working with In Section 2 of this notebook,

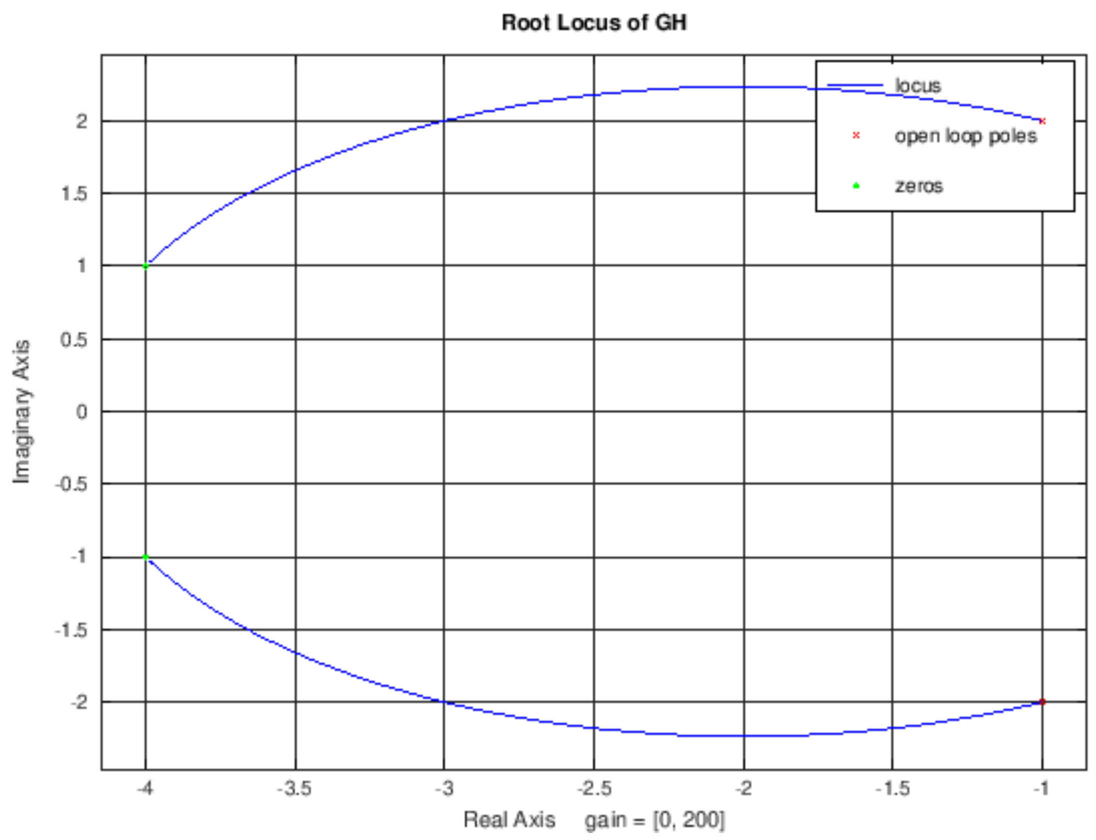
$$G(s)H(s) = \frac{(s+z_1)(s+z_2)}{(s+p_1)(s+p_2)}=\frac{(s+4-j)(s+4+j)}{(s+1-2j)(s+1+2j)}$$
We get the following root locus plot, according to Octave. Make sure to run BOTH of the following cells to see the plot.

```
In [3]:  %load_ext oct2py.ipython
```

```
The oct2py.ipython extension is already loaded. To reload it, use:
  %reload_ext oct2py.ipython
```

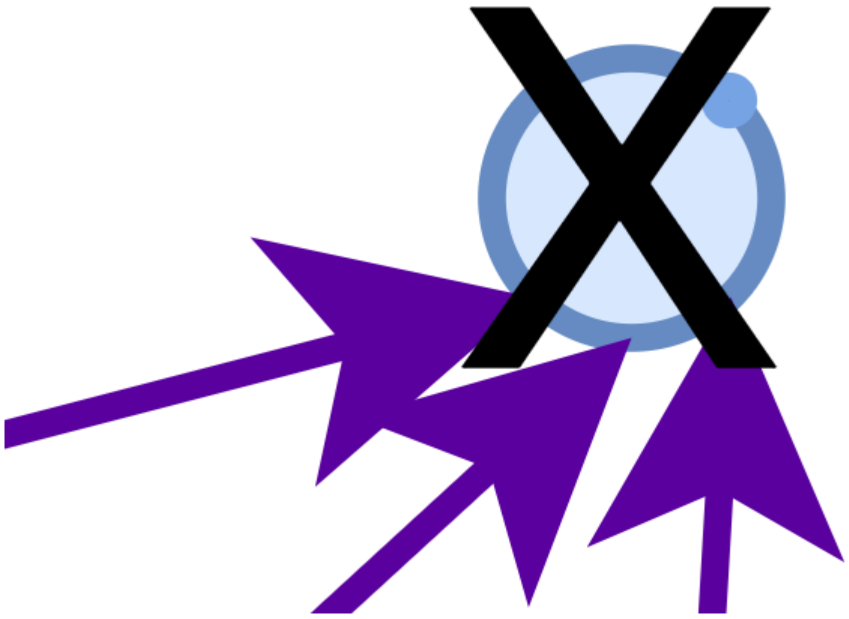
```
In [4]:  %%octave
         s = tf('s');

         GH = (s^2+8*s+17)/(s^2+2*s+5)

         rlocus(GH)
```

```
Transfer function 'GH' from input 'u1' to output ...

      s^2 + 8 s + 17
 y1:  --------------
      s^2 + 2 s + 5

Continuous-time model.
                                       Root Locus of GH

                    |------------------------------------------------------|
                    |  +       +       +#################################-+|
                  2 |-+@@@@@@@@@@########@@@@@@@@@@|***?***locus        #B||
                1.5 |-+@@@@######@@@@@@@@@@@@@@@@@@|   B   open loop poles||
                    |  @####   @       @        @  +---F---zeros----------+|
                  1 |-+F#@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@-|
                    |  @       @       @        @       @       @        @ |
                0.5 |-+@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@-|
                  0 |-+@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@-|
                    |  @       @       @        @       @       @        @ |
               -0.5 |-+@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@-|
                    |  @       @       @        @       @       @        @ |
                 -1 |-+F*@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@-|
                    |  @****   @       @        @       @       @        @ |
               -1.5 |-+@@@@******@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@-|
                 -2 |-+@@@@@@@@@@********@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@*B-|
                    |  +       +       +*********************************+ |
                    |------------------------------------------------------|
                      -4     -3.5     -3      -2.5     -2     -1.5      -1
                                 Real Axis     gain = [0, 200]
```

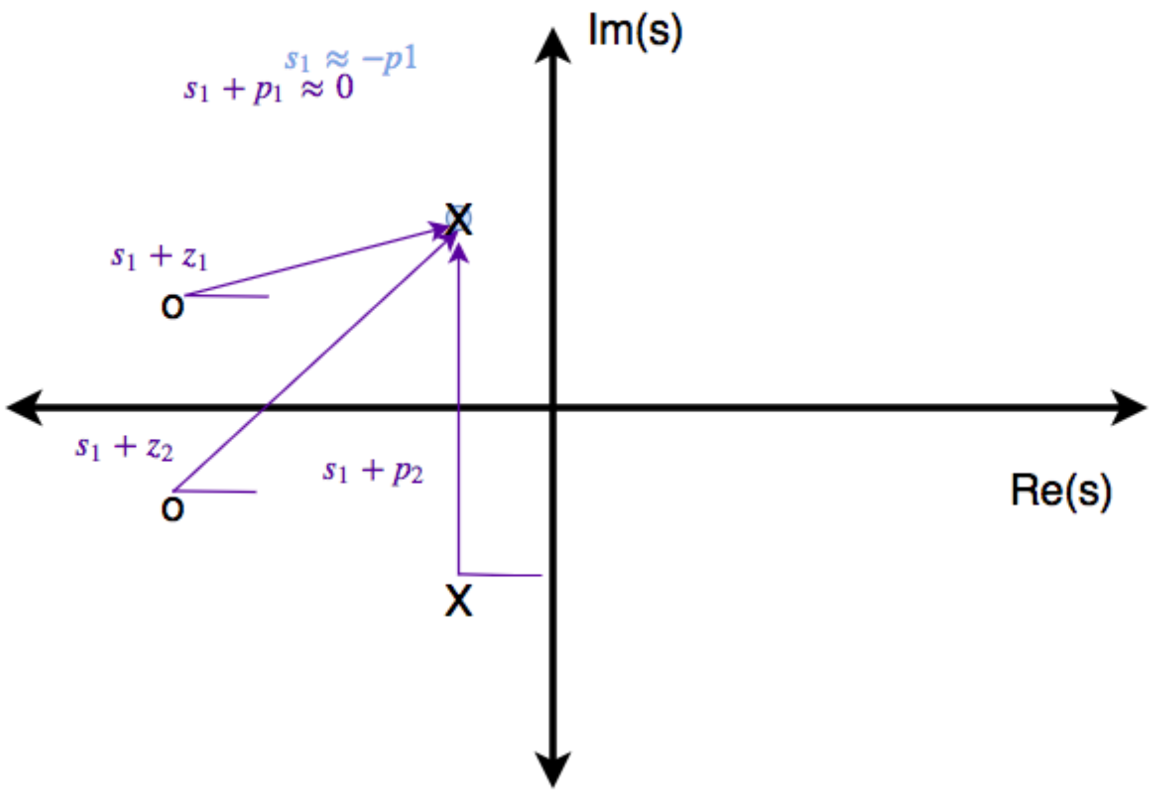


Notice that the two root locus branches leave the poles at $s=-1\pm 2j$ at a particular angle, and arrive at the zeros at $s=-4\pm j$ at a different angle. How do we calculate these angles? The key is in the exercise we just did, where we interpreted the angle criterion graphically.

Specifically, imagine that we have a test point that is **almost exactly** at the pole we're interested in calculating the angle of departure from. We don't know where it is relative to this pole except that it is very close. You could draw the location of this test point s_1 (shown as a small, light blue dot above and to the right of the open loop pole 'X') on a "ring" around our pole of interest, as shown below.



We don't know *exactly where* this trial point is yet, but we know it's on the root locus, and we know that it will have "left" the pole at the *angle of departure* we are looking for. We will call the unknown angle from our departed pole to the test point θ_d. Then, we will calculate the angles of all of the vectors from the other open loop poles and zeros to the pole that our test point just left (because the test point is 'basically at the same spot'), and solve for our unkown angle. This is shown graphically below:



Now, we're ready to enforce the angle criterion, which will tell us what angle the test point makes with the departed pole just as it leaves. To do this, we set up the angle criterion just the way we did in the previous section:

\begin{equation} \left\{ \begin{matrix} \angle(s_1+z_1)&=\tan^{-1}\frac{1}{3}&=0.32175\\ \angle(s_1+z_2)&=\tan^{-1}\frac{3}{3}&=\frac{\pi}{4}\\ \angle(s_1+p_1)&=\tan^{-1}\frac{0}{0} &=\theta_d\\ \angle(s_1+p_2)&=\tan^{-1}\frac{4}{0}&=\frac{\pi}{2} \end{matrix} \right. \end{equation}
Then, if our point \$s_1\$ is on the root locus (which we know it must be),

\$\$\angle(\left.G(s)H(s)\right|_{s=s_1}) = \sum \angle(zeros(G(s)H(s))\rightarrow s_1) - \sum \angle(poles(G(s)H(s))\rightarrow s_1)=\pm 180^\circ\$\$
Substituting in our known values here and solving for \$\theta_d\$ gives:

\$\$0.32175+\frac{\pi}{4}-\theta_d-\frac{\pi}{2}=\pi\$\$
Using Octave to calculate my angle of departure, I get:

```
In [5]:  %load_ext oct2py.ipython
```

```
The oct2py.ipython extension is already loaded. To reload it, use:
  %reload_ext oct2py.ipython
```

```
In [6]:  %%octave
         theta_d_radians = .32175+pi/4-pi/2-pi
         theta_d_degrees = theta_d_radians*180/pi + 360
```

```
theta_d_radians = -3.6052
theta_d_degrees =  153.43
```

So, according to my calculation, the angle of departure is about 153°, which matches what Octave tells us! Note that the angles of departure from complex poles and arrival to complex zeros are **symmetrical about the real axis**, so for this system, we only have to calculate one of each to know everything we need to know about them.

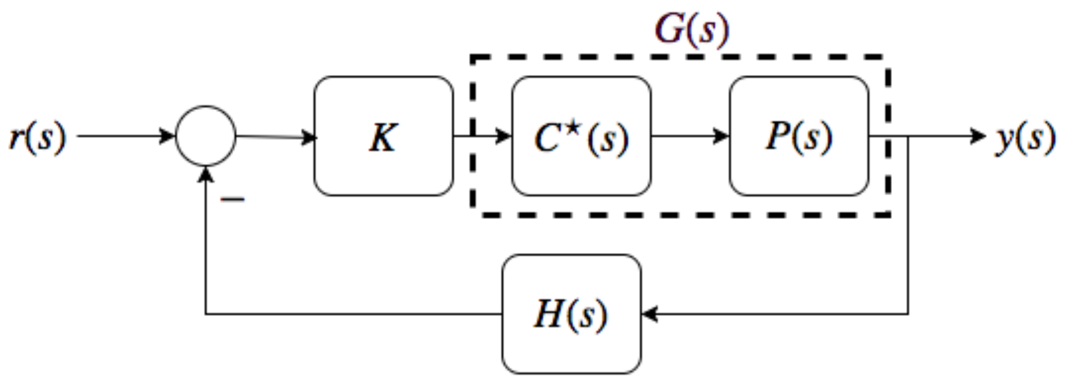
Step 8 - Confirm the correct number of branches

This last step is almost self-explanatory. The number of poles of $G(s)H(s)$ needs to match the number of branches.

Root Locus Summary

The root locus is a plot on the complex plane of how the *closed loop* poles of a feedback control system move around the complex plane when one real, positive parameter K in the forward path changes. Addtionial details of root locus analysis is covered in Chapter 13 of Schaum's Outline of Feedback and Control Systems.

For our discussions, we will assume you can write your system as a feedback control block diagram in the following form:



The root locus for a system of this form has the following important properties:

1. The characteristic equation for this system is $1+KG(s)H(s)=0$, which can be rewritten as $-\frac{1}{K}=G(s)H(s)$.
2. The location of each possible pole is governed by the *angle criterion* for the root locus: $\angle -\frac{1}{K}=\angle G(s)H(s) = \pm 180^\circ$.
3. There are exactly as many **branches** (paths for a closed loop pole to follow) as the number of **poles in G(s)H(s)**.
4. When $K=0$, the closed loop poles lie at **the poles of the 'open loop transfer function' G(s)H(s)** because of the magnitude criterion: $\left|-\frac{1}{K}\right| = \infty = \left|G(s)H(s) \right|$. The only way to make $G(s)H(s)=\infty$ is to set s to one of the poles of $G(s)H(s)$.
5. When $K=\infty$, the closed loop poles go to **either** a **zero** of $G(s)H(s)$ or infinity. This is because of the magnitude criterion. The fraction $\left|-\frac{1}{K}\right|=0= \left| G(s)H(s) \right|$. There are two ways to make $G(s)H(s)=0$. One is that s is one of the **zeros** of G(s)H(s), and the other is that the magnitude of s goes to infinity. Assuming that the transfer function is strictly proper (meaning that at least one branch of the root locus goes to infinity), setting the magnitude of s to infinity will make the transfer function's magnitude go to zero.
6. If a branch goes to infinity, it does so along an *asymptote*.
7. A branch will lie on a portion of the real axis if there is an odd number of poles and zeros of $G(s)H(s)$ to the *right* of that portion. This is required by the angle criterion.
8. The root locus branches can "break away" from and "break in" to the real axis. They always do so perpendicularly to the real axis. This is required by the angle criterion.
9. The root locus is always symmetrical about the real axis. This is required by the angle criterion.

Disciplined Process: Steps 1-8 for sketching the root locus

1. Find the Closed loop characteristic equation for your system: $$1+KG(s)H(s)=0$$
2. Write the Closed loop characteristic equation as $$\frac{-1}{K}=G(s)H(s)$$
3. Plot the poles and zeros of $G(s)H(s)$ on the s-plane. Identify any segments of the R.L. on the real axis: the root locus lies on a portion of the root locus if and only if an *odd* number of "open loop" poles and zeros (poles and zeros of $G(s)H(s)$ are to the *right* of that portion of the real axis.
4. Identify the angle(s) of any asymptotes and the point on the real axis where the asymptotes intersect: $$\sigma_a = \frac{\sum poles -\sum zeros}{n-m}$$ $$\theta_a = \frac{2k+1}{n-m} \pi$$
5. If applicable, calculate any points on the real axis where the root locus "breaks in" to the real axis or "breaks away" from the real axis, using the fact that at such a point, $$\frac{d}{ds}G(s)H(s)=0$$.
6. If you suspect that the root locus crosses into the right half of the s-plane (based on the asymptote angles), calculate the crossing frequency (where the system crosses into unstable territory) and critical gain (gain that makes this happen) using the closed loop characteristic equation. Plug in the point $s=s_{crossing}=j\omega_{crossing}$ into the open loop transfer function $G(s)H(s)$ to find the value of $K=K_{critical}$ that will result in an unstable system.
7. Calculate angles of arrival to complex zeros and angles of departure from complex poles by finding the “missing angle” to make GH have an angle of 180 degrees.
8. Sketch the R.L., making sure that there is only one branch per pole of $G(s)H(s)$.

Exercises: Root Locus Sketching: Due FRIDAY, Oct 16 before class

Execise 1 - Controller Design

Sketch the root locus for the following system, performing all 8 steps. Be very careful with step 6 (find if/when the system goes unstable). Place your sketch and your work in the markdown cell. You may use the code cell to *check* your work, but it is not graded.

$$G(s)H(s) = \frac{s^2+10s+200}{(s+1)(s+2)(s+5)}$$

STUDENT ANSWER: 20 POINTS POSSIBLE

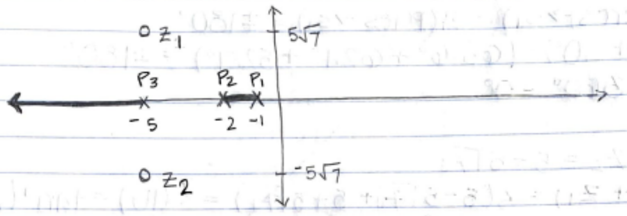
Week 9 Monday

$GH = \dfrac{s^2+10s+200}{(s+1)(s+2)(s+5)}$

Step ①: $1+KGH = 1+K\dfrac{s^2+10s+200}{(s+1)(s+2)(s+5)} = 0$

Step ②: $\dfrac{-1}{K} = \dfrac{s^2+10s+200}{(s+1)(s+2)(s+5)}$

Step ③: zeros @ $s=-5\pm 5\sqrt{7}j$; Poles @ $s=-1,-2,-5$



Step ④: $a = n-m = 3-2 = 1$ asymptote

$\theta_a = \dfrac{2K+1}{n-m}\pi \rightarrow K=0 \rightarrow \theta_a = \pi$

(already Plotted in step ③)

Step ⑤: $\dfrac{d}{ds}GH = \dfrac{d}{ds}\dfrac{s^2+10s+200}{(s+1)(s+2)(s+5)} = 0$

$\rightarrow -(s^4+20s^3+698s^2+5160s+6200)=0$ ✓

$\rightarrow s^4+20s^3+698s^2+5160s+6200=0$

$\rightarrow s = -1.49, -7.05752$ ✓ X

Step ⑥: $s = j\omega_c,\ K = K_c$

$1+KGH = 1+K\dfrac{s^2+10s+200}{(s+1)(s+2)(s+5)} = 1+K\dfrac{s^2+10s+200}{s^3+8s^2+17s+10} = 0$

$s^3+8s^2+17s+10+K(s^2+10s+200)=0$

$(j\omega_c)^3+8(j\omega_c)^2+17(j\omega_c)+10+K_c((j\omega_c)^2+10(j\omega_c)+200)=0$

imag: $-\omega_c^3 j + 17\omega_c j + 10K_c\omega_c j = 0$

real: $-8\omega_c^2+10-K_c\omega_c^2+200K_c = 0$

considering only the reasonable answers...

$\boxed{\begin{array}{l} K_c = 8.88 \text{ or } 1.42 \\ \omega_c = \pm 10.29 \text{ or } \pm 5.58 \end{array}}$ ✓ $\rightarrow$ crosses the imaginary axis and goes unstable @ $K_c = 1.42$ and crosses again and goes back to stable at $K_c = 8.88$

Step⑦: $GH = \frac{s^2+10s+200}{(s+1)(s+2)(s+5)} = \frac{(s+5+5\sqrt{7}j)(s+5-5\sqrt{7}j)}{(s+1)(s+2)(s+5)}$

$= \frac{(s+z_1)(s+z_2)}{(s+p_1)(s+p_2)(s+p_3)}$

only have complex zeros...

$s_1 = -z_1 = -(5+5\sqrt{7}j) = -5-5\sqrt{7}j$

$\angle(s_1+z_1) = \angle(-5-5\sqrt{7}j + 5+5\sqrt{7}j) = \tan^{-1}(\frac{0}{0}) = \theta_{d1}$

$\angle(s_1+z_2) = \angle(-5-5\sqrt{7}j + 5-5\sqrt{7}j) = \angle(-10\sqrt{7}j) = \tan^{-1}(\frac{-10\sqrt{7}}{0}) = -90°$

$\angle(s_1+p_1) = \angle(-5-5\sqrt{7}j+1) = \angle(-4-5\sqrt{7}j) = \tan^{-1}(\frac{-5\sqrt{7}}{-4}) = -106.8°$

$\angle(s_1+p_2) = \angle(-5-5\sqrt{7}j+2) = \angle(-3-5\sqrt{7}j) = \tan^{-1}(\frac{-5\sqrt{7}}{-3}) = -102.8°$

$\angle(s_1+p_3) = \angle(-5-5\sqrt{7}j+5) = \angle(-5\sqrt{7}j) = \tan^{-1}(\frac{-5\sqrt{7}}{0}) = -90°$

$\Sigma\angle(\text{zeros}\to s_1) - \Sigma\angle(\text{poles}\to s_1) = \pm 180$

$(\theta_{d1} + (-90°)) - (-106.8° + (-102.8°) + (-90°)) = \pm 180$

$\to \boxed{\theta_{d1} = -29.6°}$

$s_2 = -z_2 = -(5-5\sqrt{7}j) = -5+5\sqrt{7}j$

$\angle(s_2+z_1) = \angle(-5+5\sqrt{7}j + 5+5\sqrt{7}j) = \angle(10\sqrt{7}j) = \tan^{-1}(\frac{10\sqrt{7}}{0}) = 90°$

$\angle(s_2+z_2) = \angle(-5+5\sqrt{7}j + 5-5\sqrt{7}j) = \tan^{-1}(\frac{0}{0}) = \theta_{d2}$

$\angle(s_2+p_1) = \angle(-5+5\sqrt{7}j+1) = \angle(-4+5\sqrt{7}j) = \tan^{-1}(\frac{5\sqrt{7}}{-4}) = 106.8°$

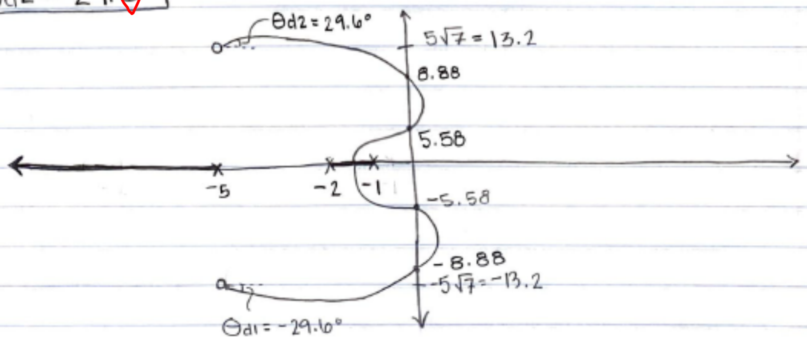
$\angle(s_2+p_2) = \angle(-5+5\sqrt{7}j+2) = \angle(-3+5\sqrt{7}j) = \tan^{-1}(\frac{5\sqrt{7}}{-3}) = 102.8°$

$\angle(s_2+p_3) = \angle(-5+5\sqrt{7}j+5) = \angle(5\sqrt{7}j) = \tan^{-1}(\frac{5\sqrt{7}}{0}) = 90°$

$\Sigma\angle(\text{zeros}\to s_2) - \Sigma\angle(\text{poles}\to s_2) = \pm 180°$

$(\theta_{d2} + 90°) - (106.8° + 102.8° + 90°) = \pm 180°$

$\boxed{\theta_{d2} = 29.6°}$ ✓



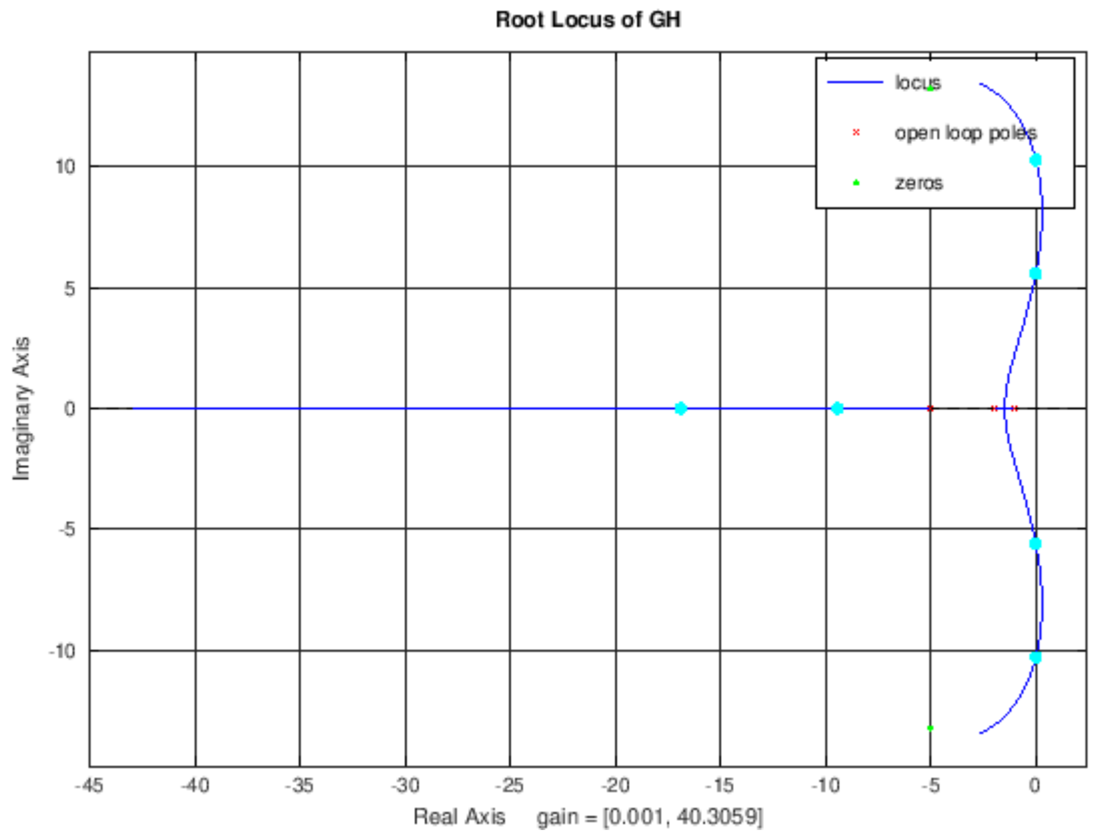
step⑧: 3 branches = 3 poles in GH ✓

STUDENT ANSWER: 0 POINTS POSSIBLE

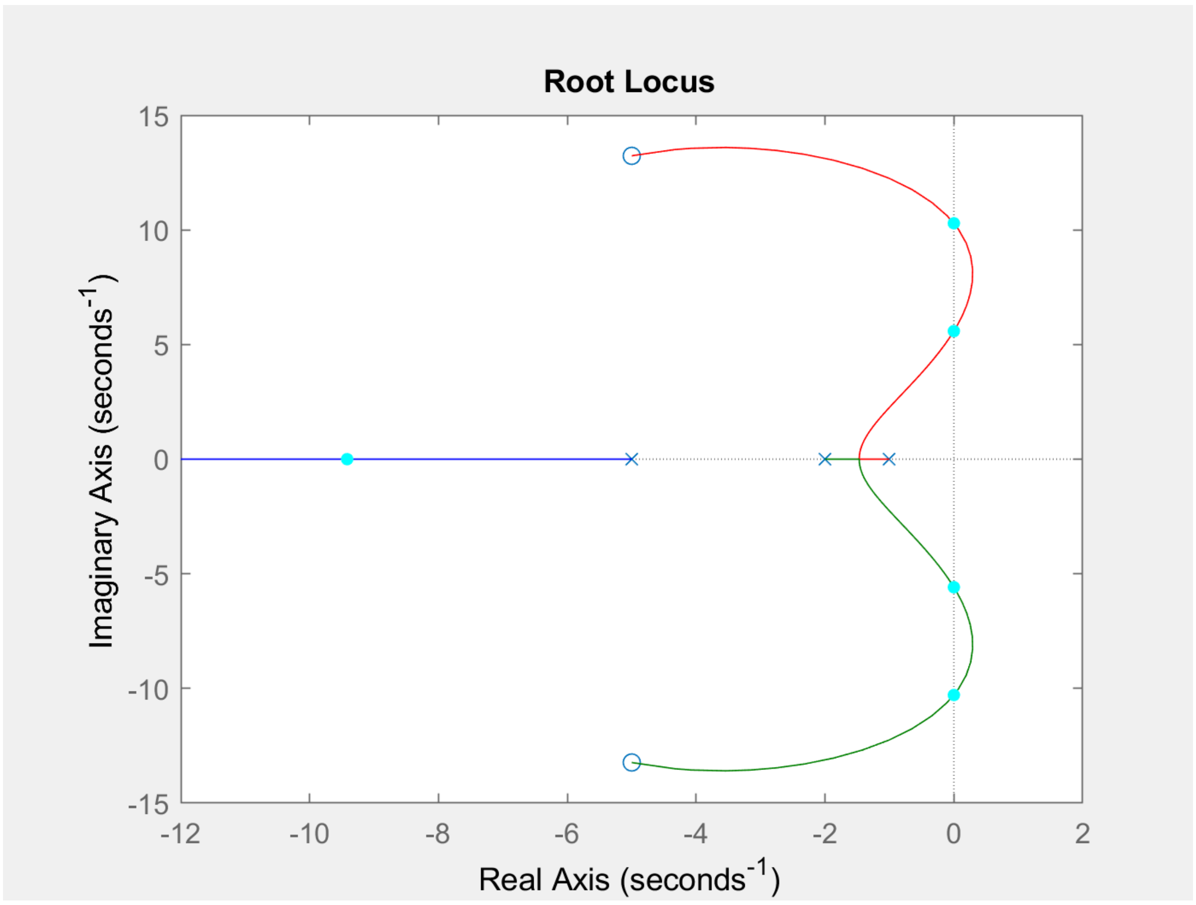
20/20

```
In [1…  %%octave
        s = tf('s');

        %set up the open loop TF
        GH = (s^2+10*s+200)/((s+1)*(s+2)*(s+5))
        rlocus(GH,1,0.001)
        hold on

        %now calculate our eigenvalues at our calculated critical gain so we c
        Kcrit1 = 1.41847
        %compute closed loop TF
        Gcl = minreal(Kcrit1*GH/(1+Kcrit1*GH));
        %pull out the denominator
        [num,den] = tfdata(Gcl,'v');
        %find the roots of the characteristic equation
        myroots = roots(den);

        %plot those roots on our root locus plot
        plot(real(myroots),imag(myroots),'c.','MarkerSize',15)

        %now calculate our eigenvalues at our calculated critical gain so we c
        Kcrit2 = 8.88129
        %compute closed loop TF
        Gcl = minreal(Kcrit2*GH/(1+Kcrit2*GH));
        %pull out the denominator
        [num,den] = tfdata(Gcl,'v');
        %find the roots of the characteristic equation
        myroots = roots(den);

        %plot those roots on our root locus plot
        plot(real(myroots),imag(myroots),'c.','MarkerSize',15)
```

```
Transfer function 'GH' from input 'u1' to output ...

         s^2 + 10 s + 200
 y1:  ----------------------
      s^3 + 8 s^2 + 17 s + 10

Continuous-time model.
Kcrit1 =  1.4185
Kcrit2 =  8.8813
                                        Root Locus of GH

                   |-------------------------------------------------------|
                   |     +     +     +     +     + +--------------F--%%%--+|
                   |     #     #     #     #     # |###?###locus       %% ||
                10 |-+#############################|   B   open loop polGs||
                   |     #     #     #     #     # +---F---zeros--------%-+|
                   |     #     #     #     #     #    #     #     #     G  |
                 5 |-+#################################################%%+-|
                   |     #     #     #     #     #    #     #     #    %#  |
                 0 |**##############################G########G####B***BB***|
                   |     #     #     #     #     #    #     #     #   $$#  |
                   |     #     #     #     #     #    #     #     #    $#  |
                -5 |-+#################################################$$+-|
                   |     #     #     #     #     #    #     #     #     G  |
                   |     #     #     #     #     #    #     #     #     $  |
               -10 |-+##################################################G+-|
                   |     #     #     #     #     #    #     #     #    $$  |
                   |     +     +     +     +     +    +     +     F  $$$+  |
                   |-------------------------------------------------------|
                  -45   -40   -35   -30   -25   -20  -15   -10   -5     0
                             Real Axis     gain = [0.001, 40.3059]
```



(The root locus produced by Octave does not accurately show the connection to the zeros so included below is a plot from MATLAB):


Root Locus
Imaginary Axis (seconds^{-1})
Real Axis (seconds^{-1})
15
10
5
0
-5
-10
-15
-12
-10
-8
-6
-4
-2
0
2

```
In [ ]:
```