EE 461: Controls Systems I

http://venus.ece.ndsu.nodak.edu/~glower/ee461/index.html

Instructor:	Prof. Jake Glower
Office Location:	EE Building 101Q
Office Phone:	231-8068
Office Hours:	t.b.d.
Class Hours:	MWF 3-4
Class Location:	FLC 313

Introduction:

Control systems is the study of dynamic systems and how to regulate the output of such systems.

Dynamic systems are any system which are described by differential equations. These include the voltage in an RLC circuit, the temperature of a person, the flow of a fluid through a jet engine, the inflation rate in a country, etc. For all such systems, the output responds to the input with a delay - a characteristic of a dynamic system. For all these systems, it is desirable to adjust the input so that the output remains constant (such as body temperature at 98.6F regardless of external disturbances such as wind or internal disturbances, such as exercise).

Control theory is the study of how to define the input so that the output behaves well.

In this class, 'classical control' theory is presented. This approach was developed in the U.S. and Europe from 1900 to 1950 for designing feedback amplifiers, anti-aircraft guns, and rockets. To this day, these approaches are the most commonly used and serve as the foundation of subsequent control theory (such as Modern Control, developed in the U.S.S.R. in the 1950's, digital control adaptive control, robust control, optimal control, etc.).

The heart of control theory is the study of differential equations (termed 'Applied Math' in the math department). Key to this study is the se of LaPlace transforms - a mathematical tool which transforms differential equations into algebraic equations in 's'. Also of use are mathematical tools, such as MATLAB, which allow you so solve 10th-order algebraic equations or use complex math with ease, or VISSIM, which allows you to simulate a differential equation with 'point and click' operations rather than programming in a differential equation in C.

Requirements:

ECE 343: Signals and Systems is required for this course. It is assumed that the students have an understanding of LaPlace transforms, solving differential equations using LaPlace transforms.

Good algebra skills are also useful, since the LaPlace transform will convert dynamic systems into sets of algebraic equations.

A calculator which can do complex math is very useful for this course. A common problem in this class (and on tests will be to find the value of $\left(\frac{1000}{s(s+5)(s+20)}\right)$ evaluated at $s = -2 + j4$.

A calculator which solves for f(x)=0 is also very useful. A common problem will be to solve for 'a' so that $angle\left(\frac{1000}{s(s+5)9s+20)}\right) = -140^0$ evaluated along the line $s = a\angle 90^0$. Graphical techniques can and will be used on tests and homework. Numerical solutions are much easier to find if your calculator can do this (MATLAB can solve such problems but is not available on tests).

Text Book:

A text book is required for this course. Each student is expected to find or borrow copy for the semester. Note that new control theory text books cost between \$150 and \$250 per copy. Instead of buying a new text, however, I'd suggest finding an older copy from

- www.amazon.com (29,000 hits for "feedback control theory)
- overture.directtextbook.com/textbooks/173515_Engineering
- www.half.com
- www.ebay.com

Grading

- Midterms: 1 unit each
- Homework & Quizzes 1 unit
- Final 1 units
- Total: Average of all above

Final Percentage:

- 100% - 90% A
- 89% - 80% B
- 79% - 70% C
- 69% - 60% D
- < 59% E

Grading will be on a straight scale to encourage working together. My objective is to see that everyone learns the material. If the class studies together and everyone gets a 90% average, I'd gladly give all A's. (After all, your competitors are at schools like UCLA, Michigan, etc. - they're not your classmates.)

Policies:

A student may take a makeup exam if he/she misses an exam due to an emergency, illness, or plant trip and notifies me in advance of the exam. Late homework will not be accepted once the solutions are posted online. All questions on the grading of a particular exam must be resolved within a week of returning the exam.

Quizzes: Each Friday, there will be a short 15 minute quiz. The topics cover what we're doing in class **or** the prerequisites of ECE 461 (namely calculus, circuits, and signals and systems). The quizzes are closed book, closed note, no calculator. Topics that are fair game for any quiz are:

ECE 343:

- Fourier Transforms: Be able to determine the spectral content of a square wave, etc.
- LaPlace Transforms: Be able to find the inverse LaPlace transform of $\left(\frac{20}{s(s+10)}\right)$ or similar
- Z transform: Be able to find the inverse z-transform of $\left(\frac{20}{z(z+10)}\right)$

ECE 311:

- Find the transfer function of an op-amp circuit with resistors, inductors, and capacitors
- Design a circuit with a gain of +10 or -10.

Homework: Homework is graded as

- 80%: You attempted the problem with an organized approach I can follow.
- 20%: You got the right answer.

Homework is practice using the tools being presented. Copying someone else's homework is sort of like watching someone else exercise. You need to do it yourself. Similarly, with grading I care most that you attempted the problem and thought about it.

Labs: Labs are a part of the homework sets (For example, problems 7-10 might be finding the transfer function for a motor you'll find in the lab.) To get credit for those homework problems, you have to attend lab.

Testing: All tests will be closed-book, closed-notes, open calculator. Midterms serve to identify who put in the time solving the homework problems. My goal in writing tests is add new twists you havn't seen yet so that

- If you did the homework and are comfortable with the concepts and tools, you'll have a shot at the midterms.
- If you copied someone else's homework, you'll be lost.

The best way to study for the midterm is to make up your own midterm. There's only so many ways to ask a question.

Special Needs - Any students with disabilities or other special needs, who need special accommodations in this course are invited to share these concerns or requests with the instructor as soon as possible.

Academic Honesty - All work in this course must be completed in a manner consistent with NDSU University Senate Policy, Section 335: Code of Academic Responsibility and Conduct. Violation of this policy will result in receipt of a failing grade.

ECE Honor Code: On my honor I will not give nor receive unauthorized assistance in completing assignments and work submitted for review or assessment. Furthermore, I understand the requirements in the College of Engineering and Architecture Honor System and accept the responsibility I have to complete all my work with complete integrity.

LaPlace Transforms

Assumption:

LaPlace transforms assume all functions are in the form of

$$y(t) = \begin{cases} a \cdot e^{st} & t > 0 \\ 0 & otherwise \end{cases}$$

This results in the derivative of y being:

$$\frac{dy}{dt} = s \cdot y(t)$$

Tables for LaPlace Transforms:

You can think of 's' as an operator meaning 'the derivative of'. Initial conditions also come into play as follows:

Table 1: LaPlace Transforms with Initial Conditions	
Time: y(t)	LaPlace: Y(s)
y	Y
y'	sY - y(0)
y"	s^2Y - y'(0) - sy(0)

From ECE 343, several common functions have the following LaPlace transform:

Table 2: Common LaPlace Transforms		
Name	Time: y(t)	LaPlace: Y(s)
delta (impulse)	$\delta(t)$	1
unit step	u(t)	$\frac{1}{s}$
	$a \cdot e^{-bt}u(t)$	$\frac{a}{s+b}$
	$2a \cdot e^{-bt}\cos(ct - \theta)u(t)$	$\left(\frac{a\angle\theta}{s+b+jc}\right) + \left(\frac{a\angle-\theta}{s+b-jc}\right)$

Example:

Find the output of a system which satisfies the following differential equation:

y" + 3'y + 2y = 4u

given that all initial conditions are zero and u is a unit step input.

Solution:

Convert to LaPlace notation

$$s^2 + 3s + 2)Y = 4U$$

Substitute for U

$$(s^2 + 3s + 2)Y = 4\left(\frac{1}{s}\right)$$

Solve for Y

$$Y = \frac{4}{s(s^2+3s+2)} = \frac{4}{s(s+1)(s+2)}$$

Use partial fraction expansion

$$Y = \left(\frac{2}{s}\right) + \left(\frac{-4}{s+1}\right) + \left(\frac{2}{s+2}\right)$$

And convert back to time using Table 2.

$$y(t) = (2 - 4e^{-t} + 2e^{-2t})u(t)$$

Example 2:

Find the y(t) given that

$$Y(s) = G \cdot U = \left(\frac{15}{s^2+2s+10}\right) \cdot \left(\frac{1}{s}\right)$$

Solution:

Factoring Y(s)

$$Y(s) = \left(\frac{15}{(s)(s+1+j3)(s+1-j3)}\right)$$

Using partial fraction expansion:

$$Y(s) = \left(\frac{1.5}{s}\right) + \left(\frac{0.7906\angle -161.56^0}{s+1+j3}\right) + \left(\frac{0.7906\angle 161.56^0}{s+1-j3}\right)$$

$$y(t) = 1.5 + 1.5812 \cdot e^{-t} \cdot \cos(3t + 161.56^0) \qquad \text{for } t>0$$

note: Control systems is actually easier than signals and systems. In controls, we limit ourselves to one dimensional causal systems (time is one dimensional and only moves forward). For motors, lights, etc. this is a physical limitation. (if you could build a system where the output happens before the input, you can make lots of money in Las Vegas.)

In Signals and Systems, you *can* have non-causal systems. For example, if you are filtering an image, you can filter left to right, right to left, or even up and down. Signals and Systems likewise uses mathematics that work for multiple non-causal dimensions.

First and Second Order Approximations

Objectives:

In this section we will look at

- Determining the dominant poles of a system
- Be able to time-scale a system
- Simplifying a transfer function by replacing it with a first or second-order approximations
- Predicting a system's behavior based upon this first or second order approximation

Definitions:

Dominant Pole(s): The pole which dominates the step response of a system. These are the poles which are closest to the jw axis.

G(s): The transfer function of a system

DC Gain: G(s) at s=0. The gain of a system for a constant input.

Settling Time: The time it takes the transients to decay to 2% of their initial value

Overshoot: The maximum of a step response divided by it's steady-state value.

Step Response: The output of a system when a unit step is applied to the input (U(s) = 1/s)

Resonance: The maximum gain vs. frequency, normalized by the DC gain.

Damping Ratio: The cosine of the angle of a complex pole as measured from the negative real axis.

Dominant Pole:

A transfer function represents a differential equation which models the response of a dynamic system. The purpose of using transfer functions is to simplify analysis while accurately describing a system's response. The complexity of a transfer function represents two conflicting goals:

- More complex transfer functions (i.e. more poles and zeros) are better able to model a system's response
- Simpler transfer functions are easier to understand and use for analysis.

When extremely accurate models are required, the system can be very high order. The Maverick Missile, for example, was modeled as a 250th order differential equation. Often times, a crude model will suffice. In this case, a simple model which captures the overall behavior of a system will do.

In order to obtain this simple model, the transfer function needs to be

- Simplified (reducing the order and complexity of the model), and
- Accurate, still capturing the system's overall behavior.

In short, you want to simplify the transfer function to one which includes the poles which dominate the system's response.

If a system has several poles, these dominant poles will be the ones which

- Have the largest initial condition (i.e. most energy), and
- Slowest decay rate (so they last the longest).

It turns out that the pole(s) closest to s=0 has the largest initial condition and the pole(s) closest to the jw axis have the slowest decay rate. Often the same pole satisfies both of these conditions.

Example: Find the dominant poles of

$$G(s) = \left(\frac{2000}{(s+1)(s+10)(s+100)}\right)$$

Solution: The step response of this system is

$$Y(s) = \left(\frac{2000}{(s+1)(s+10)(s+100)}\right)\left(\frac{1}{s}\right)$$

Using partial fraction expansion

$$Y(s) = \left(\frac{2}{s}\right) + \left(\frac{-2.222}{s+1}\right) + \left(\frac{0.2469}{s+10}\right) + \left(\frac{0.0022}{s+100}\right)$$

The step response is then

$$y(t) = (2 - 2.222e^{-t} + 0.2469e^{-10t} - 0.0022e^{-100t})u(t)$$

Note that

- The first term (2) is the forced response. It remains as long as the input remains equal to 1.
- The second term (-2.22e-t) has an initial condition 9x larger than any other term and it decays slower than the other terms.

Hence, the pole at s=-1 dominates the response (and is termed the dominant pole).

Once the dominant pole has been identified, you can simplify the transfer function by

- Keeping the dominant pole(s), and
- Keeping the DC gain unchanged.

(DC is often used since most control systems are to track constant or slowly changing inputs, such as the desired temperature in a room, flow through a pipe, etc.)

Example: Find a simplified model for G(s) from above.

Solution: Since the dominant pole is at -1

$$G(s) = \left(\frac{2000}{(s+1)(s+10)(s+100)}\right) \approx \left(\frac{a}{s+1}\right)$$

The constant 'a' is selected so that G(s=0) is the same in both cases. Setting the DC gain to be the same

$$G(s) \approx \left(\frac{2}{s+1}\right)$$

Example: Find a simplified model for a system with complex poles:

$$G(s) = \left(\frac{2000}{(s+1+j2)(s+1-j2)(s+10)(s+50+j200)(s+50-j200)}\right)$$

Solution: The dominant pole is the pole which decays the slowest and is closest to s=0 are (-1+j2, -1-j2). Since the pole is complex, its complex conjugate will also be present in any model. Both decay at the same rate, so there are two dominant poles.

The simplified model is then

$$G(s) = \left(\frac{2000}{(s+1+j2)(s+1-j2)(s+10)(s+50+j200)(s+50-j200)}\right) \approx \left(\frac{a}{(s+1+j2)(s+1-j2)}\right)$$

Setting the DC gain to be the same, 'a' is found to be 0.000941

$$G(s) \approx \left(\frac{0.000941}{(s+1+j2)(s+1-j2)}\right)$$

Time Scaling

Since a transfer function is only a model that simulates a real system, it is not necessary for one second in simulation time to correspond to one second in real time. Sometimes, it is an advantage to have different time scales. For example, if modeling the effect of interest rates on the U.S. economy, it would be very convenient if 1 month real time corresponded to 1 second in simulation time. Events which take effect in 6 months could then be observed in 6 seconds. Fast events can also be slowed down so that you can observe what is going on (such as modeling the propagation of a flame or pressure wave).

In order to time scale a transfer function

i) Define the relationship between simulation time (τ) and real time (t) as $\tau = \alpha t$.

- For example, if 1s simulation time corresponds to 1ms real time (i.e. the simulation is 1000x slower than the actual system), $\tau = 0.001t$.

ii) Substitute for $s \rightarrow \alpha s$

- The LaPlace transform assumes all functions are in the form of e^{st}. Substituting for t results in all functions being in the form of $e^{(s/\alpha)\tau}$.
- To slow up the system by 1000, replace s with $\frac{s}{0.001} = 1000s$.

iii) Simplify the transfer function.

Example: The dynamics of a servo-motor are given as $G(s) = \left(\frac{10^7}{s(s+500)(s+2000)}\right)$. Find the transfer function if time scaled by 1000x (i.e. the model is 1000x slower than the actual system.)

Solution: Replace 's' with $1000s$

$$G(s) \Rightarrow \left(\frac{10^7}{(1000s)(1000s+500)(1000s+2000)}\right) = \left(\frac{0.01}{s(s+0.5)(s+2)}\right)$$

note: In this course, most systems will have poles with a magnitude close 1. This is done since i) with time scaling, any system can be modeled as one with poles close to 1, and ii) 1 has nice numerical properties. (i.e. $1^n=1$)

First-Order Approximations:

Assume the simplified model for a system is a first-order system. Given this model, try to predict how the system will behave, including

- The steady-state gain (if an input of 1 is applied, what will the output go to?)
- The settling time (how long it takes to reach steady-state), and
- The bandwidth (if the input is a sinusoid, what frequency range will be passed?)

Consider a generic 1st-order system

$$G(s) = \left(\frac{a}{s+b}\right)$$

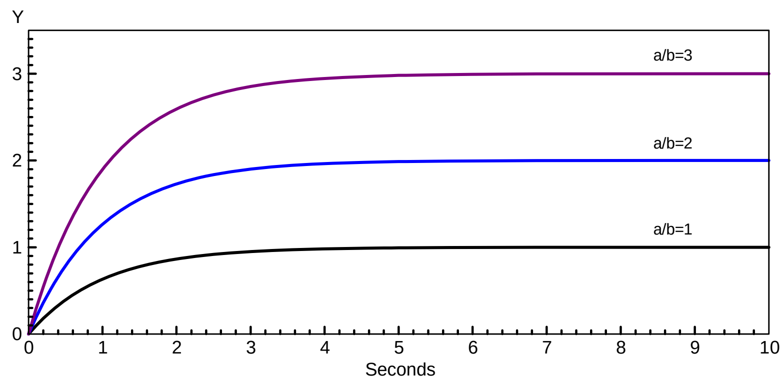
The DC gain is G(s=0), or a/b

The steady-state gain is G(s=0) = a/b

Example: Plot the step response of $\left(\frac{1}{s+1}\right), \left(\frac{2}{s+1}\right), \left(\frac{3}{s+1}\right)$

Solution: From MATLAB:

```
» G2 = tf(2,[1,1]);
» G3 = tf(3,[1,1]);
» t = [0:0.1:10]';
» y1 = step(G1,t);
» y2 = step(G2,t);
» y3 = step(G3,t);
» plot(t,y1,t,y2,t,y3)
```



Settling Time: The transient decays as e^{-bt}. This transient decays to 2% of its initial value in

$$0.02 = e^{-bt}$$

$$\ln(0.02) = -3.912 = -bt$$

$$t = \frac{3.912}{b} \text{ seconds}$$

or, since the model is only approximate anyway, this is usually defined as

$$t \approx \frac{4}{b}$$

The 2% settling time will be (approximately) $\frac{4}{b}$

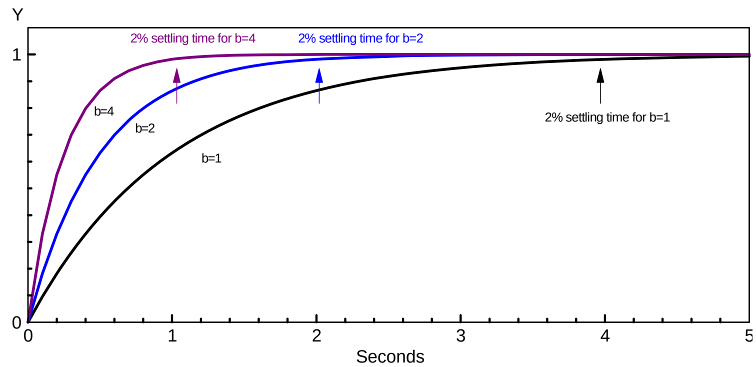
Another way to look at this is to use time scaling. If you slow down the simulation by a factor of 'b', the transfer function becomes

$$G(s) \Rightarrow \left(\frac{a}{bs+b}\right) = \left(\frac{a/b}{s+1}\right)$$

A system with a pole at -1 has a 2% settling time of about 4 seconds. Since this is a time-scaled system, the actual system will have a settling time of $\frac{4}{b}$ seconds.

Example: Plot the step response of $\left(\frac{1}{s+1}\right), \left(\frac{2}{s+2}\right), \left(\frac{4}{s+4}\right)$

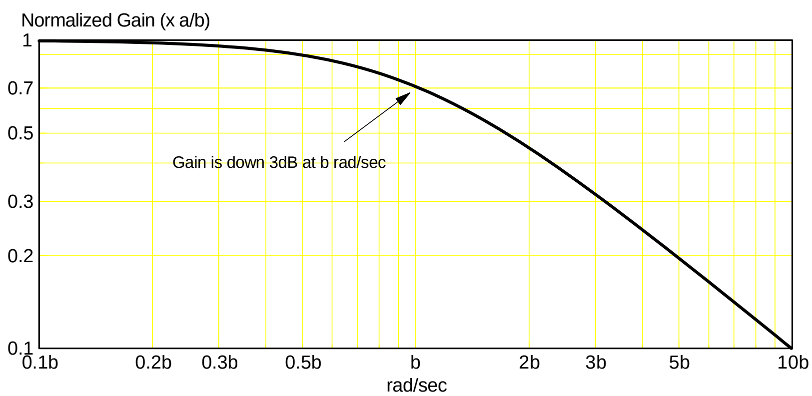
from MATLAB:



The frequency response of G(s) is found by letting s=jw. The gain vs. frequency is then

$$G(j\omega) = \left(\frac{a}{j\omega+b}\right)$$

which is plotted below.



At a frequency of b rad/sec, this gain drops to

$$|G(jb)| = \left|\frac{a}{jb+b}\right| = \left(\frac{a/b}{\sqrt{2}}\right)$$

or

- The gain is down 3dB
- The power is down 6db (1/2)
- Frequencies past 'b' rad/sec are attenuated. Those below 'b' rad/sec are passed.

The bandwidth of the system is b rad/sec

Second-Order Approximations:

Assume the simplified model for a system is second-order:

$$G(s) = \left(\frac{ac}{s^2+bs+c}\right)$$

Factoring the denominator and placing it in rectangular form gives

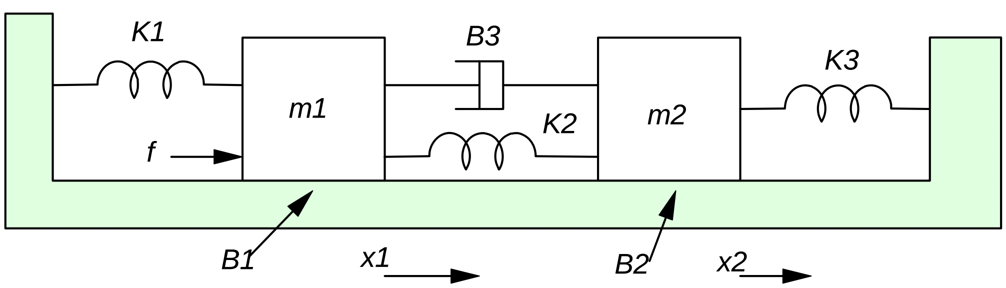
$$G(s) = \left(\frac{a\omega_n^2}{(s+\sigma+j\omega_d)(s+\sigma-j\omega_d)}\right)$$

If you take the step response of this system, the terms will be of the form

$$y(t) = a + be^{-\sigma t}\cos(\omega_d t + \phi) \qquad (t>0)$$

where 'b' and ϕ are constants. Note that the step response can be determined by inspection by looking at the different terms in the transfer function:

- a: Determines the DC gain. Doubling 'a' doubles the amplitude of the response.



- ω_d: Determines the frequency which the system oscillates at.
- σ: Determines the rate at which the exponential envelope decays.

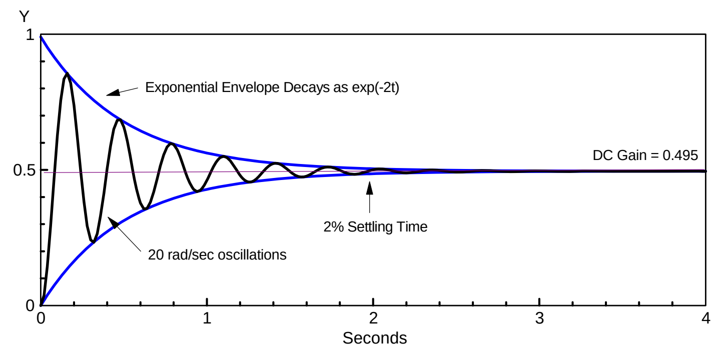
Example: Determine how $G(s) = \left(\frac{200}{(s+2+j20)(s+2-j20)}\right)$ will behave:

Solution:

- The step response will eventually go to 0.495. (G(0) = 0.495).

- It will take approximately 2 seconds to reach steady-state. (The real part of the dominant pole is -2. Hence, the transient decays as e^{-2t}), and
- The transient will oscillate at 20 rad/sec (the complex part of the pole is j20).

The actual step response follows:



2% Settling time = $\frac{4}{\sigma}$

Frequency of Oscillation = ω_d

Resonance Frequency ≈ ω_d

Often times, the overshoot in the step response if of interest rather than the frequency of oscillation. This is easiest to predict if you express the transfer function in polar form:

$$G(s) = \left(\frac{a\omega_n^2}{(s+\omega_n\angle\theta)(s+\omega_n\angle-\theta)} \right)$$

or

$$G(s) = \left(\frac{a\omega_n^2}{s^2+2\zeta\omega_n s+\omega_n^2} \right)$$

where $\zeta = \cos\theta$. If you time scale by a factor of ω_n, and scale by the DC gain (a), this system becomes

$$G(s) = \left(\frac{1}{(s+1\angle\theta)(s+1\angle-\theta)} \right) = \left(\frac{1}{s^2+2\zeta s+1} \right)$$

From this, it is clear that

- 'a' determines the amplitude of the response (double 'a' and you double the output)
- ω_n (the amplitude of the poles) determines the speed of the system (i.e. the time scaling). Double ω_n and you halve the response time or double the bandwidth. (If you respond to higher frequencies, you respond more quickly).
- The 'shape' of the step or frequency response is determined by the angle of the complex poles (θ or the damping ratio ζ)

This 'shape' determines the overshoot for a step response or amplitude of the resonance for a frequency response. This is summarized on the second-order approximations sheet. Two commonly used terms are

$$\text{Overshoot} = e^{\frac{-\pi\zeta}{\sqrt{1-\zeta^2}}}$$

$$\text{Resonance} = \text{Mm} = \begin{cases} \frac{1}{2\zeta\sqrt{1-\zeta^2}} & \zeta < 0.7 \\ 1 & \zeta > 0.7 \end{cases}$$

(If the damping ratio is greater than 0.7, there is no resonance and Mm does not apply)

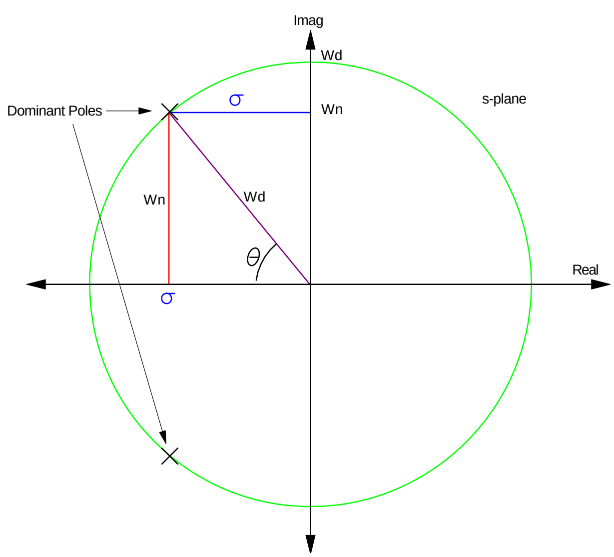
Note: Since a complex pole only has two degrees of freedom, there is some redundancy here:

- Real part of the pole (σ) determines the settling time
- Imaginary part of the pole (ω_d) determines the frequency of oscillation
- Angle of the pole (θ or $\cos\theta = \zeta$) determines the overshoot and resonance,
- The magnitude of the pole (ω_n), determines the corner frequency on a Bode plot.

These are all related by through trigonometry as shown in the following figure. Hence, if you are going to specify how a system should behave, you should only include two of the following requirements to avoid inconsistency:

- Settling Time
- Frequency of Oscillation (or resonance)
- % Overshoot (if the magnitude of the resonance was not specified)
- Magnitude of the Resonance (if % overshoot was not specified)
- Corner frequency (bandwidth)

The type of system often determines which parameters are relevant. (For example, for an amplifier, resonance and bandwidth are important. For a jet engine, settling time and overshoot are important.)



Example: Determine how mass x1 will behave when a step input is applied at F. Assume M=1, B=2, K=3.

Solution: From before, the transfer function is

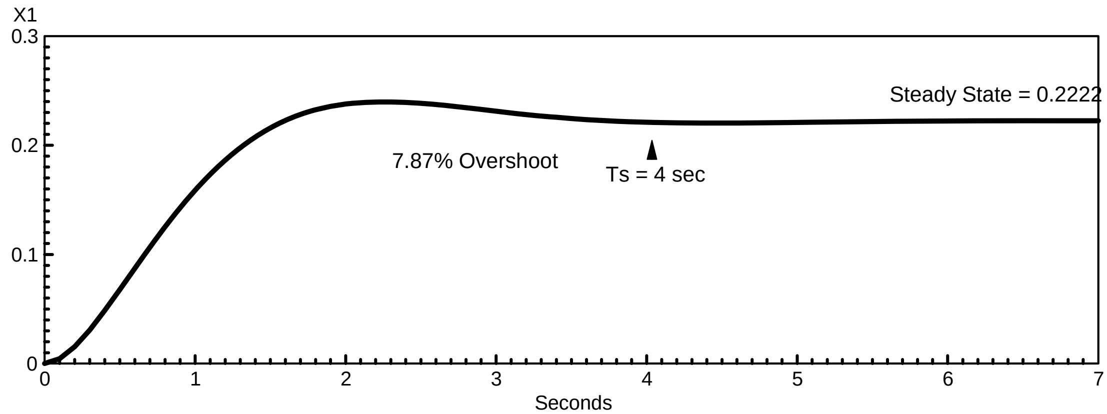
$$x_1 = \left(\frac{s^2+4s+6}{s^4+8s^3+24s^2+36s+27}\right)F$$

Factoring

$$x_1 = \left(\frac{s^2+4s+6}{(s+1+j1.412)(s+1-j1.412)(s+3)(s+3)}\right)F$$

The DC gain is 6/27. Mass x1 will eventually move 6/27 meter to the right.

The 'dominant' pole is at -1 + j1.412. The transient will take about 4 seconds. $(\sigma = 1)$ The overshoot will be about 10.8% ($\zeta = 0.578$). The actual step response is shown below. The actual response is a little different from what the second order approximations would suggest since i) this system is actually 4th order, ii) the fast poles aren't that much faster than the dominant poles, and iii) there are zeros in this system. In spite of this, the 2nd-order approximations predict the step response fairly accurately.



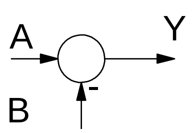
Block Diagrams & VisSim

Objective:

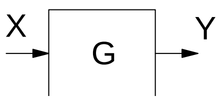
- Introduce block diagram notation
- Be able to write the equations that describe a block diagram
- Be able to simplify a block diagram

Block Diagram Notation

Block diagrams are one of the core concepts in controls systems. They are a pictorial way to represent how a feedback system is connected. Some of the symbols used are as follows:



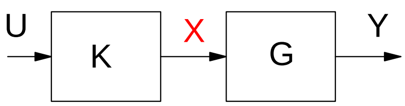
Summing block. Y = A - B



Gain block: Y = GX

With these, you can simplify a block diagram to a single gain. Sometimes it helps to add in dummy variables and then simplify.

Series Connection:



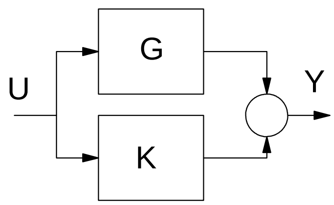
$$Y = GX$$

$$X = KU$$

simplifying:

$$Y = GK \cdot U$$

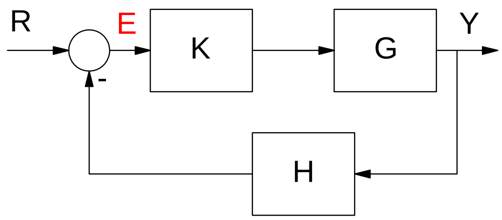
Parallel Connection:



$$Y = G \cdot U + K \cdot U$$

$$Y = (G + K) \cdot U$$

Feedback Configuration:



Add in a dummy variable, E:

$$Y = (G \cdot K) \cdot E$$

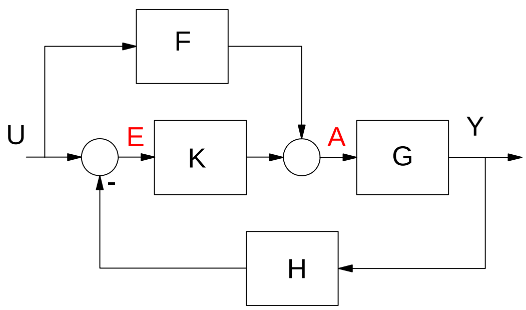
$$E = R - H \cdot Y$$

Simplifying:

$$\boxed{Y = \left(\frac{GK}{1+GKH}\right) R}$$

Memorize this one. We'll be using it a lot this semester.

Feedback + Feedforward:



Add in two dummy variables, E and A, to simplify the equations:

$$Y = GA$$

$$A = KE + FU$$

$$E = U - HY$$

Substitute and solve:

$$Y = G(K(U - HY) + FU)$$

$$(1 + GKH)Y = (GK + GF)U$$

$$Y = \left(\frac{G(K+F)}{1+GKH}\right)U$$

VisSim:

VisSim is a program from Visual Solutions Incorporated - designed specifically for the design, analysis, and simulation of dynamic systems (i.e. ECE 461). It's similar to SciLab, but VisSim came first and I prefer it. Plus, VisSim has a free departmental license (from http://www.vissim.com/products/academic.html):

> *A one year license to operate the software on an unlimited number of machines in the department. The free license for the VisSim Academic Package will be renewed at no charge simply by registering for a license extension at the termination of the first year license.*
>
> *One user manual for each licensed product.*
>
> *All department faculty members and students enrolled in the VisSim training program may make copies of the software for personal use on- or off-campus.*
>
> *Technical support is available to faculty members only..*

Note:

- You may use VisSim on a personal machine for one year.
- You may use VisSim at NDSU for one year,
- It's free for the year (!)
- If you like the product, you can buy your own version for about $50.

Help manuals are located at

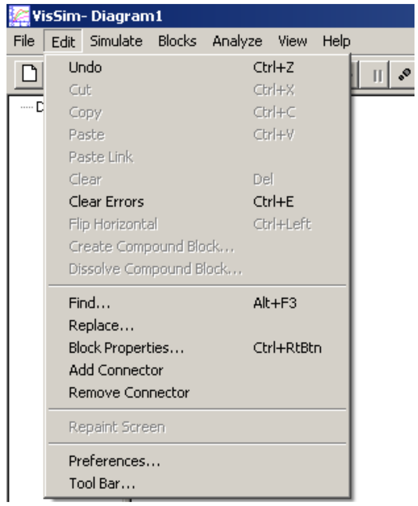
www.VisSim.com

VisSim Menu's

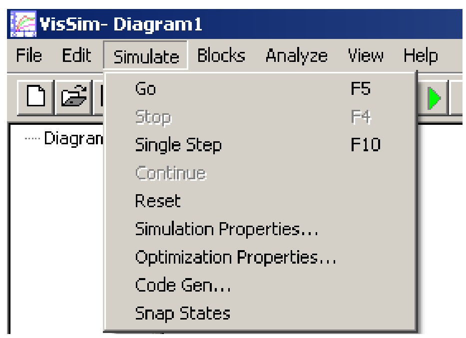
VisSim implements block diagrams used in controls systems. I've found it's very easy to use and intuitive. I've never read the user manual and have had no problem. Pretty much, use the pull down menus to find stuff that looks useful. This will illustrate some of the features we'll use in ECE 461.

Edit:

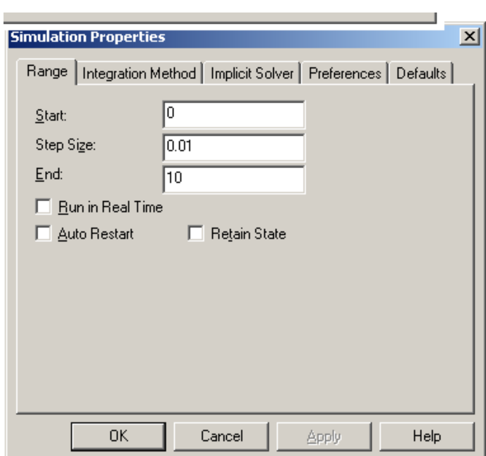
- Flip Horizontal: Flip a block left to right. It sometimes makes a screen prettier if the input is on the right.
- Create Compound Block: Highlight part of the screen and click on this. All the mess you highlighted is placed inside a block. This cleans up the screen display. If you double-click on the block, you see what's inside.
- Add / Remove Connector: Add or remove inputs and outputs to compound blocks and summing junctions.

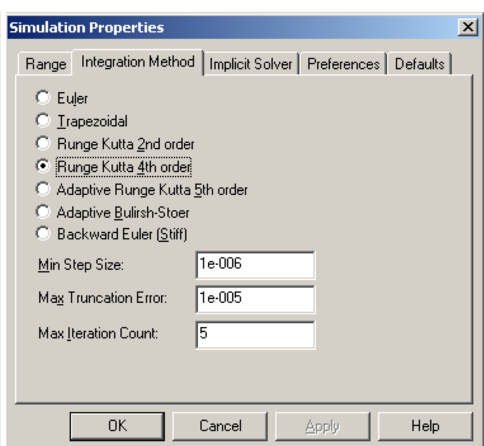


Simulate: This lets you change how the program runs.

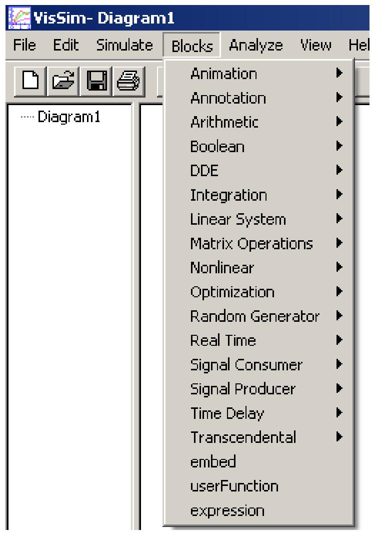


Simulation Properties lets you change the step size (0.001 or smaller usually). Integration Method is what form of numerical integration is used. Usually use Runge Kutta 4th order or 5th order.



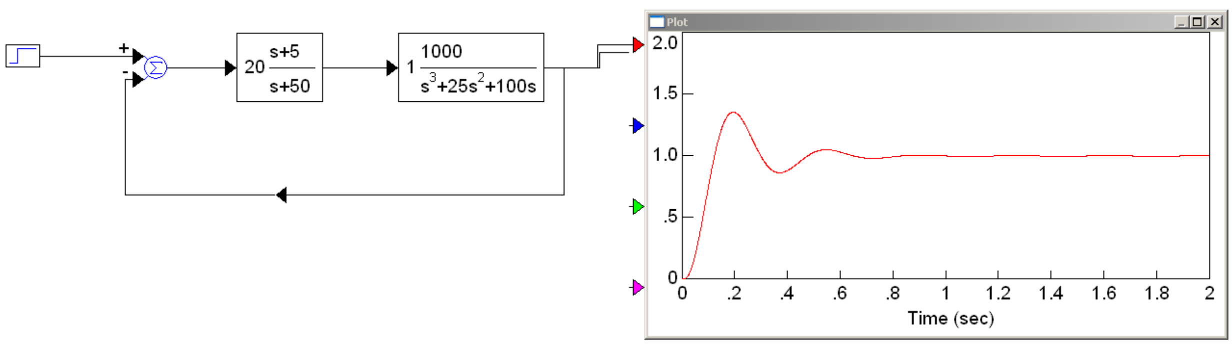


Block: Pretty much everything you're going to use. Just scroll down each item until you find something that sounds close. We'll go through some examples of more commonly used blocks.



Simulation in VisSim

Example 1. Find the step response for the following system:



For this, we need

- A transfer function, (menu Block, Linear System, Transfer Function)
- A step input, (Block, Signal Producer, Step)
- A summing junction (Block, Arithmetic, Summing Junction)

Add a transfer function. To input the one on the right, the numerator and denominator polynomials are input in decreasing powers of s:

numerator = 1000

denominator = 1 25 100 0

Once you add the blocks, connect them with arrows. (left click on the output of one block, left click on what it connects to, repeat).

Set up the simulation to go from 0 to 2 seconds with a step size of 0.001 second.

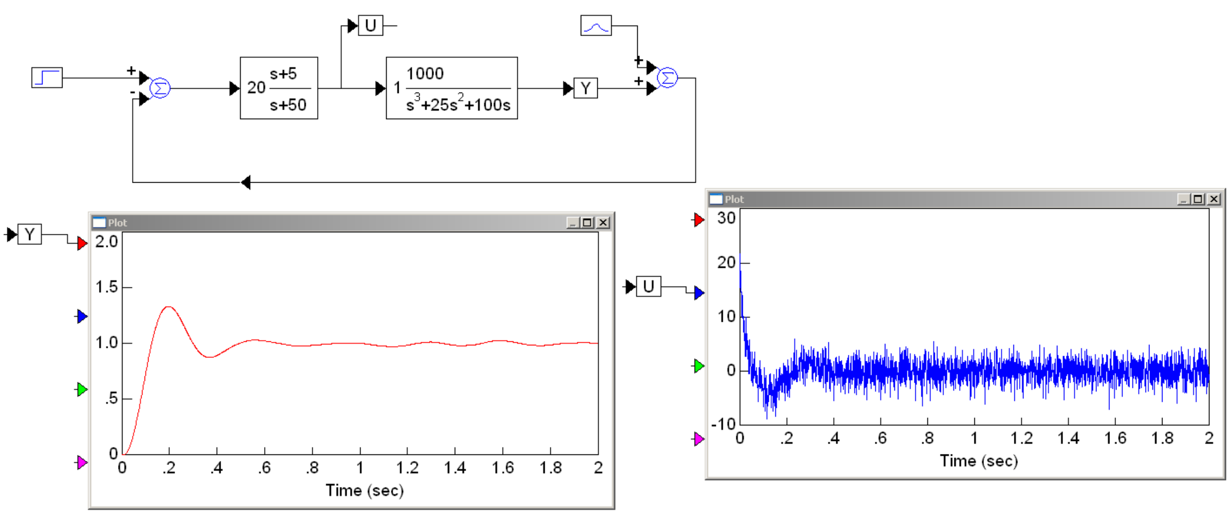
Click on the green arrow to run the simulation.

Example 2: Add Gaussian noise (N(0, 0.1)) to the measured output to simulate sensor noise. Also plot the input to the last block:

Solution: Add Gaussian noise (Block, Random, Gaussian). Left click to change the block's parameters to mean 0, standard deviation 0.1.

Add a variable to pretty up the screen. (Block, Annotation, Variable). Call these variables Y and U (output and input).

Add a summing junction to add the real output to the noise to model the sensor.



State-Space Representation of System Dynamics

State-Space is a matrix-based formulation for a system's dynamics. The standard form for the dynamics of a linear system are

$$sX = AX + BU$$

$$Y = CX + DU$$

where Y is the system's output, U is the system's input, and X are 'dummy' states (termed internal states.) With this formulation, the transfer function will be

$$X = (sI - A)^{-1}BU$$

$$Y = \left(C(sI - A)^{-1}B + D\right)U$$

MATLAB is a matrix language which has functions that let you find the transfer function of a system in state-space form. It's much easier (and less prone to errors) if you express many systems in state-space form and let MATLAB determine the transfer function. It's also an easier way to implement a system on a microcomputer.

MATLAB Commands

- G = ss(A, B, C, D); input a system in state-space form
- G = tf(num, den) input a system in transfer function form
- G = zpk(z, p, k) input a system in zeros, poles, gain form
- ss(G) determine A, B, C, D for system G (answer is not unique)
- tf(G) determine the transfer function of system G
- zpk(G) determine the zeros, poles, and gain of system G

Matrix Algebra

An nxm matrix has n rows and m columns. For example, a 2x3 matrix has 6 elements:

$$A_{2x3} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix}$$

Scalar Multiplication:

$$bA = \begin{bmatrix} ba_{11} & ba_{12} & ba_{13} \\ ba_{21} & ba_{22} & ba_{23} \end{bmatrix}$$

Matrix Addition:

Matrices with the same dimensions can be added:

$$A + B = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix} + \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \end{bmatrix} = \begin{bmatrix} a_{11}+b_{11} & a_{12}+b_{12} & a_{13}+a_{13} \\ a_{21}+b_{21} & a_{22}+b_{22} & a_{23}+b_{23} \end{bmatrix}$$

Matrix Multiplication:

To multiply two matrices, the inner dimensions must match

$$A_{xy} \cdot B_{yz} = C_{xz}$$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix} \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \\ b_{31} & b_{32} \end{bmatrix} = \begin{bmatrix} a_{11}b_{11} + a_{12}b_{21} + a_{13}b_{31} & a_{11}b_{12} + a_{12}b_{22} + a_{13}b_{32} \\ a_{21}b_{11} + a_{22}b_{21} + a_{23}b_{31} & a_{21}b_{12} + a_{22}b_{22} + a_{23}b_{32} \end{bmatrix}$$

$$= \begin{bmatrix} \sum a_{1x}b_{x1} & \sum a_{1x}b_{x2} \\ \sum a_{2x}b_{x1} & \sum a_{2x}b_{x2} \end{bmatrix}$$

Matrix Inversion

$$A \cdot A^{-1} = I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

I = the identity matrix = the matrix version of '1'

For a 2x2 matrix,

$$A^{-1} = \begin{bmatrix} \frac{a_{22}}{\Delta} & \frac{-a_{12}}{\Delta} \\ \frac{-a_{21}}{\Delta} & \frac{a_{11}}{\Delta} \end{bmatrix}$$

$$\Delta = a_{11}a_{22} - a_{12}a_{22}$$

Placing a System in State-Space Form:

i) Write N equations for the N voltage nodes

ii) Solve for the highest derivative for each equation

iii) Rewrite in matrix form.

Example. The following differential equation describe the water level in a two-tank system. Write this in state-space form and find the transfer function from U to Y.

$$\frac{dx_1}{dt} = x_2 - x_1 + 0.3u$$

$$\frac{dx_2}{dt} = x_1 - 1.3x_2$$

Write this as

$$sX = AX + BU$$

$$s\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} -1 & 1 \\ 1 & -1.3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0.3 \\ 0 \end{bmatrix} U$$

If x2 is the output,

$$Y = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + [0]U$$

To find the transfer function in MATLAB,

```
A = [-1, 1; 1, -1.3]
B = [0.3; 0]
C = [0, 1]
D = 0;

G = ss(A, B, C, D)
tf(G)

Transfer function:
       0.3
-----------------
s^2 + 2.3 s + 0.3
```

Example: The temperature along the length of a metal bar are described by

$$sx_1 = u - 2x_1 + x_2$$

$$sx_2 = x_1 - 2x_2 + x_3$$

$$sx_3 = x_2 - 2x_3 + x_2$$

$$sx_4 = x_3 - x_4$$

Find the transfer function from U to X4.

Solution: Express this in state-space form

$$s\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} -2 & 1 & 0 & 0 \\ 1 & -2 & 1 & 0 \\ 0 & 1 & -2 & 1 \\ 0 & 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} U$$

$$y = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + [0]U$$

Input this into MATLAB

```
A = [-2,1,0,0;1,-2,1,0;0,1,-2,1;0,0,1,-1];
B = [1;0;0;0];
C = [0,0,0,1];
D = 0;

G = ss(A,B,C,D);
tf(G)
```

```
Transfer function:

               1
-------------------------------
s^4 + 7 s^3 + 15 s^2 + 10 s + 1
```

or if you prefer factored form:

```
zpk(G)

Zero/pole/gain:
                 1
------------------------------------
(s+1) (s+2.347) (s+3.532) (s+0.1206)
```

Suppose you wanted to measure the average temperature of the bar. The only change is what you measure:

$$y = \begin{bmatrix} 0.25 & 0.25 & 0.25 & 0.25 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + [0]U$$

```
C = [0.25,0.25,0.25,0.25];
D = 0;

G = ss(A,B,C,D);
tf(G)


Transfer function:

0.25 s^3 + 1.5 s^2 + 2.5 s + 1
-------------------------------
s^4 + 7 s^3 + 15 s^2 + 10 s + 1
```

or if you prefer factored form:

```
» zpk(G)

Zero/pole/gain:

  0.25 (s+3.414) (s+2) (s+0.5858)
------------------------------------
(s+1) (s+2.347) (s+3.532) (s+0.1206)
```

note: Yo don't have to use MATLAB. You can also write this using LaPlace notation as four equations for four unknowns. Using algebra (and about three hours), you can simplify and get the same answer.

Canonical Forms

Objective:

- Give a block-diagram representation for a transfer function in various canonical forms
- Give a state-space representation for a transfer function in various canonical forms

State-space is the way MATLAB and other programs represent transfer functions. One feature of state-space is there are an infinite number of ways to represent the same transfer function. Some forms have standard names - these are termed 'canonical forms.'

Problem: Place the following system in state-space form

$$Y = \left(\frac{a_3s^3+a_2s^2+a_1s+a_0}{s^4+b_3s^3+b_2s^2+b_1s+b_0}\right)U$$

Controller Canonical Form:

Change the problem to

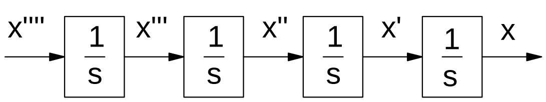
$$X = \left(\frac{1}{s^4+b_3s^3+b_2s^2+b_1s+b_0}\right)U$$

$$Y = (a_3s^3 + a_2s^2 + a_1s + a_0)X$$

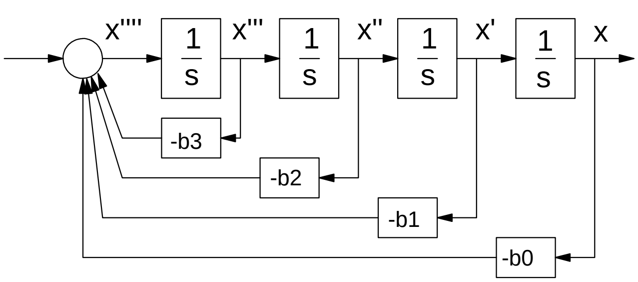
Solve for the highest derivative of X:

$$s^4X = U - b_3s^3X + b_2s^2X + b_1sX + b_0X$$

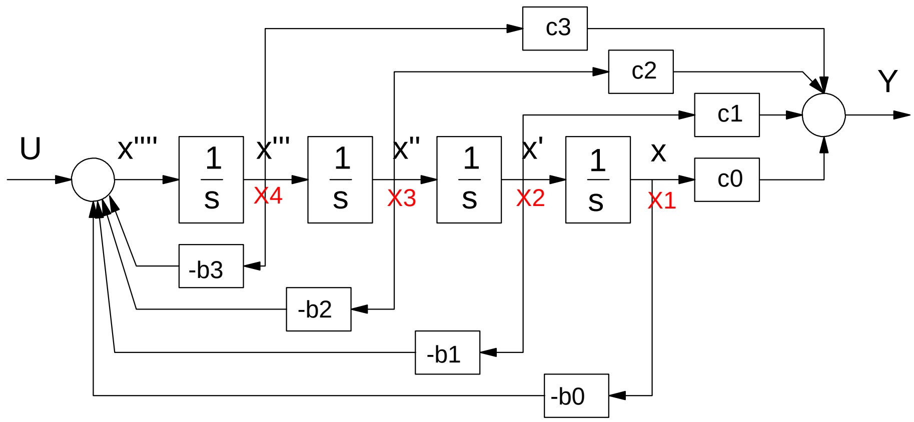
Integrate s4X four tiems to get X: (note: x' means $\frac{dx}{dt}$)



Create s4X from U and its derivatives



Create Y from the derivatives of X:



Controller Canonical Form

State Space Form: Define a state to be the output of each integrator (shown in red above). This gives

$$sX_1 = X_2$$

$$sX_2 = X_3$$

$$sX_3 = X_4$$

$$sX_4 = U - b_3X_4 + b_2X_3 + b_1X_2 + b_0X_1$$

or in matrix form:

$$s\begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -b_0 & -b_1 & -b_2 & -b_3 \end{bmatrix}\begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}U$$

$$Y = \begin{bmatrix} c_0 & c_1 & c_2 & c_3 \end{bmatrix}\begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix} + [0]U$$

Note that you can write this state-space model by inspection from the transfer function:

$$Y = \left(\frac{a_3s^3+a_2s^2+a_1s+a_0}{s^4+b_3s^3+b_2s^2+b_1s+b_0}\right)U$$

This is what MATLAB uses to store transfer functions since it's so easy to determine once you know the transfer function. It's also easy to convert from controller canonical form to the transfer function.

This form is called 'controller form' since the input, U, can set the states at will. For example, if

$$u(t) = \delta(t)$$

the output of the first integrator jumps to 1 at t=0+. If

$$u(t) = \frac{d}{dt}(\delta(t))$$

(termed a doublet), the output of the second integrator jumps to 1 at t=0+.

This form also has some of the worst numerical properties. Nothing is free.

Observer Canonical Form:

If you transpose controller canonical form you get observer form:

$$A_o = (A_c)^T$$

$$B_o = (C_c)^T$$

$$C_o = (B_c)^T$$

or

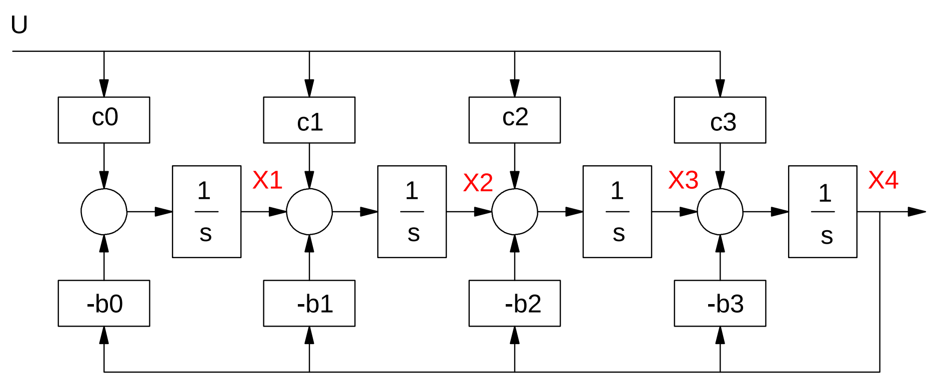
$$s\begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & -b_0 \\ 1 & 0 & 0 & -b_1 \\ 0 & 1 & 0 & -b_2 \\ 0 & 0 & 1 & -b_3 \end{bmatrix}\begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix} + \begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} U$$

$$Y = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix} + [0]U$$

The block-diagram version looks like the following:

$$sX_1 = -b_0 X_4 + c_0 U$$

etc.



Observer Canonical Form

The nice thing about observer canonical form is you can determine all four states by measuring the output, Y, and its derivatives.

Cascade Form:

Assume the transfer function factors as

$$Y = \left(\frac{c_0}{(s+p_1)(s+p_2)(s+p_3)(s+p_4)}\right)U$$

Treat this as four sytems cascaded:

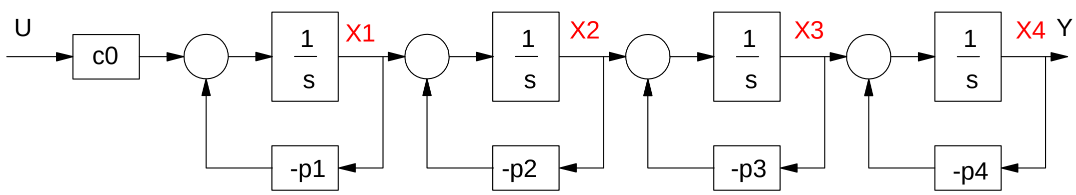
$$X_1 = \left(\frac{c_0}{s+p_1}\right)U$$

$$X_2 = \left(\frac{1}{s+p_2}\right)X_1$$

$$X_3 = \left(\frac{1}{s+p_3}\right)X_2$$

$$X_4 = \left(\frac{1}{s+p_4}\right)X_3$$

The block diagram model looks like the following:



The state-space model is:

$$s\begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix} = \begin{bmatrix} -p_1 & 1 & 0 & 0 \\ 0 & -p_2 & 1 & 0 \\ 0 & 0 & -p_3 & 1 \\ 0 & 0 & 0 & -p_4 \end{bmatrix}\begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix} + \begin{bmatrix} c_0 \\ 0 \\ 0 \\ 0 \end{bmatrix}U$$

$$Y = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix} + [0]U$$

The nice thing about cascade form is it has very good numerical properties. You can also determine the poles of the system by inspection: they're the values on the diagonal.

Jordan Form:

Assume the transfer function can be expressed using partial fractions as

$$Y = \left(\left(\frac{a_1}{s+p_1} \right) + \left(\frac{a_2}{s+p_2} \right) + \left(\frac{a_3}{s+p_3} \right) + \left(\frac{a_4}{s+p_4} \right) \right) U$$

Treat this as four separate systems:

$$X_1 = \left(\frac{c_1}{s+p_1} \right) U$$

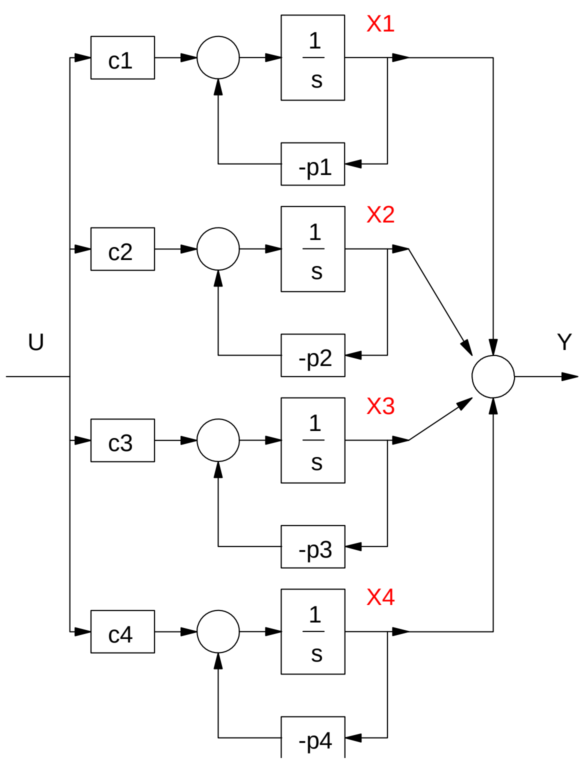
$$X_2 = \left(\frac{c_2}{s+p_2} \right) U$$

$$X_3 = \left(\frac{c_3}{s+p_3} \right) U$$

$$X_4 = \left(\frac{c_4}{s+p_4} \right) U$$

Y is then the sum of these four.

In block diagram form, Jordan form looks like the following:



In state-space, this looks like:

$$s\begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix} = \begin{bmatrix} -p_1 & 0 & 0 & 0 \\ 0 & -p_2 & 0 & 0 \\ 0 & 0 & -p_3 & 0 \\ 0 & 0 & 0 & -p_4 \end{bmatrix}\begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix} + \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{bmatrix}U$$

$$Y = \begin{bmatrix} 1 & 1 & 1 & 1 \end{bmatrix}\begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix} + [0]U$$

SciLab

Objectives:

- Be able to write your own function in SciLab
- Input a system in packed form
- Introduce the control system toolbox for SciLab

SciLab vs. Matlab:

SciLab and Matlab are pretty similar. Both have almost identical syntax. Both allow you to write your own functions. SciLab is free, however, so you can install it on your home computer (and laptop, etc.) I like free.

To get your own copy of SciLab, search for SciLab from a web browser. I think the site is:

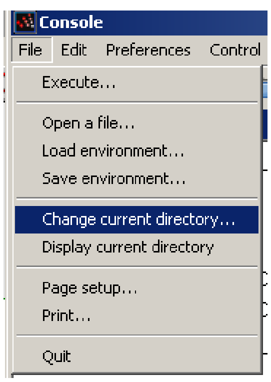
www.SciLab.org

Adding a function in SciLab:

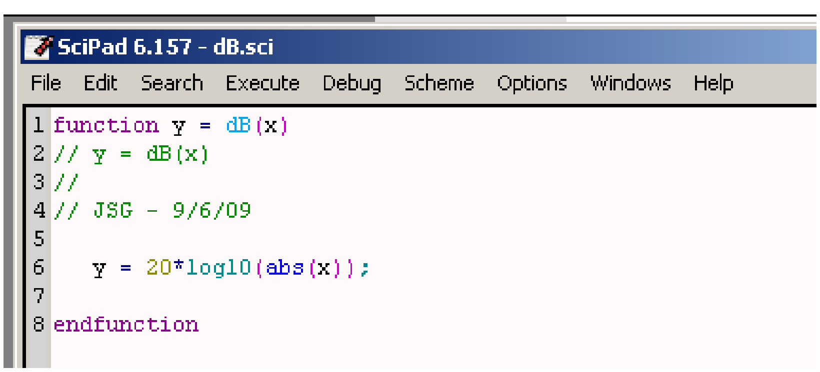
Suppose you want to add a function 'dB' which is called as

y = dB(x)

In SciLab, go to File Change Current Directory. Change it to where your files are located.



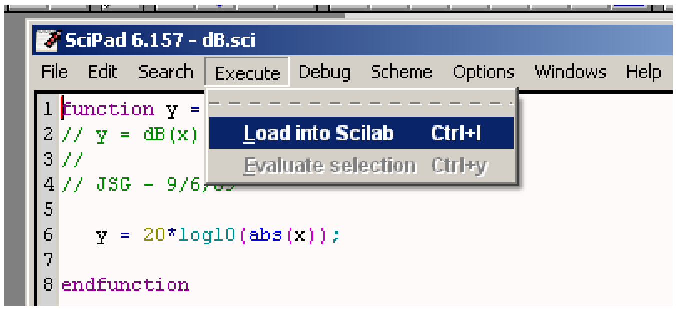
Go to File Open a file. Open up a .sci file. Change it as follows:



note: SciLab is very picky.

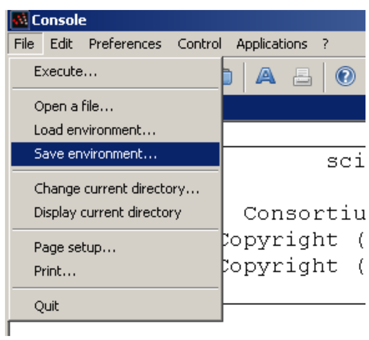
- The file name must patch the function name
- Both are case sensitive

Once you have a file you're happy with, add this to SciLab (execute - load into Scilab)



You now have the function 'dB' which you can call in Scilab!

note: If you create a bunch of files, you can load them into Scilab one at a time. Save the environment. When you restart Scilab, load the environment and all the functions that were in Scilab before are there again.



SciLab Control Systems Toolbox:

The control systems toolbox in Scilab is free for you to use. It's still a work in progress (started Sep 6, 2009), so more stuff should appear shortly.

The idea is to let you add and manipulate transfer functions in Scilab. The functions are as follows:

note: You probably have to

- Copy the control toolbox files over to your computer and unzip them
- In SciLab, load each file into Scilab one at a time.
- Save your environment.

Yea, loading in each file is a pain, but you only have to do it once. From then on, just load the environment.

poly: Find a polynomial which has a given set of poles:

```
->P = poly([-1,-2,-3,-4])
 P  =

    1.    10.    35.    50.    24.
```

$$s^4 + 10s^3 + 35s^2 + 50s + 24 = (s+1)(s+2)(s+3)(s+4)$$

sspack: Save a system in packed matrix form

```
-->G = sspack(A,B,C,D)
 G  =

    0.      1.     0.
  - 10.   - 2.     20.
    1.      0.     0.
```

note: $G = \begin{bmatrix} A & B \\ C & D \end{bmatrix}$. This assumes a single-input, single-output system, so the last row and last column gives your B, C, and D.

ssunpack: Extract the A, B, C, D matrix from G:

```
-->[A,B,C,D] = ssunpack(G)
```

tf2ss2: Input a system in transfer funciton form. For example: $G = \left(\frac{5s+6}{s^2+3s+2}\right)$

```
-->G = tf2ss2([5,6],[1,3,2])
 G  =

    0.     1.     0.
  - 2.   - 3.     1.
    6.     5.     0.
```

zp2ss: Input a system in zeros, poles, gain form: For example: $G = \left(\frac{5(s+2)}{(s+3)(s+4)}\right)$

```
-->G = zp2ss(-2,[-3,-4],5)
 G  =

    0.     1.    0.
  - 12.  - 7.    1.
    10.    5.    0.
```

eig: Compute the poles (eigenvalues) of a system in packed form:

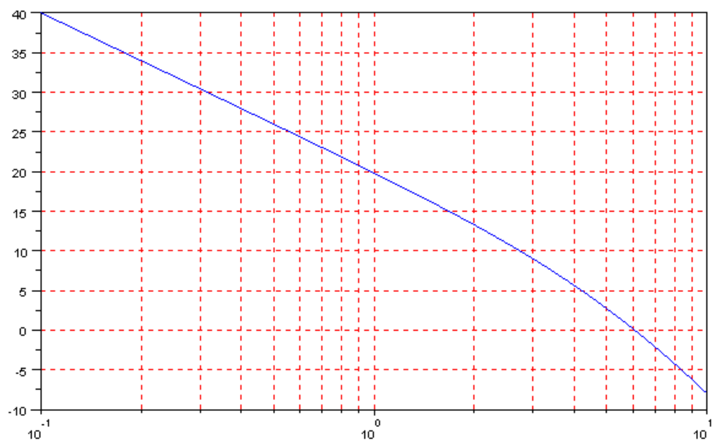
```
-->eig(G)
 ans  =

  - 3.
  - 4.
```

evalfr: Compute G(s). Find $G = \left(\frac{5(s+2)}{(s+3)(s+4)}\right)_{s=j3}$

```
-->G = zp2ss(-2,[-3,-4],5);
-->evalfr(G, j*3)

    0.7666667 - 0.3666667i
```

bode2: Compute G(jw) at a bunch of frequencies:

```
-->w = logspace(-2,2,100);
-->Gw = bode2(G,w);

-->plot(w,dB(Gw));
```



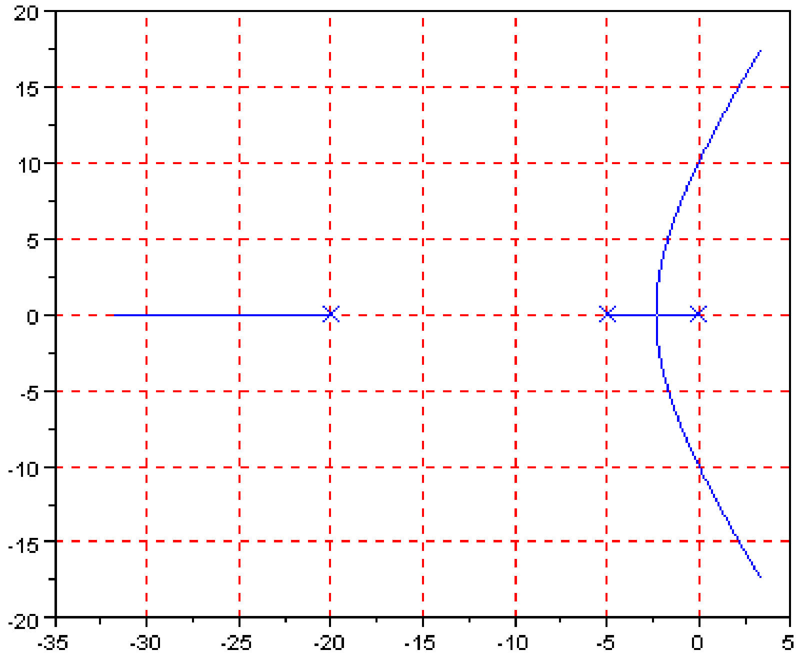
intcon: Connect two systems with unity feedback.

```
-->G

    0.     1.    0.
```

```
  - 12.  - 7.     1.
    10.    5.     0.

-->intcon(G,3)

    0.     1.     0.
  - 42.  - 22.    3.
    10.    5.     0.
```

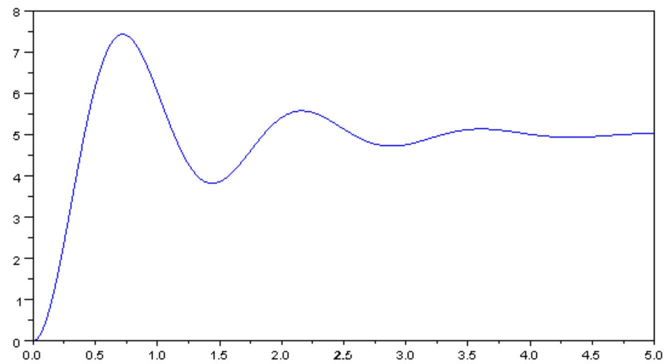
rlocus: Plot the root locus for a unity feedback system with several gains:

```
k = logspace(-2,1,100);
locus = rlocus(G,k);
```



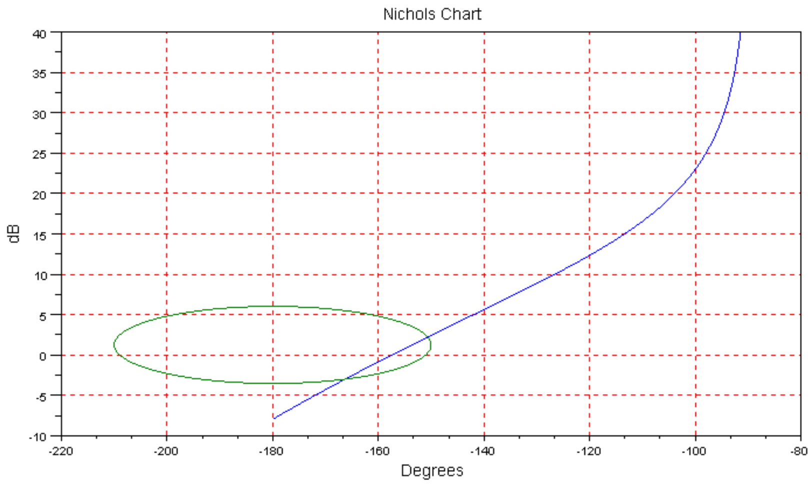
step: Compute the step response of a system

```
-->G = tf2ss2(100,[1,2,20])
-->t = [0:0.01:5]';
-->y = step(G,t);
-->plot(t,y);
```



Nichols: Plot the Nichols chart of G(jw) along with an M-circle Mm = 2 = 6dB.

```
G = zp2ss([],[0,-5,-20],1000);
w = logspace(-1,1,100);
Gw = bode2(G,w);
nichols(Gw,2);
```



leastsq(): Minimization of a Function:

note: MATLAB's version is call fminsearch().

A *very* useful tool in SciLab is the funciton *leastsq()*. This is a nonlinear optimization routine that finds the minimum of a function. For example, suppose you wanted to find the solution to

$$x\ln(x) + x = 3$$

First, create a function in SciLab which returns a minimum for x solving the above equation. One way to do this is compute the error and square it:

$$y = (x\ln(x) + x - 3)^2$$

In SciLab, create a function (I called it 'cost2.sci' for lack of a better name).

```
function y = cost2(x)
// y = cost(z)

  e = x*log(x) + x - 3;
  y = e*e;

  endfunction
```

Execute and load this into SciLab. You can use trial and error to find x. Keep guessing until cost2 returns zero

```
-->cost2(2)
    0.1492233
-->cost2(3)
    10.862541
-->cost2(2.1)
    0.4330541
-->cost2(1.9)
    0.0142856
```

The answer is close to 1.9. Or, you can use leastsqfn

```
-->[e,x] = leastsq(cost2,1.7)
 x  =
    1.8545507
 e  =
    1.458D-31
```

The answer is 1.8545507.

Problem: Find the values of a..e which make the following transfer function as close to an ideal low pass filter with a corner at 3 rad/sec

$$G(s) = \left(\frac{as^2+bs+c}{s^3+ds^2+es+c}\right)$$

Solution: Use leastsq(). Pass a 5x1 array containing a..e. Compute the difference in gain of G(s) from an ideal low pass filter. Return the sum squared difference.

```
function y = cost(z)
// y = cost(z)
// call with [e,y] = leastsq(cost,[1,2,1,4,4])

   a = z(1);
   b = z(2);
   c = z(3);
   d = z(4);
   e = z(5);
   j = sqrt(-1);
   w = [0:0.1:10]';
   s = j*w;
```

```
    Gideal = 1*(w<3);
    Gs = (a*s.^2 + b*s + c) ./ (s.^3 + d*s.^2 + e*s + c);
    plot(w,Gideal,w,abs(Gs));
    e = abs(Gideal) - abs(Gs);
    y = sum(e.^2);

endfunction
```

Note that the plot command plots the ideal low pass filter along with your current guess. This slows down the routine, but it's interesting to see how the optimization routine progresses.

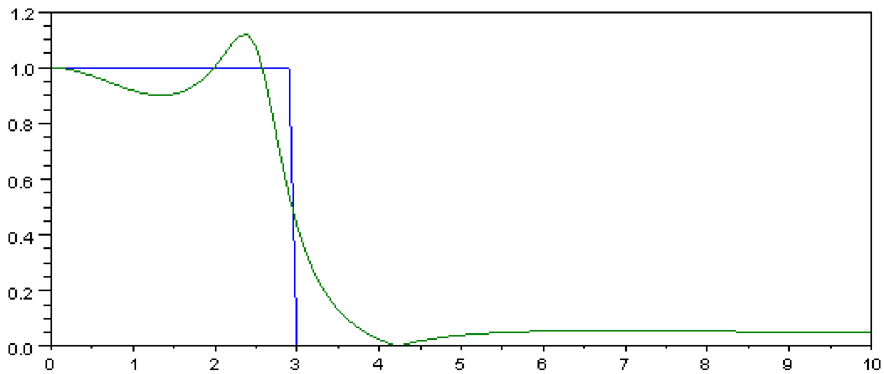
The result is

```
->[e,y] = leastsq(cost,[1,2,1,4,4])
 y  =
     0.5571236    8.901D-08   9.9814208    2.3253089   7.8440956
 e  =
     1.2097191
```

so

$$G(s) = \left(\frac{0.557s^2+9.98}{s^3+2.325s^2+7844s+9.98} \right)$$

with the following gain vs. frequency:



Modeling Electrical Circuits

Objective:

- Be able to write the voltage node equations for an electrical circuit
- Be able to place these dynamics in state-space form
- Be able to find the transfer funciton for an electrical circuit

Electronic Circuits:

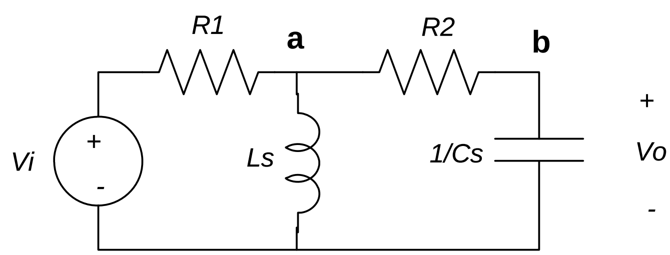
Circuits with elements which store energy (inductors and capacitors) are dynamic systems (as opposed to static systems which have only resistors). The transfer function for a circuit can be obtained by substituting the LaPlace value of the element's impedance:

	V/I Relationship	LaPlace value of Resistance
Resistor	V = I R	R
Capacitor	$V = \frac{1}{C}\int I \cdot dt$	1 / sC
Inductor	$V = L\frac{dI}{dt}$	L s

Current loops, voltage nodes, or other circuit analysis techniques can then be used to find the transfer function.

Example:

Find the transfer function from Vi to Vo:



Solution:

Writing the voltage node equations at nodes a and b:

a) $$\left(\frac{V_a - V_i}{R_1}\right) + \left(\frac{V_a}{Ls}\right) + \left(\frac{V_a - V_b}{R_2}\right) = 0$$

b) $$\left(\frac{V_b - V_a}{R_2}\right) + \left(\frac{V_b}{1/Cs}\right) = 0$$

Separating into common terms results in

a) $\left(\frac{1}{R_1}+\frac{1}{Ls}+\frac{1}{R_2}\right)V_a-\left(\frac{1}{R_1}\right)V_i-\left(\frac{1}{R_2}\right)V_b=0$

b) $\left(\frac{1}{R_2}+Cs\right)V_b-\left(\frac{1}{R_2}\right)V_a=0$

Note that a pattern emerges (which will follow in mechanical examples:

At Node a:
(The sum of the admittance's to node a)Va - (the sum of the admittances from node a to b)Vb - ...)
= (The current to node a)

Now, you can solve 2 equations for 2 unknowns by getting Va to drop out. For example,

$$AV_a-BV_i-CV_b=0 \qquad \text{(x C)}$$

$$+ \qquad DV_b-CV_a=0 \qquad \text{(xA)}$$

$$= \qquad -(BC)V_i+(AC-C^2)V_b=0$$

$$V_b=\left(\frac{BC}{AC-C^2}\right)V_i$$

$$V_b=\left(\frac{\left(\frac{1}{R_1}\right)\left(\frac{1}{R_2}\right)}{\left(\frac{1}{R_1}+\frac{1}{R_2}+\frac{1}{Ls}\right)\left(\frac{1}{R_2}\right)-\left(\frac{1}{R_2}\right)^2}\right)V_i=G(s)\cdot V_i$$

If you live long enough, you can simplify G(s).

Placing the Dynamics in State Space Form

If MATLAB is available, you can find the transfer funciton using state-space representation. The goal is to place the system in the form of

$$sX=AX+BU$$

$$Y=CX+DU$$

For example, take the circuit above. Rewrite the two dynamics equations as

$$\left(\frac{Ls}{R_1}+1+\frac{Ls}{R_2}\right)V_a-\left(\frac{Ls}{R_1}\right)V_i-\left(\frac{Ls}{R_2}\right)V_b=0$$

$$\left(\frac{L}{R_1}+\frac{L}{R_2}\right)sV_a-\left(\frac{L}{R_1}\right)sV_i-\left(\frac{L}{R_2}\right)sV_b=-V_a \qquad \text{(a)}$$

$$(C)sV_b=\left(\frac{-1}{R_2}\right)V_b+\left(\frac{1}{R_2}\right)V_a=0 \qquad \text{(b)}$$

In matrix form

$$\begin{bmatrix}\left(\frac{L}{R_1}+\frac{L}{R_2}\right) & \left(\frac{-L}{R_1}\right)\\ 0 & C\end{bmatrix}\begin{bmatrix}sV_a\\ sV_b\end{bmatrix}=\begin{bmatrix}-1 & 0\\ \frac{1}{R_2} & \frac{-1}{R_2}\end{bmatrix}\begin{bmatrix}V_a\\ V_b\end{bmatrix}+\begin{bmatrix}\frac{L}{R_2}\\ 0\end{bmatrix}sV_i$$

Inverting the matrix on the left:

$$\begin{bmatrix}sV_a\\ sV_b\end{bmatrix}=\begin{bmatrix}\left(\frac{L}{R_1}+\frac{L}{R_2}\right) & \left(\frac{-L}{R_1}\right)\\ 0 & C\end{bmatrix}^{-1}\begin{bmatrix}-1 & 0\\ \frac{1}{R_2} & \frac{-1}{R_2}\end{bmatrix}\begin{bmatrix}V_a\\ V_b\end{bmatrix}+\begin{bmatrix}\left(\frac{L}{R_1}+\frac{L}{R_2}\right) & \left(\frac{-L}{R_1}\right)\\ 0 & C\end{bmatrix}^{-1}\begin{bmatrix}\frac{L}{R_2}\\ 0\end{bmatrix}sV_i$$

$$Y=\begin{bmatrix}0 & 1\end{bmatrix}\begin{bmatrix}V_a\\ V_b\end{bmatrix}+[0]sV_i$$

(trick): The input is the derivative of Vi. If you ignore this 's' term times Vi, you're off by a derivative. Let's be off by a derivative for now.

$$sX=AX+BU$$

$$Y^*=CX$$

If you differentiate the output, you get the derivative back

$$sY^*=CsX=C(AX+BU)$$

so change the output to add this derivative:

$$Y=(CA)X+(CB)U$$

The state space dynamics are then

$$A=\begin{bmatrix}\left(\frac{L}{R_1}+\frac{L}{R_2}\right) & \left(\frac{-L}{R_1}\right)\\ 0 & C\end{bmatrix}^{-1}\begin{bmatrix}-1 & 0\\ \frac{1}{R_2} & \frac{-1}{R_2}\end{bmatrix}$$

$$B=\begin{bmatrix}\left(\frac{L}{R_1}+\frac{L}{R_2}\right) & \left(\frac{-L}{R_1}\right)\\ 0 & C\end{bmatrix}^{-1}\begin{bmatrix}\frac{L}{R_2}\\ 0\end{bmatrix}$$

$$C=\begin{bmatrix}0 & 1\end{bmatrix}A$$

$$D=\begin{bmatrix}0 & 1\end{bmatrix}B$$

Systems Modeled by the Heat Equation

Objectives:

Be able to

- Write the differential equations for a system modeled by the heat equation
- Write the electrical RC circuit equivalent for such a system
- Write the state-space equations for a heat system
- Find the transfer function using MATLAB
- Given the transfer function, predict how the system will respond to a step input

Definition: The Heat Equation:

If you want to model the temperature behaviour of a system, you could break the system into a large number of finite elements. The temperature at each element is proportional to the energy in that element. The energy at this node can flow to neighboring elements and is described by a first-order differential equation.

At each node:

$$\frac{dx_i}{dt} = -a_{ii}x_i + \sum_{i \neq j} a_{ij}x_j$$

where xi is the temperature at node i and aij are real positive constants:

$$a_{ij} > 0$$

$$a_{ii} > \sum_{i \neq j} a_{ij}$$

The last requirement is conservation of energy: the energy gained by neighboring nodes cannot be greater than the energy lost at node i. If could be less (if you have a lossy, i.e. poorly insulated) system, however.

RC Circuit Analogy:

The electrical circuit which also satisfied the heat equation is a passive RC network with capacitors to ground and resistors connecting the capacitors. For example, suppose you wanted to model the temperature along a long rod. If you treat this as five finite elements, each element has

- Thermal intertia: the energy stored is proportional to the temperature at that node, and
- Thermal conductivity: Between elements, energy can flow and is proportional to the temperature difference between adjacent nodes times the thermal conductivity.

The electrical analogy is an RC network where each voltage node has

- A capacitor to ground: the energy stored is porportional to the voltage at the node, and
- Resistance: between each capacitor, resistors are palced. The current flow is proportional to the voltage between adjacent nodes times the electrical conductuvity of that resistor.

The analogy is

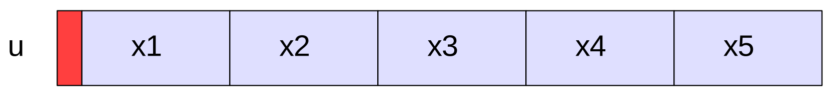
Heat	Thermal Inertia (J / degree)	Temperature (degrees C)	Heat Flow (Watts)	Thermal Resistance (degree C / Watt)

Electrical	Capacitance (J / volt)	Voltage (Volts)	Current (Amps)	Resistance (R) (Ohm = Volts / Amp)

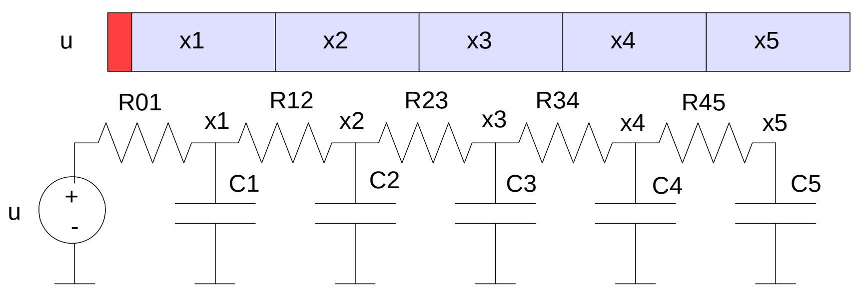
1-Dimenstional Example:

For example, find the mathematical model for the temperature along a long thin rod. For convenience, assume you split this rod into five finite elements. For each element,

- Assume each element has a thermal inertia of C Joules / degree. (the volume of each element times the thermal capacitance of the material)
- Between elements, assume that 1 Watts flows if the temperature difference is R degrees. (Thermal conductivity times area divided by distance).



The elecrical analogy is an RC circuit is as follows:



The differential equations for this system are then:

$$C_1 \frac{dx_1}{dt} = -\left(\frac{1}{R_{01}} + \frac{1}{R_{12}}\right) x_1 + \left(\frac{1}{R_{12}}\right) x_2 + \left(\frac{1}{R_{01}}\right) u$$

$$C_2 \frac{dx_2}{dt} = -\left(\frac{1}{R_{12}} + \frac{1}{R_{23}}\right) x_2 + \left(\frac{1}{R_{12}}\right) x_1 + \left(\frac{1}{R_{23}}\right) x_3$$

$$C_3 \frac{dx_3}{dt} = -\left(\frac{1}{R_{23}} + \frac{1}{R_{34}}\right) x_3 + \left(\frac{1}{R_{23}}\right) x_2 + \left(\frac{1}{R_{34}}\right) x_4$$

$$C_4 \frac{dx_4}{dt} = -\left(\frac{1}{R_{34}} + \frac{1}{R_{45}}\right) x_4 + \left(\frac{1}{R_{34}}\right) x_3 + \left(\frac{1}{R_{45}}\right) x_5$$

$$C_5 \frac{dx_2}{dt} = -\left(\frac{1}{R_{45}}\right) x_5 + \left(\frac{1}{R_{45}}\right) x_4$$

In State Space form:

$$s\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} -\left(\frac{1/R_{01}+1/R_{12}}{C_1}\right) & \left(\frac{1/R_{12}}{C_1}\right) & 0 & 0 & 0 \\ \left(\frac{1/R_{12}}{C_2}\right) & -\left(\frac{1/R_{12}+1/R_{23}}{C_2}\right) & \left(\frac{1/R_{23}}{C_2}\right) & 0 & 0 \\ 0 & \left(\frac{1/R_{23}}{C_3}\right) & -\left(\frac{1/R_{23}+1/R_{34}}{C_3}\right) & \left(\frac{1/R_{34}}{C_3}\right) & 0 \\ 0 & 0 & \left(\frac{1/R_{34}}{C_4}\right) & -\left(\frac{1/R_{34}+1/R_{45}}{C_4}\right) & \left(\frac{1/R_{45}}{C_4}\right) \\ 0 & 0 & 0 & \left(\frac{1/R_{45}}{C_5}\right) & -\left(\frac{1/R_{45}}{C_5}\right) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} + \begin{bmatrix} \left(\frac{1}{C_1 R_{01}}\right) \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} u$$

If C = 0.01F and R = 10 Ohms,

$$s\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} = \begin{bmatrix} -20 & 10 & 0 & 0 & 0 \\ 10 & -20 & 10 & 0 & 0 \\ 0 & 10 & -20 & 10 & 0 \\ 0 & 0 & 10 & -20 & 10 \\ 0 & 0 & 0 & 10 & -10 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} + \begin{bmatrix} 10 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} u$$

Note that this is a lossless system: there is no thermal conductivity bewteen the rod and the environment. This shows up in the above matrix with each row adding to zero. If there were losses, the diagonal elements would be more negative and the rows would sum to a negative number (lossy).

Problem: Estimate how this system will behave:

Solution: I need to know the systems dominant pole(s) and DC gain. In MATLAB:

```
A = [-20,10,0,0,0;10,-20,10,0,0;0,10,-20,10,0;0,0,10,-20,10;0,0,0,10,-10];
B = [10;0;0;0;0];
C = [0,0,0,0,1];
D = 0;

eig(A)
 {-0.8101, -6.9028, -17.1537, -28.3083, -36,8251}

DC = C*inv(A)*B
   DC = 1.00000
```

The system has a DC gain of one. If the input is increased to 100C, the output will go to 100C as well.

The system has a dominant pole at -0.8101. It will take approximately 4.93 seconds (4/0.8101) to reach steady state.

The dominant pole is real. There should be no overshoot or oscillations in the step response.

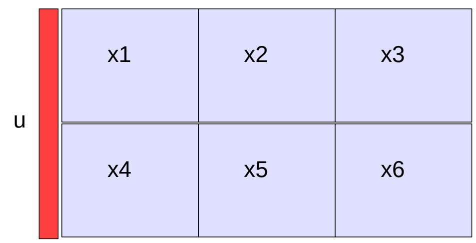
Problem: Determine the actual step response:

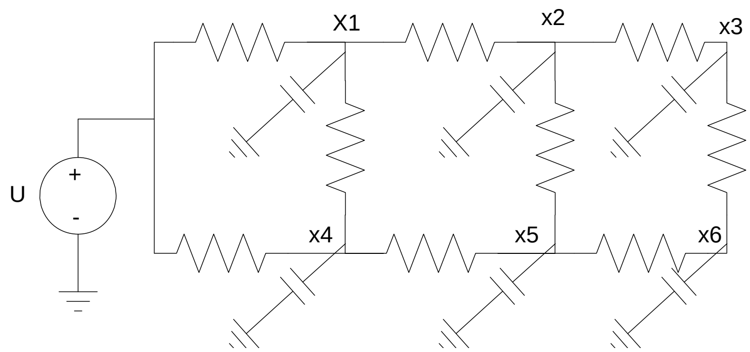
Solution: Using MATLAB:

```
G = ss(A,B,C,D);
zpk(G)

t = [0:0.1:10]';
y = step(G,t);
plot(t,y)
```

2-Dimenstional Heat Equations:

The above also works for plates (2 dimenstions), and solids (3-dimenstions). For example, find the heat flow for a plate, modeled as five finite elements:





Assuming all resitances are 10 Ohms and C = 0.01F again,

$$s\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} = \begin{bmatrix} -30 & 10 & 0 & 10 & 0 & 0 \\ 10 & -30 & 10 & 0 & 10 & 0 \\ 0 & 10 & -20 & 0 & 0 & 10 \\ 10 & 0 & 0 & -30 & 10 & 0 \\ 0 & 10 & 0 & 10 & -30 & 10 \\ 0 & 0 & 10 & 0 & 10 & -20 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} + \begin{bmatrix} 10 \\ 0 \\ 0 \\ 10 \\ 0 \\ 0 \end{bmatrix}$$

Problem: Estimate what the step response from U to x3 will look like

Solution: In MATLAB:

```
A = [-30,10,0,10,0,0;10,-30,10,0,10,0;0,10,-20,0,0,10];
A = [A; A(:,[4,5,6,1,2,3]));
B = [10;0;0;10;0;0];
C = [0,0,1,0,0,0];
D = 0;
eig(A)
 -32.46, -35.54, -21.98, -15.54, -52.46, -1.98

DC = -C*inv(A)*B
   DC = 1.0000
```

- The system has a DC gain of one. If the input is increased to 100C, the output will go to 100C as well.

- The system has a dominant pole at -1.98. It will take approximately 2.02 seconds (4/1.98) to reach steady state.
- The dominant pole is real. There should be no overshoot or oscillations in the step response.

Translational Mechanical Systems:

Mass and spring systems also are dynamic systems. One way to model these systems is to

- 1) Replace the mass, spring, and friction terms with their LaPlace admittance,
- 2) Redraw the system as an electric circuit, and
- 3) Write the voltage node equations.

The LaPlace admittance's come from modeling the system as

Force = mass * acceleration

F = Z X (F = force, Z = admittance, X = displacement)

which has the electrical analog

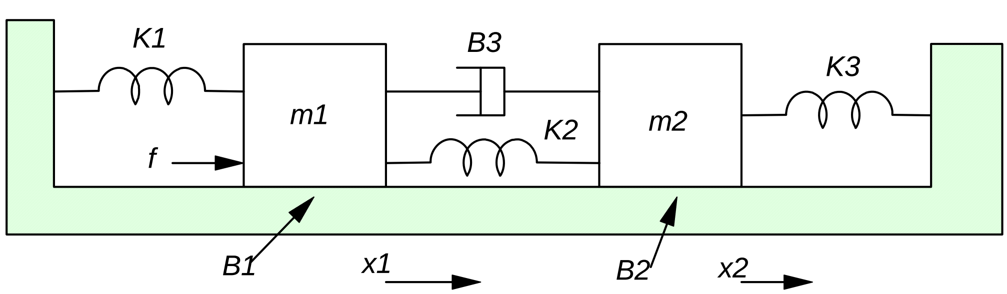
I = G V (current = admittance * voltage).

Mechanical System	**Electrical Analog**
Force in the positive direction	Current to the node
Displacement in the positive direction	Positive voltage
Mass, (see below)	Admittance

	F = Z X Relationship	LaPlace value of Addmittance
Mass	f = m x"	s^2 m
Spring	f = k x	k
Friction	f = B x' f = f_v x'	sB s f_v

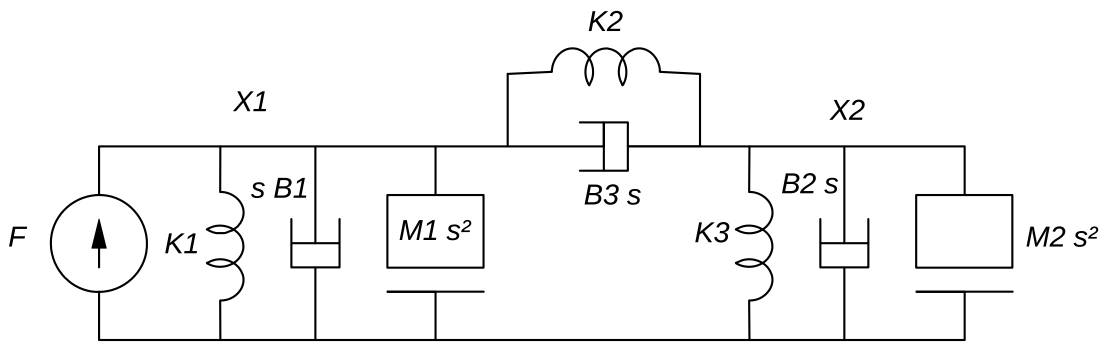
Example:

Write the equations of motion for the following mass and spring system:



Solution:

Step 1: Draw the circuit equivalent:



Step 2: Write the voltage node equations. From before

(The sum of the admittance's to node a)Va - (the sum of the admittances from node a to b)Vb - ...)

= (The current to node a)multiply a: times (1/R2) and

$$(K_1 + B_1 s + M_1 s^2 + K_2 + B_3 s)X_1 - (K_2 + B_3 s)X_2 = F$$

$$(M_2 s^2 + B_2 s + K_3 + K_2 + B_3 s)X_2 - (K_2 + B_3 s)X_1 = 0$$

Step 3: Solve to find the output as a function of the input. If X_2 is the output and F is the input, solve to find

$$X_1 = G(s) \cdot F$$

For this system use dummy variables for the coefficients of X_1 and X_2 in the above equations:

$AX_1 - BX_2 = F$ (multiply by B)

$-BX_1 + CX_2 = 0$ (multiply by A and add)

= $(-B^2 + AC)X_2 = BF$

$$X_2 = \left(\frac{B}{AC-B^2}\right)F$$

or substituting in the values of A, B, C

$$X_2 = \left(\frac{K_2+B_3 s}{\left(K_1+B_1 s+M_1 s^2+K_2+B_3 s\right)\left(M_2 s^2+B_2 s+K_3+K_2+B_3 s\right)-(K_2+B_3 s)^2}\right)F$$

Rotational Systems

Dynamics of Rotational Systems

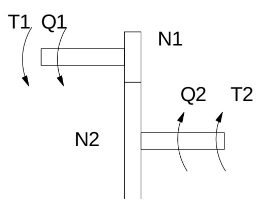
Rotational systems are very similar to translational systems. The analogy used is to treat inertia, springs, and friction as admittanced. The only difference is the standard symbol for intertia (J vs M), spring (K vs k) and friction (D vs B).

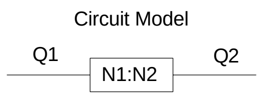
	F = Z X Relationship	LaPlace value of Addmittance
Mass	f = J x"	Js^2
Spring	f = K x	K
Friction	f = D x' $f = f_v$ x'	Ds $s\ f_v$

Rotational systems often times have gears as well. A gear will speed up or slow down the rotation of a system by a factor N. Two nodes conected with a single gear only have one degree of freedom: once the angle of one node is defined, the angle of the other is fexed by the gear. The relationship between the angle of two nodes connected with a gear is

$$N_1\theta_1 = N_2\theta_2$$

which is signified as:





For example, if N2 is larger then N1, Q2 will be slower than Q1 by the turn ratio

$$N_1 Q_1 = N_2 Q_2$$

$$Q_2 = \left(\frac{N_1}{N_2}\right) Q_1$$

Torqe increases if you go from a small (fast spinning) gear to a larger one as the turn ratio

$$T_2 = \left(\frac{N_2}{N_1}\right) T_1$$

If there is an admittance at Q1

$$T_1 = Y_1 Q_1$$

this will be seen at node 2 as

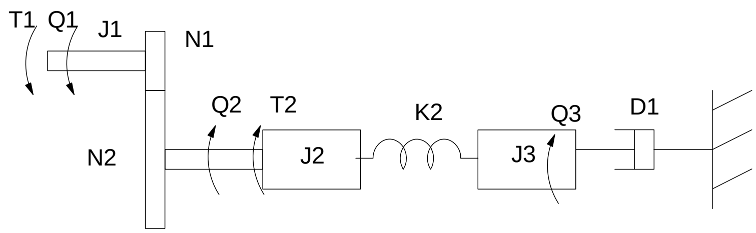
$$\left(\frac{N_1}{N_2}\right) T_2 = Y_1\left(\left(\frac{N_2}{N_1}\right) Q_2\right)$$

$$T_2 = \left(\frac{N_2}{N_1}\right)^2 Q_2$$

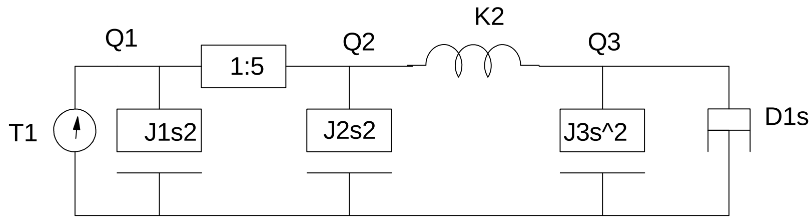
The admittance seen through a gear is changed by the turn ratio squared.

The impedance seen on the rapidly spinning side is reduced by the turn ratio squared.

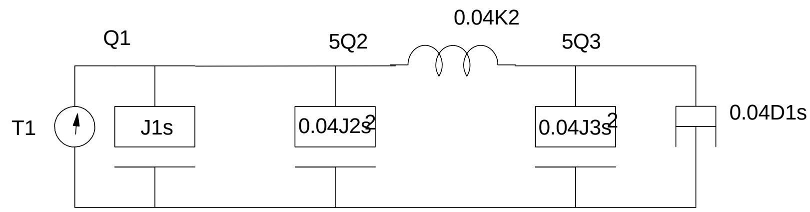
Example: Determine the transfer function from T1 to Q1:



First draw the circuit diagram using LaPlace admittances.



Next, transfer all the admittances through the gear to Q1's side. Since you're going right to left, the admittances will transfer as $\left(\frac{1}{5}\right)^2$



You can now write your voltage node equations

$$(J_1 s^2 + 0.04 J_2 s^2 + 0.04 K_2) Q_1 - (0.04 K_2) Q_3^* = T_1$$

$$(0.04 J_3 s^2 + 0.04 D_1 s + 0.04 K_2) Q_3^* - (0.04 K_2) Q_1 = 0$$

Since the admittance translates as the gear ratio squared, it allows you to minimize the effect of whatever is on the other side of the gear by using a large gear reduction. This is often done in robotics, where 300:1 gear reduction is not uncommon (the motor sees the outside world, scaled down by a factor of 90,000!)

Servo Motors

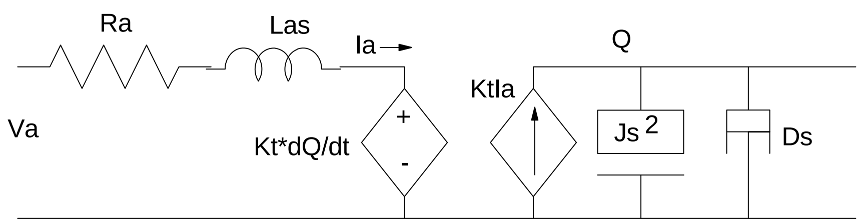
Transfer Functions for DC Servo Motors:

The equations that describe a DC servo motor are

$$\text{Back EMF} = K_t \omega = K_t \frac{dQ}{dt}$$

$$\text{Torque} = K_t I_a$$

where Kt is the torque constant, Q is the angle, and Ia is the armature current. The circuit model for a DC servo motor is then



or

$$V_a = K_t(sQ) + (R_a + L_a s)I_a$$

$$K_t I_a = (Js^2 + Ds)Q$$

Solving gives

$$Q = \left(\frac{1}{s}\right)\left(\frac{K_t}{(Js+D)(L_a s+R_a)+K_t^2}\right)V_a$$

if the output is angle (Q) or

$$\omega = \frac{dQ}{dt} = \left(\frac{K_t}{(Js+D)(L_a s+R_a)+K_t^2}\right)V_a$$

if the output is speed.

Example: Determine the transfer function for a DC servo motor with the following specifications:

(http://www.servosystems.com/electrocraft_dcbrush_rdm103.htm)

- $K_t = 0.03 Nm/A$
- Terminal resistance = 1.6 (Ra)
- Armature inductance = 4.1mH (La)
- Rotor Inertia = 0.008 oz-in/sec/sec *ugh - english units*
- Damping Contant = 0.25 oz-in/krpm *double ugh*
- Torque Constant = 13.7 oz-in/amp
- Peak current = 34 amps
- Max operating speed = 6000 rpm

You need to translate to metric:

Kt: $$13.7\frac{oz-in}{amp}\left(\frac{1m}{39.4in}\right)\left(\frac{1lb}{16oz}\right)\left(\frac{0.454kg}{lb}\right)\left(\frac{9.8N}{kg}\right) = 0.0967\left(\frac{N\cdot m}{A}\right)$$

D: $$0.25\left(\frac{oz\cdot in\cdot \min}{krev}\right)\left(\frac{1Nm}{141.7oz-in}\right)\left(\frac{60\sec}{\min}\right)\left(\frac{krev}{1000rev}\right)\left(\frac{rev}{2\pi rad}\right) = 1.68\cdot 10^{-5}\left(\frac{Nm}{rad/\sec}\right)$$

J: $$0.008\left(\frac{oz-in}{\sec^2}\right)\left(\frac{Nm}{141.7oz-in}\right)\left(\frac{kg}{9.8N}\right) = 5.76\cdot 10^{-6}\left(\frac{kg\cdot m}{s^2}\right)$$

Plugging in numbers:

$$Q = \left(\frac{10.31\cdot 630^2}{s(s^2+393s+630^2)}\right)V = \left(\frac{10.31(630)^2}{s(s+196.5+j598)(s+196.5-j598)}\right)V$$

This means....

- If you apply a constant +12V to the motor, the motor spins at a speed of 123.72 rad/sec. (10.31*12)
- It takes about 20ms for the motor to get up to speed (Ts = 4/196)

The motor will overshoot it's final speed (123 rad/sec) by 35% (damping ratio = 0.3119)

LaGrangian Formulation of System Dynamics

Find the dynamics of a nonlinear system:

Circuit analysis tools work for simple lumped systems. For more complex systems, especially nonlinear ones, this approach fails. The Lagrangian formulation for system dynamics is a way to deal with any system.

Definitions:

KE Kinetic Energy in the system

PE Potential Energy

$\frac{\partial}{\partial t}$ The partial derivative with respect to 't'. All other variables are treated as constants.

$\frac{d}{dt}$ The full derivative with respect to t.

$$\frac{d}{dt} = \frac{\partial}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial}{\partial y}\frac{\partial y}{\partial t} + \frac{\partial}{\partial z}\frac{\partial z}{\partial t} + \ldots$$

L Lagrangian = KE - PE

Procedure:

1) Define the kinetic and potential energy in the system.

2) Form the Lagrangian:

$$L = KE - PE$$

3) The input is then

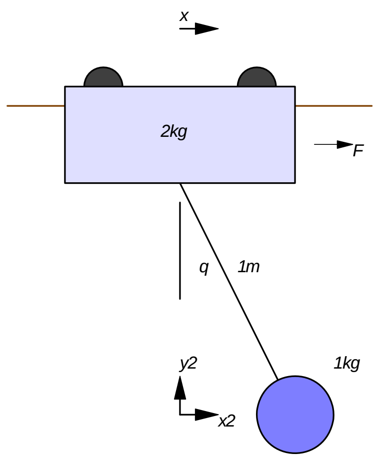
$$F_i = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}_i}\right) - \frac{\partial L}{\partial x_i}$$

where F_i is the input to state x_i. Note that

- If x_i is a position, F_i is a force.
- If x_i is an angle, F_i is a torque

Example:

Find the dynamics for the following system: A 1kg mass swings from a gantry which weighs 2kg. The length of the rope is 1m.



1) Write the energy in the system

Mass #1:

- $x_1 = x$
- $y_1 = 0$

$$KE_1 = \tfrac{1}{2}mv^2 = \tfrac{1}{2}\cdot 2\cdot \dot{x}^2$$

$$PE_1 = mgh = 0$$

Mass #2

- $x_2 = x_1 + \sin\theta$ $\quad\quad \dot{x}_2 = \dot{x}_1 + (\cos\theta)\dot{\theta}$
- $y_2 = 1 - \cos\theta$ $\quad\quad \dot{y}_2 = (\sin\theta)\dot{\theta}$

$$KE_2 = \tfrac{1}{2}\left(\dot{x}_2^2 + \dot{y}_2^2\right)$$

$$KE_2 = \tfrac{1}{2}\left(\dot{x}_1^2 + 2\dot{x}_1\dot{\theta}_1\cos\theta + \cos^2\theta\cdot\dot{\theta}^2 + \sin^2\theta\cdot\dot{\theta}^2\right)$$

$$KE_2 = \tfrac{1}{2}\left(\dot{x}_1^2 + 2\dot{x}_1\dot{\theta}_1\cos\theta + \dot{\theta}^2\right)$$

$$PE_2 = mgh = gy_2$$

$$PE_2 = g - g\cos\theta$$

So, the Lagrangian is

$$L = \left(\dot{x}^2 + \tfrac{1}{2}\left(\dot{x}^2 + 2\dot{x}\dot{\theta}\cos\theta + \dot{\theta}^2\right)\right) - (g - g\cos\theta)$$

The force on the base is

$$F = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) - \frac{\partial L}{\partial x}$$

$$F = \frac{d}{dt}\left(2\dot{x} + \dot{x} + 2\dot{\theta}\cos\theta\right) - 0$$

$$F = 3\ddot{x} + 2\ddot{\theta}\cos\theta - 2\dot{\theta}^2\sin\theta$$

The torque on the rod is

$$T = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) - \frac{\partial L}{\partial \theta}$$

$$T = \frac{d}{dt}\left(\dot{x}\cos\theta + \dot{\theta}\right) - \dot{x}\dot{\theta}\sin\theta - 9.8\sin\theta$$

$$T = \ddot{x}\cos\theta + \dot{x}\dot{\theta}\sin\theta + \ddot{\theta} - \dot{x}\dot{\theta}\sin\theta - g\sin\theta$$

So, the dynamics are

$$\begin{bmatrix} 3 & 2\cos\theta \\ \cos\theta & 1 \end{bmatrix}\begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 2\dot{\theta}^2\sin\theta \\ g\sin\theta \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} F$$

Linear Model

For small perturbations about 0,

$$\sin\theta \approx \theta$$

$$\cos\theta \approx 1$$

$$\dot{\theta}^2 \approx 0$$

$$g = 9.8 \text{ m/s}^2$$

The dynamics linearized about zero are then

$$\begin{bmatrix} 3 & 2 \\ 1 & 1 \end{bmatrix}\begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 \\ 9.8\theta \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} F$$

$$\begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 1 & -2 \\ -1 & 3 \end{bmatrix}\left(\begin{bmatrix} 0 \\ 9.8\theta \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} F\right)$$

or

$$\begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} -19.6\theta + F \\ 29.4\theta - F \end{bmatrix}$$

which is a linear differential equation. You can put this in state-space form:

$$s\begin{bmatrix} x \\ \theta \\ x' \\ \theta' \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -19.6 & 0 & 0 \\ 0 & 29.4 & 0 & 0 \end{bmatrix}\begin{bmatrix} x \\ \theta \\ x' \\ \theta' \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \\ -1 \end{bmatrix} F$$

Note that the eigenvalues of 'A' are

$$\{0, 0, 5.42, -5.42\}$$

corresponding to an unstable system. Here, gravity is pulling up, causing the pendulum to fall. (A sign error in 'g').

Let g = -9.8 m/s^2:

$$s\begin{bmatrix} x \\ \theta \\ x' \\ \theta' \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 19.6 & 0 & 0 \\ 0 & -29.4 & 0 & 0 \end{bmatrix}\begin{bmatrix} x \\ \theta \\ x' \\ \theta' \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \\ -1 \end{bmatrix} F$$

results in the poles being

$$\{0, 0, j5.42, -j5.42\}$$

The corresponding eigenvectors are (same order as above)

$$\left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} -0.1 \\ 0.15 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ -0.54 \\ 0.81 \end{pmatrix} \right\} \qquad \text{note: states are } \begin{pmatrix} x \\ \theta \\ \dot{x} \\ \dot{\theta} \end{pmatrix}$$

- The first eigenvalue says that the position decays as e^{0t}. If you displace the cart by 1m, it sits there.
- The second eigenvalue says that the cart's velocity decays as e^{0t}. If the swinging motion has stopped and the cart is moving right at 1m/s, it keeps moving right at 1m/s.
- The third eigenvalue says that something oscillates at 5.42 rad/sec. The eigenvector says that if you displace the cart 0.1m left and the angle 0.15 rad right, the cart will swing back and forth in place at a frequency of 5.42 rad/sec.
- The fourth eigenvalue says that something oscillates at 5.42 rad/sec. The eigenvector says that if the cart is at (x=0, angle=0), but has an initial velocity of (0.54m/s left, 0.81 rad/sec right), the cart will swing back and forth in place at a frequency of 5.42 rad/sec.

Error Constants and Steady-State Error

In addition to being stable, you might want to know how good your feedback system is. Steady-state error describes how well a feedback system can track certain types of inputs. Error constants relate to steady-state error and allow you to assign a number to a feedback system, the bigger the number the better the system is.

Error Constants: Most control systems are designed to track constant or slowly changing inputs. Likewise, the DC gain of a system tells you how the system will behave in steady-state (assuming the system is stable, of course.) Error constants are little more than the DC gain of a system.

One problem arises, however, if a system has a pole at s=0. If this is the case, the DC gain is infinity:

$$G(s)_{s\to 0} = \lim_{s\to 0}\left(\frac{G'(s)}{s}\right) = \infty$$

So that you can assign meaningful numbers to this type of system, several types of error constants are defined:

Definitions:

Type N System: The plant, G(s), has N poles at s=0.

Error Constants: Kp, Kv, Ka. The DC gain of a plant:

Kp: The DC gain of a type-0 system

- $K_p = \lim_{s\to 0}(G(s))$
- $\lim_{s\to 0}(G(s)) = K_p$

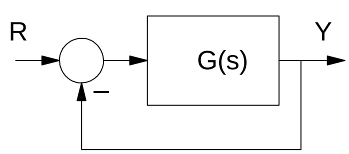
Kv: The DC gain of a type-1 system:

- $K_v = \lim_{s\to 0}(s \cdot G(s))$
- $\lim_{s\to 0}(G(s)) = \frac{K_v}{s}$

Ka: The DC gain of a type-2 system:

- $K_a = \lim_{s\to 0}(s^2 \cdot G(s))$
- $\lim_{s\to 0}(G(s)) = \frac{K_z}{s^2}$

Steady-State Error: The steady-state error is the difference between the desired output and actual output (R-Y) for a feedback system:



Three standard inputs are used to test how well a system behaves:

Unit Step:

- $R(s) = \frac{1}{s}$
- Historically modeled trying to point an anti-aircraft gun at a German bomber in WWII.

- Models tracking a constant setpoint, such as holding the temperature of a room at 72F, regulating the flow through a pipe to 0.1m3/s, holding the speed of a car at 55mph, etc.

Unit Ramp:

- $R(s) = \frac{1}{s^2}$
- Historically modeled trying to track a German bomber moving at a constant speed across the sky.
- Models tracking a setpoint which is rising or falling at a constant rate.

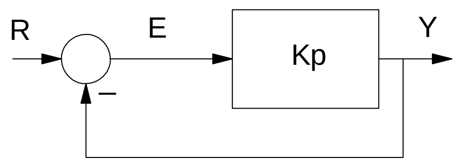
Unit Parabola

- $R(s) = \frac{1}{s^3}$
- I'm not sure what this models.

Error constants tell you how well the output tracks the input.

Type-0 Systems:

Assume that at DC G(s)=Kp.



The error wil then be

$$Y = \left(\frac{K_p}{K_p+1}\right)R$$

$$E = R - Y = \left(\frac{1}{K_p+1}\right)R$$

i) if R(s) is a unit step:

$$E = \left(\frac{1}{K_p+1}\right)\left(\frac{1}{s}\right)$$

$$e = \left(\frac{1}{K_p+1}\right) \quad \text{(t>0)}$$

A type-0 system will have a constant error of $\left(\frac{1}{K_p+1}\right)$ when tracking a unit step input

ii) if R(s) is a unit ramp:

$$E(s) = \left(\frac{1}{K_p+1}\right)\left(\frac{1}{s^2}\right)$$

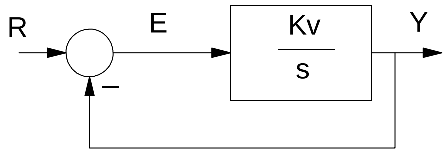
$$e(t) = \left(\frac{1}{K_p+1}\right)t$$

As $t \to \infty$, the error goes to ∞.

A type-0 system cannot track a unit ramp input.

Type-1 Systems:

Assume at DC $G(s) = \frac{K_v}{s}$:



Then

$$Y = \left(\frac{\frac{K_v}{s}}{\frac{K_v}{s}+1}\right)R = \left(\frac{K_v}{K_v+s}\right)R$$

$$E = R - Y = \left(\frac{s}{K_v+s}\right)R$$

i) R = a unit step:

$$E = \left(\frac{s}{K_v+s}\right)\left(\frac{1}{s}\right) = \left(\frac{0}{s}\right) + \left(\frac{1}{K_v+s}\right)$$

$$e(t) = 0 + transient \qquad (t>0)$$

$$e(\infty) = 0$$

A type-1 system can track a constant set point with no error.

ii) R = a unit ramp:

$$E = \left(\frac{s}{K_v+s}\right)\left(\frac{1}{s^2}\right) = \left(\frac{1}{K_v+s}\right)\left(\frac{1}{s}\right) = \left(\frac{\frac{1}{K_v}}{s}\right) + \left(\frac{c}{s+K_v}\right)$$

$$e(t) = \left(\frac{1}{K_v}\right) + transients$$

$$e(\infty) = \left(\frac{1}{K_v}\right)$$

A type-1 system can track a unit ramp with a constant error of $\left(\frac{1}{K_v}\right)$

iii) R = a unit parabola:

$$E = \left(\frac{s}{K_v+s}\right)\left(\frac{1}{s^3}\right) = \left(\frac{1}{K_v+s}\right)\left(\frac{1}{s^2}\right) = \left(\frac{\frac{1}{K_v}}{s^2}\right) + \left(\frac{c}{s}\right) + \left(\frac{d}{s+K_v}\right)$$

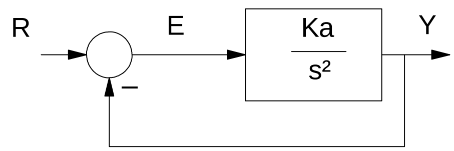
$$e = \left(\frac{1}{K_v}\right)t + \ldots$$

$$e(\infty) = \infty$$

A type 1 system cannot track a unit parabola.

Type-2 systems:

Assume G(s) is a type-2 system:



In this case

$$Y = \left(\frac{\frac{K_a}{s^2}}{\frac{K_a}{s^2}+1}\right)R = \left(\frac{K_a}{K_a+s^2}\right)R$$

$$E = R - Y = \left(\frac{s^2}{K_a+s^2}\right)R$$

i) If R is a unit step:

$$E = \left(\frac{s^2}{K_a+s^2}\right)\frac{1}{s} = 0 + \left(\frac{c}{s^2+K_a}\right)$$

$$e(t) = 0 + (transients)$$

$$e(\infty) = 0$$

A type-2 system can track a constant set-point with no error.

ii) if R is a unit ramp:

$$E = \left(\frac{s^2}{K_a+s^2}\right)\frac{1}{s^2} = 0 + \left(\frac{1}{s^2+K_a}\right)$$

$$e(t) = 0 + (transients)$$

$$e(\infty) = 0$$

A type-2 system can track a ramp input with no error

iii) Unit parabola:

$$E = \left(\frac{s^2}{K_a+s^2}\right)\frac{1}{s^3} = \left(\frac{\frac{1}{K_z}}{s}\right) + \left(\frac{c}{s^2+K_a}\right)$$

$$e(t) = \left(\frac{1}{K_z}\right) + transients$$

$$e(\infty) = \left(\frac{1}{K_a}\right)$$

A type-2 system can track a unit parabola with a constant error of $\left(\frac{1}{K_a}\right)$

Summary:

Computation of Error Constants			
	Kp	Kv	Ka
Type 0	$\lim_{s\to 0}(G(s))$	0	0
Type 1	∞	$\lim_{s\to 0}(s\cdot G(s))$	0
Type 2	∞	∞	$\lim_{s\to 0}(s^2\cdot G(s))$

Steady-State Error			
	Unit Step	Unit Ramp	Unit Parabola
Type 0	$\left(\frac{1}{K_p+1}\right)$	∞	∞
Type 1	0	$\left(\frac{1}{K_v}\right)$	∞
Type 2	0	0	$\left(\frac{1}{K_a}\right)$

Routh Criteria & Stability

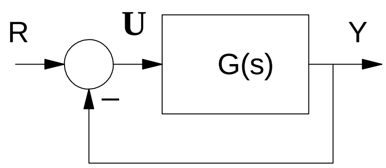
Objectives:

In this section we will look at

- Finding the transfer function for a feedback system
- Determining the stability of a system using Routh criteria.

Stability:

One problem with using feedback is that the transfer function chances. For example, consider the following feedback system:



Assume that G(s) is a stable system, such as

$$Y = \left(\frac{100}{(s+1)(s+2)(s+3)}\right)U$$

With feedback, the transfer function becomes

$$Y = \left(\frac{G}{1+G}\right)R$$

$$Y = \left(\frac{100}{(s+1)(s+2)(s+3)+100}\right)R$$

or factoring

$$Y = \left(\frac{100}{(s-0.3567+j3.9575)(s-0.3567-j3.9575)(s+6.7134)}\right)R$$

Note that the closed-loop system is unstable in spite of the open-loop system being a well behaved stable system.

In general, feedback can improve the response of a system. It can also make the system worse, such as the above example where it made the system unstable. The question then may arise as to when a system will be stable.

The first method we'll use to determine stability uses a Routh Table. Note that stability is determined by when the denominator polynomial is negative definite. The general problem is then

Problem: Determine if a given polynomial

$$a_n s^n + a_{n-1} s^{n-1} + a_{n-2} s^{n-2} + ... + a_1 s + a_0 = 0$$

is negative definite.

Solution: Set up a Routh Table.

1) The first two rows take the even and coefficients of the polynomial and odd terms respectively:

an	an-2	an-4	an-6	...
an-1	an-3	an-5	an-7	...

2) Using the two previous rows, create the next row as

Previous two rows:	a	b	c	d	...
	e	f	g	h	...
New Row	$\frac{-\begin{vmatrix} a & b \\ e & f \end{vmatrix}}{e}$	$\frac{-\begin{vmatrix} a & c \\ e & g \end{vmatrix}}{e}$	$\frac{-\begin{vmatrix} a & d \\ e & h \end{vmatrix}}{e}$	etc.	

3) Repeat step 2 for the next row. Repeat until all entries are zero.

4) Count the number of sign flips in column #1. The number of sign flips is equal to the number of positive (unstable) roots to this polynomial.

Example: Determine if

$$(s+1)(s+2)(s+3)(s+4) = s^4 + 10s^3 + 35s^2 + 50s + 24$$

has any unstable roots.

Solution: Set up the Routh table. The first two rows are

1	35	24	0	0
10	50	0	0	0

The next rows are

1	35	24	0
10	50	0	0
$\frac{-\begin{vmatrix} 1 & 35 \\ 10 & 50 \end{vmatrix}}{10} = 30$	$\frac{-\begin{vmatrix} 1 & 24 \\ 10 & 0 \end{vmatrix}}{10} = 24$	$\frac{-\begin{vmatrix} 1 & 0 \\ 10 & 0 \end{vmatrix}}{10} = 0$	0
$\frac{-\begin{vmatrix} 10 & 50 \\ 30 & 24 \end{vmatrix}}{30} = 42$	$\frac{-\begin{vmatrix} 10 & 0 \\ 30 & 0 \end{vmatrix}}{30} = 0$	$\frac{-\begin{vmatrix} 10 & 0 \\ 30 & 0 \end{vmatrix}}{30} = 0$	
$\frac{-\begin{vmatrix} 30 & 24 \\ 42 & 0 \end{vmatrix}}{42} = 24$	$\frac{-\begin{vmatrix} 30 & 0 \\ 42 & 0 \end{vmatrix}}{42} = 0$		
$\frac{-\begin{vmatrix} 42 & 0 \\ 24 & 0 \end{vmatrix}}{24} = 0$	$\frac{-\begin{vmatrix} 42 & 0 \\ 24 & 0 \end{vmatrix}}{24} = 0$		

There are no sign flips. Hence, this polynomial is stable. (The system is stable).

Example 2: Determine the range of K for which the following system is stable:

$$Y = \left(\frac{K}{(s+1)(s+2)(s+3)+K}\right)R$$

Solution:

i) Factor the denominator polynomial:

$$(s+1)(s+2)(s+3)+K = s^3+6s^2+11s+6+K$$

ii) Insert the coefficients into the first two rows of a Routh Table:

1	11	0	
6	6+K	0	

iii) Fill in the rest of the table:

1	11	0	
6	6+K	0	
$\frac{60-K}{6}$	0	0	

One property of Routh Tables is you can scale any row by a positive constant and not change the results. Multiplying by 6 to keep the numbers pretty:

1	11	0	
6	6+K	0	
60-K	0	0	
6+K	0	0	
0	0	0	

iv) Select K so that there are *no* sign flips in column #1:

1	11	0	
6	6+K	0	
60-K	0	0	K < 60
6+K	0	0	K > -6
0	0	0	

The valid range of K will be those values which satisfy every condition in the right column:

(-6 < K < 60)

Note: at K = -6, the roots to $s^3+6s^2+11s+6+K$ are {0, -3+j1.4142, -3-j1.4142}. (One pole is on the jw axis)

At K=+60, the roots are {-6, +j3.3166, -j3.3166}. (Two poles are on the jw axis)

Root Locus

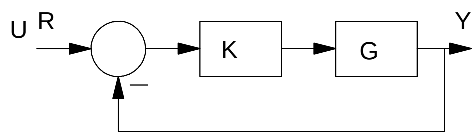
Objectives:

Understand

- With feedback, the poles of the closed-loop system shift
- The shift is a well defined curve, termed a root locus plot

Definitions:

Assume a feedback system of the following form:



where

$$k \cdot G(s) = \left(\frac{k \cdot z(s)}{p(s)}\right)$$

Open-Loop Gain: k G(s)

Closed-Loop Gain: $\left(\frac{kG}{1+kG}\right)$

Open-Loop Poles: solutions to p(s) = 0

Open-Loop Zeros: solutions to z(s) = 0.

Closed-Loop Poles: solutions to p(s) + kz(s) = 0.

Root Locus: Plot of the closed-loop poles as k varies from 0 to infinity.

Background:

Feedback is a very useful method for controlling the output of a system. With feedback, it is much easier to regulate the output and automatically compensate for disturbances, such as noise, aging, manufacturing variations, etc. Feedback can also make a system behave worse - it can even make a system go unstable.

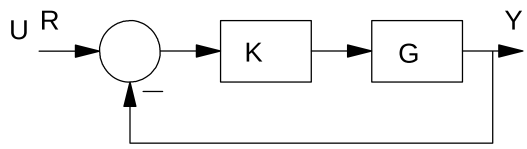
One property feedback has is that the dynamics of the closed-loop system change as you vary the feedback gain. Root locus techniques allow you to

- Predict how the closed-loop system will behave as you vary the feedback gain (k), and
- Pick the 'best' value of k to get the 'best' response possible.

In addition, root locus techniques help provide insight as to how to improve the system's response with a pre-filter.

What is a root locus plot

Assume you have a unity feedback system as follows:



where

$$KG = \left(\frac{kz(s)}{p(s)}\right)$$

As you vary k from 0 to infinity, the dynamics of the closed-loop system will vary. The closed-loop system will be

$$\left(\frac{kG}{1+kG}\right) = \left(\frac{k \cdot z(s)}{p(s)+k \cdot z(s)}\right) .$$

The roots of the closed-loop system determine how the system will behave. Hence, we are interested in the solutions to

$$p(s) + k \cdot z(s) = 0$$

A root locus plot is a plot of the solutions to p(s)+kz(s)=0 for 0<k<∞.

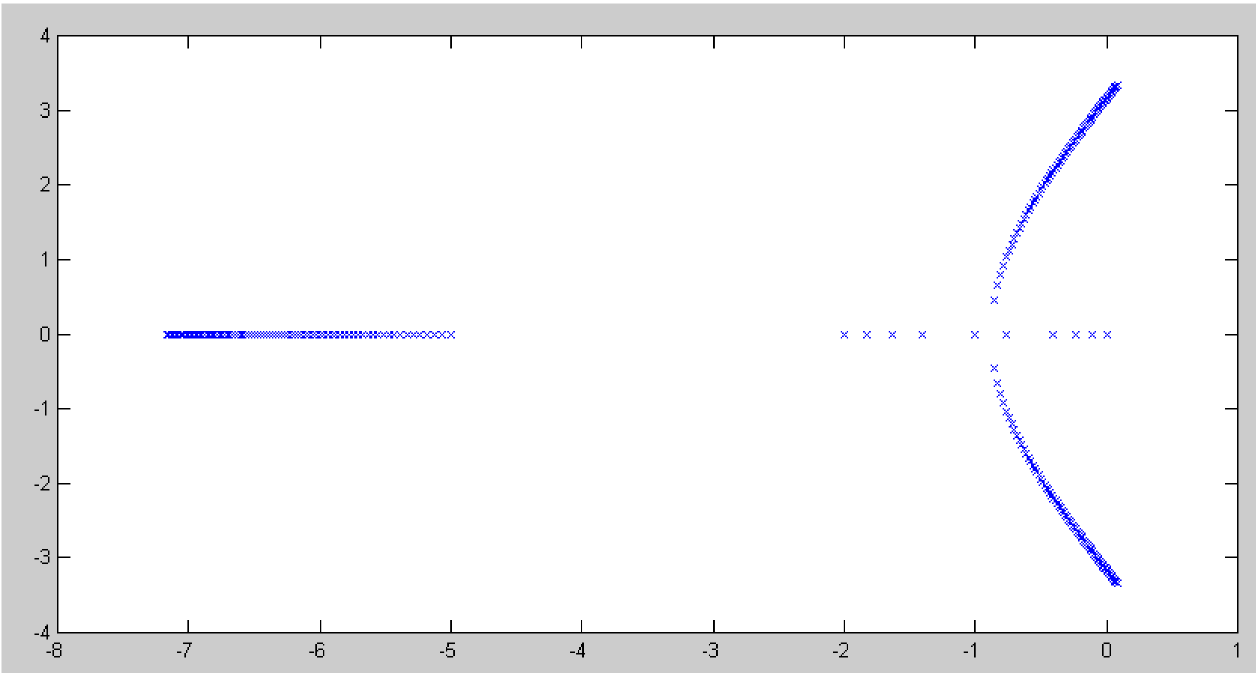
Example: Plot the root locus of

$$kG(s) = \left(\frac{k}{s(s+2)(s+5)}\right)$$

or, equivalently, the solution to

$$s(s+2)(s+5) + k = 0$$

Using MATLAB to do this with 1000 values of k results in the following plot:



Note the following:

- The roots follow a well defined path
- At k=0, the root locus starts at the poles of the open-loop system (0, -2, -5)
- As k increases, the roots slide together
- As k increases further the poles become complex
- As k gets really large, the poles go into the right half plane and the system becomes unstable.

This root locus plot is essentially your shopping list: you can place the closed-loop poles anywhere on the above root locus plot. If you pick a point on the root locus plot, there is a gain, k, which results in that solution. If you pick a point off the root locus plot, however, there is no solution for k.

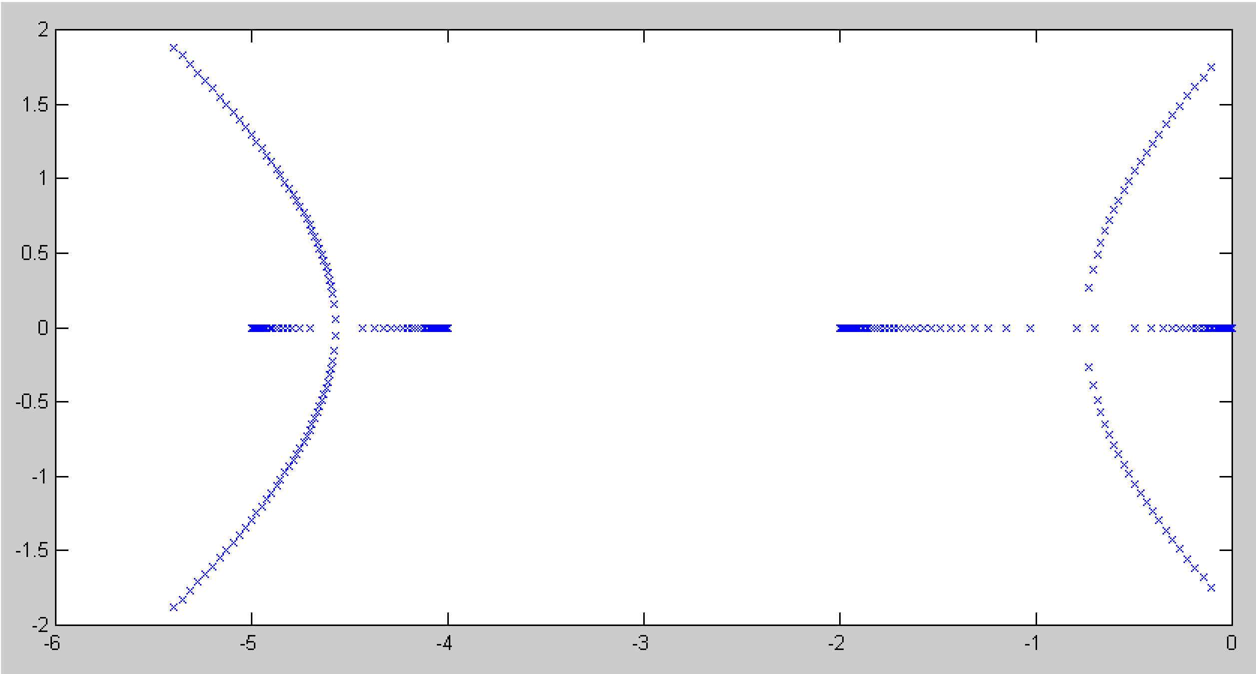
Example: Plot the root locus for the following system

$$G = \left(\frac{10}{s(s+2)(s+4)(s+5)} \right)$$

or equivalently, the roots of

$$s(s+2)(s+4)(s+5) + 10k = 0$$

Solution: Picking 1000 points for k results in the following root locus plot



Note again:

- The roots follow a well defined path
- At k=0, the root locus starts at the poles of the open-loop system (0, -2, -4, -5)
- As k increases, the roots slide together
- As k increases further the poles become complex
- As k gets really big, two of the poles become unstable

Sketching a Root Locus Plot

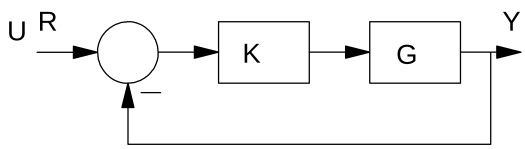
Objectives:

To be able to

- Predict how a feedback system will behave as you increase the feedback gain using a root-locus plot.
- Determine the 'best' feedback gain to use via root-locus techniques.
- Design a compensator to improve the response of a system using root-locus techniques.

Sketching a Root Locus Plot

Assume you have a unity feedback system as follows:



where

$$KG = \left(\frac{kz(s)}{p(s)}\right)$$

As you vary k from 0 to infinity, the dynamics of the closed-loop system will vary. The closed-loop system will be

$$\left(\frac{kG}{1+kG}\right) = \left(\frac{k\cdot z(s)}{p(s)+k\cdot z(s)}\right) .$$

The roots of the closed-loop system determine how the system will behave. Hence, we are interested in the solutions to

$$p(s) + k \cdot z(s) = 0$$

A root locus plot is a plot of the solutions to p(s)+kz(s)=0 for 0<k<∞.

Rewriting this

$$\frac{p(s)}{z(s)} = -k = k\angle 180^0$$

Since 's' can be complex, this graphical interpretation of $-k = \frac{p(s)}{z(s)}$ provides two pieces of information: one for amplitude and one for the angle:

1) The amplitude of both sides must match, meaning

$$k = \frac{\Pi(\text{distantes from the poles to point s})}{\Pi(\text{distantes from the zeros to point s})} \tag{1}$$

2) The angles must match:

$$180^0 = \sum(\text{angles from the poles to point s}) - \sum(\text{angles from the zeros to point s}) \tag{2}$$

Plotting all points on the 's' plane which satisfies the angle condition produces the root-locus. The root locus tells you how the system can behave. Once you pick a point on the root locus, k is found using the first equation.

Rules for Plotting the Root Locus:

Recall that the root-locus plot is a plot of all points which satisfy

$$p(s) + k \cdot z(s) = 0 \tag{3}$$

which is also a plot of all points which satisfy the angle condition. From (3), two solutions are obvious:

1. Poles:

At k=0, the roots start at the open-loop poles. (i.e. the solution to p(s)=0). Mark an 'X' at these points.

2. Zeros:

As $k \to \infty$, the roots go the open-loop zeros (i.e. the solution to z(s)=0). Mark an 'O' at these points.

3. Real Axis Loci:

Any point on the real axis where the angles add to 180 degrees will be on the root locus. Note that

- If a real pole is to the left of point 's', the angle from the pole to this point is zero degrees.
- If a real pole is to the right of point 's', the angle from the pole to point 's' is 180 degrees.
- Complex poles have no affect. Two poles to the left add zero degrees (twice). Two poles to the right add $360^0 = 0^0$.
- Zeros act like poles since $+180^0 = -180^0$.
- Adding 1800 an odd number of times produces 1800.
- Adding 1800 an even number of times produces 00.

Hence,

Real axis loci will be whenever there are an odd number of poles and zeros to the right.

4. Number of Asymptotes:

Assume there are m zeros and n poles. m of the poles will go the m zeros. The remainder go to infinity.

The number of asymptotes = n-m. (# of poles - # of zeros)

5. Asymptote Angle:

Assume that there are m zeros and n poles. Then, k is from

$$-k = \frac{p(s)}{z(s)} = \frac{s^n+...}{s_m+...}$$

If $k \rightarrow \infty$ then $s \rightarrow \infty$ as well. In this case, only the highest power of 's' matters:

$$-k = \frac{s^n}{s^m} = s^{n-m}$$

The angle is then

$$180^0 = (n-m)\angle\phi$$

where ϕ is the angle of 's'.

The asymptote angles are: $\phi = \frac{180^0 + k \cdot 360^0}{n-m}$ (k=0, 1, 2, ...)

6. Asymptote Intersect:

At infinity, the poles and zeros look like they are at the same spot on the 's' plane (relatively). This will be the 'average' pole and zero location

$$\text{Asymptote Intersect} = \frac{\sum(poles) - \sum(zeros)}{\#poles - \#zeros}$$

7. Breakaway Point:

If two real-axis poles come together, they will move away from the real axis at a point. Two ways to compute these points are

- Treat each pole as a - charge and each zero as a + charge. An electron at equilibrium on the real axis will be at a breakaway point.
- Solve $\frac{d}{ds}\left(\frac{p(s)}{z(s)}\right) = 0$

8. jw Crossings:

These are important since they tell you what gain makes the system go unstable:

$$angle(G(s))_{s=j\omega} = 180^0$$

9. Departure Angle:

If a pole or zero is complex, the departure angle is the angle at which the root locus leaves the pole or enters the zero. This is found by summing the angles to 180 degrees.

10. Gain at a point on the root-locus:

The root locus plots all possible locations that the system's roots can be for $0 < k < \infty$. The value of **k** at any particular point on the root locus comes from

$$\left(\frac{kz(s)}{p(s)}\right) = kG(s) = -1$$

.

The minus sign means that the angles must add to 180 degrees. The amplitude tells you the value of k.

Root Locus Summary p(s) + k z(s) = 0	
Poles	solution to p(s)=0
Zeros	solution to z(s)=0
Real Axis Loci:	There are an off number of poles or zeros to the right
Breakaway Point(s)	$\frac{d}{ds}\left(\frac{p(s)}{z(s)}\right) = 0$
jw crossing	ang(G(jw)) = 180^0
Number of asymptotes	n-m (#poles - #zeros)
Asymptote Angles	$\frac{180^0 + k \cdot 360^0}{n-m}$
Asymptote Intersect	$\frac{\sum poles - \sum zeros}{n-m}$
Gain at any point on the root-locus	$k \cdot G(s) = -1$

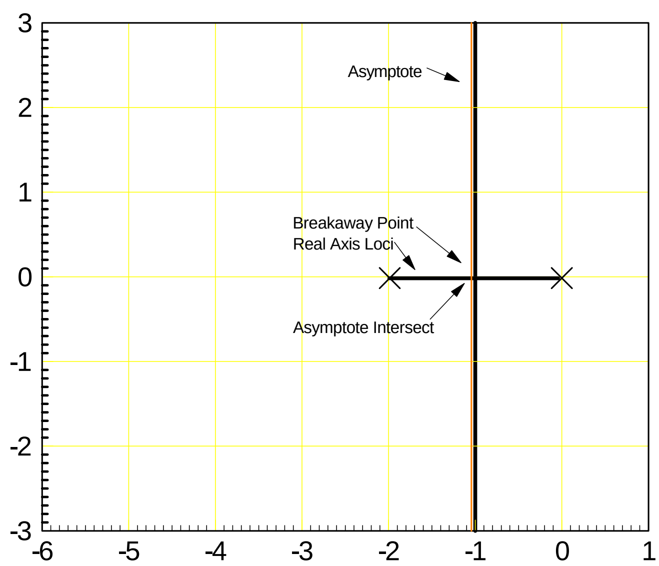
Problem 1a: Sketch the root locus of

$$G(s) = \left(\frac{1}{s(s+2)}\right)$$

Solution:

- Number of poles = 2, number of zeros = 0.
- Open-Loop poles at {0, -2}. Mark an 'X' at these points.
- Real Axis Loci: (0, -2)
- Number of asymptotes = 2
- Asymptote Angles = $\pm 90^0$ (note that the asymptotes split the s-plane into 2 equal regions)
- Asymptote Intersect: $\left(\frac{\sum poles - \sum zeros}{n-m}\right) = \left(\frac{-2}{2}\right) = -1$
- Breakaway Point: $\frac{d}{ds}(s(s+2)) = 2s+2 = 0.$ s=-1.
- jw crossings: s=j0.

The following is then the root locus plot:



Problem 1b: Describe how the previous system behaves as k increases from 0 to infinity:

Solution: From the root-locus, the closed loop poles start at the open-loop poles and move along the root locus.

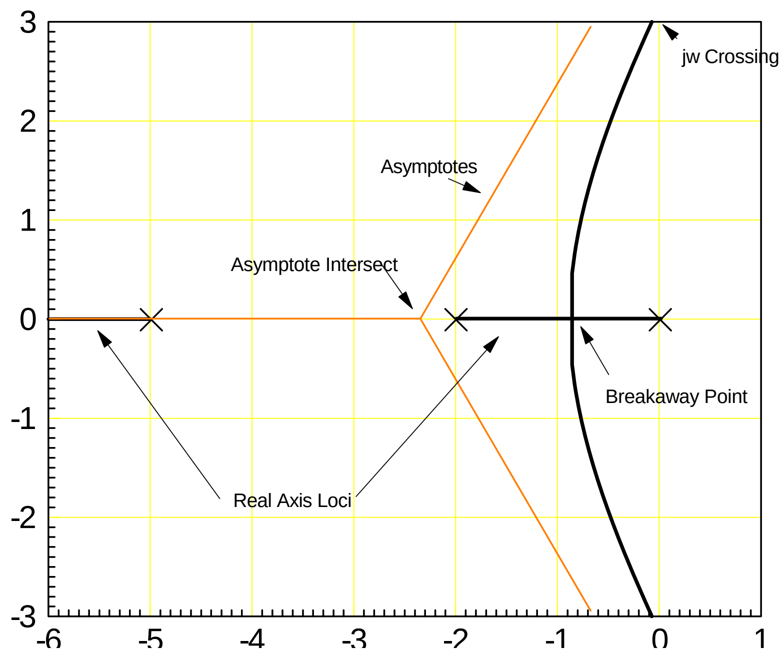
- For **k** small, the system behaves as a slow first-order system.
- As **k** increases, the system speeds up until it has a settling time of 4 seconds. (the pole at s=0 moves left to s=-1).
- When **k** becomes bigger than 1, the system has a settling time of 4 but starts to have some overshoot in the step response. (The real part of the dominant pole is at -1 but the complex part increases.)
- For large **k**, the system becomes more and more oscillatory but still has a 4 second settling time.

Problem #2a: Sketch the root locus of

$$\left(\frac{1}{s(s+2)(s+5)}\right)$$

Solution: This will be similar to the previous root locus only with a third pole at -5. This pole repels the root locus, pushing it into the right-half plane.

- Number of poles = 3, number of zeros = 0.
- Open-Loop poles at {0, -2, -5}. Mark an 'X' at these points.
- Real Axis Loci: (0, -2), (-5, $-\infty$)
- Number of asymptotes = 3
- Asymptote Angles = $\pm 60^0, 180^0$ (note that the asymptotes split the s-plane into 3 equal regions)
- Asymptote Intersect: $\left(\frac{\sum poles - \sum zeros}{n-m}\right) = \left(\frac{(0)+(-2)+(-5)}{3-0}\right) = -2.3333$
- Breakaway Point: $\frac{d}{ds}(s(s+2)(s+5)) = \frac{d}{ds}(s^3+7s^2+10s) = 3s^2+14s+10 = 0$ (s=-0.88, =3.79) The point (s=-0.88) is on the root locus and is the breakaway point.
- jw crossings: s=j3.16. $(G(j3.16)) = 0.0143\angle 180^0$
- The following is then the root locus plot



Note: The third pole at -5 has

- Pushed the root-locus to the right (causing it to go into the right-half plane)
- Pushed the breakaway point away from -1 (the midpoint between the former breakaway point)

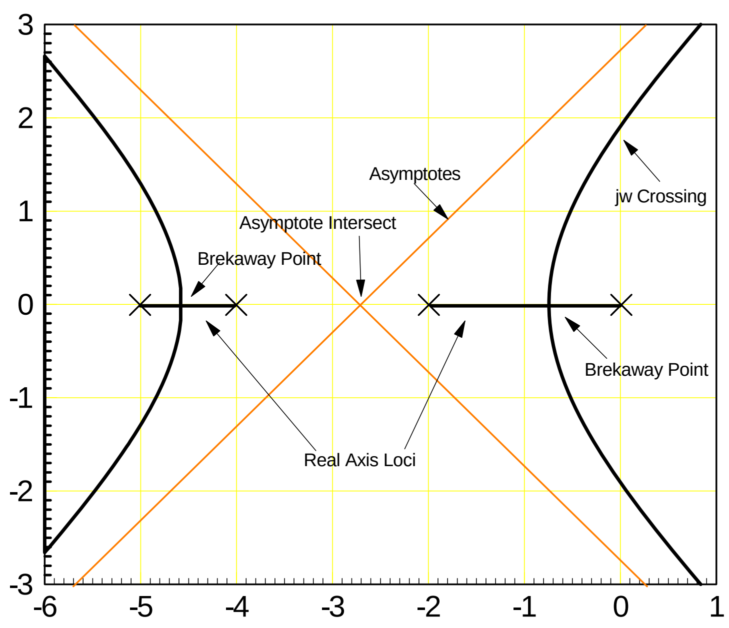
Example 3: Sketch the root locus of

$$\left(\frac{1}{s(s+2)(s+4)(s+5)}\right)$$

Solution:

- Number of poles = 4, number of zeros = 0.
- Open-Loop poles at {0, -2, -4, -5}. Mark an 'X' at these points.
- Open-Loop zeros: none
- Real Axis Loci: (0, -2), (-4, -5)
- Number of asymptotes = 4
- Asymptote Angles = $\pm 45^0, \pm 135^0$
- Asymptote Intersect: $\frac{((0)+(-2)+(-4)+(-5))}{4} = -2.75$
- Breakaway Point: $\frac{d}{ds}\left(\frac{s(s+2)(s+5)(s+10)}{s^2+2s+2}\right) = 0$ s = {-0.7438, -4.5771}
- jw crossings: $s = j1.9069$ $G(j1.9069) = 0.0080\angle 180^0$

The following is then the root locus plot

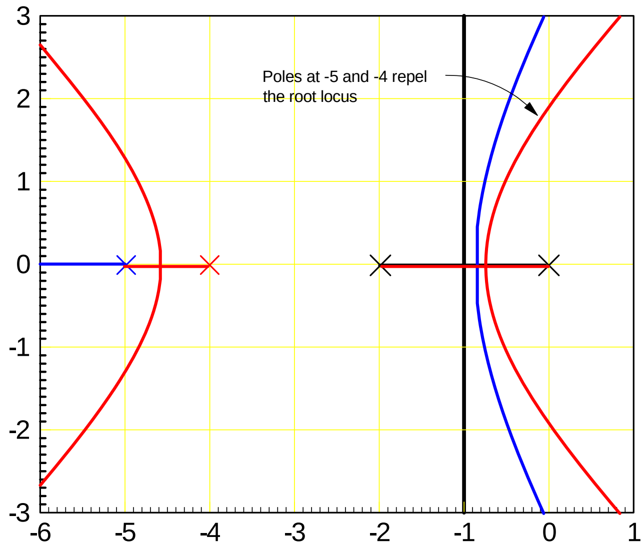


Note: The two pole at -4, and -5 have

- Pushed the root locus to the right even harder (now going right at an angle of 45 degrees)
- Pushed the breakaway point right of -1 (the former breakaway point).

Summary:

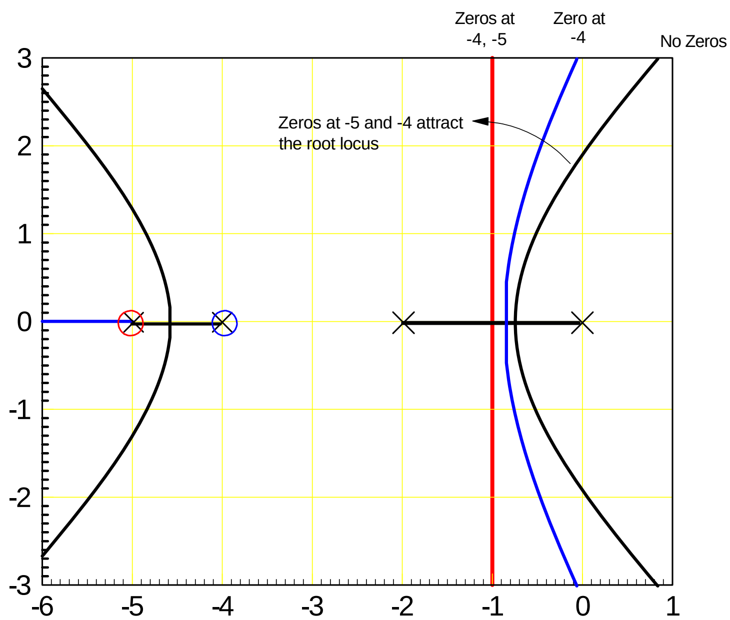
Note that poles repel the root locus. As you add poles at -4 and -5, the root locus is pushed further and further to the right.



Zeros behave the opposite way of poles: they attract the root locus. For example, if you compare

$$G(s) = \left(\frac{1}{s(s+2)(s+4)(s+5)}\right) \text{ vs } \left(\frac{(s+4)}{s(s+2)(s+4)(s+5)}\right) \text{ vs } \left(\frac{(s+4)(s+5)}{s(s+2)(s+4)(s+5)}\right)$$

the zeros at -4 and -5 will pull the roots towards them as shown below.



This will be a useful tool later on:

- If you want to push the root-locus to the right, add a pole to the left or slide fast pole right.
- If you want to pull a root-locus to the left, add zeros to the left to attract the root-locus.

Gain Compensation using Root Locus

Objectives:

- Pick a feedback gain t give a desired response (such as %OS)
- Specify how the resulting closed-loop system will behave

Intro

The root-locus plot tells you where you can place the poles of a feedback (ζ)system (i.e. it is the locus of the closed-loop system's roots - hence its name). Once you know how the system can behave, it is up to the design engineer to pick the 'best' behavior from these choices. The gain, k, that places you there is then found from

$$k = -\left(\frac{p(s)}{z(s)}\right)_{s*}$$

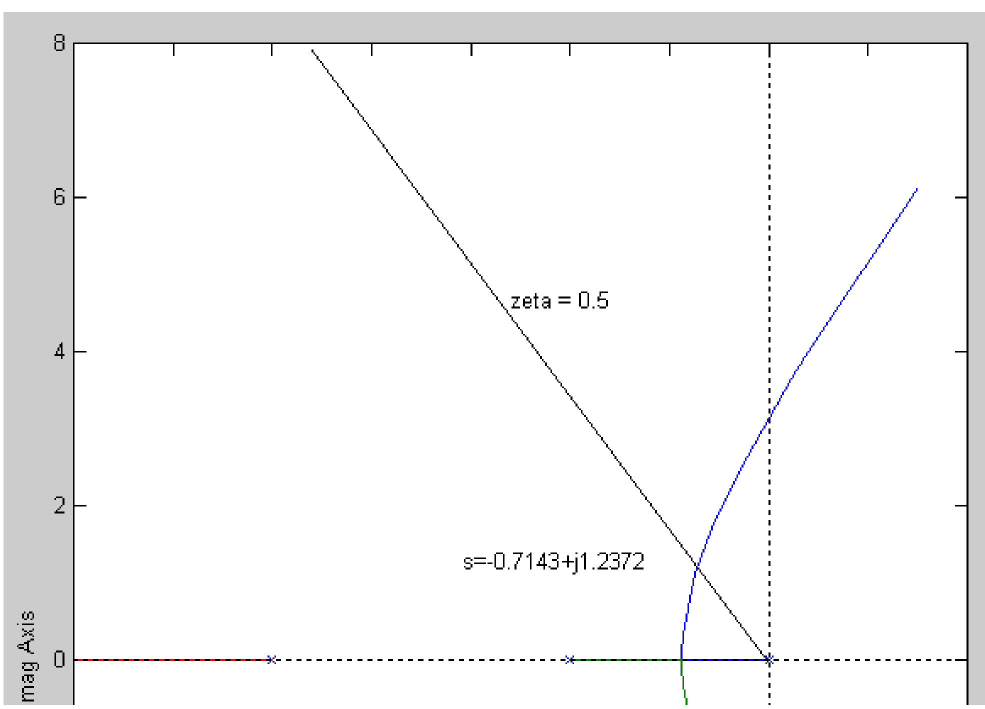
where s* is the spot on the root-locus you'd like to place the poles.

Example 1: Assume $G(s) = \left(\frac{20}{s(s+2)(s+5)}\right)$. Design a feedback system so that

- The closed-loop system has a damping ratio (ζ) of 0.5.

Solution:

Step 1: Sketch the root locus of G(s):



The root locus plot is similar to a shopping list: for various values of 'k', you can get any response on the root locus plot.

Step 2: Pick a point on the root locus plot.

The design specifications require that the damping ratio be 0.5. The point on the root locus plot which meets the design specs is

s = -0.7143 + j1.2372

Step 3: Pick 'k' so that the closed-loop system has a pole at this spot

At any point on the root locus plot,

$$(G \cdot k)_s = -1$$

plugging in numbers

$$\left(\left(\frac{20}{s(s+2)(s+5)} \right) k \right)_{s=-0.7143+j1.2372} = -1$$

$$k = 0.5680$$

Step 4: Describe the response of the closed-loop system with this value of k:

The closed-loop dominant poles will be

$$s = -0.7143 \pm j1.2372$$

I know this because I picked k to place the poles here. This means

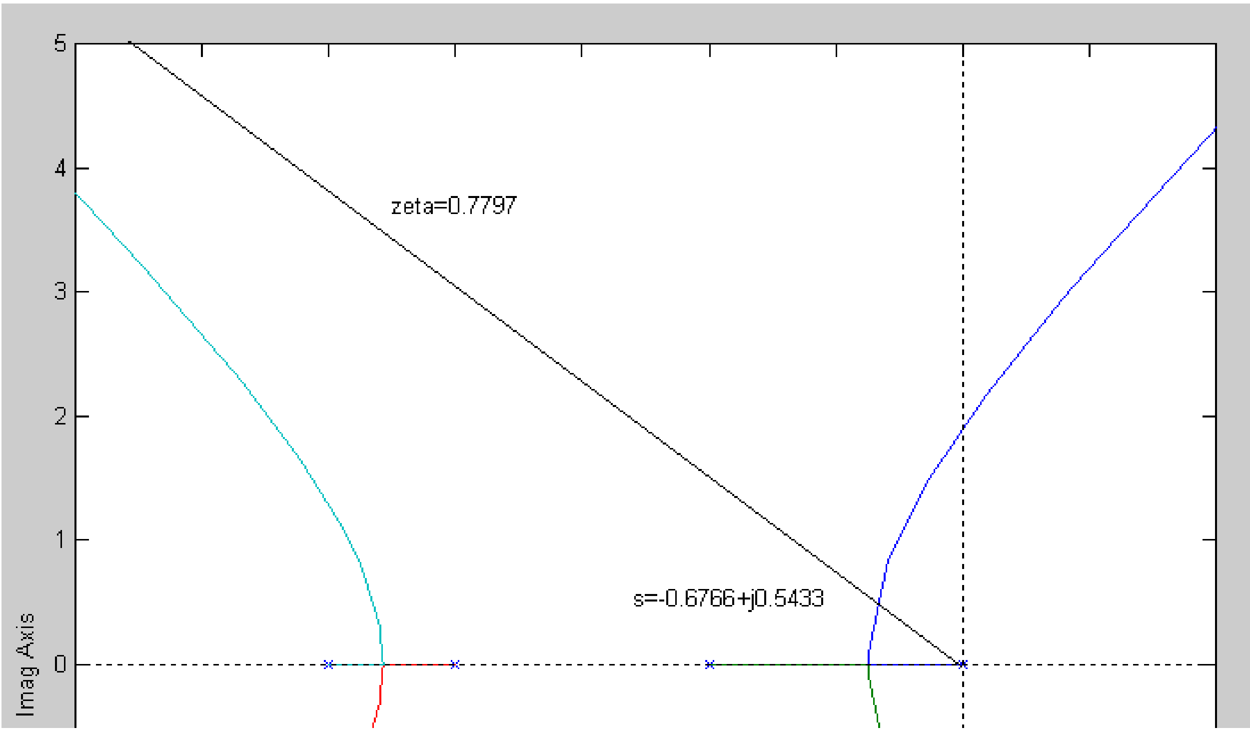
- The 2% settling time will be 5.60 seconds
- There will be 16.3% overshoot in the step response $(\zeta = 0.5)$
- There is no error for a unit step input (the system is type-1)
- $K_v = (sGK)_{s \to 0} = \left(s \cdot \frac{20 \cdot k}{s(s+2)(s+5)} \right)_{s \to 0} = 2 \cdot k = 1.1370$

Sidelight: 'k' should be a real positive number. If you get a complex number, it means the point you picked is not on the root locus.

Example #2: Design a gain compensator which results in the closed-loop system having 2% overshoot for a step input. Assume

$$G(s) = \left(\frac{20}{s(s+2)(s+4)(s+5)}\right)$$

Step 1: Sketch the root locus for G(s)



Step 2: Pick a point on the root locus which satisfies the design criteria.

You want 2% overshoot in the step response. This means you want the damping ratio to be

$$\zeta = 0.7797$$

The point on the root locus which intersects this damping ratio is

$$s = -0.6766 + j0.5433$$

Step 3: Pick k to place the closed-loop poles at this point

$$(Gk)_{s=--0.6766+j0.5433} = -1$$

$$k = 0.9180$$

Step 4: Describe the closed-loop system's response with this value of k:

The closed-loop system's dominant poles will be at

$$s = -0.6766 + j0.5433$$

I know this because I picked k to place the closed-loop dominant poles here.

- The 2% settling time will be 5.9119 seconds.
- There will be 2% overshoot in the step response $(\zeta = 0.7797)$
- There is no error for a unit step input (type 1)
- $K_v = (sGk)_{s \to 0} = \left(\frac{20 \cdot k}{s(s+2)(s+4)(s+5)} \right)_{s=0} = 0.5k = 0.4590$
- There will be 76.08% steady-state error in the step input.

Lead Compensator Design Using Root Locus Techniques

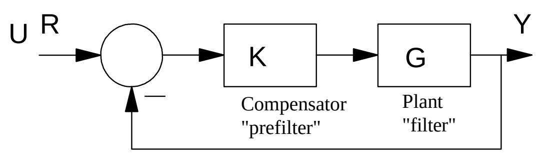
Objectives:

To be able to

- Design a lead compensator to speed up the closed-loop system
- Provide a circuit to implement compensators

Background:

Assume a feedback system of the following form:



A lead compensator is a prefilter of the form:

$$K(s) = k\left(\frac{s+a}{s+b}\right) \qquad b > a$$

The idea behind a lead compensator is as follows. Given a system, G(s), you have a corresponding root locus plot. The root locus plot tells you how fast you can make the system using gain compensation (see the previous lecture notes).

If you could shift the root locus left, you could get a faster closed-loop system. Typically, there is a (stable) pole which is causing you problems: it pushes the root locus plot to the right, limiting the response of the system. If you could cancel this stable pole and move it further left (faster), the root locus plot would shift left. This lets you speed up the system.

Example: Design a lead compensator that results in a damping ratio of 0.5 for the following system:

$$G(s) = \left(\frac{20}{s(s+2)(s+5)}\right)$$

Step 1: Pick the form of K(s):

The pole at -2 is causing problems and pushes the root locus to the right. If you could cancel this pole and move it out ot -20, you'd have a better system.

So, let

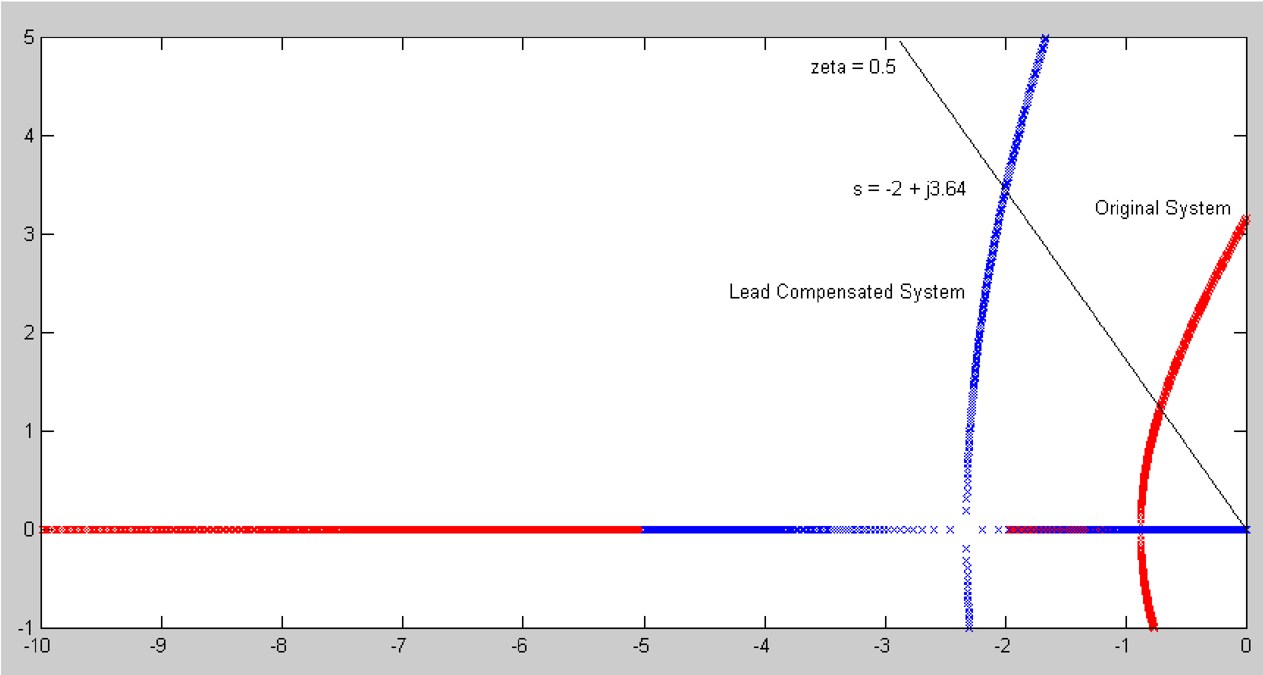
$$K(s) = k\left(\frac{s+2}{s+20}\right)$$

This results in the open-loop system being

$$GK = \left(\frac{20(s+2)k}{s(s+2)(s+5)(s+20)}\right) \approx \left(\frac{20k}{s(s+5)(s+20)}\right)$$

Once the form of K(s) is specified, the selection of 'k' is just like before:

Step 2: Sketch the root locus of the compensated system



Step 3: Pick a point on the root locus which satisfies the design criteria:

The point on the root locus plot which intersects the 0.5 damping ratio line is

$$s = -2.000 + j3.4641$$

Step 4: Find k to place the closed-loop poles here:

$$(GK)_{s=-2+j3.4641} = -1$$

$$k = 16.80$$

Step 5: Complete K(s)

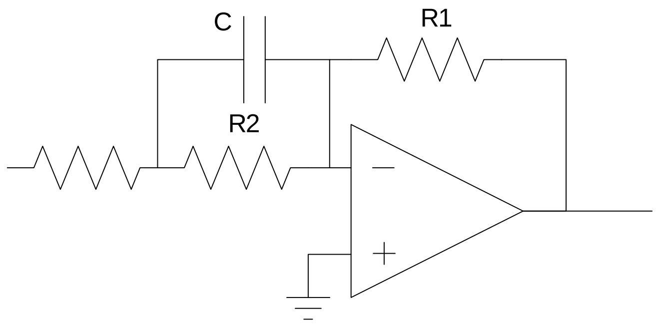
$$K(s) = 16.80\left(\frac{s+2}{s+20}\right)$$

Step 6: Describe how the closed-loop system will respond:

- Ts =2 seconds (if was 5.60 seconds using only gain compensation)
- No error for a step input (it's still a type-1 system)
- $K_v = (sGK)_{s=0} = 3.360$ (it was 1.1370 with gain compensation)

Note that the lead compensator sped up the system and reduced the stead-state error.

Step 7: Design a circuit to implement K(s):



- Let R1 = 1M
- At $s \to \infty$, $K(s) = 16.80 = \left(\frac{R_1}{R_3}\right)$. $R_3 = 59.5k\Omega$
- At $s \to 0$, $K(s) = 1.68 = \left(\frac{R_1}{R_2+R_3}\right)$. $R_2 = 535.7k\Omega$
- The zero tells you where C starts to short out R2: $\left(\frac{1}{R_2C}\right) = 2$. $C = 0.9333\mu F$

Problem: Design a lead compensator to speed up the system

$$G(s) = \left(\frac{336}{s(s+5)(s+20)}\right)$$

which results in a damping ratio of 0.5. (note: this is the same as adding a second lead compensator to speed up the system from before. You might do this if a settling time of 2 seconds was too slow).

Solution:

Step 1: Pick the form of K(s):

The pole at -5 is causing problems and pushes the root locus to the right. If you could cancel this pole and move it out to -50, you'd have a better system.

So, let

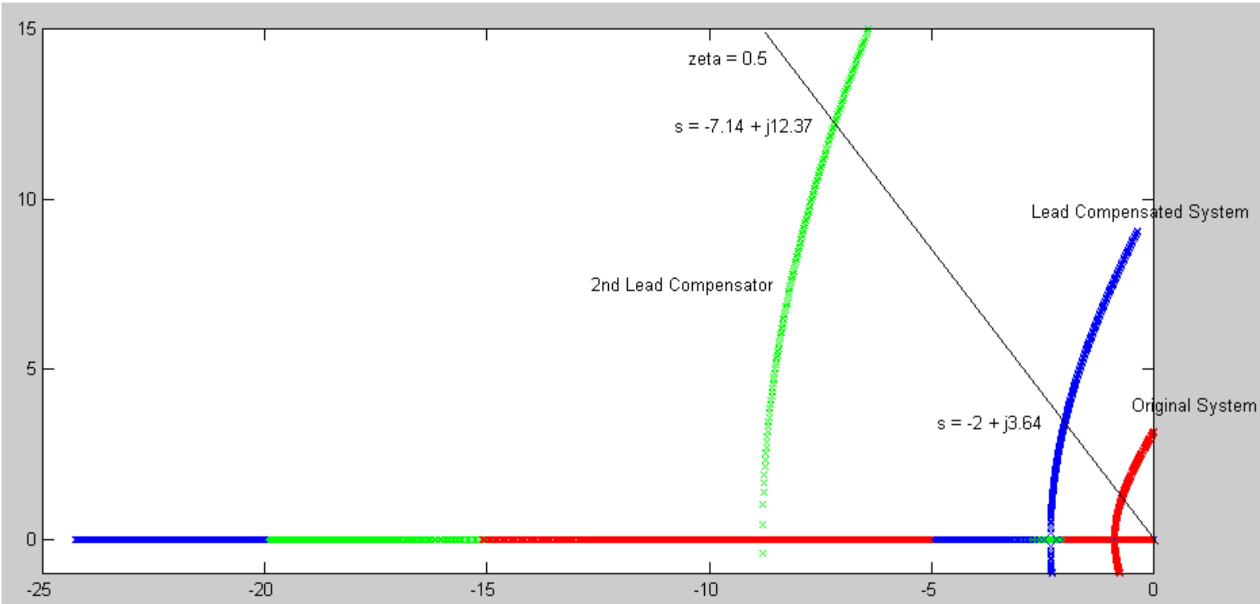
$$K(s) = k\left(\frac{s+5}{s+50}\right)$$

This results in the open-loop system being

$$GK = \left(\frac{336k}{s(s+20)(s+50)}\right)$$

Once the form of K(s) is specified, the selection of 'k' is just like before:

Step 2: Sketch the root locus of the compensated system



Step 3: Pick a point on the root locus which satisfies the design criteria:

The point on the root locus plot which intersects the 0.5 damping ratio line is

$$s = -7.1429 + j12.3718$$

Step 4: Find k to place the closed-loop poles here:

$$(GK)_{s=-7.1429+j12.3718} = -1$$

$$k = 33.84$$

Step 5: Complete K(s)

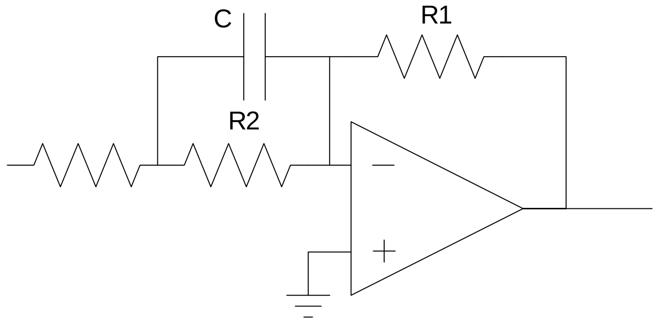
$$K(s) = 33.84\left(\frac{s+5}{s+50}\right)$$

Step 6: Describe how the closed-loop system will respond:

- Ts =0.56 seconds (if was 2 seconds previously)
- No error for a step input (it's still a type-1 system)
- $K_v = 11.37$ (it was 3.36)

Again, the lead compensator sped up the system and reduced the stead-state error.

Step 7: Design a circuit to implement K(s):



- Let R1 = 1M
- At $s \to \infty$, $K(s) = 33.84 = \left(\frac{R_1}{R_3}\right)$. $R_3 = 29.6k\Omega$
- At $s \to 0$, $K(s) = 3.384 = \left(\frac{R_1}{R_2+R_3}\right)$. $R_2 = 266k\Omega$
- The zero tells you where C starts to short out R2: $\left(\frac{1}{R_2 C}\right) = 5$. $C = 0.7519\mu F$

PID Compensation in the z-Domain

Objectives:

- Design a PID compensator in the z-domain to speed up a system

Discussion:

Type-1 systems are pretty useful: they track constant set points without any error. If a system starts out as type-0, you often add a pole at s=0 to make it type 1. You can do this in the z-domain as well.

- If the system is type-0, add a pole at z = +1. Place the rest of the poles out of harms way (at z=0).
- If there is already a pole at s=0, don't add any more poles at s=1 (or z = +1).

Example: Find a gain compensator for G(s) to give a damping ratio of 0.5 and results in no error for a step input.

$$G(s) = \left(\frac{1}{s^3+5s^2+6s+1}\right) = \left(\frac{1}{(s+0.1981)(s+1.555)(s+3.2469)}\right)$$

From before, the discrete-time equivalent with a sampling rate of 0.1 second is

$$G(z) = \left(\frac{0.0001147(z+0.2358)(z+3.3011)}{(z-0.9803)(z-0.8560)(z-0.7227)}\right)$$

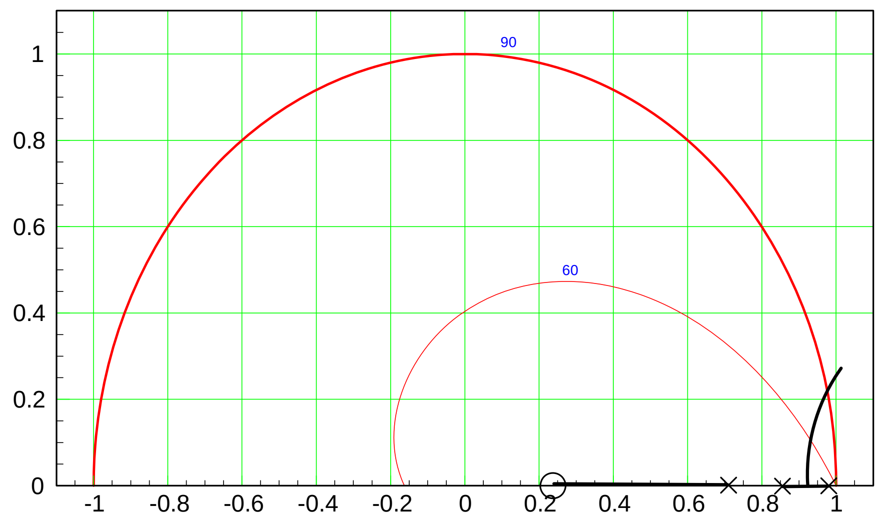
Integral Compensation

Proportional-Integral (PI) Compensation

This system is type-0. Add a pole at z = +1. If you add a pole, you can add a zero at no charge. Pick this zero to canel the slowest stable pole (z = 0.9803):

$$K(z) = k\left(\frac{z-9803}{z-1}\right)$$

The root locus for this system is show below:



For a damping ratio of 0.6, you want to place the dominant pole at -0.9471 + j0.0831. At this point

GK = -0.1967

so

k = 5.0841

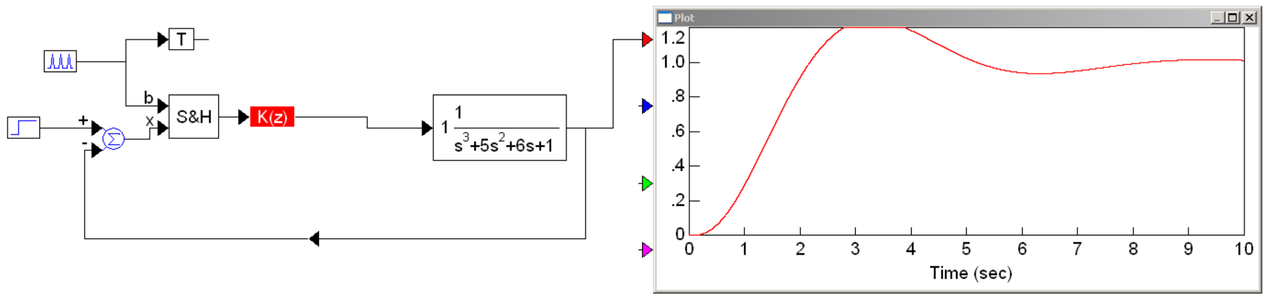
The 2% settling time should be

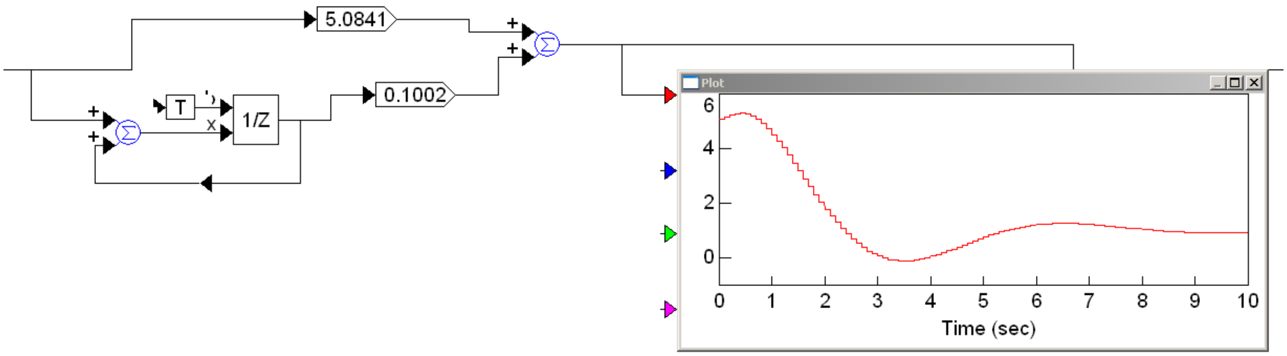
$$|pole| = |-0.9471 + j0.0831| = 0.9507$$

$$(0.9507)^k = 0.02$$

$$k = 77.3 \text{ samples}$$

$$t = kT = 7.7 \text{ seconds}$$




5.0841
0.1002
T
1/Z
x
Plot
6
4
2
0
0 1 2 3 4 5 6 7 8 9 10
Time (sec)

Meeting Design Specs using Root Locus

Objectives:

To be able to

- Design a compensator to provide a specified response

Really, when you design a compensator, K(s), you are free to add poles and zeros as you like. Some forms have special names (like lead, PI, PID, etc.) All you're doing is adding poles and zeros, however.

In general,

- If the system needs to be type one and there isn't a pole at s=0, add a pole at s=0.
- If the system is too slow, cancel a slow stable pole and replace it with a faster pole
- If the system is too fast, slow down one of the fast poles you added (or add a pole which you can slide around).

The last one only comes in if there is a very specific response you want.

For example, design a compensator for

$$G(s) = \left(\frac{10}{(s+1)(s+2)(s+4)(s+5)}\right),$$

so that the closed-loop system has

- No error for a step input,
- 10% overshoot for a unit step input, and
- A 2% settling time less than 4 seconds.

(The case of a 2% settling time equal to 4 seconds will be taken shortly).

Solution: This is similar to the last problem, except that you are placing the closed-loop poles at a specific spot.

Step 1. Translate the requirements into root-locus terms

- No error means make the system type 1. Since the system is initially type-0, K(s) will add a pole at s=0.
- 10% overshoot means that the dominant pole will have a damping ratio of 0.59.
- A 4 second settling time means the real part of the dominant pole should be **less than** -1.

Step 2: Specify the form of K(s)

You need a pole at s=0, so add a pole at s=0. If you add a pole, you can also add a zero at no cost, so also add a zero at -1.

$$K(s) = k\left(\frac{s+1}{s}\right)$$

This results in a root locus which crosses the $\zeta = 0.59$ damping line at

s = -0.5934 + j0.8119

This is too slow.

So, cancel the next slowest pole and replace it with a pole 10x further out:

$$K(s) = k\left(\frac{(s+1)(s+2)}{s(s+20)}\right)$$

This results in a root locus which crosses the damping line at

s = -1.1134+j1.5235

This meets the design specs, so the form of K(s) is correct. Now you only have to specify k to place the closed-loop poles here:

$$GK = \left(\frac{10k}{s(s+4)(s+5)(s+20)}\right)$$

$$(GK)_{s=-1.1134+j1.5235} = -1$$

$$k = 48.7170$$

and

$$\boxed{K(s) = 48.7170\left(\frac{(s+1)(s+2)}{s(s+20)}\right)}$$

Note that this compensator doesn't have a name (it's not a true PID, PI, lead, etc.) It's just a compensator that meets the design specs. You could build it as a cascaded PI and lead if you like.

Problem: Change the previous problem so that the 2% settling time is **equal** to 4 seconds.

In this case you want to place the closed-loop poles at a specific spot: **s = -1.000 + 1.3685**

Step 1: Design a compensator which makes the closed-loop system too fast.

See the previous solution

Step 2: Adjust the fast pole so that the root locus passes through -1 + j1.3685.

Let

$$K(s) = k\left(\frac{(s+1)(s+2)}{s(s+a)}\right)$$

so

$$GK = \left(\frac{10k}{s(s+4)(s+5)(s+a)}\right)$$

We know the point **s = -1.000 + 1.3685** should be on the root locus. This means the phase of GK must be 180 degrees at this point. Plugging it in:

$$\angle\left(\frac{10k}{s(s+4)(s+5)}\right)_{s=-1+j1.3685} = -169.56^0$$

so, to make the angles add up to 180 degrees,

$$\angle(s+a) = 10.4354^0$$

$$a = 1 + \frac{1.3685}{\tan\left(10.4354^0\right)} = 8.4305$$

and

$$K(s) = k\left(\frac{(s+1)(s+2)}{s(s+8.4305)}\right)$$

$$GK = \left(\frac{10k}{s(s+4)(s+5)(s+8.4305)}\right)$$

Finding k:

$$\left(\frac{10k}{s(s+4)(s+5)(s+8.4305)}\right)_{s=-1+j1.3685} = -1$$

$$k = 17.8517$$

and

$$\boxed{K(s) = 17.8517\left(\frac{(s+1)(s+2)}{s(s+8.4305)}\right)}$$

Root Locus for Systems with Delays

Objectives:

To be able to design a feedback compensator for a system with a delay.

Delays:

A delay can model fast dynamics which you ignore to simplify the model. It can also be part of the actual system.

A delay has a LaPlace transform of

$$\text{delay(T seconds)} \Leftrightarrow e^{-sT}$$

This is a problem for root locus analysis since you need to know the poles and zeros to sketch the root loucs. e^{-sT} doesn't have poles or zeros.

One way around this is to the a Pade approximation for a delay:

Pade Approximation:

$$e^{-sT} = \left(\frac{e^{-\frac{sT}{2}}}{e^{\frac{sT}{2}}} \right)$$

Using the Taylors series

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + ...$$

a delay of T seconds has a LaPlace transform of

$$e^{-sT} = \left(\frac{1+\left(\frac{-sT}{2}\right)+\left(\frac{-sT}{2}\right)^2\left(\frac{1}{2!}\right)+\left(\frac{-sT}{2}\right)^3\left(\frac{1}{3!}\right)+\left(\frac{-sT}{2}\right)^4\left(\frac{1}{4!}\right)+...}{1+\left(\frac{sT}{2}\right)+\left(\frac{sT}{2}\right)^2\left(\frac{1}{2!}\right)+\left(\frac{sT}{2}\right)^3\left(\frac{1}{3!}\right)+\left(\frac{sT}{2}\right)^4\left(\frac{1}{4!}\right)+...} \right)$$

or

$$e^{-sT} = \left(\frac{1-\left(\frac{T}{2}\right)s+\left(\frac{T^2}{8}\right)s^2-\left(\frac{T^3}{48}\right)s^3+\left(\frac{T^4}{384}\right)s^4+...}{1+\left(\frac{T}{2}\right)s+\left(\frac{T^2}{8}\right)s^2+\left(\frac{T^3}{48}\right)s^3+\left(\frac{T^4}{384}\right)s^4+...} \right)$$

The more terms you add, the better the approximation.

Example: Find 'k' so that the following system has 10% overshoot for its step response: (A DC servo motor with a 1/2 second delay)

$$Y = \left(\frac{1000 \cdot e^{-0.5s}}{s(s+5)(s+20)} \right) U$$

Solution #1: Let's use a Pade approximation:

$$e^{-sT} = \left(\frac{1 - \left(\frac{T}{2}\right)s + \left(\frac{T^2}{8}\right)s^2 - \left(\frac{T^3}{48}\right)s^3 + \left(\frac{T^4}{384}\right)s^4 + ...}{1 + \left(\frac{T}{2}\right)s + \left(\frac{T^2}{8}\right)s^2 + \left(\frac{T^3}{48}\right)s^3 + \left(\frac{T^4}{384}\right)s^4 + ...} \right)$$

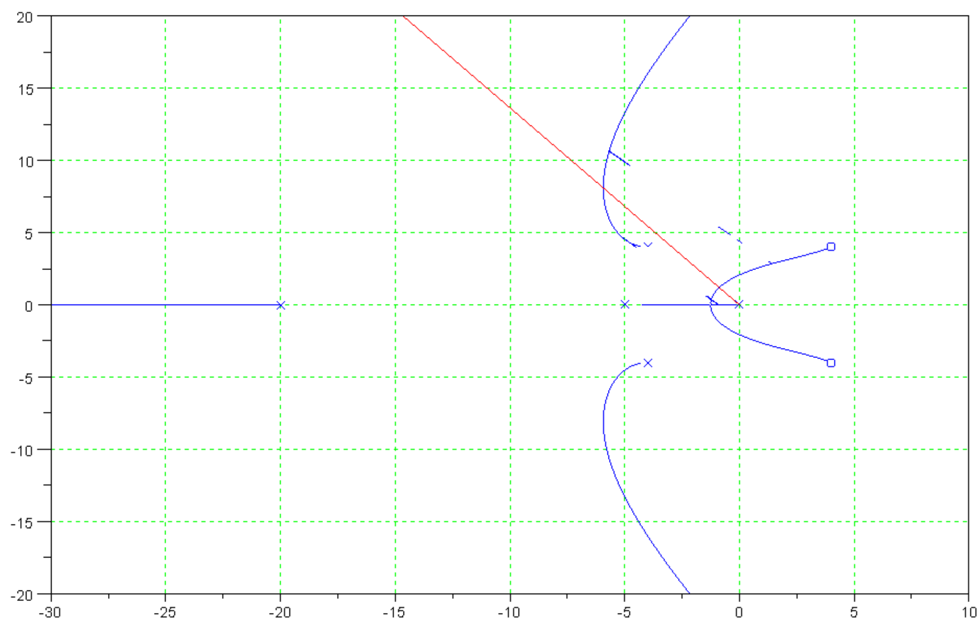
Pluggint in T = 0.5 and using the first two terms results in

$$e^{-0.5s} \approx \left(\frac{1 - 0.25s + 0.0313s^2}{1 + 0.25s + 0.0313s^2} \right)$$

$$e^{-0.5T} \approx \left(\frac{(s-4+j4)(s-4-j4)}{(s+4+j4)(s+4-j4)} \right)$$

In SciLab:

```
-->G = zp2ss([4+j*4,4-j*4],[0,-5,-20,-4+j*4,-4-j*4],1000);
-->k = logspace(-2,2,1000)';
-->R = rlocus(G,k,0.5910);
```



Root locus for G(s) with a 0.5 second delay

Note that there are poles at -4 +/- j4 and zeros at +4 +/- j4. These model the delay.

There are two solutions for a damping ratio of 0.5910 (for 10% overshoot):

s = -0.8767 + j1.1966 k = 0.0775

s = -5.9293 + j8.0930 k = 0.4370

The former is the dominant pole (the one we care about). Hence,

$$\boxed{K(s) = 0.0775}$$

Problem: Repeat for a 4th-order Pade Approximation

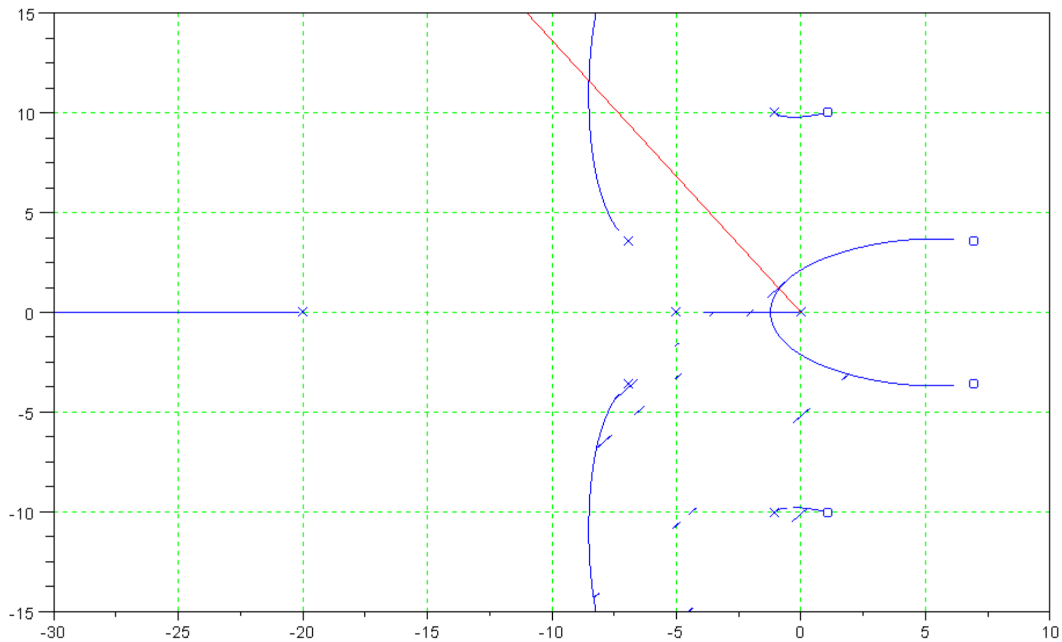
$$e^{-sT} = \left(\frac{1 - \left(\frac{T}{2}\right)s + \left(\frac{T^2}{8}\right)s^2 - \left(\frac{T^3}{48}\right)s^3 + \left(\frac{T^4}{384}\right)s^4 + \ldots}{1 + \left(\frac{T}{2}\right)s + \left(\frac{T^2}{8}\right)s^2 + \left(\frac{T^3}{48}\right)s^3 + \left(\frac{T^4}{384}\right)s^4 + \ldots} \right)$$

$$e^{-0.5s} \approx \left(\frac{1 - 0.25s + 0.0313s^2 - 0.0026s^3 + 0.002s^4}{1 + 0.25s + 0.0313s^2 + 0.0026s^3 + 0.002s^4} \right)$$

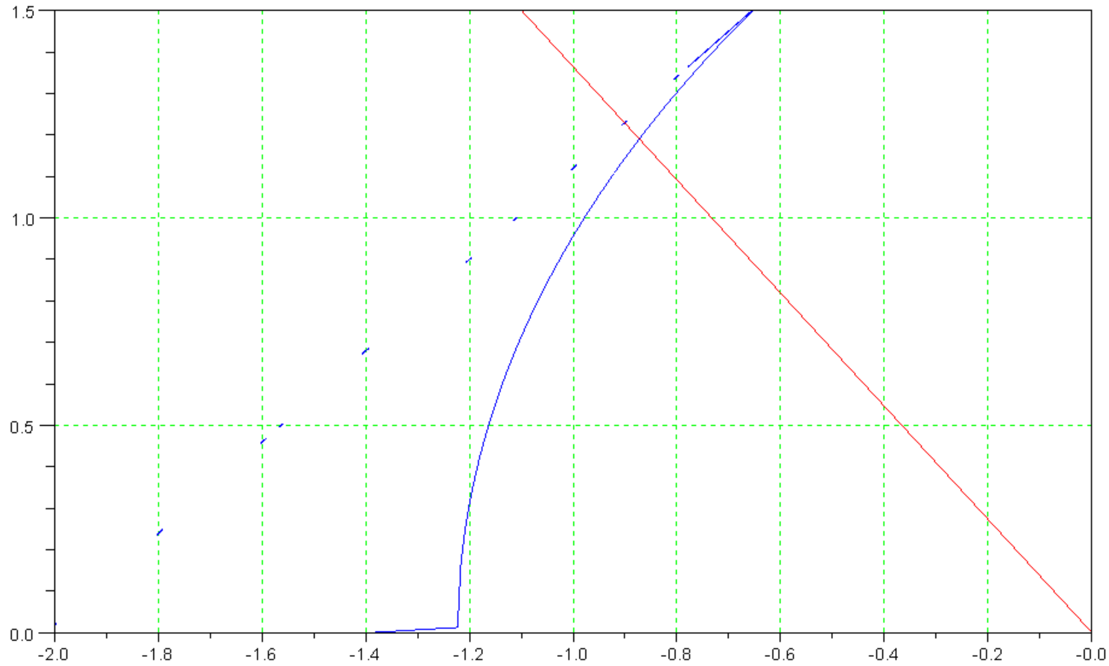
$$e^{-0.5s} \approx \left(\frac{(s - 1.08222 \pm j10.019)(s - 6.9177 \pm j3.5559)}{(s + 1.08222 \pm j10.019)(s + 6.9177 \pm j3.5559)} \right)$$

The root locus of G + delay is a bit more complicated:

```
-->P = roots({0.5^4/384,0.5^3/48,0.5^2/8,0.25,1]);
-->Z = roots({0.5^4/384,-0.5^3/48,0.5^2/8,-0.25,1]);
-->G = zp2ss(Z,[P;0;-5;-20],1000);
-->R = rlocus(G,k,0.5910);
```



Zooming in on the dominant pole:



4th order Pade Model:

s = -0.8724 + j1.1907 k = 0.0786

Option #2:

This option doesn't really have a name. It's based upon root locus techniques. The idea is, for a damping ratio of 0.5910, you're searching along the line

$$s = \alpha\angle 126.228^0$$

until the angle of G(s) is 180 degrees. You don't really need to use a Pade approximation to do this. Simply find the point where

$$angle\left(\frac{1000\cdot e^{-0.5s}}{s(s+5)(s+20)}\right)_{s=\alpha\angle 126.228^0} = 180^0$$

This method doesn't have a name and might not be taught anywhere else other than NDSU.

- It isn't really root locus design, since you're not drawing the root locus
- It sort of is root locus design, since you're finding the point on the root locus which intersects the desired damping line. That's the only point you care about anyway, so you don't need (and won't use) the rest of the root loucs.
- It's a lot easier and more accurate since you using e^{-sT} to model a delay, not an approximation. (Typing in four poles and four zeros for the 4th-order model was a bit of a pain. Typing in e^{-sT} was easy.)

Anyway, if you use this method, the result is

exact solution:

s = -0.8725 + j1.1909 k = 0.0786

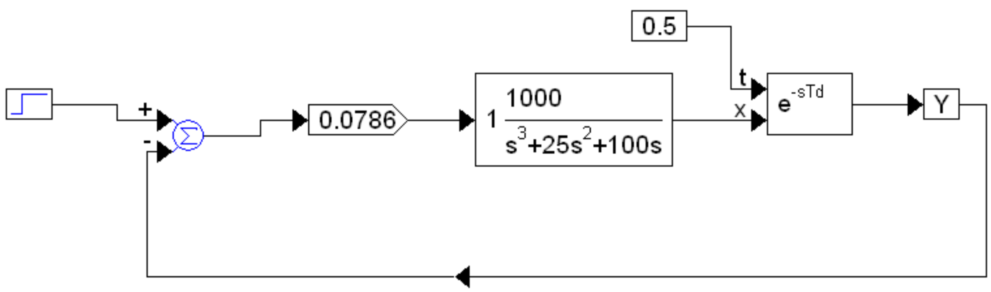
4th order Pade:

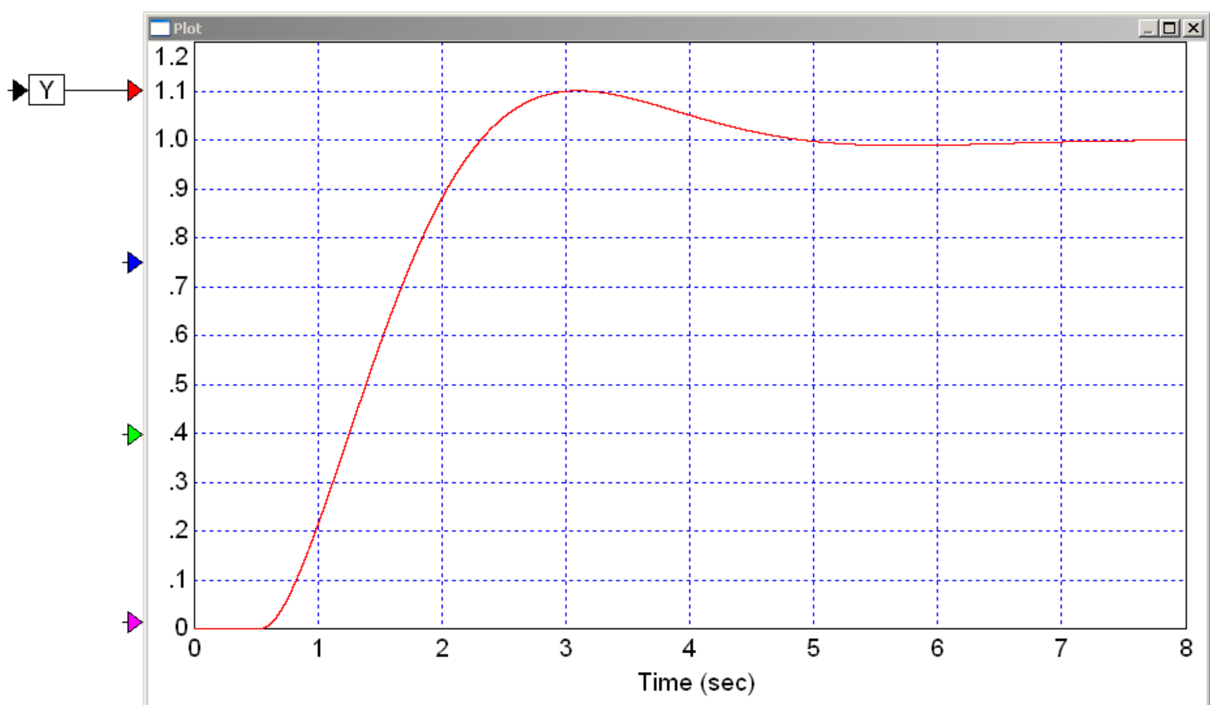
s = -0.8724 + j1.1907 k = 0.0786

2nd Order Pade Model:

s = -0.8767 + j1.1966 k = 0.0775

The real test of a design is if it works. Using VisSim with an actual 1/2 second delay, we do get 10% overshoot:





Multi-Loop Feedback

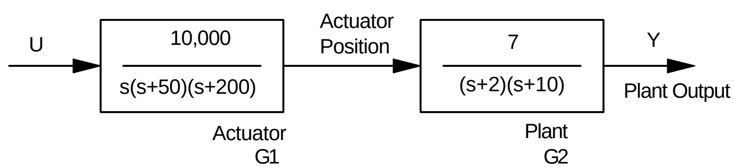
Objectives:

To be able to

- Design a compensator for a system which cannot (or is difficult) to control using a single feedback loop.

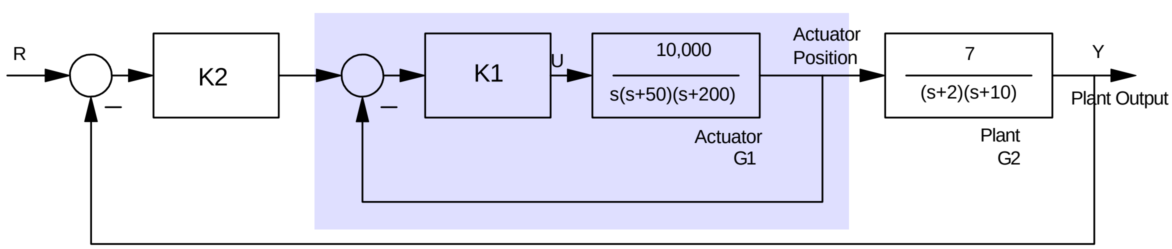
Multi-Loop Feedback

Problem: Design a compensator for the following system: This could model the dynamics of a jet engine where a DC motor opens or closes a valve (the actuator position). The valve then feeds fuel to the engine, which provides thrust (Y). The goal is to force the output (thrust) to track a set point quickly and accurately.



Solution: Solve this in two steps:

- i) Close the feedback loop around the actuator. This results in the output of the actuator following its set point. (This is also easy to test: drive the actuator to 30 degrees and see if it goes there.)
- ii) Close the feedback loop around the plant.



First System to Design

i) Closing the feedback loop around the actuator.

If you only look at the actuator, the only dynamics you care about are G1:

$$G_1(s) = \left(\frac{10{,}000}{s(s+50)(s+200)}\right)$$

This is not a very convenient transfer function since, if you apply a constant input, the integrator will cause the output to drift. Unity feedback can speed up this part of the system. Placing the dominant pole at -20 (chosen so that the actuator is fast) results in

$$G_1K_1 = \left(\frac{10{,}000k}{s(s+50)(s+200)}\right)_{s=-20} = -0.0926k = -1$$

or

$$k = 10.8$$

The actuator's dynamics then become

$$\left(\frac{G_1 K_1}{1+G_1 K_1}\right) = \left(\frac{10{,}000k}{s(s+50)(s+200)+10{,}000k}\right) = \left(\frac{108{,}000}{(s+20)(s+26.54)(s+203.46)}\right)$$

ii) Combine the plant and the actuator:

$$G_{21} = \left(\frac{7}{(s+2)(s+10)}\right)\left(\frac{108{,}000}{(s+20)(s+26.54)(s+203.46)}\right) = \left(\frac{756{,}000}{(s+2)(s+10)(s+20)(s+26.54)(s+203.46)}\right)$$

iii) Add a compensator.

- Add a pole at s=0 to make the system type-1
- Add a zero at s = -2 to cancel the slowest stable pole

$$K_2(s) = k\left(\frac{s+2}{s}\right)$$

$$G_{21}K_2 = \left(\frac{756{,}000k}{s(s+10)(s+20)(s+26.54)(s+203.46)}\right)$$

iv) Place the poles for this system. For a damping ratio of 0.8, the poles should be placed at

$$s = -3.3995 + j2.5496$$

(this is found by sketching the root locus for $G_{21}K_2$ and finding the point which crosses the 0.8 damping line).

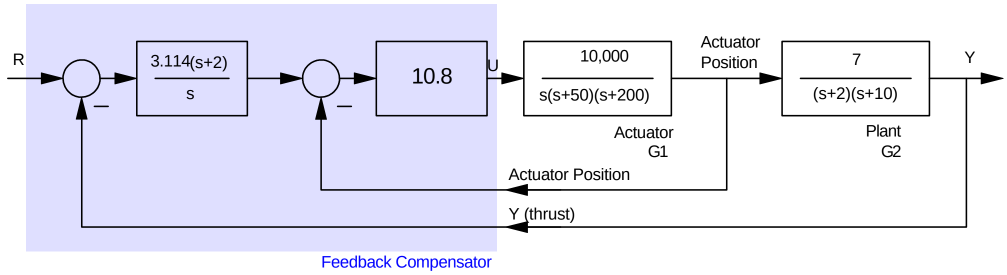
$$(G_{21}K_2)_{s=-3.3995+j2.5496} = -0.3214k = -1$$

$$k = 3.1114$$

and

$$K_2(s) = 3.1114\left(\frac{s+2}{s}\right) .$$

The net feedback control system is then as follows. Note that you are using two measurements in feedback compensator: actuator position and system output (Y). Even though root-locus techniques are designed for systems with only a single feedback loop, you can design a system with several measurements if you close the feedback loops in stages.



MATLAB Code

input the systems into MATLAB

```
G1 = zpk([],[0,-50,-200],10000);
G2 = zpk([],[-2,-10],7);
s = -20;
k1 = -1 / evalfr(G1,s)
   10.8000
Gcl1 = (G1*k) / (1+G1*k);
```

compute the gain to place a pole at -20

so $K_1 = 10.8$

close the feedback loop around the actuator

$$G_{1:CL} = \left(\frac{(10.8)(10{,}000)}{(s+20)(s+26.541)(s+203.459)} \right)$$

```
G21 = G2*Gcl1;
K2 = zpk(-2,0,1);
G21K2 = G21 * K2;
```

combine the plant and the actuator and the compensator to get the feedforward system

```
s = -3.3995 + j*2.5496;
```

place the poles of the net system here

```
k2 = -1 / evalfr(G21K2, s)
   3.1115 - 0.0000i
K2 = zpk(-2,0,3.1115);
```

so $K_2(s) = \left(\frac{3.1115(s+2)}{s} \right)$

```
Gcl2 = (G21*K2) / (1+G21*K2)
```

and the net system with feedback is

$$G_{net} = \left(\frac{2{,}352{,}842(s+2)}{(s+203.46)(s+24.87\pm j4.66)(s+3.4\pm j2.55)(s+2)} \right)$$

Root Locus for Systems with Complex Poles
Gantry System

Find the dynamics of a nonlinear system:

Circuit analysis tools work for simple lumped systems. For more complex systems, especially nonlinear ones, this approach fails. The Lagrangian formulation for system dynamics is a way to deal with any system.

Definitions:

KE Kinetic Energy in the system

PE Potential Energy

$\frac{\partial}{\partial t}$ The partial derivative with respect to 't'. All other variables are treated as constants.

$\frac{d}{dt}$ The full derivative with respect to t.

$$\frac{d}{dt} = \frac{\partial}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial}{\partial y}\frac{\partial y}{\partial t} + \frac{\partial}{\partial z}\frac{\partial z}{\partial t} + \ldots$$

L Lagrangian = KE - PE

Procedure:

1) Define the kinetic and potential energy in the system.
2) Form the Lagrangian:

$$L = KE - PE$$

3) The input is then

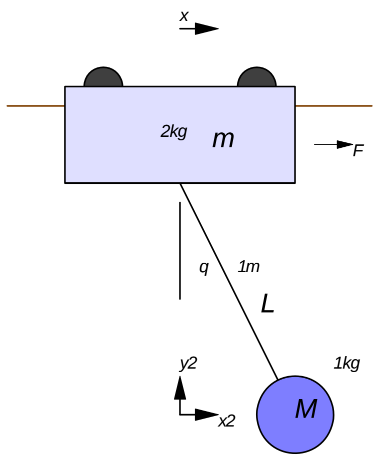
$$F_i = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}_i}\right) - \frac{\partial L}{\partial x_i}$$

where F_i is the input to state x_i. Note that

- If x_i is a position, F_i is a force.
- If x_i is an angle, F_i is a torque

Example:

Find the dynamics for the following system: A 1kg mass swings from a gantry which weighs 2kg. The length of the rope is 1m.



1) Write the energy in the system

Mass #1:

- $x_1 = x$
- $y_1 = 0$

$$KE_1 = \frac{1}{2}mv^2 = \frac{1}{2}m\dot{x}^2$$

$$PE_1 = mgh = 0$$

Mass #2

- $x_2 = x_1 + L\sin\theta$ $\qquad \dot{x}_2 = \dot{x}_1 + L(\cos\theta)\dot{\theta}$
- $y_2 = L(1-\cos\theta)$ $\qquad \dot{y}_2 = L(\sin\theta)\dot{\theta}$

$$KE_2 = \frac{1}{2}M\left(\dot{x}_2^2 + \dot{y}_2^2\right)$$

$$KE_2 = \frac{1}{2}M\left(\dot{x}_1^2 + 2L\dot{x}_1\dot{\theta}_1\cos\theta + L^2\cos^2\theta\cdot\dot{\theta}^2 + L^2\sin^2\theta\cdot\dot{\theta}^2\right)$$

$$KE_2 = \frac{1}{2}M\left(\dot{x}_1^2 + 2L\dot{x}_1\dot{\theta}_1\cos\theta + L^2\dot{\theta}^2\right)$$

$$PE_2 = mgh = gy_2$$

$$PE_2 = mgL(1-\cos\theta)$$

So, the Lagrangian is

$$L = \tfrac{1}{2}m\dot{x}^2 + \tfrac{1}{2}M\left(\dot{x}_1^2 + 2L\dot{x}_1\dot{\theta}_1\cos\theta + L^2\dot{\theta}^2\right) - MgL(1-\cos\theta)$$

The force on the base is

$$F = \tfrac{d}{dt}\left(\tfrac{\partial L}{\partial \dot{x}'}\right) - \tfrac{\partial L}{\partial x}$$

$$F = \tfrac{d}{dt}\left((M+m)\dot{x} + LM\dot{\theta}\cos\theta\right) - 0$$

$$F = (M+m)\ddot{x} + LM\ddot{\theta}\cos\theta - LM\dot{\theta}^2\sin\theta$$

The torque on the rod is

$$T = \tfrac{d}{dt}\left(\tfrac{\partial L}{\partial \dot{\theta}}\right) - \tfrac{\partial L}{\partial \theta}$$

$$T = \tfrac{d}{dt}\left(LM\dot{x}\cos\theta + LM\dot{\theta}\right) - LM\dot{x}\dot{\theta}\sin\theta - MgL\sin\theta$$

$$T = LM\ddot{x}\cos\theta + LM\dot{x}\dot{\theta}\sin\theta + LM\ddot{\theta} - LM\dot{x}\dot{\theta}\sin\theta - MgL\sin\theta$$

Substituting M=1, m=2, L=1, the dynamics are

$$\begin{bmatrix} 3 & \cos\theta \\ \cos\theta & 1 \end{bmatrix}\begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} \dot{\theta}^2\sin\theta \\ g\sin\theta \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix}F + \begin{bmatrix} 0 \\ 1 \end{bmatrix}T$$

Linearizing near x=0, θ=0.

$$\begin{bmatrix} 3 & 1 \\ 1 & 1 \end{bmatrix}\begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 0 \\ g\theta \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix}F + \begin{bmatrix} 0 \\ 1 \end{bmatrix}T$$

$$\begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} -0.5g\theta \\ 1.5g\theta \end{bmatrix} + \begin{bmatrix} 0.5 \\ -0.5 \end{bmatrix}F + \begin{bmatrix} -0.5 \\ 1.5 \end{bmatrix}T$$

Let g = -9.8 m/s2

$$s\begin{bmatrix} x \\ \theta \\ x' \\ \theta' \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 4.9 & 0 & 0 \\ 0 & -14.7 & 0 & 0 \end{bmatrix}\begin{bmatrix} x \\ \theta \\ x' \\ \theta' \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0.5 \\ -0.5 \end{bmatrix}F$$

This gives the transfer funciton of

```
-->A = [0,0,1,0;0,0,0,1;0,4.9,0,0;0,-14.7,0,0]
 A  =

    0.     0.        1.     0.
    0.     0.        0.     1.
```

```
    0.    4.9     0.    0.
    0.  - 14.7    0.    0.

-->B = [0;0;0.5;-0.5]
 B  =

    0.
    0.
    0.5
  - 0.5

-->C = [1,0,0,0]
 C  =

    1.    0.    0.    0.

-->D = 0
 D  =

    0.

-->G = sspack(A,B,C,D)
 G  =

    0.    0.      1.    0.    0.
    0.    0.      0.    1.    0.
    0.    4.9     0.    0.    0.5
    0.  - 14.7    0.    0.  - 0.5
    1.    0.      0.    0.    0.

-->[z,p,k] = ss2zp(G)
 k  =

    0.5
 p  =

    0
    0
    3.8340579i
  - 3.8340579i
 z  =

    1.090D-14 + 3.1304952i
    1.090D-14 - 3.1304952i
```

$$X \approx \left(\frac{0.5(s+j3.13)(s-j3.13)}{s^2(s+j3.83)(s-j3.83)} \right) F$$

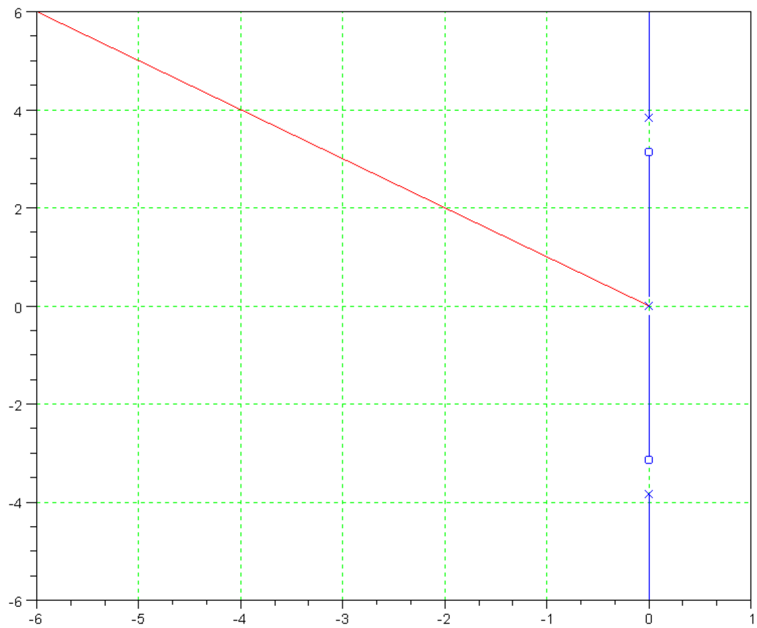
or to velocity:

$$sX \approx \left(\frac{0.5(s+j3.13)(s-j3.13)}{s(s+j3.83)(s-j3.83)} \right) F$$

Compensator Design:

$$X \approx \left(\frac{0.5(s+j3.13)(s-j3.13)}{s^2(s+j3.83)(s-j3.83)} \right) F$$

Let's start simple. Let K(s) = k. The root locus looks like the following:



There's no way to dissipate energy in this system. This shows up as the root locus remaining on the jw axis.

Lets add derivative feeedback to dissipate the energy:

$$K(s) = ks$$

Normally you don't want to do this since differentiation is a noise amplifier. Assuming you can measure cart velocity (use a tachometer), you can feed back the speed without differentiation.

You also don't want to add the 's' term in the numeator

- The open-loop zeros are also the closed-loop zeros
- Placing a zero at s=0 in the feedforward path creates a zero at s=0 in the closed-loop system

You can get around this by placing K(s) in the feedback path:

$$Y = \left(\frac{G}{1+GK} \right) R$$

If $G(s) = \left(\frac{z(s)}{p(s)} \right)$ (poles and zeros)

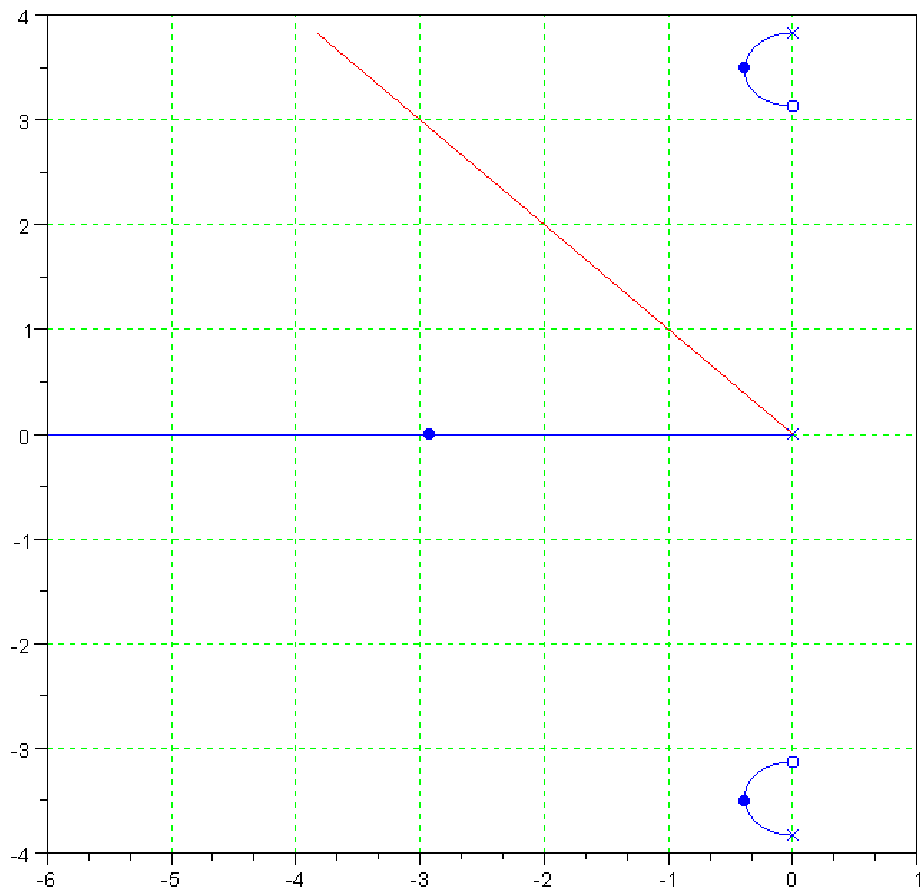
$$\left(\frac{G}{1+ksG} \right) = \left(\frac{z(s)}{p(s)+ks \cdot z(s)} \right)$$

Note that the denominator's roots are what we normally see: the poles are the poles of G and K. The zeros are the zeros of K and G. The numerator avoids a zero at s=0 this way.

Now, find how much friction to add (find k).

First, sketch the root locus of

$$sX \approx \left(\frac{0.5(s+j3.13)(s-j3.13)}{s(s+j3.83)(s-j3.83)} \right) F$$



Pick 'k' to make the system as stable as possible. Note that

- For no friction (k=0), there is a root on the jw axis at j3.83.
- For infinite friction (k = infinity), the root moves back to the jw axis at j3.13.
- The 'optimal' k places this pole at -0.5 + j3.5 (about).

To find k,

```
-->k = 1/abs(evalfr(G,-0.4+j*3.5))
    7.4105997
```

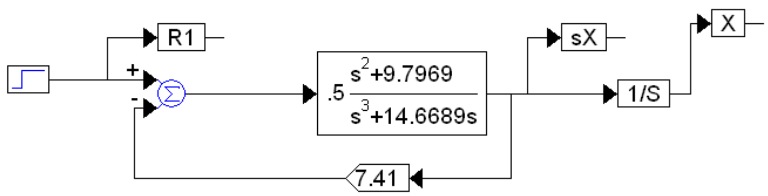
The closed-loop system is then:

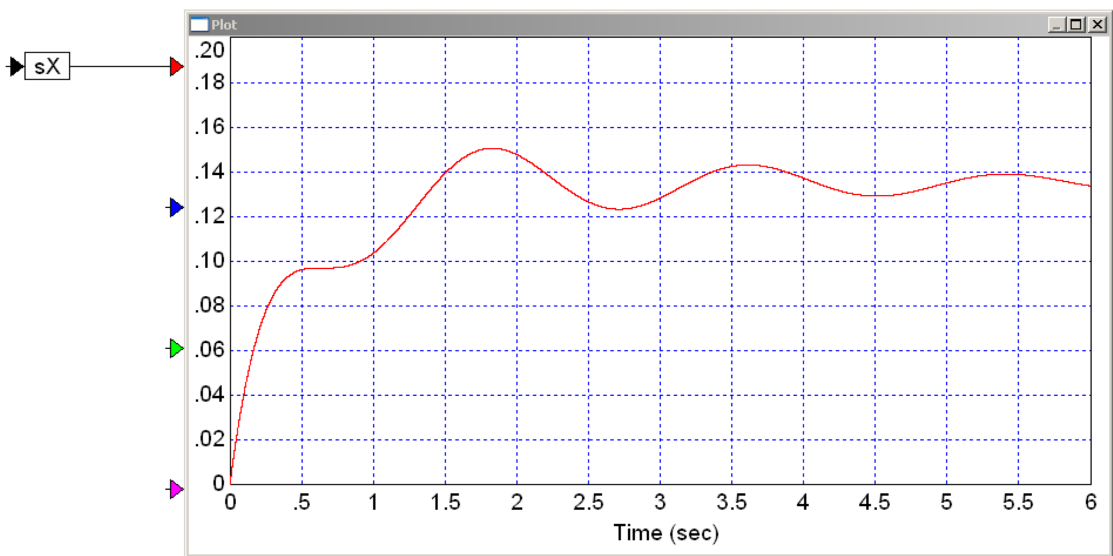
```
-->G2 = intcon(G,k);
-->[zero,pole,gain] = ss2zp(G2)
 gain  =

    36300456. - 3.814D-08i
 pole  =

  - 0.3882796 + 3.499114i
  - 2.9287407 + 2.826D-15i
  - 0.3882796 - 3.499114i
 zero  =

   1.0D-11 *

    0.2644023
  - 0.4318139
```

$$sX = \left(\frac{z(s)}{p(s)+ks\cdot z(s)}\right)R_1$$

$$sX = \left(\frac{0.5(s+j3.13)(s-j3.13)}{(s+0.388+j3.5)(s+0.388-j3.5)(s+2.93)}\right)R_1$$

The step response to velocit is now stable. Not good, bus stable:





Step 2: Now that you have a stable system, close the feedback loop arount X.

$$X = \left(\frac{0.5(s+j3.13)(s-j3.13)}{s(s+0.388+j3.5)(s+0.388-j3.5)(s+2.93)} \right) R_1$$

First add a zero to the prevous solution

```
-->G1 = tf2ss2(1,[1,0])
 G1  =

    0.    1.
    1.    0.

-->G3 = ssmult(G1,G2)
 G3  =

    0     4.89845      0          0.5          0
    0     0            1.         0            0
    0     0            0          1.           0
    0   - 36.300452  - 14.6689  - 3.7052998    7.4105997
    1.    0            0          0            0

-->eig(G3)
 ans  =

    0
  - 0.3882796 + 3.499114i
  - 2.9287407 + 2.826D-15i
  - 0.3882796 - 3.499114i
```

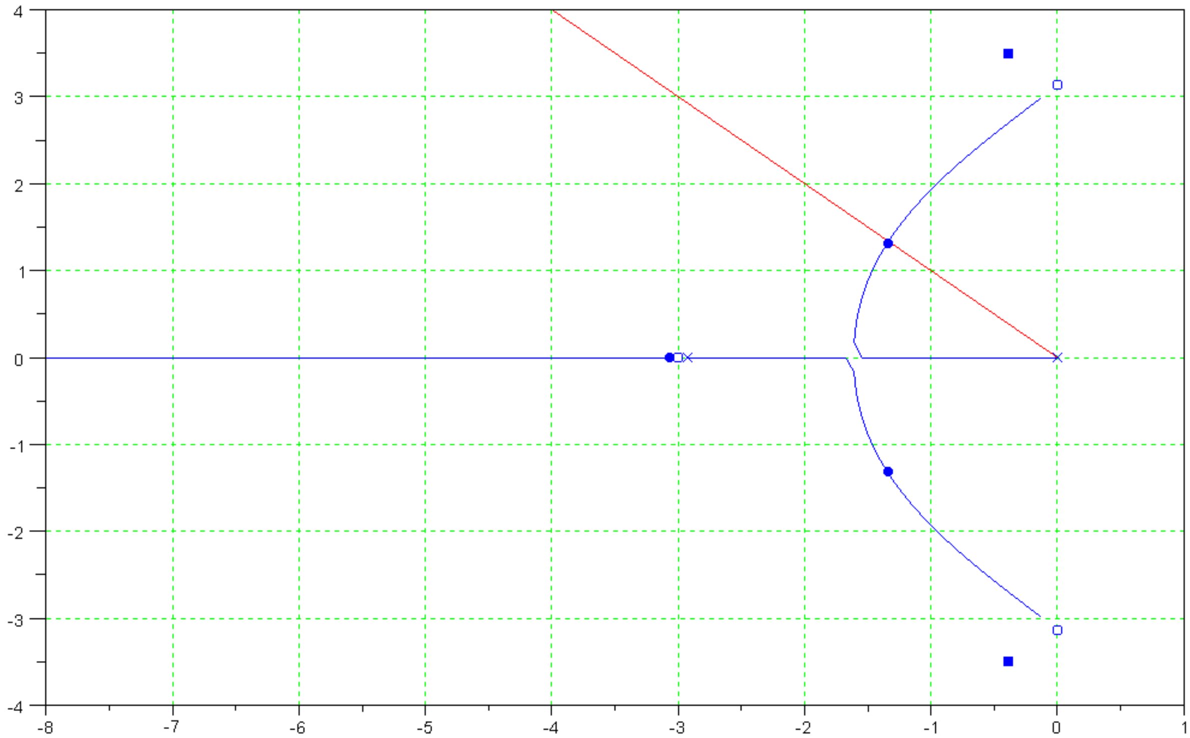
Add a compensator. Let's cancel the slow poles and move them out to -10, -11, -12:

$$K_3(s) = \left(\frac{k(s+3)(s+0.388+j3.5)(s+0.388-j3.5)}{(s+10)(s+11)(s+12)} \right)$$

```
-->K3 = zp2ss([-0.388+j*3.5,-0.388-j*3.5,-3],[-10,-11,-12],1);
```

Sketch the resulting root locus:

```
-->G3K3 = ssmult(G3,K3);
-->k = logspace(-2,2,1000)';
-->R3 = rlocus(G3K3,k);
```



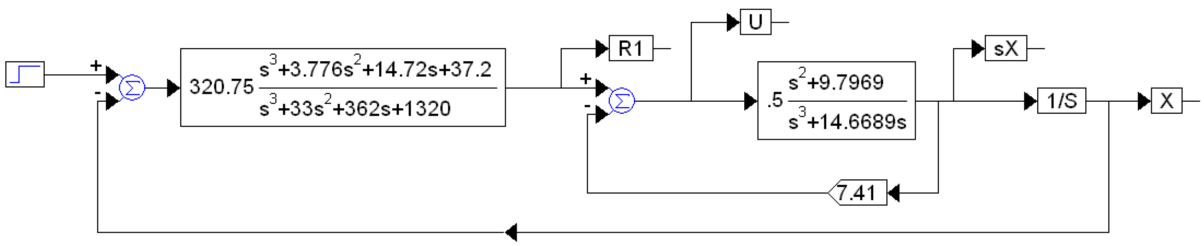
Pick a point on the root locus, such as -1.3 + j1.3: (marked with the blue dot above)

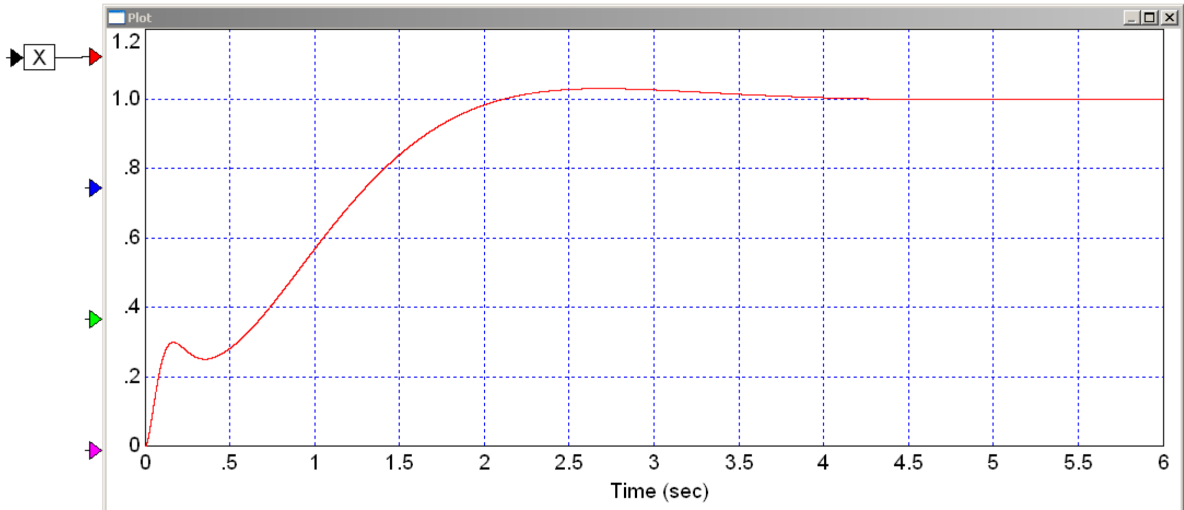
```
-->gain = 1/abs(evalfr(G3K3,-1.3+j*1.3))
 gain  =

    320.37505
```

As a check, look at the closed-loop poles:

```
-->G4 = intcon(G3K3, gain);

-->eig(G4)
 ans  =

  - 15.086998 + 14.345848i
  - 15.086998 - 14.345848i
  - 1.3431398 - 1.3183664i
  - 1.3431398 + 1.3183664i        this is the pole you placed at -1.2 + j1.3
  - 3.0682936 - 1.623D-15i
  - 0.3883652 + 3.4991768i
  - 0.3883652 - 3.4991768i
```

and check the step response:



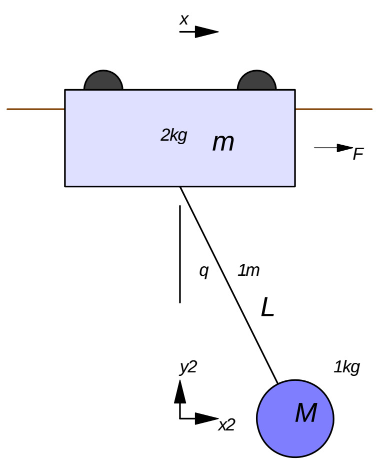


The net result is you can move the gantry system over one meter in about 3 seconds.

Root Locus for Unstable Systems
Inverted Pendulum

Example:

Find the dynamics for the following system: A 1kg mass swings from a gantry which weighs 2kg. The length of the rope is 1m. (upside down figure)



$$F = (M+m)\ddot{x} + LM\ddot{\theta}\cos\theta - LM\dot{\theta}^2\sin\theta$$

$$T = LM\ddot{x}\cos\theta + LM\dot{x}\dot{\theta}\sin\theta + LM\ddot{\theta} - LM\dot{x}\dot{\theta}\sin\theta - MgL\sin\theta$$

Assuming the force on the cart is produced by a motor driving a wheel with radius r

$$Fr = T$$

Substituting M=1, m=2, L=1, r = 0.1, the dynamics are

$$\begin{bmatrix} 3 & \cos\theta \\ \cos\theta & 1 \end{bmatrix}\begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} \dot{\theta}^2\sin\theta \\ g\sin\theta \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix}F + \begin{bmatrix} 0 \\ 1 \end{bmatrix}0.1F$$

Let g = +9.8 m/s2 and linearize about zero:

$$s\begin{bmatrix} x \\ \theta \\ x' \\ \theta' \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -4.9 & 0 & 0 \\ 0 & 14.7 & 0 & 0 \end{bmatrix}\begin{bmatrix} x \\ \theta \\ x' \\ \theta' \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0.45 \\ -0.35 \end{bmatrix}F$$

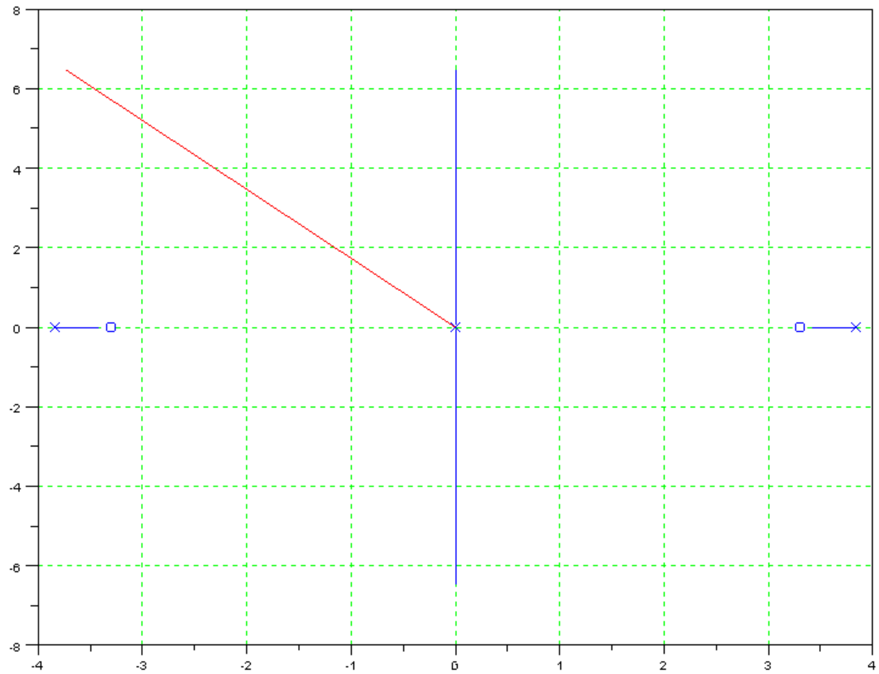
$$y = x = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \theta \\ x' \\ \theta' \end{bmatrix}$$

Now, let's try to control the position while stabilizeing the person riding the cart. The transfer function from F to x is:

```
->G = sspack(A,B,C,D)

    X      Q       sX     sQ      B:D
    0.     0.       1.     0.     0.
    0.     0.       0.     1.     0.
    0.   - 4.9      0.     0.     0.45
    0.     14.7     0.     0.   - 0.35
    1.     0.       0.     0.     0.

-->k = logspace(-2,2,100);

-->[z,p,k] = ss2zp(G)
 k  =

    0.45
 p  =

    0
    0
    3.8340579
  - 3.8340579
 z  =

  - 3.2998316
    3.2998316
```

$$x = \left(\frac{0.45(s+3.3)(s-3.3)}{s^2(s+3.83)(s-3.83)} \right) F$$

with the following root locus:



At this point, you're really stuck. The pole at +3.83 goes to the zero at +3.3 and there not much you can do about it.

The math is trying to tell you something. The problem is figuring out what it's trying to say.

The message is there just isn't enough information just measuring the position. Without enough information, you can't stabilize the system. You need to measure the angle as well.

So, let's first try to stabilze the angle. Changing the output to the angle, θ, the transfer function becomes:

$$y = \theta = \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \theta \\ x' \\ \theta' \end{bmatrix}$$

```
->Cq = [0,1,0,0]
    0.    1.    0.    0.

-->Gq = sspack(A,B,Cq,D)

    X     Q      sX    sQ     B:D
    0.    0.      1.    0.    0.
    0.    0.     0.    1.    0.
    0.  - 4.9    0.    0.    0.45
    0.    14.7   0.    0.  - 0.35
    0.    1.     0.    0.    0.
```

(note: the C matrix is in bold)

```
->[z,p,k] = ss2zp(Gq)
```

```
 k  =

  - 0.35
 p  =

    0
    0
    3.8340579
  - 3.8340579
 z  =

    0
    0
```

or

$$\theta = \left(\frac{-0.35}{(s+3.83)(s-3.83)} \right) F$$

(The poles and zeros at s=0 cancel. You can't measure position using just angle measurements.) This you can do something about.

The root locus goes up the jw axis. To pull it left, cancel a ***stable*** pole

$$K_1(s) = \left(\frac{s+3.83}{s+10} \right)$$

```
-->K1 = zp2ss(-3.83,-10,1)
 K1  =

  - 10.     1.
  - 6.17    1.

-->GqK1 = ssmult(Gq,K1)

    x     q      sx    sq     z
    0.    0.      1.    0.    0.        0.
    0.    0.      0.    1.    0.        0.
    0.  - 4.9     0.    0.  - 2.7765    0.45
    0.    14.7    0.    0.    2.1595  - 0.35
    0.    0.      0.    0.  - 10.       1.
    0.    1.      0.    0.    0.        0.
```

(Again, the C matrix is in bold. The states are labeled above. An additional state from K(s) is added to the right.)

Plot the root locus of this system.

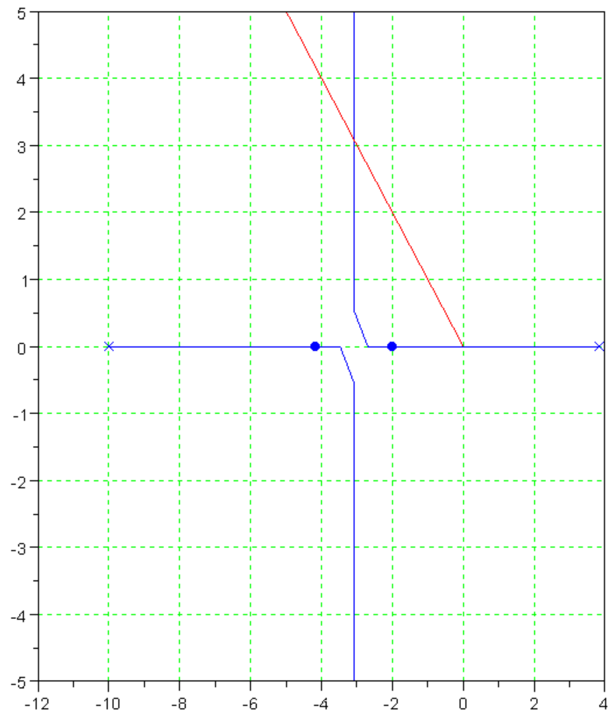
Pick a point on the root locus where the system is stable. You're not done yet so the actual response doesn't matter that much.

```
-->G8 = zp2ss([],[3.83,-10],0.35);
-->k = logspace(-2,2,1000)';
-->R8 = rlocus(G8,k*100,0.707);
```

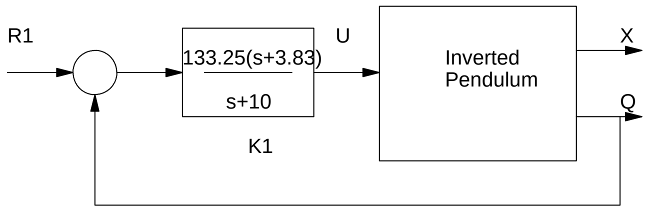
Let's place the dominant pole at -2 (shown with the dot on the following root locus plot).

k at this point is -133.25714. (It's negative to cancel the negative gain on the plant).

```
-->k1 = 1/abs(evalfr(G8,-2))

  133.25714
```



The closed loop system from R to θ is then:




```
-->Gcl1 = intcon(GqK1,k1)
 Gcl1  =

    X      Q              sX     sQ     Z           B:D
    0.     0.             1.     0.     0.          0.
    0.     0.             0.     1.     0.          0.
    0.     55.065714      0.     0.   - 2.7765    - 59.965714
    0.   - 31.94          0.     0.     2.1595      46.64
    0.     133.25714      0.     0.   - 10.       - 133.25714
    0.     1.             0.     0.     0.          0.
```

and the closed-loop transfer function is

```
-->[z,p,k] = ss2zp(Gcl1)
```

```
 k  =

    46.64
 p  =

    0
    0
  - 4.0297498 + 0.2483031i
  - 4.0297498 - 0.2483031i
  - 1.9405004
 z  =

    0
    0
  - 3.83
```

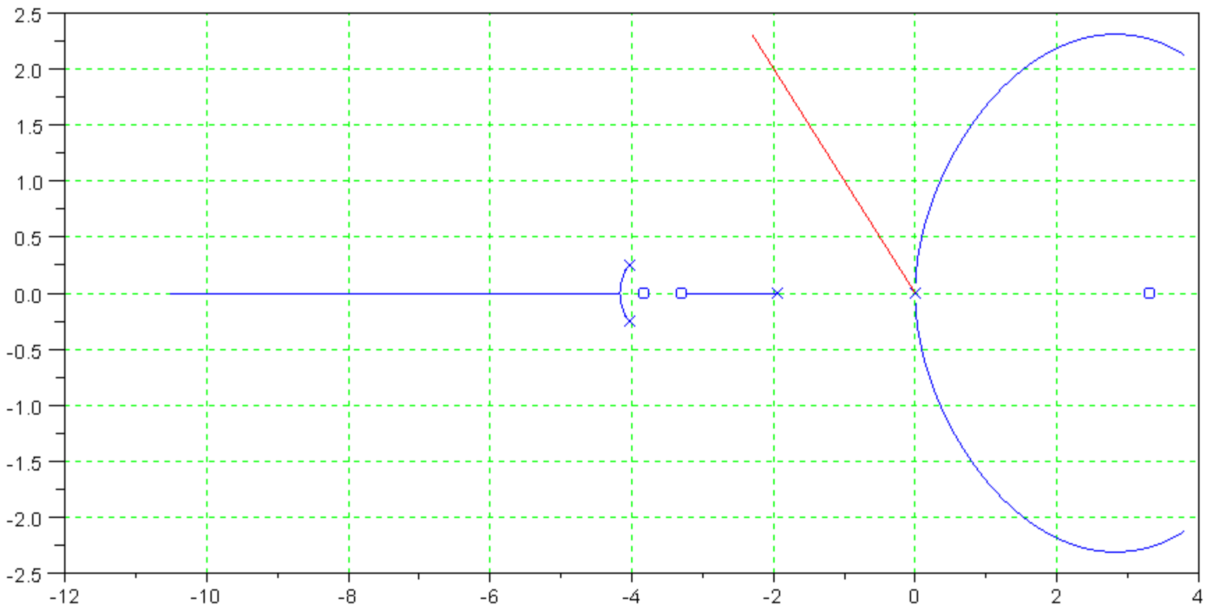
$$\theta = \left(\frac{46.64(s+3.83)}{(s+2)(s+4.02+j0.25)(s+4.02-j0.25)} \right) R$$

Now the system is stable, let's close the feedback loop around displacement, x. Find the transfer function from R to x: Change the C matrix:

```
 Gcl2  =

    X      Q             sX    sQ     Z          B:D
    0.     0.            1.    0.     0.         0.
    0.     0.            0.    1.     0.         0.
    0.     55.065714     0.    0.   - 2.7765   - 59.965714
    0.   - 31.94         0.    0.     2.1595     46.64
    0.     133.25714     0.    0.   - 10.      - 133.25714
    1.     0.            0.    0.     0.         0.

-->[z,p,k] = ss2zp(Gcl2)

 k  =

  - 59.965714
 p  =

    0
    0
  - 4.0297498 + 0.2483031i
  - 4.0297498 - 0.2483031i
  - 1.9405004
 z  =

    3.2998316
  - 3.2998316
  - 3.83
```

$$x = \left(\frac{-59.96(s+3.83)(s-3.29)(s+3.29)}{s^2(s+2)(s+4.02+j0.25)(s+4.02-j0.25)} \right) R$$

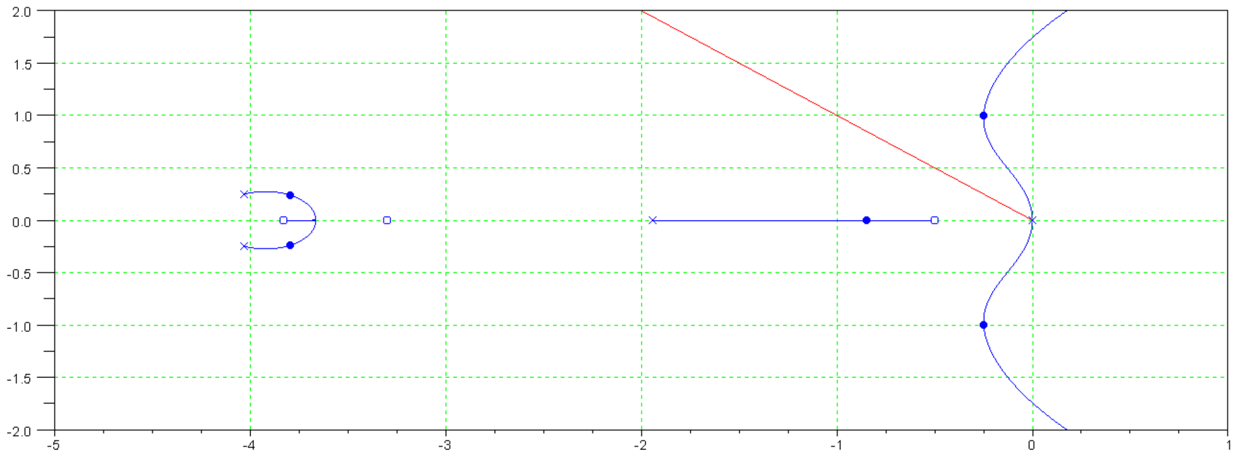
The root locus looks like the following:



The poles at s=0 go unstable as they are attracted by the zero at +3.8.

Add a zero to pull this left:

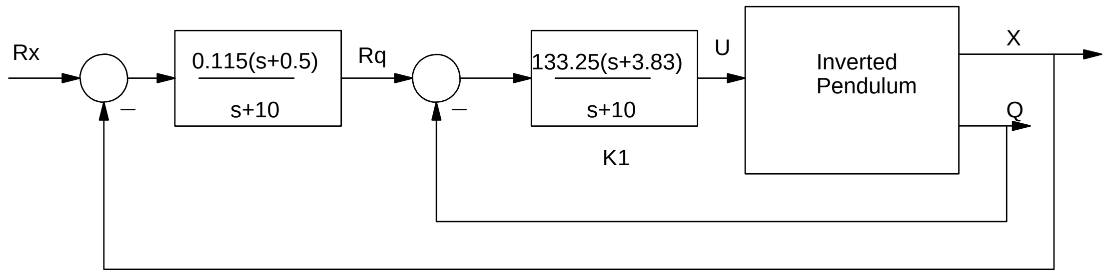
$$K_2(s) = k\left(\frac{s+0.5}{s+10}\right)$$



Place the dominant pole at -0.25 + j1:

```
-->k = 1/abs(evalfr(GK2,-0.25+j))
 k  =

    0.1153717
```

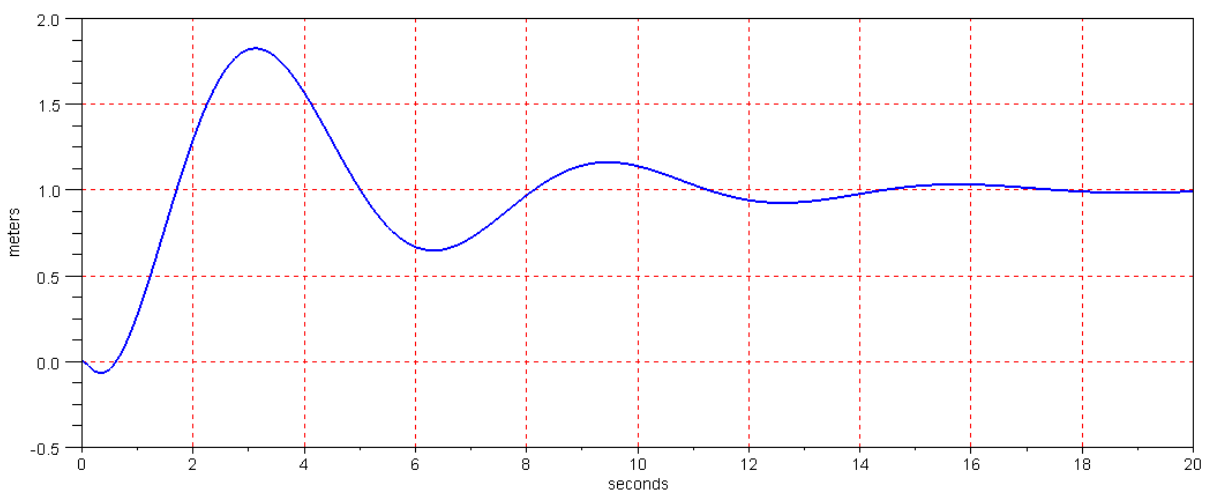
so



The closed-loop system is:

```
-->Gcl3

      x              q              sx     sq      z1          z2            B:D
      0.             0.             1.     0.     0.          0.            0.
      0.             0.             0.     1.     0.          0.            0.
      6.9140469      55.065714      0.     0.   - 2.7765      569.67429     -6.914
    - 5.377592     - 31.94          0.     0.     2.1595    - 443.08        5.377
      15.364549      133.25714      0.     0.   - 10.         1265.9429     -15.36
    - 0.1153         0.             0.     0.     0.        - 10.           0.1153
C:    1.             0.             0.     0.     0.          0.            0
```

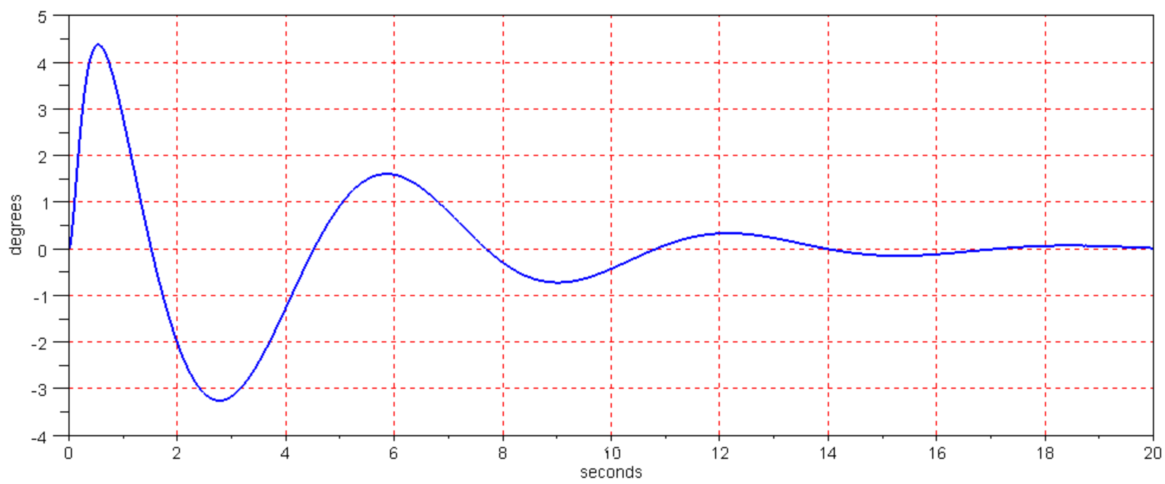
The step response to x is



The step response to angle is.... First change the C matrix to measure angle instead of displacement:

```
 Gcl3  =
 
    x             Q             sX     sQ     Z1         Z2           B:D
    0.            0.            1.     0.     0.         0.          0.
    0.            0.            0.     1.     0.         0.          0.
    6.9140469     55.065714     0.     0.   - 2.7765     569.67429   -6.91
  - 5.377592    - 31.94         0.     0.     2.1595   - 443.08      5.377
    15.364549     133.25714     0.     0.   - 10.        1265.9429   -15.36
  - 0.1153        0.            0.     0.     0.       - 10.         0.1153
    0.            1.            0.     0.     0.         0.          0.
```

```
Q = step(Gcl3,t);

plot(t,Q*180/%pi);          (plot in units of degrees)
```



The person leans less than 5 degrees to move 1 meter.

Root Locus for Lightly Damped Systems

Objectives:

To be able to design a feedback compensator for a system with poles on the jw axis.

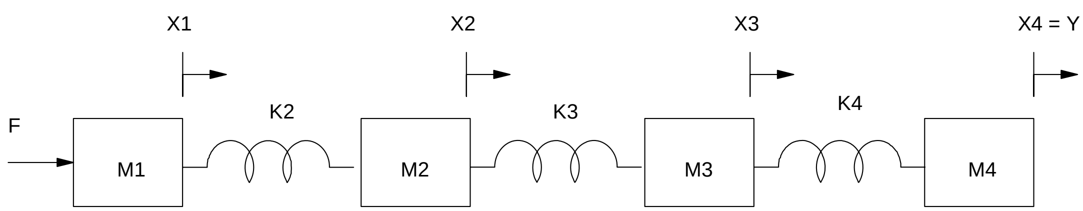
Problem:

Some systems have lots of poles on or near the jw axis. These include:

- Long flexible beams (such as the space station)
- Long strings (such as a space anchor)
- Cars drafting eachother on the freeway.

Example:

Find the transfer function for the following mass spring system. This models a vibrating string with 4 finite elements or 4 cars going down the road (where the ground is the position of the lead car)



The equations of motion are:

$$(M_1 s^2 + K_2)X_1 - K_2 X_2 = F$$

$$(M_2 s^2 + K_2 + K_3)X_2 - K_2 X_1 - K_3 X_3 = 0$$

$$(M_3 s^2 + K_3 + K_4)X_3 - K_3 X_2 - K_4 X_4 = 0$$

$$(M_4 s^2 + K_3)X_4 - K_4 X_3 = 0$$

Assuming M=K=1, the state-space model is

$$s\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ sx_1 \\ sx_2 \\ sx_3 \\ sx_4 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ sx_1 \\ sx_2 \\ sx_3 \\ sx_4 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} F$$

The poles of this system are: (from SciLab)

First, input A:

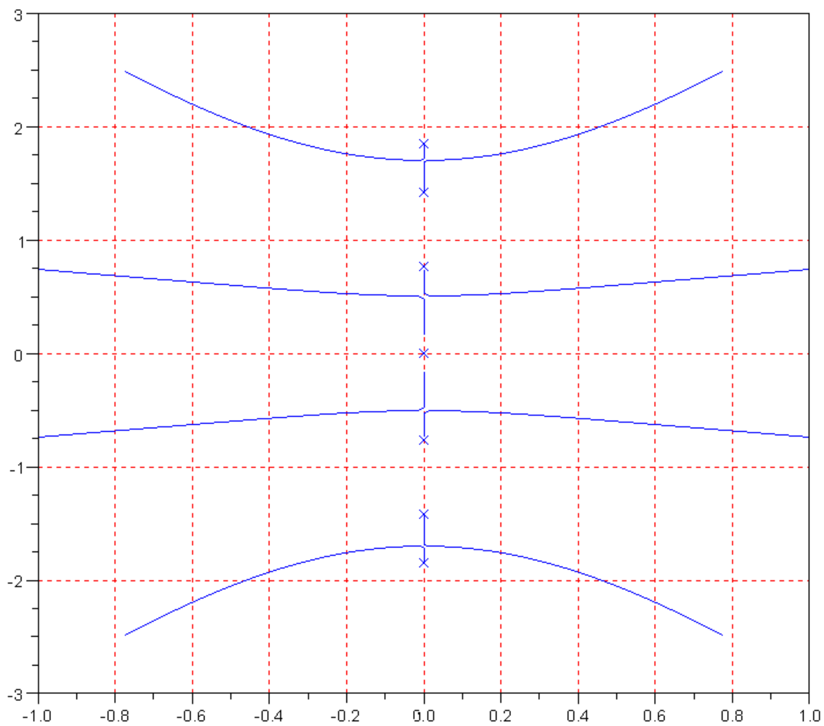
```
-->Z = zeros(4,4);
-->I = eye(4,4)
-->K = [-2,1,0,0;1,-2,1,0;0,1,-2,1;0,0,1,-1];
-->A = [Z,I;K,Z];
-->B = [0;0;0;0;0;0;0;1];
-->C = [0,0,0,1,0,0,0,0];
```

The eigenvalues of A are the poles: (spec in SciLab-speak)

```
-->spec(A)
 0 + 1.8477591i
 0 - 1.8477591i
 0 + 1.4142136i
 0 - 1.4142136i
 0 + 0.7653669i
 0 - 0.7653669i
 0 + 1.524D-08i
 0 - 1.524D-08i
```

The zeros is where the poles go as k goes to infinity. Ignore large roots (they're going to infinity via the asymptotes). There aren't any zeros.

```
If you use proportional feedback, the root locus doesn't stabilize:
 k = logspace(-2,2,1000)';
 R = rlocus(G, k, 0.707);
```

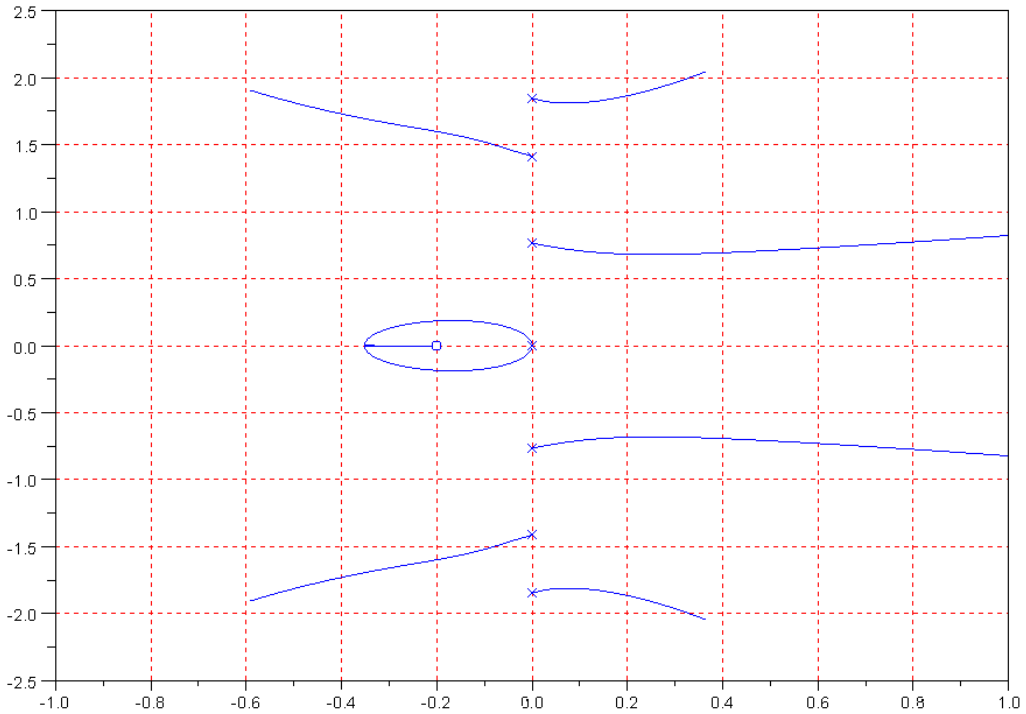


Note that you're always unstable and making things works when you add feedback

Let's add a compensator of the form

$$K(s) = k\left(\frac{s+0.2}{s+2}\right)$$

The root locus now looks like:

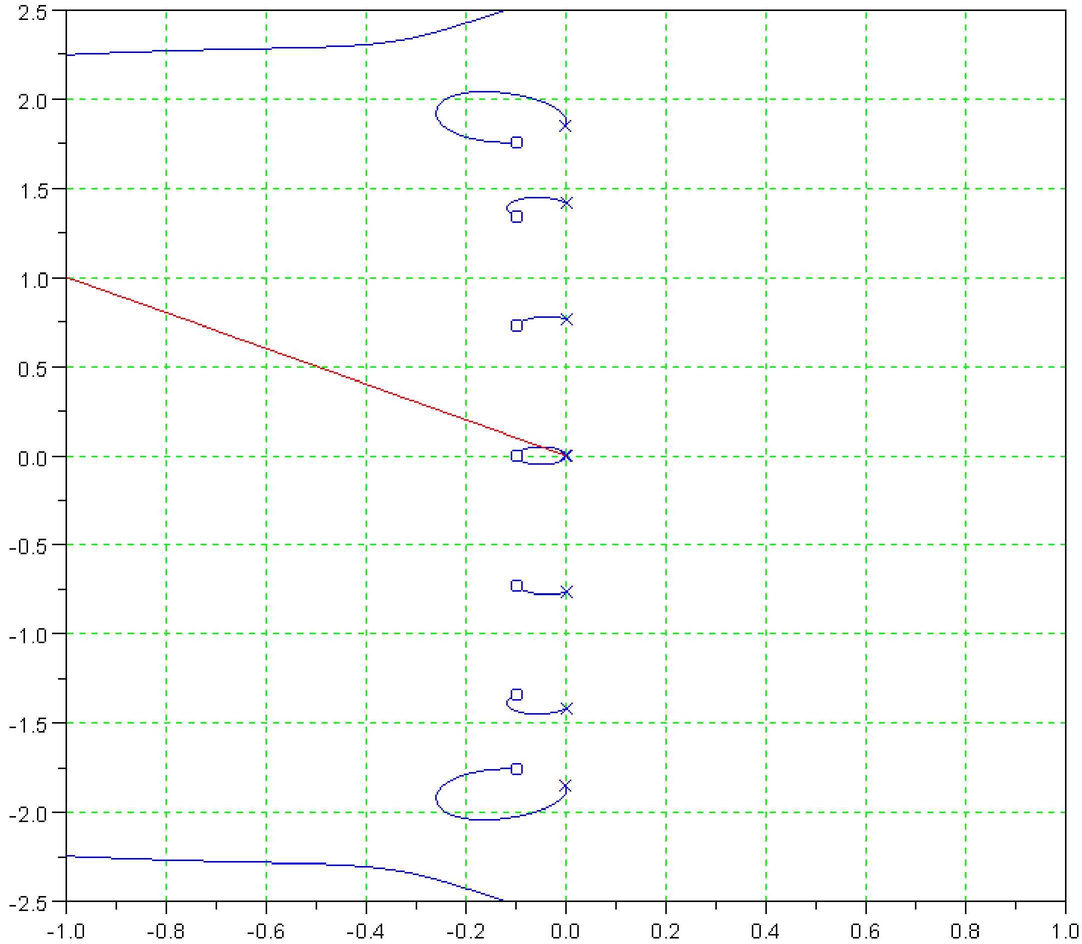


Again, more gain makes it more unstable.

Let's add a bunch of zeros near to the poles to attract them. Add a corresponding number of poles out of the way (-5 and left).

$$K(s) = k\left(\frac{(s+0.1)^2(s+0.1\pm j0.727(s+0.1\pm j1.34)(s+0.1\pm j1.75)}{(s+10)^8}\right)$$

The root locus now looks like:

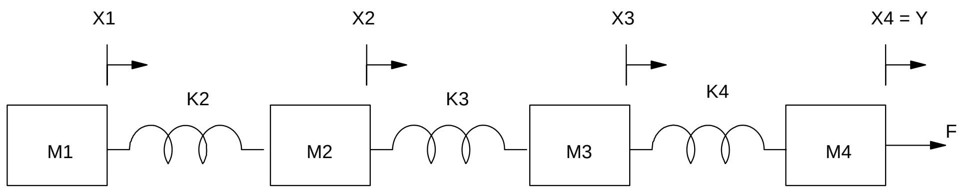


K(s) is called a comb filter. The zeros make the gain zero at several frequencies - so the gain vs. frequency plot for K(s) looks like a comb. Note that this K(s) is rather complicated.

This is one of the methods we're using to stabilze Space Station Freedom.

Solution #2:

Change where you apply the input. Use co-located feedback (the input and the output are at the same point):

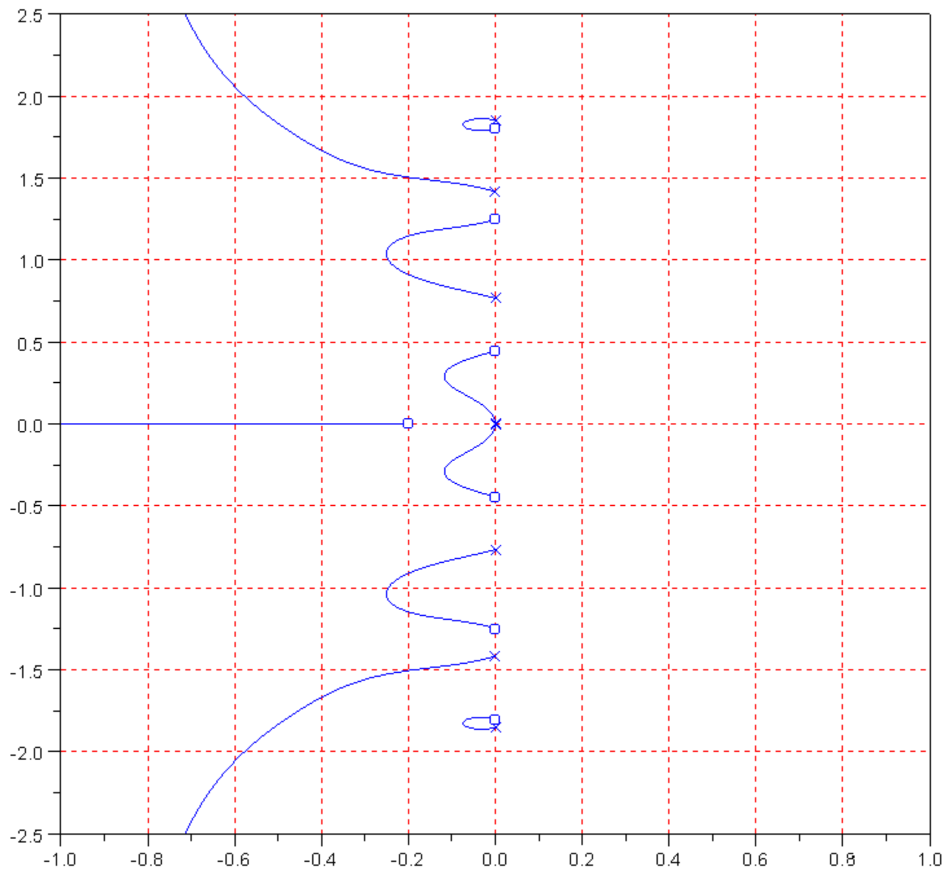


(Just change the B matrix in the previous solution). You now have poles and zeros.

Add a lead compensator for K(s)

$$K(s) = k\left(\frac{s+0.2}{s+2}\right)$$

The root locus is now stable with a *much* simpler compensator.



The optimal value for 'k' moves the roots as far left as possible:

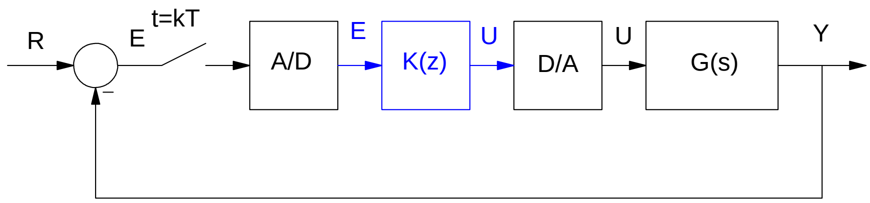
- k too small results in an oscillatory system
- k too large makes the right mass stiff - and can't dissipate energy
- 'just right' optimizes the spring and friction tied to the right to let it move and dissipate energy.

This is in essence impedance matching from ECE 351. Placing the controller anywhere but the endpoint with colocated feedback results in an impedance mismatch. Impedance mismatches cause traveling waves be reflected into the system.

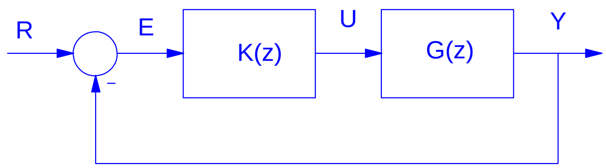
Z Transform

Introduction:

Another way to implement a controller, K(s), is to use a microcontroller. This results in the microcontroller reading the error signal at discrete times, kT, doing some number crunching (implementing K(z)), and outputting a control input, U.



Actual Feedback Control System



The world as seen by the microcontroller

When designing the software to implment K(z), its best ot view the world as the microcontroller sees it: a discrete-time system which satisfies some differece equation:

$$y(k) = f(y(k-1), y(k-1), ..., x(k), x(k-1), ...)$$

where T is the sampling rate, k is how many times you gone through the main loop, and t is the time in seconds:

$$t = kT$$

Given the right program, you can implement a filter, K(z), with a microcontroller with the same frequency response as an RC active or passive filter. The mathematical tool needed to analyze such filters is the z-transform.

Z-Transform

The LaPlace transform assumes all functions are in the form of est. This allows you to convert differential equations into algebraic equations in 's'.

In contrast,

The z-transform assumes all functions are of the form z^k.

This results in a time advance becoming multiplication by z:

$$y(k) = z^k$$

$$y(k+1) = z^{k+1} = zy(k)$$

Difference equations then become algebraic equations in 'z'. For example, the difference equation

$$y(k+1)+5y(k)=5u(k)$$

can be written as

$$zY+5Y=5U$$

or as a transfer function

$$Y=\left(\frac{5}{z+5}\right)U$$

Some Select z-Transforms.

Note that zY means 'the next value of Y'. Similarly, z-1Y means 'the previous value of Y' or Y delayed one sample. With this, you can derive some z-transforms.

i) Delta Function $\delta(k)$

The discrete-time delta function is

$$\delta(k)=\begin{cases}1 & k=0\\ 0 & otherwise\end{cases}$$

It's z-transform is '1', just like the s-domain.

ii) Unit Step:

The unit step is

$$u(k)=\begin{cases}1 & k\geq 0\\ 0 & otherwise\end{cases}$$

It's z-transform can be derives as follows. The unit step is:

k	0	1	2	3	4	5	6	7
u(k)	1	1	1	1	1	1	1	1
(1/z)*u(k)	0	1	1	1	1	1	1	1
Subtract								
(1-1/z)u(k)	1	0	0	0	0	0	0	0

So,

$$\left(1-\frac{1}{z}\right)u(k)=1$$

$$\left(\frac{z-1}{z}\right)u(k)=1$$

$$u(k)=\frac{z}{z-1}$$

iii) Decaying exponential

Let

$$x(k)=a^k u(k)$$

k	0	1	2	3	4	5	6	7
x(k)	1	a	a^2	a^3	a^4	a^5	a^6	a^7
a*(1/z)*x	0	a	a^2	a^3	a^4	a^5	a^6	a^7
Subtract								
(1-a/z)x	1	0	0	0	0	0	0	0

so

$$(1 - \tfrac{a}{z})X = 1$$

$$(\tfrac{z-a}{z})X = 1$$

$$X = (\tfrac{z}{z-a})$$

Table of z-Transforms

Similar to LaPlace transforms, you can convert from the time domain to the z-domain using just a few transforms:

y(k)	Y(z)
$\delta(k)$	1
$u(k)$	$\frac{z}{z-1}$
$a^k u(k)$	$\frac{z}{z-a}$
$2b \cdot a^k \cdot \cos(k\theta + \phi) \cdot u(k)$	$\left(\frac{(b\angle\phi)z}{z-(a\angle\theta)}\right) + \left(\frac{(b\angle-\phi)z}{z-(a\angle-\theta)}\right)$

For the last entry, assume you have a complex conjugate pair:

$$Y(z) = \left(\frac{(b\angle\phi)z}{z-(a\angle\theta)}\right) + \left(\frac{(b\angle-\phi)z}{z-(a\angle-\theta)}\right)$$

Take the inverse z-transform

$$y(k) = \left((b\angle\phi)(a\angle\theta)^k + (b\angle-\phi)(a\angle-\theta)^k\right)u(k)$$

Simplifying

$$y(k) = b(a^k)(e^{j(k\theta+\phi)} + e^{-j(k\theta+\phi)})u(k)$$

Using Euler's identity

$$y(k) = 2b(a^k)\left(\frac{e^{j(k\theta+\phi)} + e^{-j(k\theta+\phi)}}{2}\right)u(k)$$

$$y(k) = 2b \cdot a^k \cdot \cos(k\theta + \phi) \cdot u(k)$$

Solving Difference Equations Using z-Transforms

Problem #1: Suppose you put \$100 into a savings account every month. The account makes 6% interest per year (0.5% per month). Find the value of your savings account each month.

Solution: Let the sampling rate be 1 month (T=1 month). Let X be the value of your savings account. The equation for x(k) is

$$x(k+1) = 1.005x(k) + 100$$

Converting to the z-doman

$$zX = 1.005X + 100\left(\frac{z}{z-1}\right)$$

Solving

$$(z - 1.005)X = 100\left(\frac{z}{z-1}\right)$$

$$X = \left(\frac{100z}{(z-1)(z-1.005)}\right)$$

Using partial fractions - keeping a z on top since we'll need it later

$$X = z\left(\left(\frac{-20000}{z-1}\right) + \left(\frac{20000}{z-1.005}\right)\right)$$

Taking the inverse z-transform

$$x(k) = 20000(1.005^k - 1)u(k)$$

After 12 months, you have \$1233.55 in your savings account.

Problem #2: Assume you borrow \$100,000 for a house. How much do you have to pay each month to pay off the loan in 10 years? Assume 6% interest per year (0.5% per month).

Solution: The amount you owe next month (x(k)) is

$$x(k+1) = 1.005x(k) - p$$

where 'p' is your monthly payment. Converting to the z-domain

$$zX - X(0) = 1.005X - p\left(\frac{z}{z-1}\right)$$

$$(z - 1.005)X = X(0) - p\left(\frac{z}{z-1}\right)$$

$$X = \left(\frac{X(0)}{z-1.005}\right) - p\left(\frac{z}{(z-1)(z-1.005)}\right)$$

Using partial fractions

$$X = \left(\frac{X(0)}{z-1.005}\right) + pz\left(\left(\frac{2000}{z-1}\right) - \left(\frac{2000}{z-1.005}\right)\right)$$

Converting back to the time domain

$$x(k) = 1.005^k X(0) - 2000p(1.005^k - 1)u(k)$$

After 10 years (k=120 payments), x(k) should be zero

$$x(120) = 0 = \$181,939 - 2000p(0.8194)$$

$$p = \$111.02$$

Your monthly payments are \$111.02.

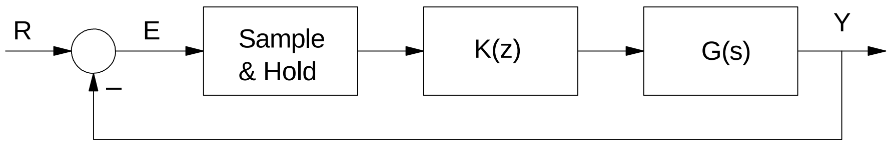
Root Locus in the z-Domain

Objectives:

- Sketch a root locus in the z-plane

Discussion:

Relative to the microcontroller, a feedback system looks like it's a discrete-time system. It looks like is's in the z-domain. The goal is to find the compensator, K(z), which gives a 'good' response.



Mathematically, the open-loop transfer function is

$$Y = H(z)G(z)K(z)E$$

where H(z) is the transfer function of the sample and hold. This is approximately a 1/2 sample delay which is ofen ignored or lumped into G():

$$H() = e^{-sT/2}$$

If you ignore H, the closed-loop transfer function is

$$Y = \left(\frac{GK}{1+GK}\right)R$$

If GK has zeros and poles:

$$GK = k\frac{z}{p}$$

the closed-loop transfer function becomes

$$Y = \left(\frac{kz}{p+kz}\right)E$$

The roots of the closed-loop system are then just as they were in the s-plane:

$$p(z) + kz(z) = 0$$

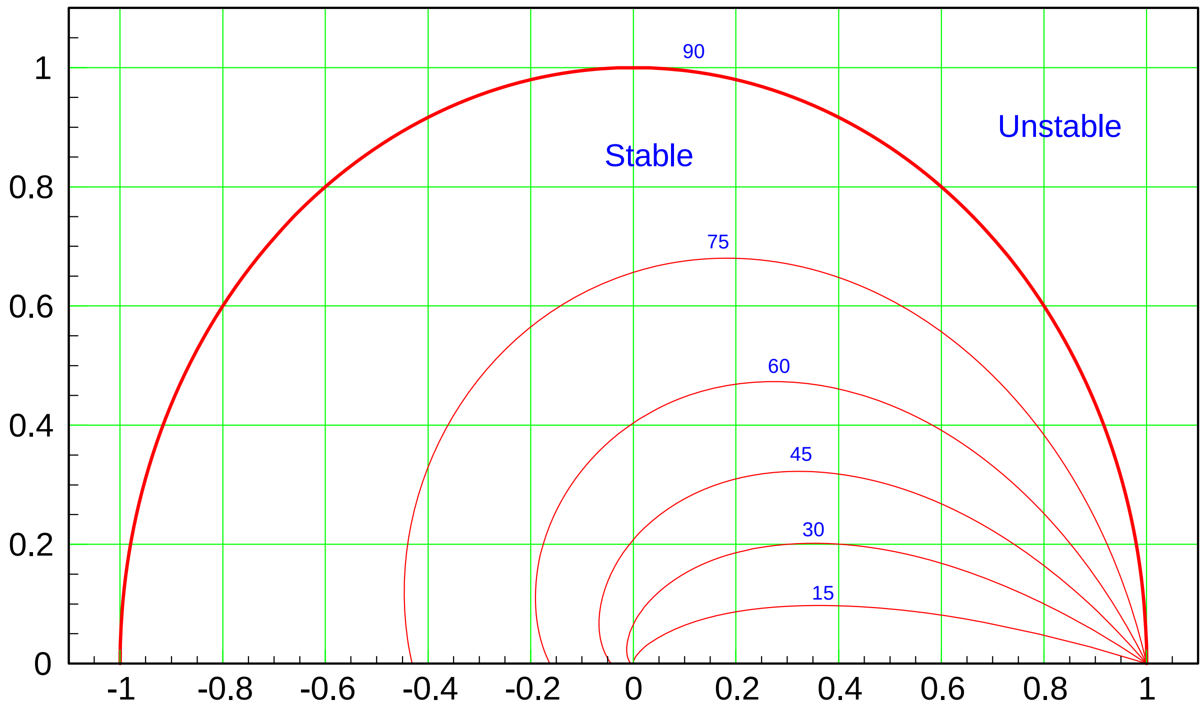
Mathematically, the roots to this polynomial don't care if the variable is 's', 'z', or anything else. The roots (and hence root locus plot) behave just the same in the s-plans as they do in the z-plane.

The only difference in the z-plans is how you interprit the meaning of the pole locations.

Recall the relationship between the s-pland and the z-plane is

$$z = e^{sT}$$

This causes the damping lines in the s-plane to spiral in the z-plane as follows:



The point you want to pick in the z-plane for a good response is then:

- The dominant pole is the pole closest to s=0. In the z-plane, it's the pole closest to z=1.
- Pick 'k' so that you are stable. All poles are inside the unit circle.
- Better yet, pick 'k' so that the dominant pole has a desired damping ratio.

Example: A microcontroller is used to control a system with the following dynamics

$$Y = \left(\frac{100}{(s+10)(s+20)}\right) U$$

Assume a sampling rate of 10ms. Find the maximum 'k' for stability and 'k' for a damping ratio of 0.5.

Solution: First, convert G(s) to G(z).

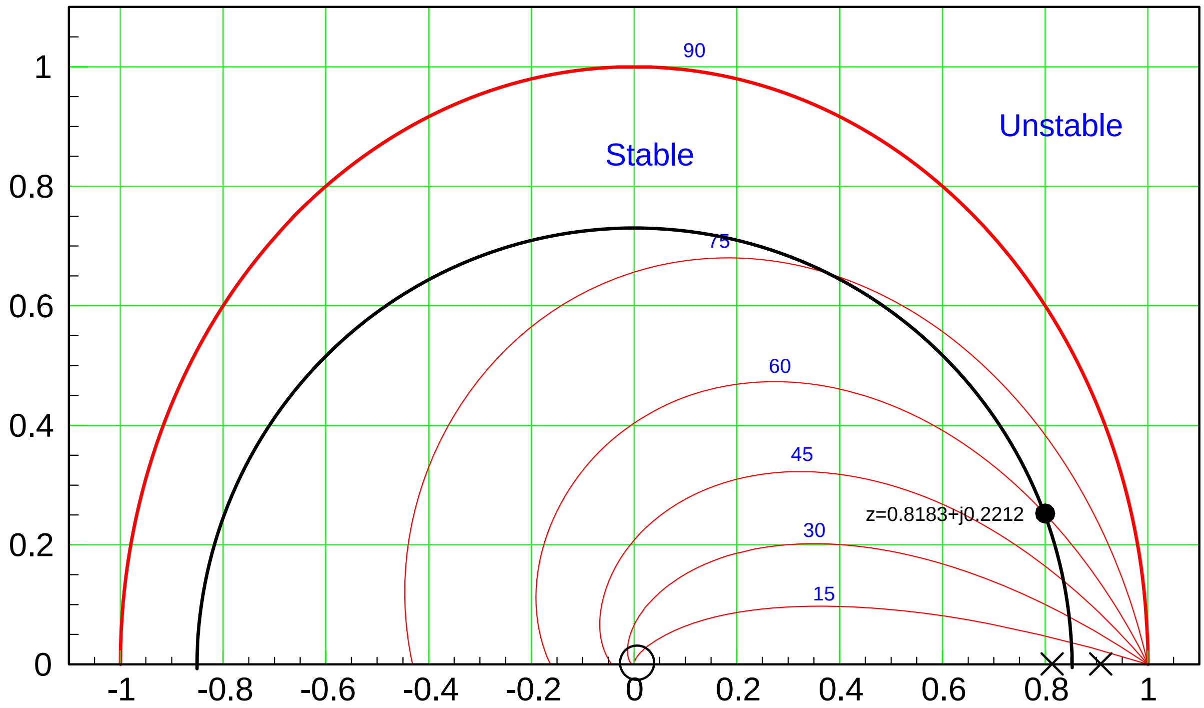
$$s = -10 \Rightarrow z = e^{sT} = 0.9048$$

$$s = -20 \Rightarrow z = e^{sT} = 0.8187$$

Define G(z) to have poles at {0.9048, 0.8187}. Add a gain to make the DC gain match G(s)

$$G(z) = \left(\frac{0.0173z}{(z-0.9048)(z-0.8187)}\right)$$

Next, sketch the root locus. Include in the root-locus plot the 60 degree damping ratio line



For a damping ratio of 0.5, search along the line

$$s = \alpha\angle 120^0$$

$$z = e^{sT}$$

until the angles add up to 180 degrees:

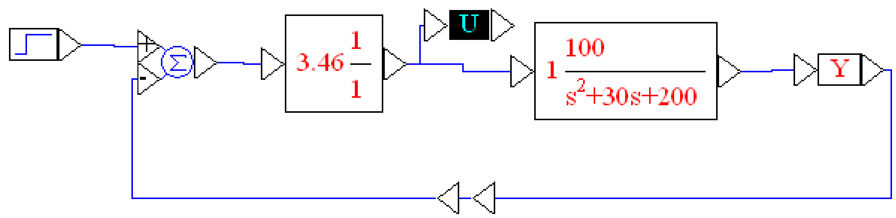
$$z = 0.8318 + j0.2212$$

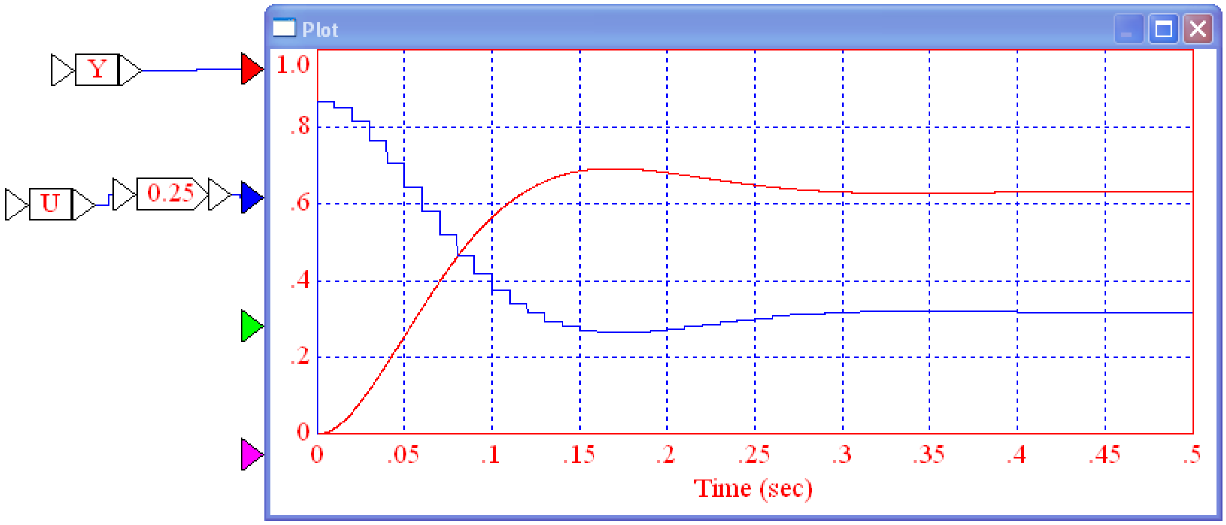
$$G(z) = -0.2886\backslash$$

Pick k to make the gain equal to -1 at this point on the z-plane

$$k = 3.46$$

Verifying that this does result in a damping ratio of 0.5 (meaning you should have 16.3% overshoot), simulate this in VisSim. The plant is in the s-domain (it's a continuous time system.) The contrller sees the error, sampled every 10ms:

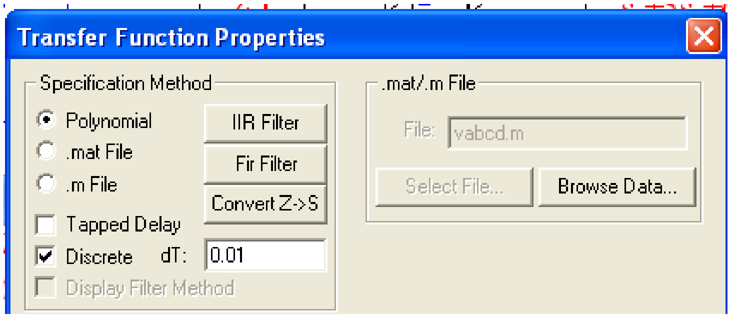




Note that the overshoot is a little off due to ignoring the 1/2 sample delay from the sample and hold.

As a sidelight, VisSim is a pretty friendly package. To include the 10ms sampling rate,

- Right click on the gain block, K(z) (the 3.46 block above)
- Click on Discrete (this tells VisSim that this block is in the z-domain)
- Set the sampling rate to 0.01 (10ms)



VisSim allows you to mix analog (s-domain) and discrete (z-domain) systems. You can see this with the red line above being continuous while the blue line has a sampling rate of 10ms. If you mix systems like this, you have a more accurate model for how the actual system will behave:

- The controller, K, is implemented with a microcontroller and hence has a discrete sampling rate
- The plant, G, is probably an analog system, such as a mass spring system, room temerature, etc.

Setting the Samping Rate:

For this system, the sampling rate was given as 10ms. What sampling rate should you pick if it wasn't given?

If you sample too slowly, you miss the dynamics. G(s) has a dominant pole at -10. If you sample at T = 1 second, these poles in the z-plane are

s = -10 $\quad z = e^{sT} = e^{-10} \approx 0$

If you sample too fast, you get numerical problems. If you sample at 1us, for example,

s = -10 $\quad z = e^{sT} = 0.99999$

In one sample, the output changes by 0.001%. Noise in the system will probably be more than the signal. Furthermore, you're asking the controller to make 300,000 adjustments before you reach steady-state.

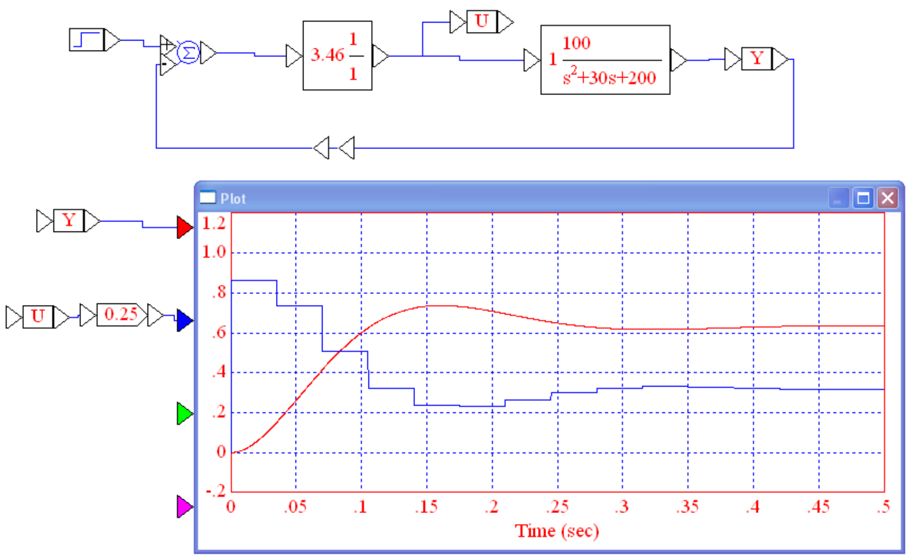
A 'good' sampling rate places the dominant pole in the z-plane around 0.7 to 0.95.

- A dominant pole at z = 0.7 results in a 2% settling time of 10 samples. The controller adjusts the input 10 times before reaching steady-state, which is pretty reasonable.
- If you're going to add a gain compensator, picking the sampling rate so that the dominant pole is around 0.7 to 0.8.
- If you're going to add a lead compensator to speed up the system, the open-loop poles will be slower (since you're going to speed up the system), meaning you want them to be around 0.9 to 0.95.

Pick T so that

$$e^{sT} \approx 0.7 \text{ to } 0.95$$

For this system, you could have used a sampling rate of 35ms (resulting in the dominant pole being at z = 0.7) rather than 10ms. That wouldn't change the resulting systems performance very much - but would allow the microcontroller to run 3.5x slower.



Lead Compensation in the z-Domain

Objectives:

- Design a lead compensator in the z-domain to speed up a system

Discussion:

The purpose of a lead compensator is to speed up a system. In the s-plane, this is done by pulling the root locus to the left. In the z-plane, this is done by pulling the root locus to the origin.

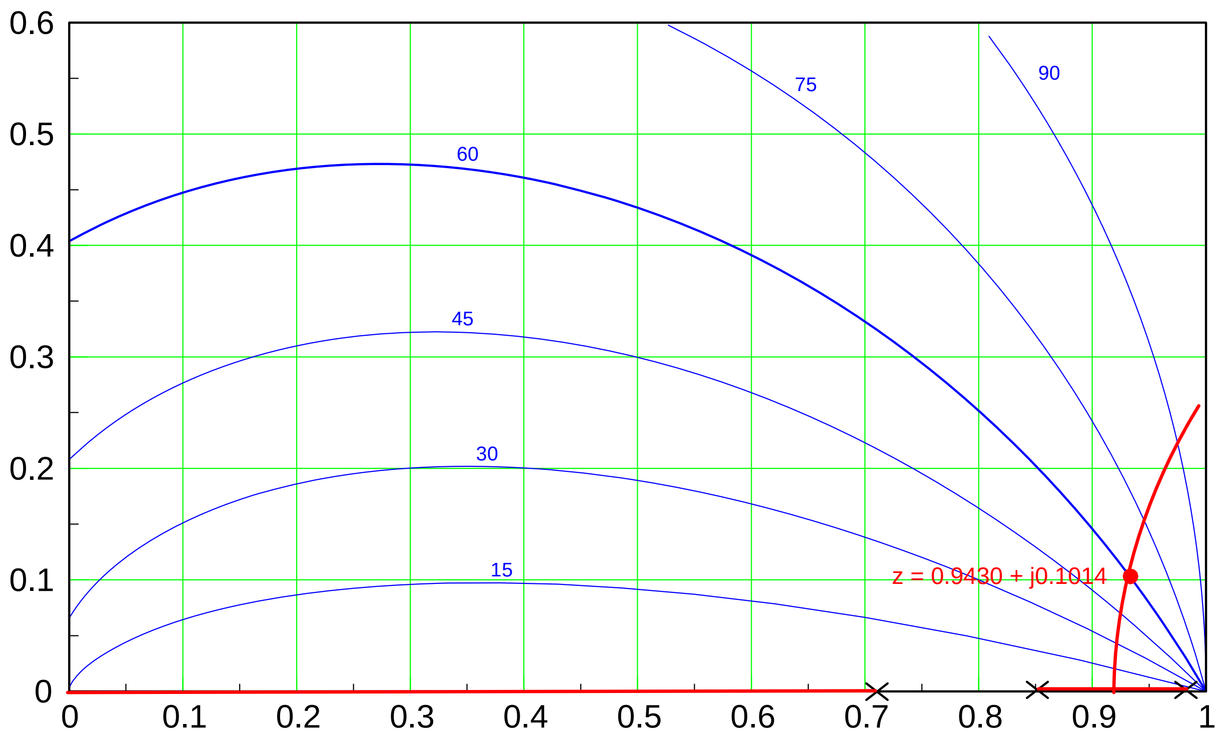
Example: Find a gain compensator for G(s) to give a damping ratio of 0.5.

$$G(s) = \left(\frac{1}{s^3+5s^2+6s+1}\right) = \left(\frac{1}{(s+0.1981)(s+1.555)(s+3.2469)}\right)$$

From before, the discrete-time equivalent with a sampling rate of 0.1 second is

$$G(z) = \left(\frac{0.0001147(z+0.2358)(z+3.3011)}{(z-0.9803)(z-0.8560)(z-0.7227)}\right)$$

The root locus for this system is show below:



For a damping ratio of 0.6, you want to place the dominant pole at -0.9390+j0.0944. At this point

GK = -1

k = 4.5093

The 2% settling time should be

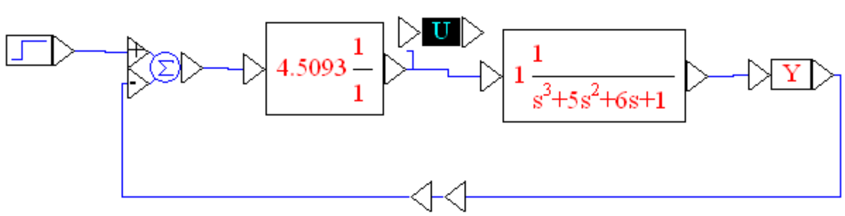
$$|pole| = |0.9390 + j0.1014| = 0.9395$$

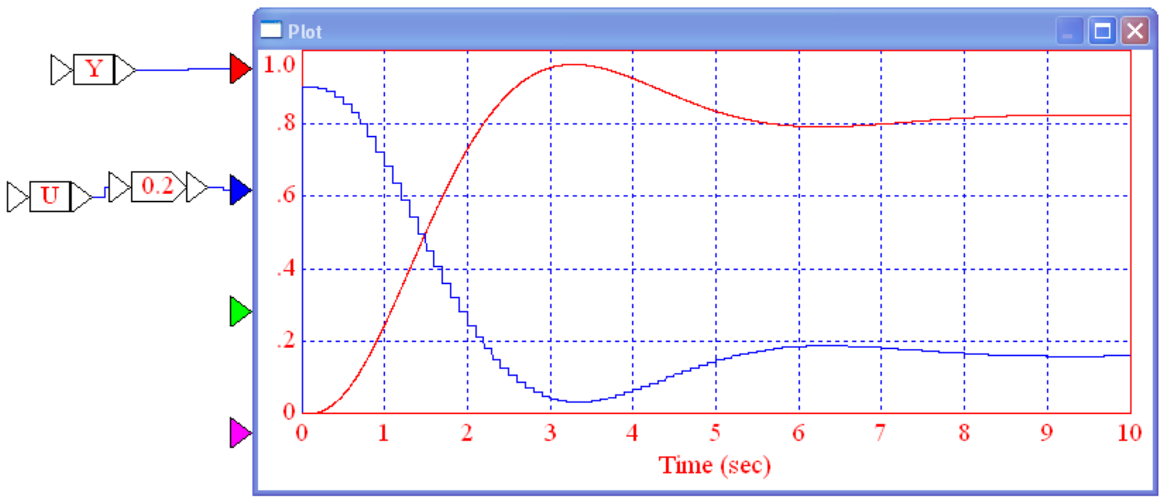
$$(0.9395)^k = 0.02$$

$$k = 62.7 \text{ samples}$$

$$t = kT = 6.27 \quad \text{seconds}$$

The step response is shown below (from VisSim):





Note that the overshoot is a little too large. This is due to ignoring the 1/2 sample delay associated with the zero-order hold. You can get better results if you include this delay:

$$G(s) = \left(\frac{1}{s^3+5s^2+6s+1}\right) \cdot e^{-0.5Ts}$$

Search along the line

$$s = \alpha\angle\theta$$

where

$$\cos\theta = \zeta = 0.6$$

until the angle of G(s) is 180 degrees. Pick 'k' to make the gain one at that value of 's'. This gives more accurate results, but makes the root locus hard to draw. A 1/2 sample delay adds half of a pole at z=0. I don't know how to add half of a pole on a root locus plot.

Regardless of whether I can draw the root locus, I can find the numerical solution to

$$GK + 1 = 0$$

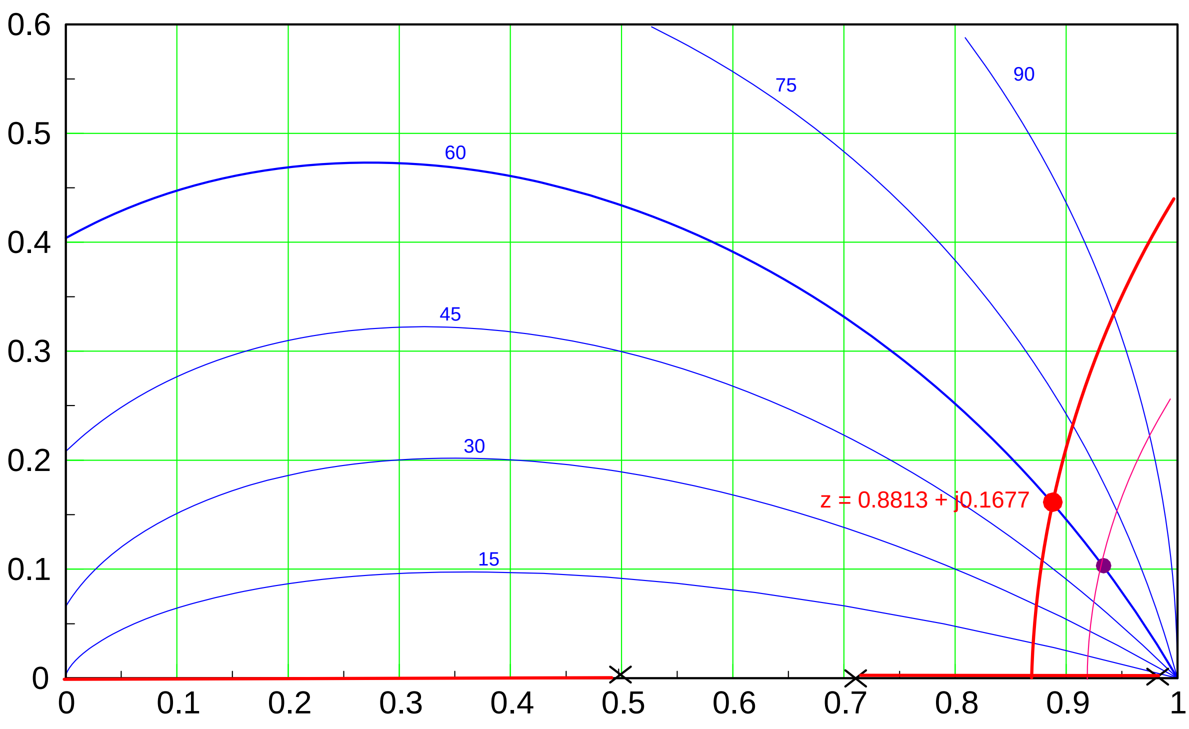
which is the one point on the root locus (whatever it looks like) that you care about. That's what you're doing when you solve for s that makes the angle of G(s) equal to 180 degrees:

$$GK = -1 = 1\angle 180^0$$

To speed up the system, pull the root locus closer to the origin. The pole near z=1 is good: it keeps the tracking error low. The pole at 0.8560 is slowing down the system, though. Let's cancel it and move it closer to the origin. Also lets cancel the zero at 0.223 to make the sketch easier.

$$K(z) = k\left(\frac{z-0.8560}{z-0.5}\right)$$

Sketching the new root locus, the intersect withthe 60 degree damping line is at 0.8813+j0.1667:



Picking 'k' to make the gain equal to -1 at this point:

GK(z)=-0.0374

k = 26.7486

so

$$K(z) = 26.7486\left(\frac{z-0.8560}{z-0.5}\right)$$

This speeds up the system, resulting in the 2% settling time being:

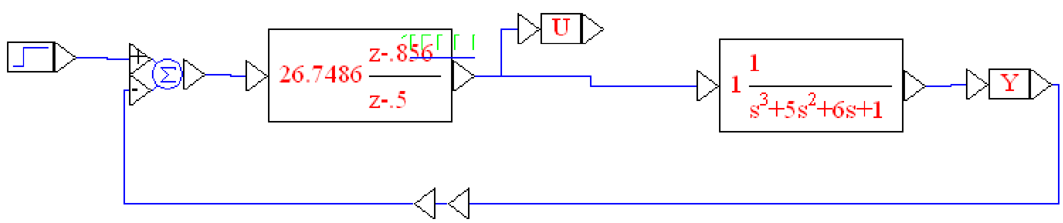
$$|0.8813 + j0.1677| = 0.8971$$

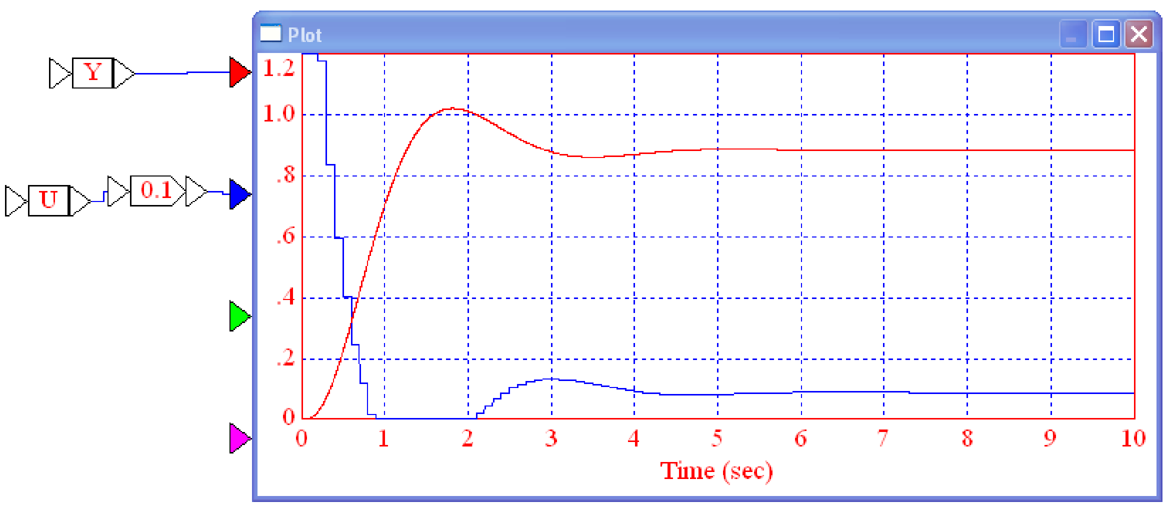
$$(0.8971)^k = 0.02$$

$$k = 36.0$$

$$t = kT = 3.60 \text{ seconds}$$

The resulting step reponse now settles out in 3.6 seconds. As expected, the lead compensated system is faster





Note that the price you pay for a faster response is a larger input. U(k) went from a peak of 4.5 with K(z)=k to 26.7 with a lead compensator.

Sidelight: Again, the overshoot is a little too large. This is due to ignoring the 1/2 sample delay. You can get better results by analyzing

$$G(s)K(z) = \left(\frac{1}{s^3+5s^2+6s+1}\right) \cdot e^{-0.5Ts} \cdot \left(\frac{z-0.8560}{z-0.5}\right)$$

Search along the line in the s-plane

$$s = \alpha\angle\theta$$

$$\cos\theta = \zeta = 0.6$$

$$z = e^{sT}$$

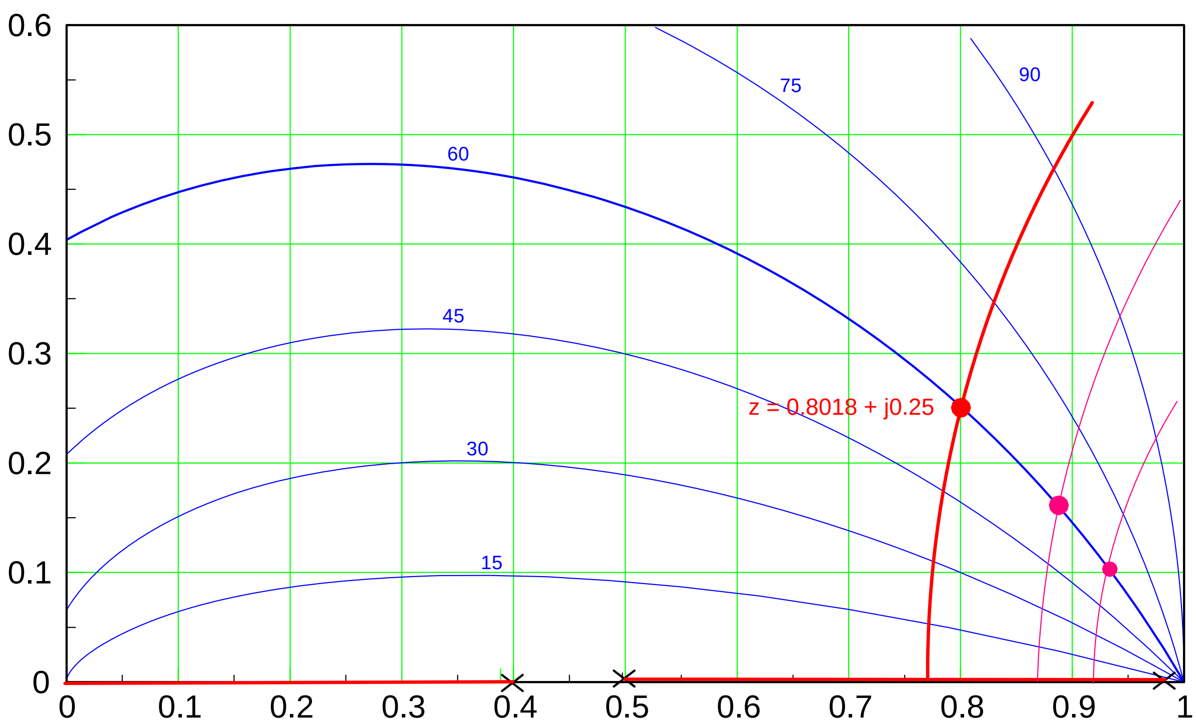
until the angle adds up to 180 degrees. Pick 'k' to make the gain one at this point. This is a more accurate model (it includes the 1/2 sample delay from the sample and hold) and likewise gives a more accurate value of 'k'.

If this isn't fast enough, cancel the next slowest pole (the one at 0.7227)

$$K(z) = k\left(\frac{(z-0.8560)(z-0.7227)}{(z-0.5)(z-0.4)}\right)$$

The point on the root locus which crosses the 60 degree damping line is

z = 0.8018 + j0.25



GK = -0.0011

k = 87.74

$$K(z) = 87.74\left(\frac{(z-0.8560)(z-0.7227)}{(z-0.5)(z-0.4)}\right)$$

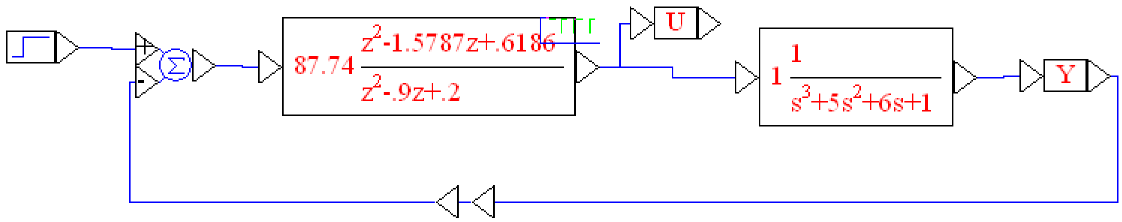
Now, the 2% settling time should be

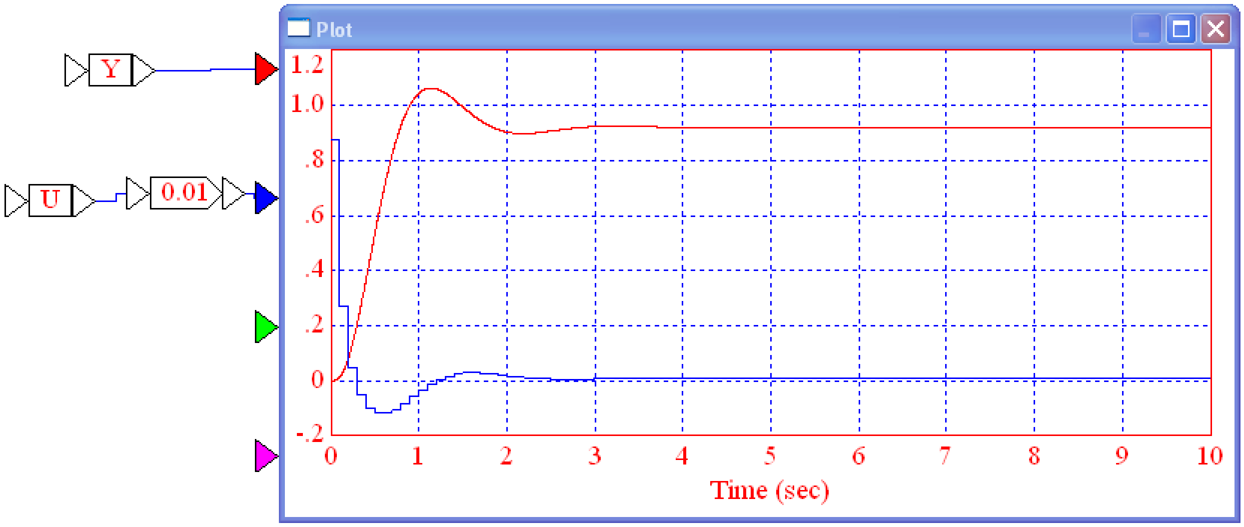
$$|pole|^k = 0.8399^k = 0.02$$

$k = 22.4$ samples

t = 2.24 seconds

The step response is as you would expect for a damping ratio of 0.6 (17% overshoot):





Note that the price you pay for a faster response is a larger input. U(k) went from a peak of

- 4.5 with K(z)=k to
- 26.7 with a lead compensator.
- 87.74 with a 2-stage lead compensator.

PID Compensation in the z-Domain

Objectives:

- Design a PID compensator in the z-domain to speed up a system

Definition:

- P: Proportional control: K(z) = k
- I: Integral control: $K(z) = \left(\frac{k}{z-1}\right)$
- D: Delay control: $K(z) = \frac{k}{z}$

Discussion:

The purpose of a lead compensator is to speed up a system. In the s-plane, this is done by pulling the root locus to the left. In the z-plane, this is done by pulling the root locus to the origin.

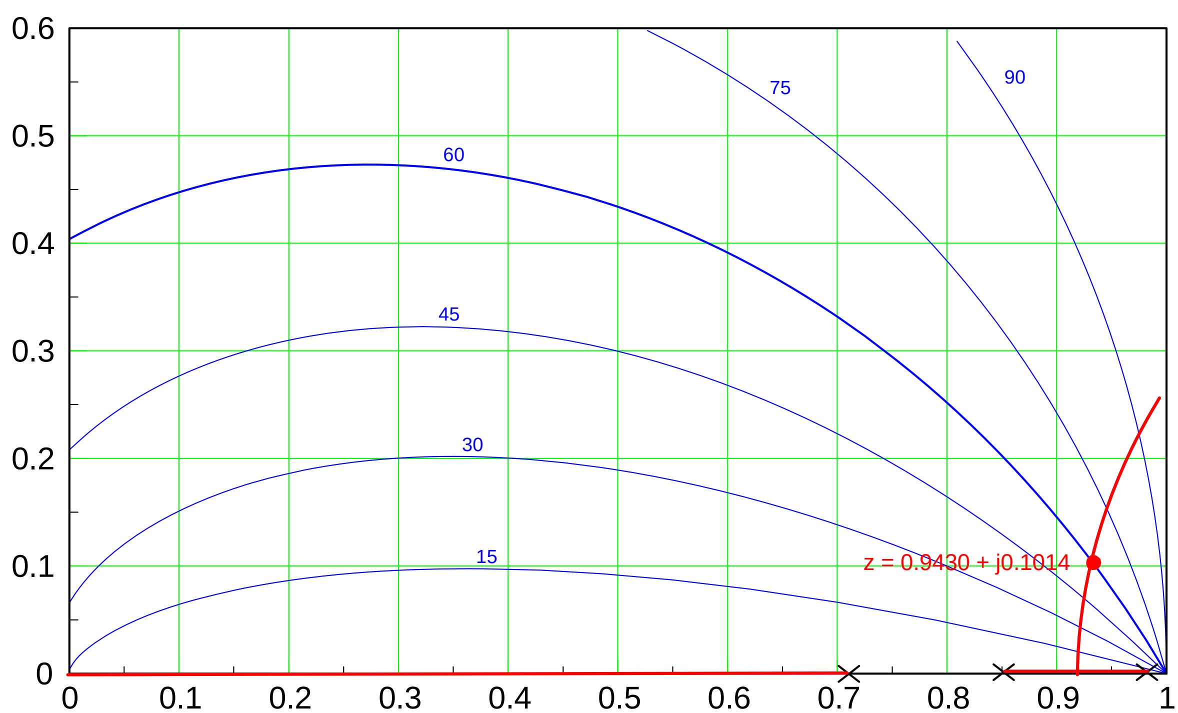
Example: Find a gain compensator for G(s) to give a damping ratio of 0.5.

$$G(s) = \left(\frac{1}{s^3+5s^2+6s+1}\right) = \left(\frac{1}{(s+0.1981)(s+1.555)(s+3.2469)}\right)$$

From before, the discrete-time equivalent with a sampling rate of 0.1 second is

$$G(z) = \left(\frac{0.0001147(z+0.2358)(z+3.3011)}{(z-0.9803)(z-0.8560)(z-0.7227)}\right)$$

The root locus for this system is show below:



For a damping ratio of 0.6, you want to place the dominant pole at -0.9390+j0.0944. At this point

GK = -1

k = 4.5093

The 2% settling time should be

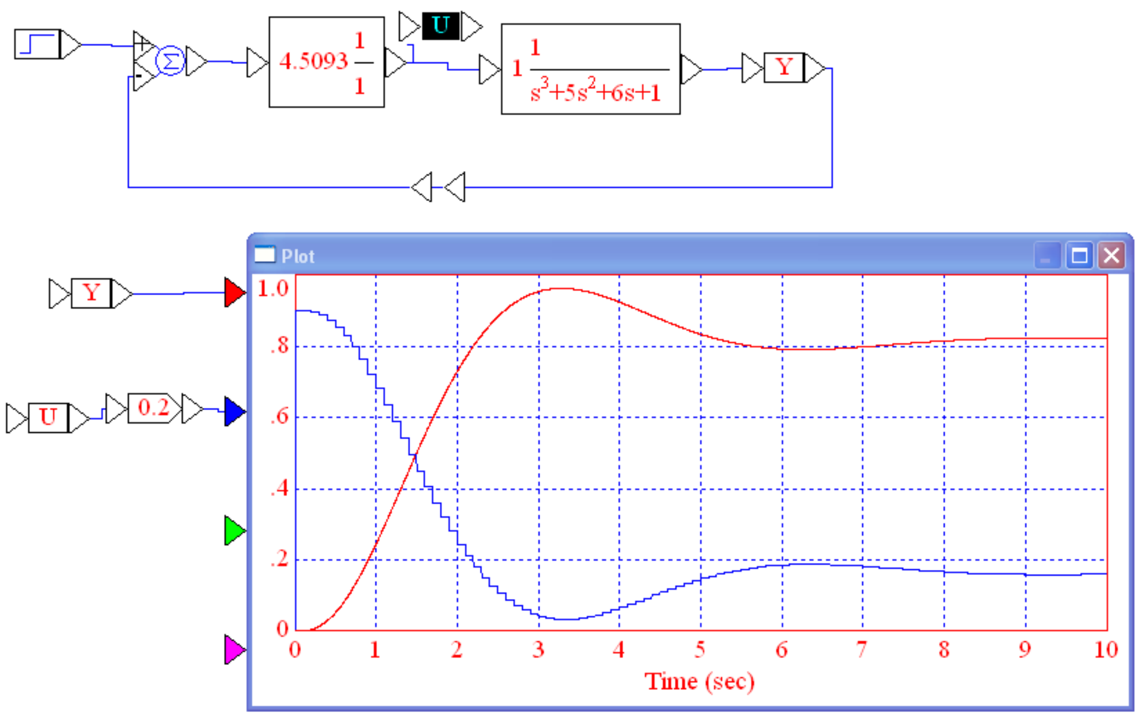
$$|pole| = |0.9390 + j0.1014| = 0.9395$$

$$(0.9395)^k = 0.02$$

$$k = 62.7 \text{ samples}$$

$$t = kT = 6.27 \text{ seconds}$$

The step response is shown below (from VisSim):



Note that the output doesn't go to 1.000. This is a type-0 system, so that's not surprising. The overshoot is a little too high due to ignoring the 1/2 sample delay from the zero-order hold.

Integral Feedback

To force the error to zero, add a pole at s=0 (z=1).

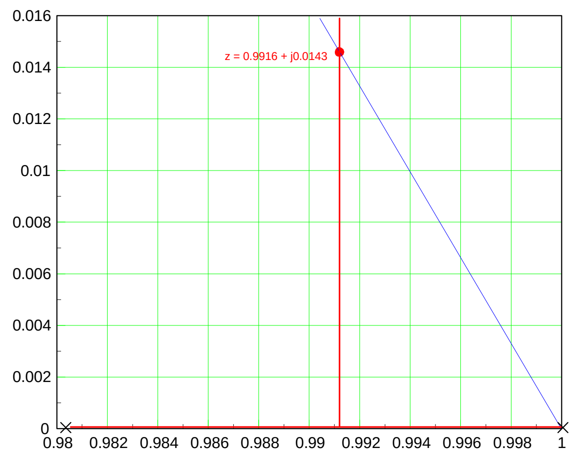
$$K(z) = k\left(\frac{1}{z-1}\right)$$

Sketching the new root locus, the intersect with the 60 degree damping line is at 0.9916 + j0.0831.

Picking 'k' to make the gain equal to -1 at this point:

k = 0.0143

This results in a settling time of 792 samples (79.2 seconds).



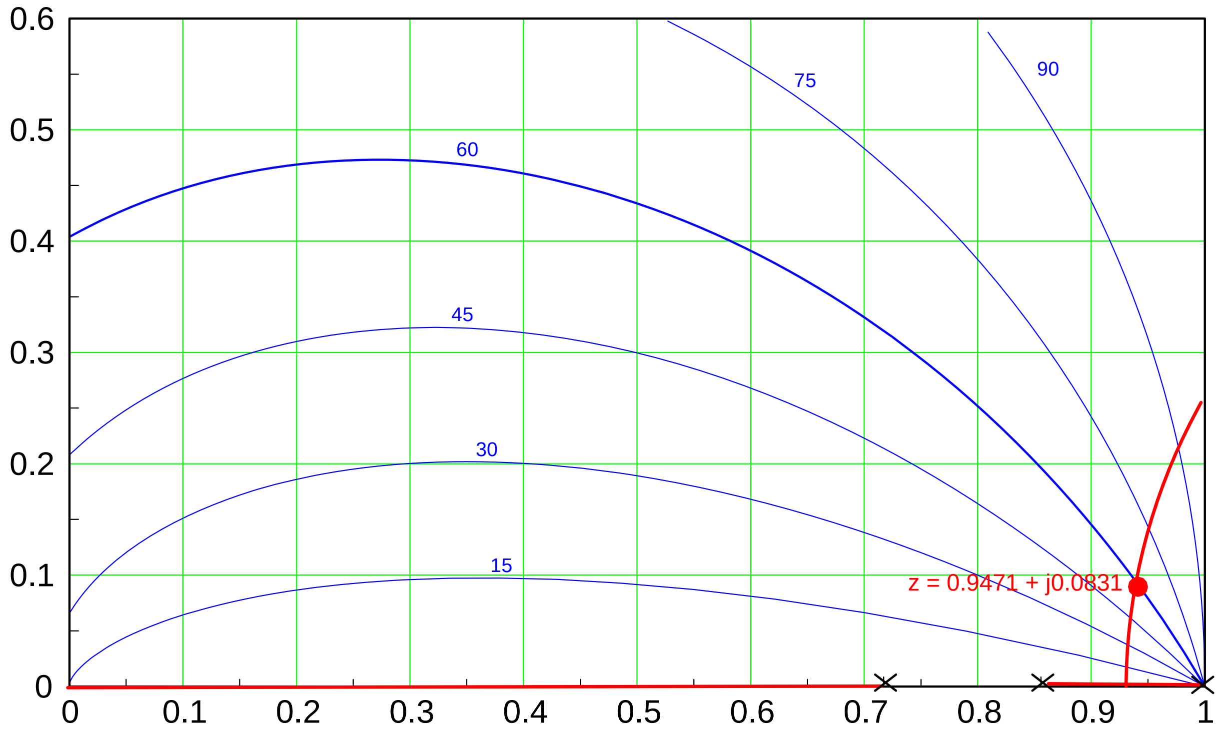
Root Locus with Integral Feedback:

Proportional - Integral (PI) Feedback

If you add a pole, you can add a zero. Pick the zero to cancel the pole at 0.9803:

$$K(z) = k\left(\frac{z-0.9803}{z-1}\right)$$

The root locus intersects the 60 degree damping line 0.9471 + j0.0831



This speeds up the system, resulting in the 2% settling time being:

$$|0.9471 + j0.0.0431| = 0.9507$$

$$(0.9507)^k = 0.02$$

$$k = 77.4$$

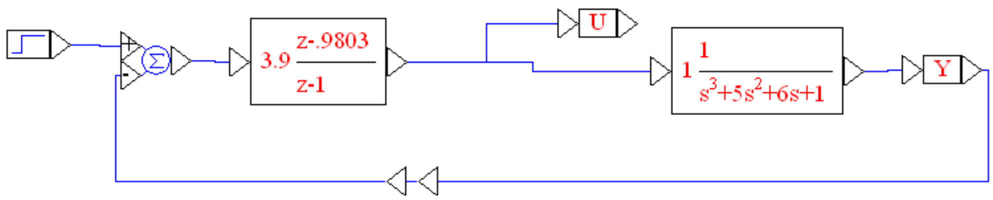
$$t = kT = 7.74 \text{ seconds}$$

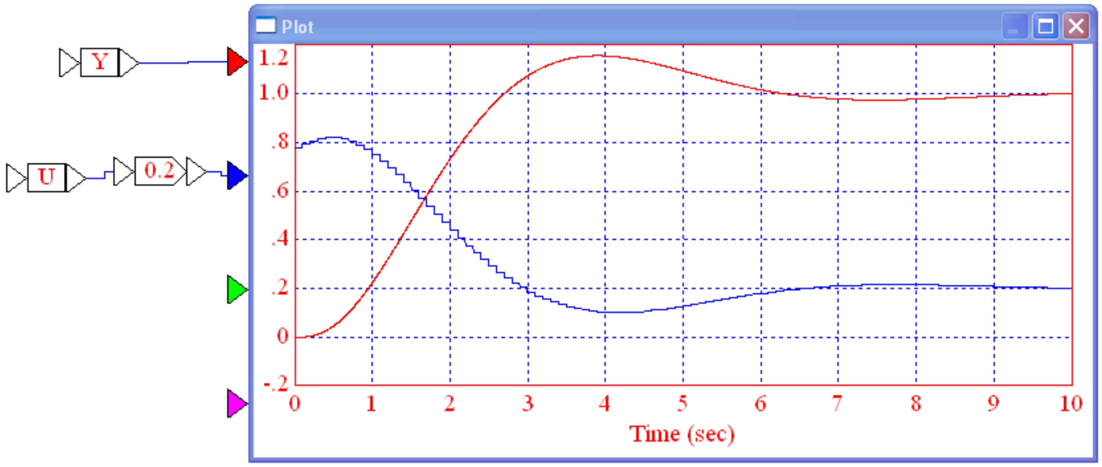
The resulting step reponse now settles out in 7.7 seconds. As expected, the lead compensated system is faster

K(z) is implemented in two parts:

$$K(z) = 3.9\left(\frac{z-9803}{z-1}\right) = 3.9 + \frac{0.0768}{z-1}$$

The sampling rate of 0.1 second is seen at the output of K(z):





Note that adding the zero sped up the system almost 10x. (!)

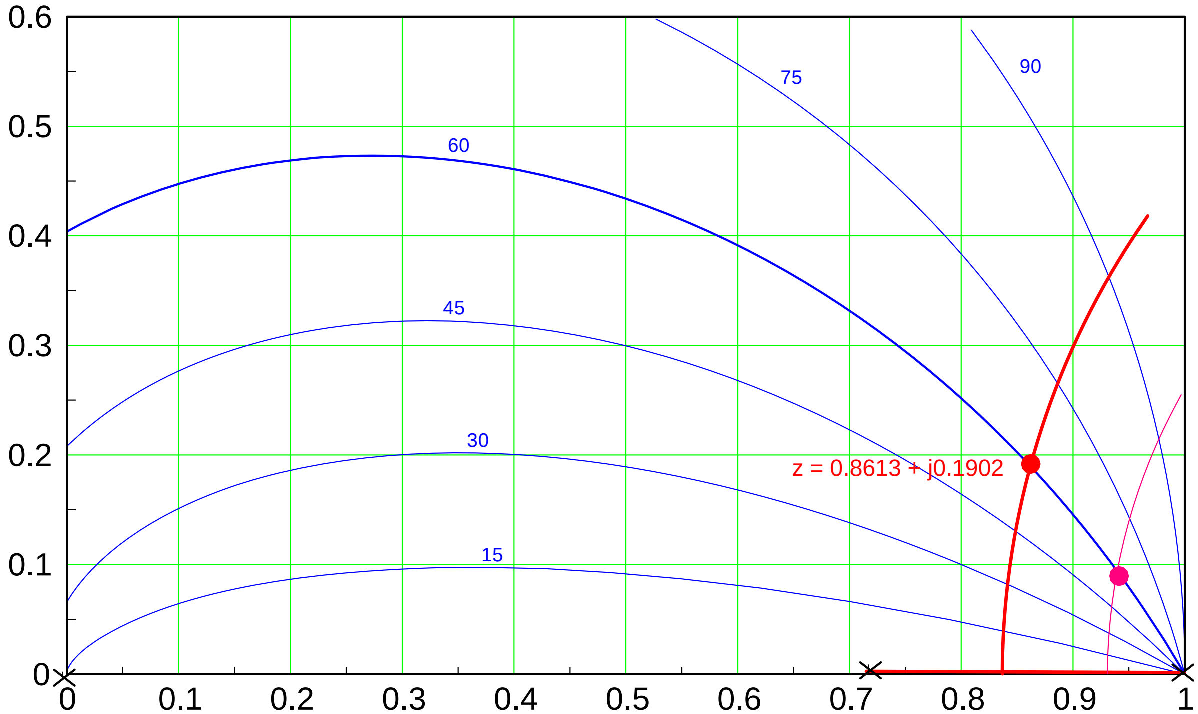
PID Control (Proportional - Integral - Delay)

If this isn't fast enough, cancel the next slowest pole (the one at 0.0.8560). Replace it with a fast pole. The fastest you can get in the z-plane is z=0, so place a pole there:

$$K(z) = k\left(\frac{(z-0.9803)(z-0.8560)}{(z-1)(z)}\right)$$

The point on the root locus which crosses the 60 degree damping line is

z = 0.0.8613+j0.1902



GK = -0.0014

k = 71.18

$$K(z) = 71.18\left(\frac{(s-0.9803)(z-0.8560)}{(z-1)(z)}\right)$$

$$K(z) = 71.18 + \frac{0.2037}{z-1} - \frac{59.73}{z}$$

These are the proportional, integral, and delay terms. Hence the name, PID.

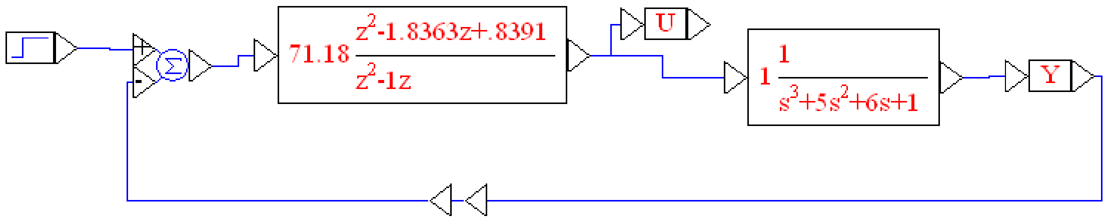
Now, the 2% settling time should be

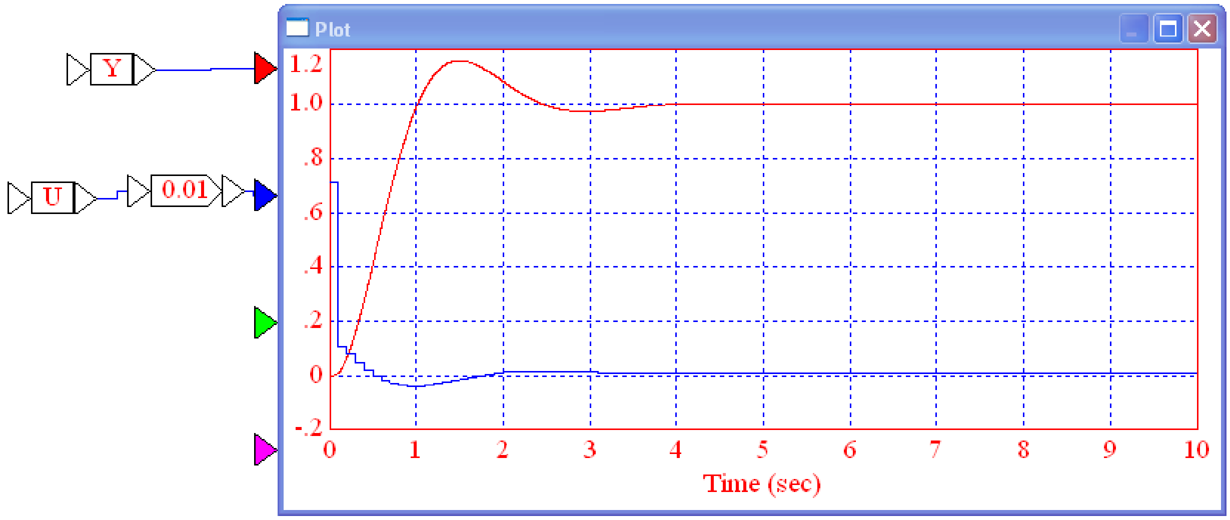
$$|pole|^k = 0.8821^k = 0.02$$

$$k = 31.2 \text{ samples}$$

t = 3.12 seconds

The step response is as you expect - 17% overshoot and a faster response than before:





The controller, K(z), consists of two delays and three gains:

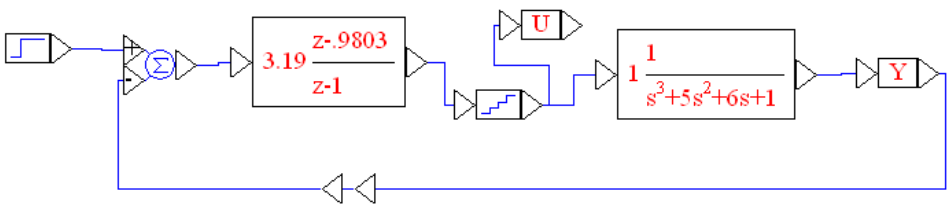
Quantization Noise:

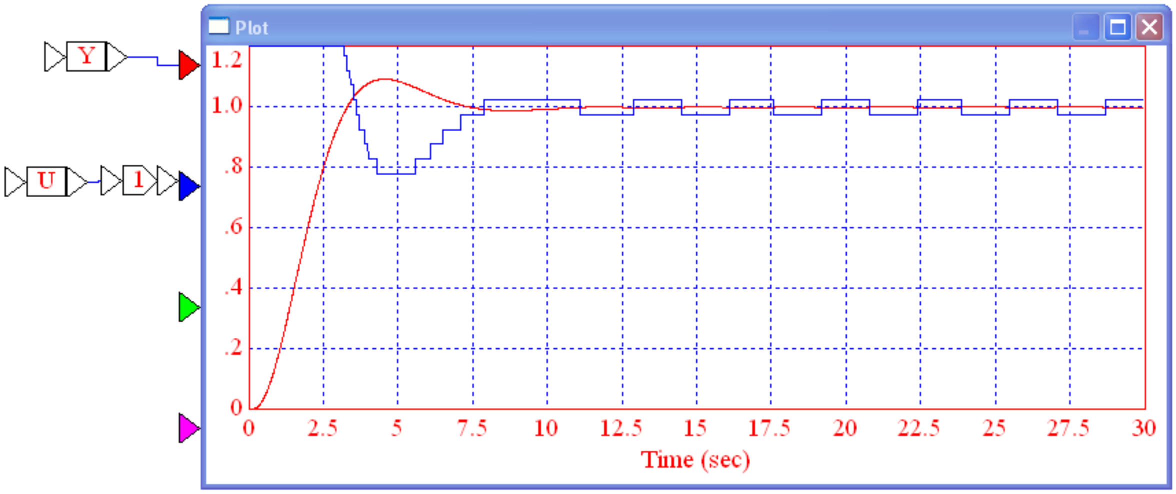
The previous simulations assumed an infinite-bit A/D and D/A. In practice, you have a limited number of bits, meaning the input and output are rounded to 2N values (where N is the number of bits in the A/D and D/A).

Assume a 10-bit A/D and D/A. Assume the voltages range from +24V to -24V. This results in the quantization level being

$$\frac{48V}{2^{10}} = 0.0469V$$

In the below figure, the stairstep block on the output of K(z) simulates the 10-bit D/A - rounding the input to the nearest 0.0469V. This results in the following step response:





It looks like the system is find, up until 9.5 seconds. Then, it starts to oscillate as seen by the input chattering.

What's happening is the ideal input (the input that holds the output equal to the set point) is inbetween bit levels. The controller tries to find that value but can't (since it's inbetween bit levels.)

- For a while, it rounds down, resulting in an output that's too small.
- The integrator starts winding up, until you get to the next bit.
- The output then jumps up one count, which causes the output to increase too much.
- This then repeats over and over.

The operator usually can't tell that the input is changing by one-bit. People just can't detect such small effects and noise usually causes more chatter than one bit. With computer interfaces, the customer might be able to zoom in on the input (the blue line above) and see that the input isn't constant - like an analog input would be. (But then, you can't zoom in on an analog dial.)

Giving the customer too much information can result in your product being returned - even though it's working fine.

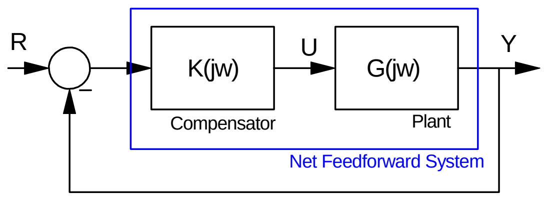
Bode Plots

Objectives:

- Sketch the frequency response (Bode plot) of a system given G(s)
- Determine G(s) given its frequency response.

Definitions:

Assume a feedback system of the following form. Unity feedback is used to force the output, Y, to track the reference signal, U. A compensator, K, is used to improve the response of the feedback system:



Open-Loop Gain:	k G(jw)
Closed-Loop Gain:	$\left(\frac{kG(jw)}{1+kG(jw)}\right)$
dB	20 log(Gain)
0dB Gain Frequency:	Frequency where |kG(jw)| = 0dB. Approximately equal to the bandwidth of the system. Approximately equal to the amplitude of the dominant pole (in rad/sec).
Gain Margin:	How far G(jw) is away from -1 when the phase is 180 degrees. Used to determine stability.
Phase Margin	How far G(jw) is away from -1 when the gain is one. Used as an approximation to resonance.
Resonance: (Mm)	The closest point from G(jw) to -1 Determines the resonance.
M-Circle	Contour on a Nichols, Nyquist, or Inverse Nyquist plot where the closed-loop gain is a constant.
Nichols Chart:	Plot of G(jw) in Cartesian coordinates. X = phase. Y = gain (dB). Graphical way to map open-loop gain to closed-loop gain Graphical way to determine the closest point to -1.
Nyquist Chart:	Plot of G(jw) in Cartesian coordinates. X = real, Y = imag. Graphical way to map open-loop gain to closed-loop gain Graphical way to determine the closest point to -1.
Inverse Nyquist Chart	Plot of 1/G(jw) in Cartesian coordinates. X = real, Y = imag. Graphical way to map open-loop gain to closed-loop gain Graphical way to determine the closest point to -1.

Bode Plot	Two plots. i) Plot of Gain (dB) vs. log(frequency). ii) Plot of phase (degrees) vs. log(frequency).

Background:

Given a dynamic system, it will have a transfer function, G(s). It will also have a frequency response, G(jw). In this light, any dynamic system acts as a filter, amplifying some frequencies, attenuating other frequencies.

Some systems behave well when feedback is added. Other system do not. Conceptually (as with root-locus), if you add a pre-filter to G(jw), you can warp the frequency response to be that of a system which *does* behave well with feedback. This section presents a way to design this pre-filter (K(s)) using frequency-domain techniques.

Finding the frequency response of from its transfer function

Assume you have a transfer function G(s). The frequency response of this system is obtained by substituting $s \rightarrow j\omega$.

Example1, assume $G(s) = \left(\frac{5000(s+10)}{(s+1)(s+60+j80)(s+60-j80)}\right)$. Find the gain at 3 rad/sec.

Solution: Substitute $s \rightarrow j3$

$$G(j3) = 1.65\angle -57^0$$

The output will be 1.65 times as large as the input, delayed by 57 degrees. (note: one cycle = 360 degrees. The output will be delayed $\frac{57}{360} = 16\%$ of a full cycle.)

Example 2: Find y(t) is $u(t) = 2\cos(3t)$.

Solution: Output = gain * input. In phasor notation:

$$y = (1.65\angle -57^0)(2\angle 0^0)$$

$$y = 3.30\angle -57^0$$

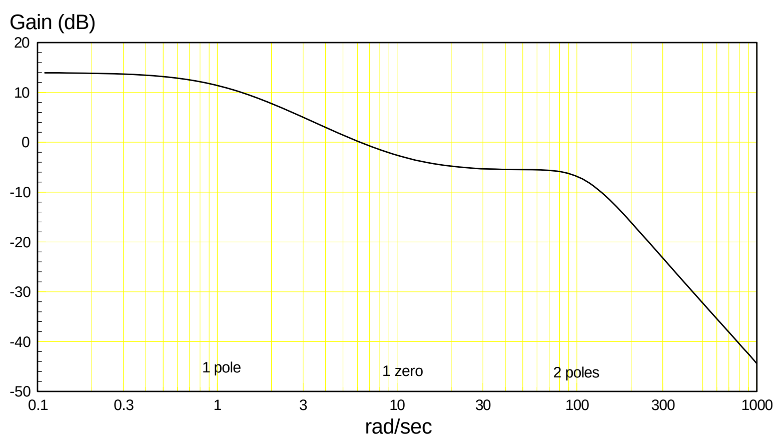
In time

$$y(t) = 3.30\cos(3t - 57^0).$$

Since G(jw) has a different gain and phase shift at all frequencies, it is convenient to display this information on a graph. A Bode plot is a standard type of graph where

- Frequency (w) is plotted on a log-scale on the X-axis
- Gain (dB) is plotted on the Y axis, and
- Phase (degrees) is plotted on the Y axis of a second plot.

For example, the frequency response of $G(s) = \left(\frac{5000(s+10)}{(s+1)(s+60+j80)(s+60-j80)}\right)$ is plotted in the following figure:



Straight-Line Approximations to a Bode Plot

The reason that this type of plot is standard is that the gain of s^n plots as a straight line:

$$dB((j\omega)^n) = 20 \cdot \log(|j\omega|^n) = 20n \cdot \log(\omega)$$

(When ω increases by 10x, the gain increases 20n times.) If $G(s) \approx s^n$ over some range, then the Bode Plot will be a straight line with a slope of 20n dB per decade. Back in the days before MATLAB, complex numbers were a pain to deal with. This straight-line property made Bode Plots a convenient way to approximate the gain of a transfer function.

The root of a Straight-Line Bode Approximation is to approximate

$$(s+a) \approx \begin{cases} s & |s| < |a| \\ a & |s| > |a| \end{cases}$$

Then, a transfer function can be approximated as $G(s) \approx s^n$ in different regions of the Bode plot. For example,

$$G(s) = \left(\frac{1}{s+1}\right) \approx \begin{cases} 1 & |s| < 1 \\ \frac{1}{s} & |s| > 1 \end{cases}$$

This produces two regions

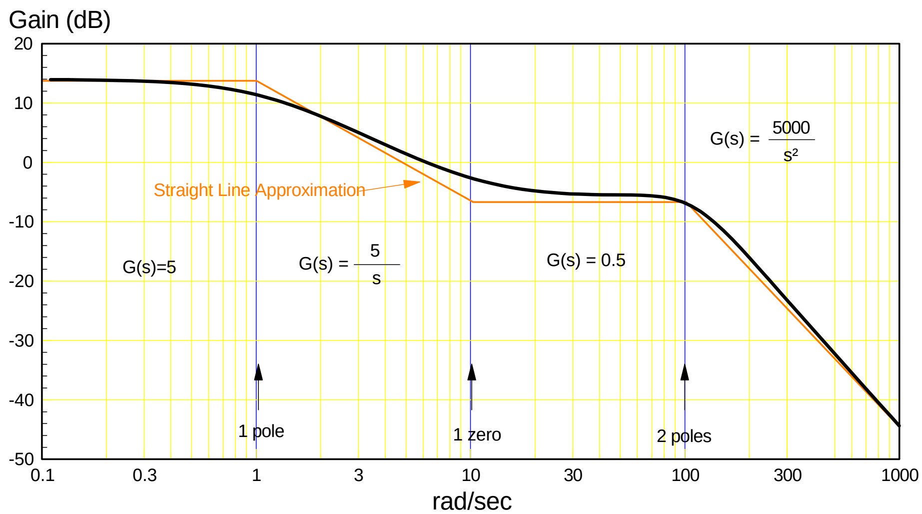
- For frequencies below 1 rad/sec, $G(s) \approx 1$ (gain is constant)
- For frequencies above 1 rad/sec, $G(s) \approx s^{-1}$ (gain drops 20dB / decade)

At 1, these two straight lines meet, making a corner. Due to the approximation used, this corner will always be at the amplitude of the pole.

Using this approximation for $G(s) = \left(\frac{5000(s+10)}{(s+1)(s+60+j80)(s+60-j80)}\right)$

$$G(s) = \left(\frac{5(s+10)}{(s+1)(s+100)}\right) \approx \begin{cases} \left(\frac{5000(10)}{(1)(100)(100)}\right) = 5 & s < 1 \\ \left(\frac{5000(10)}{s(100)(100)}\right) = \frac{5}{s} & 1 < s < 10 \\ \left(\frac{5000(s)}{(s)(100)(100)}\right) = 0.5 & 10 < s < 100 \\ \left(\frac{5000(s)}{(s)(s)(s)}\right) = \frac{5000}{s^2} & s > 100 \end{cases}$$

These plot as four straight lines - each one being valid over a small range of frequencies. Note that, although not exact, they do approximate the gain vs. frequency fairly well.



Rules to Sketch a Straight-Line Approximation to a Bode Plot

i) Mark on the Bode plot the frequencies equal to the amplitude of the poles and zeros.

- G(0) = 5 = 14dB. Start at a gain of 14dB at the left side of the Bode plot.

ii) Move right.

- The gain is constant over this range. Draw a straight line to the right with a slope of zero.

iii) If you hit a zero, the gain increases 20dB per decade per zero.

iv) If you hit a pole, the gain drops by 20dB per decade per pole.

- At 1 rad/sec, you hit a pole. The gain drops 20dB/decade.
- At 10 rad/sec, you hit a zero, increase the slope by 20dB /decade.
- At 100 rad/sec you hit two poles. The gain drops 40dB / decade (2 x 20dB / decade).

Gain at the Corner Frequency:

Note that the actual gain vs frequency is a smooth curve going from asymptote to asymptote. You can improve the accuracy of the straight-line approximation to the frequency response by drawing such a smooth curve. At the corners, the gain should be:

- Down 3dB if you hit a single pole:
- Down by $\left(\frac{1}{2\zeta}\right)$ if you are at a complex pole,
- Up a like amount for zeros.

For example, at the corners in the above graph,

- G(j1) is down 3dB from the straight-line approximation (a pole is at 1 rad/sec)
- G(j10) is up 3dB from the straight-line approximation (a zero is at 10 rad/sec),
- G(j100) is down 1.5dB from the corner. (The damping ratio of the complex poles is 0.6. $\left(\frac{1}{2\cdot 0.6}\right) = 0.83 = -1.5dB.$)

Finding the transfer function of a system from its frequency response

The straight-line Bode approximation also allows you to find the transfer function of a system given its frequency response.

i) Draw in the asymptotes to approximate G(jw). Make sure that all lines have a slope of 20n dB/decade.

(note: if you draw a line with a slop of 10dB/decade, that implies you have 1/2 of a zero. Recall that G(s) is the transfer function for a system which is described by a differential equation. 's' here means a derivative. $s^{1/2}$ means that you are taking 1/2 of a derivative - which is nonsense. A given transfer function will have

- An integer number of derivatives, implying it will have
- An integer number of poles or zeros, implying that the slope on a Bode plot will be
- An integer number of 20dB/decade.

ii) Mark the corner frequencies.

- If the slope increases by 20n dB/decade, there are n zeros with this amplitude.
- If the slope decreases by 20n dB/decade, there are n poles with this amplitude.

iii) If you have two poles or zeros at a given frequency, the damping ratio can be found by noting the gain at the corner:

- Gain = $\left(\frac{1}{2\zeta}\right)$

iv) Once all the poles and zeros are added, add a gain. Pick this gain so that gain of G(jw) matches the graph at some frequency.

Example: For the previous Bode Plot, determine G(s).

Solution: The straight-lines are drawn in red. The corners tell you that

- There is a pole at 1 (the slope drops 20dB / decade)
- A zero at 10, (the slope increases 20dB / decade)
- and two poles at 100 (the slope drops 40dB / decade)

The damping ratio for the complex poles is found from the gain at the corner:

At 100 rad/sec, G(jw) is down 1.5dB

$$-1.5dB = 0.84 = \frac{1}{2\zeta}$$

$\zeta \approx 0.59$ Gain

Hence, $G(s) = k\left(\frac{(s+10)}{(s+1)(s+100\angle\pm 53^0)}\right)$,

Pick k to set the gain at some frequency. Selecting 0.1 rad/sec (arbitrarily),

$$14dB = \left(\frac{k(s+10)}{(s+1)(s+100\angle 53^0)(s+100\angle -53^0)}\right)_{s=j0.1}$$

$$k \approx 5,011$$

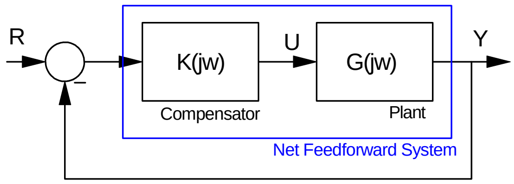
so

$$G(s) \approx \left(\frac{5011(s+10)}{(s+1)(s+100\angle 53^0)(s+100\angle -53^0)}\right) = \left(\frac{5011(s+10)}{(s+1)(s+60.2+j79.9)(s+60.2-j79.9)}\right)$$

Nichols Charts

Stability & Feedback

Consider the following feedback system.



If this models the speaker system at a rock concert, a problem can arise if the gain becomes too large. If this happens, the system will go unstable and a loud screech will be heard. Essentially, the system has gone unstable - meaning the gain has gone to infinity at some frequency (the frequency of the screech).

For this system to be useful (at least not annoying), it needs to be stable. The problem is to find the condition where the system is stable.

Assume that at some frequency the signal adds in phase. In this case, the loop gain will be 'a' where 'a' is a positive constant. Since the signal feeds back upon itself, the overall gain will be

$$\left(\frac{GK}{1+GK}\right) = a + a^2 + a^3 + a^4 + \ldots$$

If 'a' is less than one, this converges. If 'a' is greater or equal to one, this adds up to infinity. Essentially this is what happens at a rock concert.

- At some frequency, the signal adds to itself.
- When you crank up the gain, the gain at this frequency increases as well.
- When the loop gain at this frequency equals one, the signal builds upon itself, creating a loud screeching noise.

To fix this problem, you back off on the gain until the screech stops. This is analogous to the stability condition using frequency domain techniques:

For a system to be stable, the gain must be less than one when the net phase around the loop is zero degrees.

Since negative feedback is used, the feedback adds 180 degrees phase shift. For the net phase to be zero degrees, the phase of GK must be 180 degrees as well. Hence,

For a system to be stable, the gain GK must be less than one when the phase of GK is 180 degrees.

Since stability is rather important, a term, Gain Margin, is used to tell you how stable the system is

Gain Margin: How much you can increase the gain before the system just goes unstable.

This is similar to a safety margin: a 6dB gain margin means that the gain could be off by 6dB without causing the system to go unstable.

In addition to being stable, you would like the system to behave well as well. From a root-locus standpoint, 'behave well' means that the dominant poles are a certain distance from the jw axis, as defined by the damping ratio. From a frequency-domain standpoint, 'behave well' means that the resonance isn't just finite, but it is limited to some number, such as 6dB as well. From the second-order approximations specifying the resonance also specifies the damping ratio

$$M_m = \begin{cases} 1 & \zeta > 0.7 \\ \frac{1}{2\zeta\sqrt{1-\zeta^2}} & 0 < \zeta < 0.7 \end{cases}$$

- so it's just another way of saying the same thing.

Since the closed-loop system goes unstable when $G(j\omega) = -1 = 1\angle 180^0$, you can think of this constraint on Mm as being a constraint on how close G(jw) can get to -1. The problem is that 'close' is defined by the closed-loop gain: $\left(\frac{G}{1+G}\right)$, which is a nonlinear function of G. As a result, some graphical tools are defined for defining 'close'.

In each of these tools, two sets of coordinates are used

- One set defines G(jw)
- The other defines $\frac{G}{1+G}$

By plotting G(jw) one set of coordinates, you can read off $\left(\frac{G}{1+G}\right)$ by looking at the other. The various tools (Nichols Charts, Nyquist Diagrams, Inverse Nyquist Diagrams) are simply different ways to plot G(jw). Their goal in all cases is to show you what 'close to -1' means.

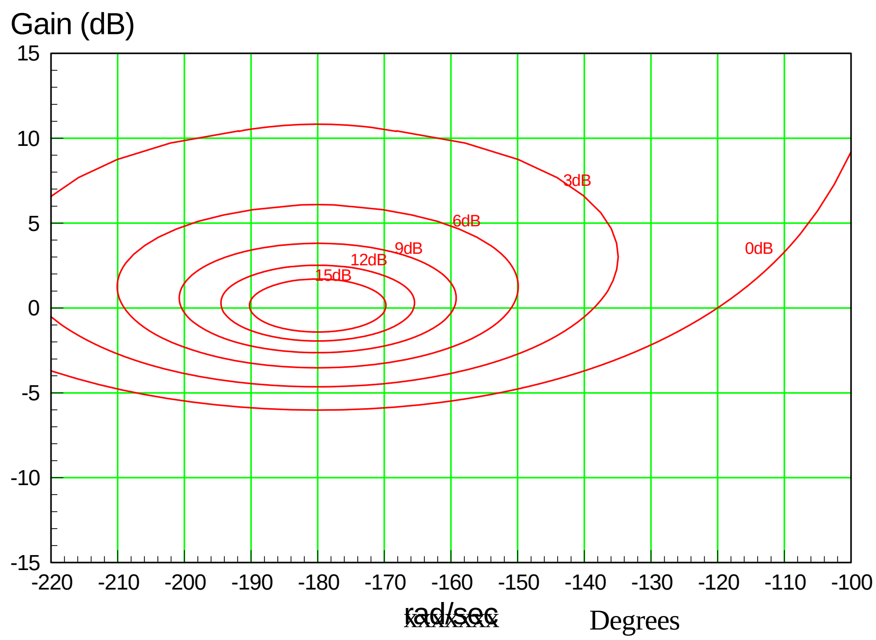
Gain Compensation Using Nichols Charts:

A Nichols Chart is a plot where G(jw) is graphed with

- The X-axis plotting the phase of G(jw), and
- The Y-axis plotting the gain of G(jw) in dB.

In addition, M-circles are drawn on top of this plot. These M-circles define all points where the closed-loop gain $\left(\frac{G}{1+G}\right)$ will be constant. This tells you that all points on the same M-circle are the same distance from -1.

Example: The following ia a Nichols Chart. The ovals (red) M-circles tell you all points which are equal distance to -1.



Nichols Chart: M-circles plot points where $\left|\frac{G}{1+G}\right|$ is constant

To use a Nichols chart

- Plot G(jw) on the Nichols chart using the outer coordinates.
- Adjust the gain of G(jw) by sliding it up and down. Adjust it so that the system is stable (The gain is less than 0dB at 180 degrees) and it is a specified distance from -1.

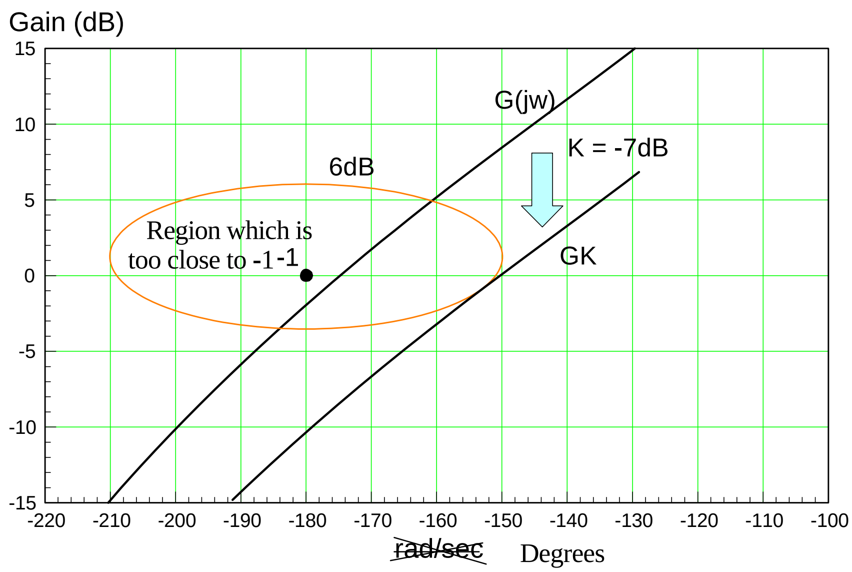
Example: Find a gain, k, so that $G(s) = \left(\frac{2000}{s(s+5)(s+20)}\right)$

- Has a resonance of 6dB or less, and
- K is as large as possible.,

Solution: Plot G(jw) on a Nichols chart along with the 6dB M-circle. This is shown below.

- Adjusting K is equivalent to sliding G(jw) up and down. If K=10, G(jw) is slid up 20dB. If K=0.1, G(jw) is slid down 20dB.
- To be sable, you need to keep the gain less than 0dB when the phase is 180 degrees.
- To keep 6dB away from -1, G(jw) should not go inside the 6dB M-circle.
- The 'best' design is to adjust K (slide G(jw) up and down) until it is tangent to the 6dB M-circle. This results in K being -7dB.

K = -7dB



SciLab Example: Using the controls system toolbox. First, guess that the resonance will be between 0.1 and 10 rad/sec:

```
-->w = logspace(-1,1,100);
```

Input G(s) and compute G(jw):

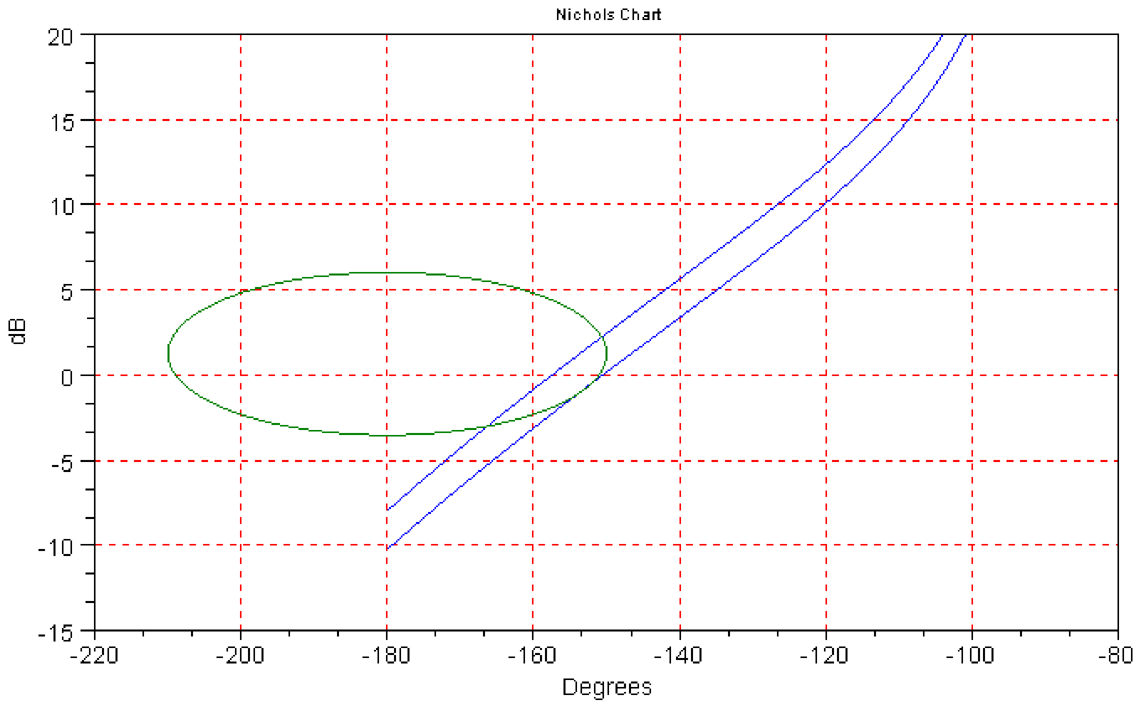
```
-->G = zp2ss([],[0,-5,-20],1000);
-->Gw = bode2(G,w);
```

Transfer G(jw) to a Nichols chart.

```
-->nichols(Gw,2);
```

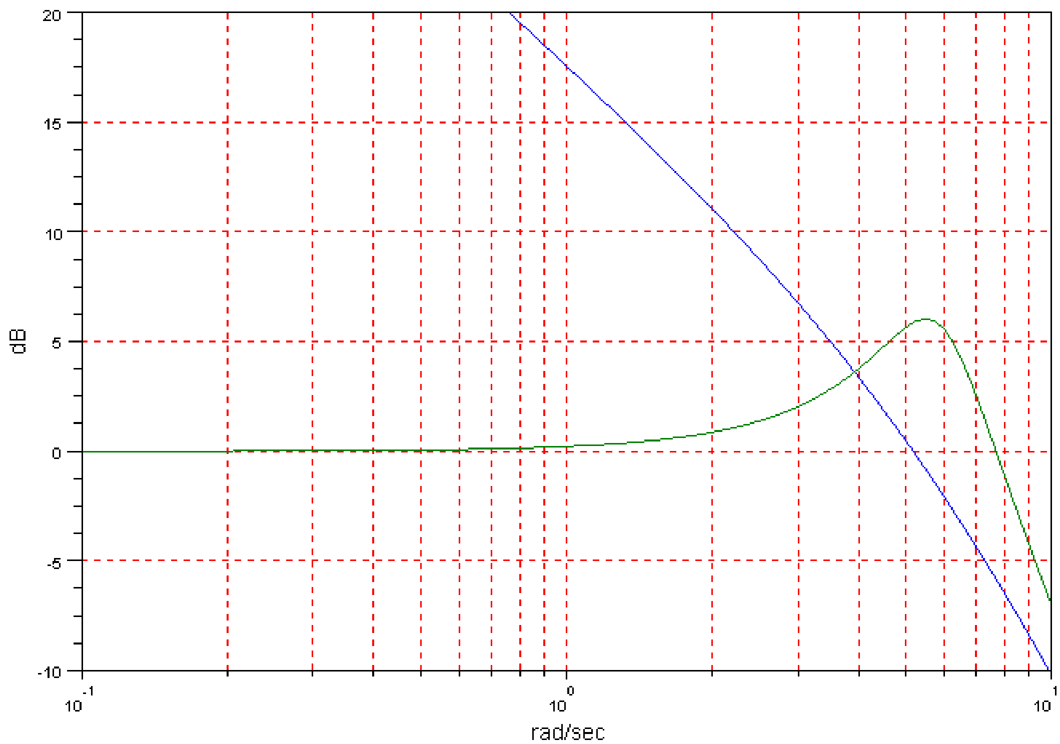
Adjust the gain until you're tangent to the M-circle.

```
-->nichols(Gw*0.77,2);
-->xgrid(5)
```



As a check, compute and plot the gain of the closed-loop system:

```
-->k = 0.77;

-->plot(w,dB(Gw*k),w,dB(Gw*k ./ (1+Gw*k)));

-->xgrid(5)

-->xlabel('rad/sec');

-->ylabel('dB')
```



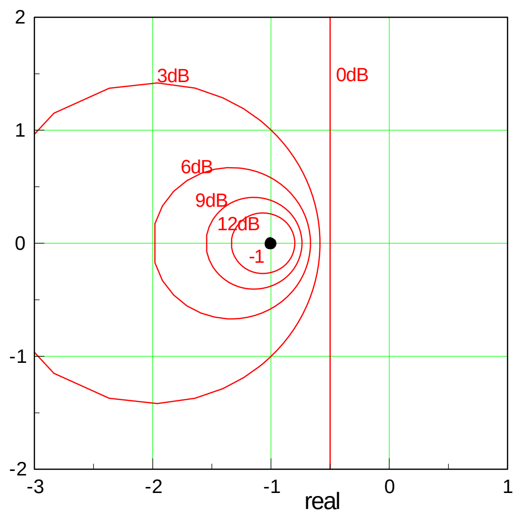
Note the gain of the closed-loop system (green line) has a resonance of 6dB. This is what you expect since k was selected so that you're tangent to the 6dB M-circle.

Gain Compensation Using a Nyquist Plot

On a Nyquist plot

- The X-axis plots real(G(jw))
- The Y axis plots imag(G(jw)).

M-circles define a constant distance to -1. An example is shown below.



Nyquist diagram. M-circles mark points where $\left|\frac{G}{1+G}\right|$ = constant.

Two ways to use a Nichols chart to find a feedback gain follow:

Method #1: (useful if using MATLAB)

i) Plot G(jw) as imag(G(jw)) vs. real(G(jw)).

ii) Scale G(jw) up or down until

- The system is stable ($|G(j\omega)| < 1$ when crossing 180^0.)
- $G(j\omega)$ is tangent to the desired M-circle.

iii) K is equal to the scaling used to make G tangent to the M-circle. If G(jw) is 2x larger, K=2.

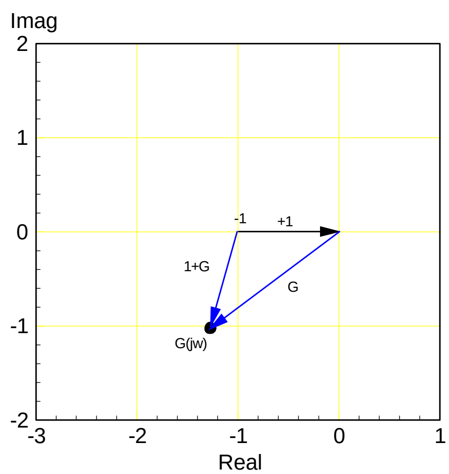
Method #2: (useful on tests)

The above procedure is difficult to use since you need to redraw G(jw) every time you guess K. A trick you can use to save time follows:

First, note that if K=2, G(jw) becomes twice as large. Equivalently, relative to G(jw), the axis looks 1/2 as large.

Second, note that you only care about the ratio of G to (1+G) to find the resonance: $\left(\frac{G}{1+G}\right)$. G is the vector from the origin to G(jw). (1+G) is the vector +1 added to G, or equivalently, the vector from -1 to G.

Hence, $\left(\frac{G}{1+G}\right)$ can be thought of as the ratio of the vector from G(jw) to the origin divided by the vector from G(jw) to -1. The resonance will be when this ratio is its largest value.



With this observation, you can design a gain compensator (i.e. find K) by scaling the axis instead.

i) Draw G(jw) as real(G) vs. imag(G).

ii) Find a point on the minus real axis, X, where the maximum of $\left(\frac{\text{The distance from G to the origin}}{\text{The distance from G to X}}\right)$ is your desired resonance.

iii) Scale the graph until this point, X, is -1: $K = \frac{1}{X}$.

For example, find K so that $G(s) = \left(\frac{2000}{s(s+5)(s+20)}\right)$ has

- A resonance of 6dB or less, and
- K is as large as possible.

Solution:

i) Plot G(jw) as real(G) vs imag(G). This is shown below.

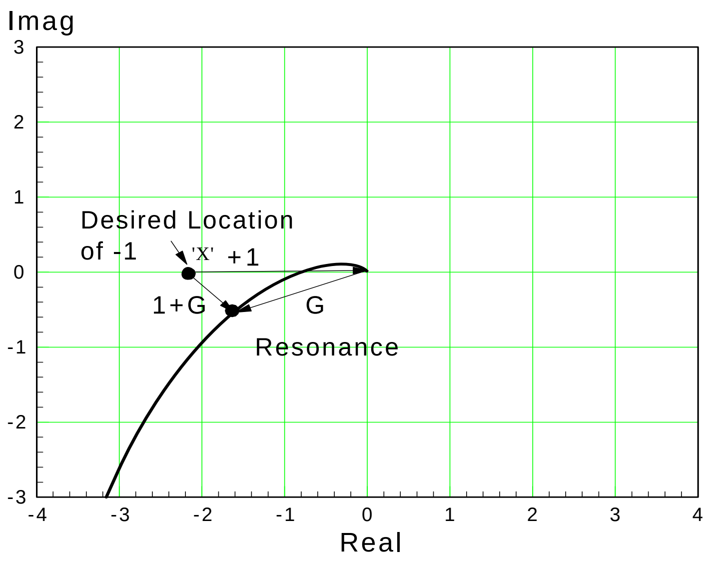
ii) Find a point on the minus real axis so that

- G(jw) passes 180 degrees to the right of this point (the gain will be less than 1 when the phase is 180 degrees), and
- $\left(\frac{\text{The distance from G to the origin}}{\text{The distance from G to X}}\right)$ is equal to 6dB.

This point is approximately -2.2.

iii) Scale the axis (i.e. scale G) so that this point is -1. If the axis is larger, G is smaller, and

$$K = \frac{1}{2.2} = 0.45 - 7dB$$

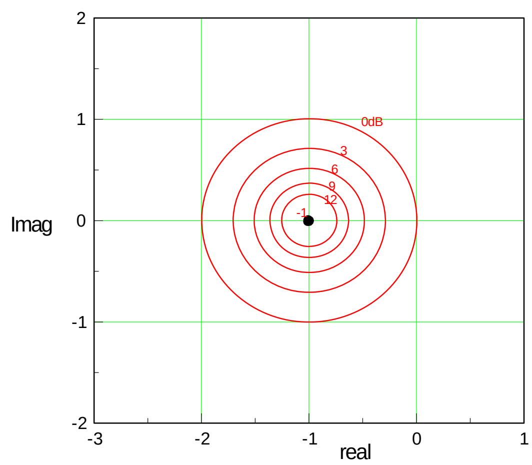


Gain Compensation using an Inverse Nyquist Plot

Inverse Nyquist Plots use

- X axis = real(1/G(jw))
- Y axis = imag(1/G(jw)).

M-circles again define the distance to -1 as shown below. Note that with an inverse-Nyquist plot, the M-circles are actually circles centered about -1. This makes inverse Nyquist plots much easier to use than Nyquist plots (although still not as easy to use as Bode plots coming up next). This feature resulted in Inverse Nyquist Plots being classified during WWII by the U.S. Army. Similarly, not many people are familiar with this technique.



Inverse Nyquist Diagram. M-Circles mark points where $\left(\frac{G}{1+G}\right)^{-1}$ = constant,

Like a Nyquist diagram, two methods can be used to find K using an Inverse Nyquist plot:

Method #1:

Method #1: (useful if using MATLAB)

i) Plot $\frac{1}{G(j\omega)}$ as $imag\left(\frac{1}{G(j\omega)}\right)$ vs $real\left(\frac{1}{G(j\omega)}\right)$.

ii) Scale $\frac{1}{GK}$ up or down until

- The system is stable: $\left|\frac{1}{GK}\right| < 1$ when crossing 180^0, or equivalently, $|GK| > 1$.
- $\frac{1}{GK}$ is tangent to the desired M-circle.

iii) K is equal to the scaling used to make $\frac{1}{GK}$ tangent to the M-circle. If $\frac{1}{G}$ is 2x larger, $\frac{1}{K}$=2.

Method #2: (useful on tests)

The above procedure is difficult to use since you need to redraw $\frac{1}{GK}$ every time you guess K. A trick you can use to save time follows:

First, note that if K=2, $\frac{1}{GK}$ becomes half as large. Equivalently, relative to $\frac{1}{G}$, the axis looks 2x as large.

Second, note that you only care about the closed-loop gain $\left(\frac{G}{1+G}\right)$, or in this case, the inverse of this:

$$\left(\frac{G}{1+G}\right)^{-1} = \left(\frac{1+G}{G}\right) = \left(1 + \frac{1}{G}\right)$$

This is simply the vector from $\frac{1}{G}$ to -1.

Hence, the inverse of the closed-loop gain is simply the distance from $\frac{1}{G}$ to -1. The resonance will be when this distance is the smallest (and its inverse is its largest).

Procedure:

i) Plot $G(j\omega)$ as $real\left(\frac{1}{G}\right)$ vs $imag\left(\frac{1}{G}\right)$.

ii) Find a point on the - real axis, X, where

- The system is stable ($\frac{1}{G}$ passes to the left of X, meaning the gain will be less than 1 when the phase is 180 degrees).
- The minimum value of $\left(\frac{\text{The distance from 1/G to -1}}{\text{The distance from X to the origin}}\right)$ is $\frac{1}{M_m}$

iii) K will then be X.

For example, find K so that $G(s) = \left(\frac{2000}{s(s+5)(s+20)}\right)$ has

- A resonance of 6dB or less, and
- K is as large as possible.

Solution:

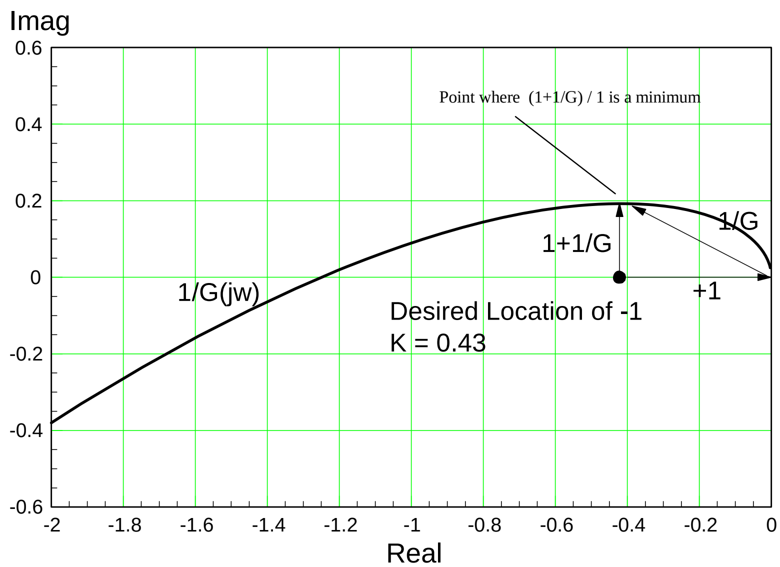
i) Plot 1/G(jw) as real(1/G) vs imag(1/G). This is shown below.

ii) Find a point on the minus real axis so that

- 1/G(jw) passes 180 degrees to the left of this point (the gain will be less than 1 when the phase is 180 degrees), and
- $\left(\frac{\text{The distance from 1/G to -1}}{\text{The distance from X to the origin}}\right)$ is equal to -6dB.

This point is approximately -0.43.

iii) Scale the axis (i.e. scale G) so that this point is -1. K = 0.43 (-7dB)



Gain Compensator Design using Bode Plots

Objectives:

To be able to

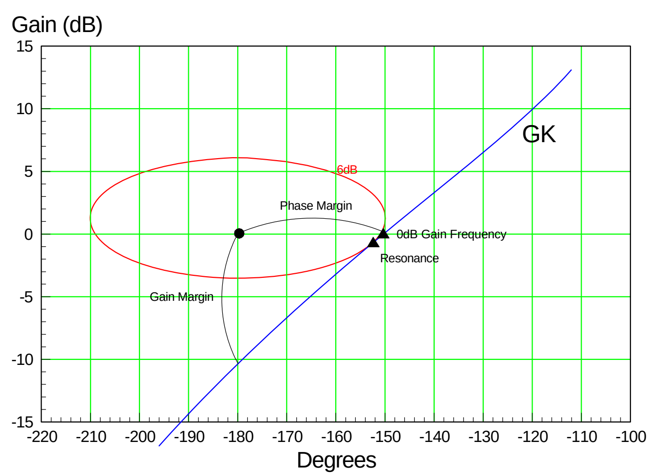
- Predict how a feedback system will behave as you increase the feedback gain using a Bode Plots and Nichols Charts
- Determine the 'best' feedback gain to use via a Bode plot or Nichols Chart
- Design a compensator to improve the response of a system using Bode Plot techniques.

Gain Compensation Using Bode Plots

Nichols charts are useful since it shows directly what you are trying to do when designing a compensator: you are trying to keep away from -1 to limit the resonance. The problem with Nichols charts is

- They are difficult to draw
- The resonance is difficult to find: the frequency where G(jw) is tangent to the M-circle changes as you change the gain.

Note on the above Nichols chart, however, that the point where G(jw) is closest to -1 (and tangent to the M-circle) is very close to the point where the gain of GK = 0dB.



If you assume that these two points are identical, it makes finding K much easier.

- i) Instead of drawing M-circles, find the point where $\left(\frac{G}{1+G}\right) = \left(\frac{1\angle\phi}{1+1\angle\phi}\right) = M_m$
- ii) Find the frequency where $\angle G = \phi$
- iii) Adjust K so that at this frequency, $|GK| = 1$

In this case, you are defining how far G(jw) is from -1 by how far it is away when the gain is 0dB. This is a much easier way to design a gain compensator, and hence has a special name: phase margin

Phase Margin: Distance of G(jw) to -1 when the gain of G(jw) = 1.

Relationship between Mm and Phase Margin:

The phase margin which approximately corresponds to a certain resonance is from

$$G(j\omega) = 0dB\angle\phi = 1\angle\phi$$

$$\left|\frac{G}{1+G}\right| = \left|\frac{1\angle\phi}{1+1\angle\phi}\right| = M_m$$

$$|1+(\cos\phi + j\sin\phi)| = \frac{1}{M_m}$$

$$(1+\cos\phi)^2 + (\sin\phi)^2 = \frac{1}{M_m^2}$$

$$1 + 2\cos\phi + \cos^2\phi + \sin^2\phi = \frac{1}{M_m^2}$$

$$\boxed{2 + 2\cos\phi = \frac{1}{M_m^2}}$$

Phase Margin = $180^0 - |\phi|$

Example: Find K so that $G(s) = \left(\frac{2000}{s(s+5)(s+20)}\right)$ has

- Mm < 6dB, and
- Minimal error for a step and ramp input.

Solution:

i) Find the phase margin corresponding to Mm = 6dB:

$$\cos\phi = \left(\frac{\frac{1}{M_m^2} - 2}{2}\right)$$

$$\phi = -151.04^0$$

Phase Margin = 28.96^0

ii) Find the frequency where G(jw) has a phase shift of -151.04^0

$$G(j5.2362) = 2.5474\angle -151.04^0$$

iii) Adjust k so that the gain at this frequency is one.

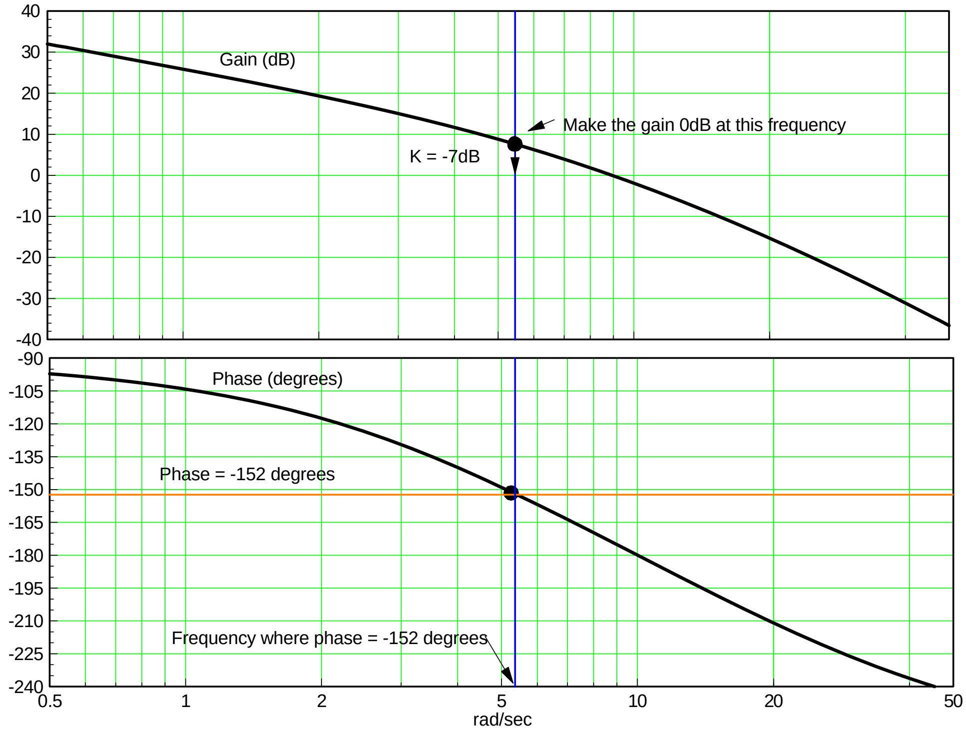
$$k = \frac{1}{2.5474}$$

$$\boxed{k = 0.3926 = -8.1dB}$$

With k=0.3926, the system will behave as

- Mm ≈ 6dB (from the phase margin being 28.96^0)
- $\zeta = 0.2588$ (from Mm = 6dB)
- $\omega_n = 5.2420$ (the corner frequency is approximately equal to the amplitude of the poles)
- Dominant pole = $5.2420\angle(180 - 75^0) = -1.3567 + j5.0633$
- $T_s \approx \frac{4}{1.3567} = 2.948$ seconds.

Example 2: Repeat where G(s) is not given. Instead, assume that only the Bode Plot of G(s) is given:



i) Find the phase margin corresponding to Mm = 6dB:

$\phi = -151.04^0$

Phase Margin = 28.96^0

ii) Find the frequency where G(jw) has a phase shift of -152^0. This is shown on the above Bode plot.

w = 5.2 rad/sec

iii) Adjust k so that the gain at this frequency is one. This shift is shown on the above Bode plot as well.

G(j5.2) = 7dB

GK(j5.2) = 0dB

so

K = - 7dB

Note that the same data can be found from the graph as from the transfer function - although it is difficult to get four decimal places of accuracy from a graph.

Lead Compensator Design Using Bode Plots

Objectives:

To be able to

- Design a lead compensator using Bode plot techniques,
- Design a lag compensator using Bode plot techniques

Definitions:

- Lead Compensator: $K(s) = k\left(\frac{s+a}{s+b}\right)$ where b>a. The phase of K(s) > 0.
- Lag Compensator: $K(s) = k\left(\frac{s+a}{s+b}\right)$ where a >b. The phase of K(s) < 0.
- Gain Compensation: $K(s) = k$.

Lead Compensation:

Background

If you are using gain compensation to find K, the procedure is to

- Crank up the gain, K, until
- The desired phase margin is obtained.

The resulting bandwidth will be whatever the system allows. For example, if $G(s) = \left(\frac{2000}{s(s+5)(s+20)}\right)$ and you want a 40 degree phase margin, then

i) Find the frequency where you are 40 degrees from unstable

$$G(j4.0031) = 3.8243\angle -140^0$$

ii) Make the gain at this frequency equal to one

$$K = \frac{1}{3.8243} = 0.2615 .$$

This results in a system with a 0dB gain frequency of 4.0031, or approximately,

- The amplitude of the dominant pole is 4.0031
- The resonance is approximately 1.46 $\left(\max\left(\frac{G}{1+G}\right) \approx \left(\frac{1\angle -140}{1+1\angle -140}\right) = 1.46\right)$
- The damping ratio is 0.3684. $\left(M_m = \frac{1}{2\zeta\sqrt{1-\zeta^2}}\right)$
- The dominant pole is approximately $-4.0031\angle \pm 68^0$ or in rectangular coordinates, $-1.47 \pm j3.72$,
- The 2% settling time will be approximately 2.72 seconds.

Note that, once the phase margin is defined (i.e. ζ) is specified, all these terms are determined by the 0dB gain frequency. If you want a faster system, you need to increase the bandwidth.

If you could crank up the gain, you could make the system faster. What limits the gain is the phase of G being too close to -180 degrees at 4 rad/sec.

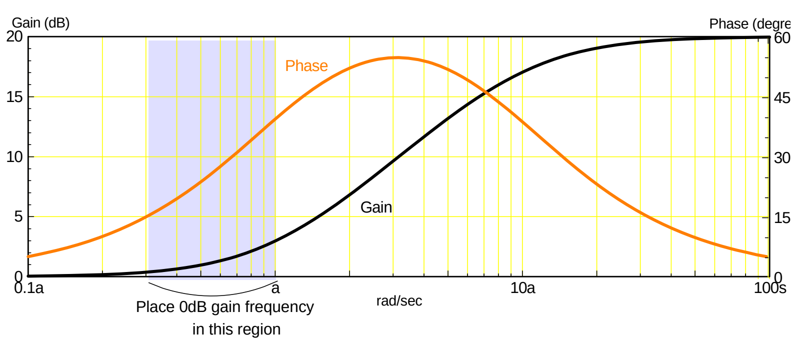
Lead Compensator Design

The purpose of a lead compensator is to increase the phase margin. This implies that the lead compensator needs to add phase (pushing G(jw) to the right, away from -1 as shown on the Nichols chart but not gain (which would push G(jw) up towards -1).

The standard form for a lead compensator is

$$K_{lead} = \left(\frac{10(s+a)}{s+10a}\right)$$

where the pole is made 10x as large as the zero for convenience and the DC gain is set to one. The gain and phase of K(s) is shown below.



The first step in designing a lead compensator is determining how to pick the pole and zero, 'a'. In the previous example, G(s) has problems with phase at 4 rad/sec. At this frequency, you would like to add phase but not gain (i.e increase the phase margin of G). Note that

- If 'a' is too large (say $\frac{a}{10} = 4$), you are adding minimal gain and minimal phase. The lead compensator will not help the phase margin much.
- If 'a' is too small (say $100a = 4$) you are adding gain but not phase. The added gain will push G(jw) closer to -1, making the system behave worse.
- If 'a' is just right (say $4 = \left(\frac{1}{3} \text{ to } 1\right)a$, you'll be adding a significant amount of phase without a lot of gain.

Since this is the purpose of a lead compensator

Rule: Pick the zero of the lead compensator to be 1 to 3 times the 0dB gain frequency of your system.

This should increase the phase margin. You can then use gain compensation to speed up the system.

Example: A gain compensator was designed for $G(s) = \left(\frac{0.2615(2000)}{s(s+5)(s+20)}\right)$ so that the system had a 40 degree phase margin. Add a lead compensator of the form $\left(\frac{10(s+a)}{s+10a}\right)$ and find the system's phase margin with this lead compensator.

Solution: Step 1) Find the 0dB gain frequency. From before, this is 4.0031 rad/sec.

Step 2) Pick the zero to be 1 to 3 times this frequency. Let a=4.

$$K_{gain}K_{lead} = (0.2615)\left(\frac{10(s+4)}{s+40}\right)$$

$$GK = \left(\frac{5230(s+4)}{s(s+5)(s+20)(s+40)}\right)$$

Step 3) The new 0dB gain frequency is 5.7165 rad/sec. The gain at this frequency is

$$GK(j5.7165) = 1.000\angle -107^0$$

so the system now has a 72 degree phase margin. The larger phase margin means the system is more stable / it will have a smaller resonance / the damping ratio is larger.

note: If root locus techniques were used, you would pick the zero to cancel the slowest stable pole at -5 (a=5). You could also do this here since 5 is in the range of (1 to 3) x 4.

Example: Add a gain + lead compensator to the previous system to obtain a 40 degree phase margin:

Solution: Add a lead compensator as before. Now find the new frequency where the phase is 40 degrees from 180:

$$GK(j13.8325) = 0.3596\angle -140^0$$

Now add gain so that the overall gain is one at this frequency

$$K_2 = \frac{1}{-0.3596} = 2.7806$$

so the overall compensator will be

$$K(s) = (0.2615)\left(\frac{10(s+4)}{s+40}\right)(2.7806) = \left(\frac{7.2171(s+4)}{s+40}\right)$$

Note that the 0dB gain frequency is now 13.83, or 3.45 times larger than it was with gain compensation. This implies that the compensated system will

- Be 3.45 times faster, or
- Have 3.45 times wider bandwidth.

In addition, the DC gain of the compensator is 2.7806 times larger, meaning that

- The error constant will be 2.7806 times larger, or
- The steady-state error will be 2.7806 times smaller.

Lead + Gain compensators speed up the system and decrease the steady-state error.

If this system was still not fast enough, a second lead compensator could be added. In this case, the zero would be (1 to 3) times 13.83 rad/sec. (Alternatively, you could cancel the next slowest stable pole, -20, which is in this range.)

Variation of Lead Compensator Design:

If you are given a specific 0dB gain frequency to shoot for, the procedure is to

- i) Add lead compensators until the system is too fast,
- ii) Slide one (or more) of the poles you added in the lead compensator until the desired 0dB gain

Problem: Design a compensator for $G(s) = \left(\frac{2000}{s(s+5)(s+20)}\right)$ so that the system has

- A 40 degree phase margin, and
- A 0dB gain frequency of 10 rad/sec.

Solution: First, see if the simplest solution (K(s)=k) works. If K has no phase, then at 10 rad/sec,

$$GK(j10) = 0.8k\angle -180^0 \text{ .}$$

The phase shift is too much.

Second, try a lead compensator. Canceling the slowest stable pole, let

$$K(s) = \left(\frac{k(s+5)}{s+50}\right)$$

(note: The 10 is left out since gain compensation will be added later. This factor of 10 will be lumped into k in the last step).

Now at 10 rad/sec

$$GK(j10) = 0.1254k\angle -127^0$$

The phase is too small in this case. This can be cured by sliding the pole at -10 in until the phase adds up to -140 degrees at 10 rad/sec

$$\angle\left(\frac{2000k(s+5)}{s(s+5)(s+20)(s+x)}\right)_{s=j10} = -140^0$$

$$\angle(j10) + \angle(j10+20) + \angle(j10+x) = 140^0$$

$$\angle(j10+x) = 23.43^0$$

$$\arctan\left(\frac{10}{x}\right) = 23.43^0$$

$$x = 23.07$$

so

$$K(s) = \left(\frac{k(s+5)}{(s+23.07)}\right)$$

For the system to have a 40 degree phase margin at 10 rad/sec,

$$GK(j10) = \left(\frac{2000k(s+5)}{s(s+5)(s+20)(s+23.07)}\right)_{s=j10} = 0.3557k\angle -140^0 = 1\angle -140^0$$

$$k = \frac{1}{0.3557} = 2.8112$$

$$\boxed{K(s) = \left(\frac{2.8112(s+5)}{s+23.07}\right)}$$

Lag Compensator Design Using Bode Plots

Objectives:

To be able to

- Design a lead compensator using Bode plot techniques,
- Design a lag compensator using Bode plot techniques

Definitions:

- Lead Compensator: $K(s) = k\left(\frac{s+a}{s+b}\right)$ where b>a. The phase of K(s) > 0.
- Lag Compensator: $K(s) = k\left(\frac{s+a}{s+b}\right)$ where a >b. The phase of K(s) < 0.
- Gain Compensation: $K(s) = k$.

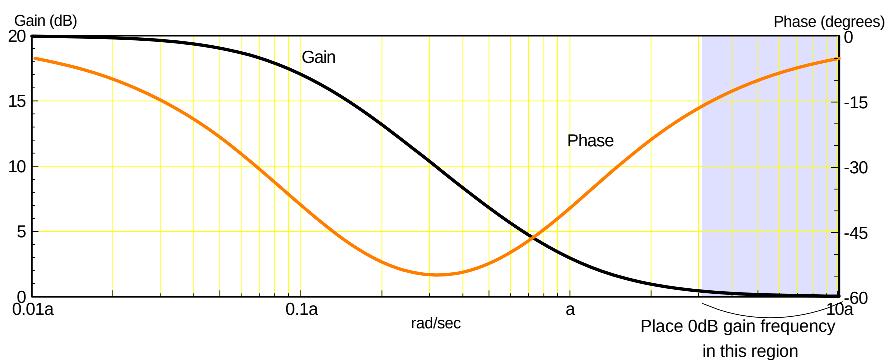
Lag Compensator Design:

Lag compensators serve to increase the DC gain, increasing the error constants and decreasing the tracking error. The idea behind lag compensators is to apply a Band-Aid to a system which is already designed, but does not meet a design criteria for Kp or Kv. In this case, you would like to keep your previous design, but crank up the DC gain.

The form of a lag compensator is

$$K_{lag} = \left(\frac{s+a}{s+0.1a}\right)$$

with its frequency response shown below.



Note that the DC gain will be 10 regardless of how you pick the zero, 'a'. Likewise, this will always increase the value of the error constant by 10x. Also note that

- The gain is always positive. This will push G(s) towards -1 on a Nichols chart, making the system less stable.
- The phase is always negative. This will push G(s) towards -1 on a Nichols chart, making the system less stable.

Since the lag compensator makes the system worse, the design is to

- Pick 'a' as large as possible, making the lag compensator as responsive as possible, but
- Not large enough to affect the system's response.

The standard rule for 'a' is

Pick the zero of the lag compensator to be 1/3rd to 1/10th of the 0dB gain frequency.

Note that even at 1/10th, the phase of the lag compensator is still negative. Letting $\omega_{0dB} = 1$ and a=0.1,

$$K_{lag}(s) = \left(\frac{s+0.1}{s+0.01}\right)$$

$$K_{lag}(j1) = 1.0049\angle -5.1^0$$

This will hurt your phase margin by 5.1 degrees. If you prefer the 1/3 rule

$$K_{lag}(s) = \left(\frac{s+0.33}{s+0.033}\right)$$

$$K_{lag}(j1) = 1.05\angle -16.3^0$$

This hurts the phase margin by 16.3 degrees.

Gain & Lag Compensator Design

Step 1: Pick a design rule for selecting the zero for your lag compensator (such as the zero will be 1/10 or 1/3 of the 0dB gain frequency.)

Step 2: Design a gain compensator with an extra 5.1 degrees (if using the 1/10 rule) or 16.3 degrees (if using the 1/3 rule) of phase margin. Determine the 0dB gain frequency.

Step 3: Add the lag compensator. This should bring the phase margin back to where you wanted it.

Example: Design a gain & lag compensator for $G(s) = \left(\frac{2000}{s(s+5)(s+20)}\right)$ so that the system has a 40 degree phase margin.

Solution:

1) Assume the lag zero will be 1/10th of the 0dB gain frequency.

2) Find the frequency where the phase of G(s) is 45.1 degrees away fro 180 degrees (40 degree phase margin plus an extra 5.1 degrees).

$$G(j3.4983) = 4.6144\angle -134.9^0$$

3) Add the lag compensator

$$K_{lag} = \left(\frac{s+0.34983}{s+0.034983}\right)$$

4) Crank up the gain until $GK = 1\angle -140^0$

$$GK = \left(\frac{2000}{s(s+5)(s+20)}\right)(k)\left(\frac{s+0.34983}{s+0.034983}\right)$$

$$GK(j3.4983) = 4.6372k\angle -140^0$$

k = 0.2156

and the compensator is

$$K(s) = \left(\frac{0.2156(s+0.34983)}{s+0.034983}\right)$$

Note that the DC gain is about 10x larger than what you obtained with gain compensator alone (a little less than 10x since you had to back off on the gain to get an extra 5.1 degrees of phase margin). The cost is a slightly slower system.

Summary

Comparing gain, Lead & Gain, Lag & Gain compensation results in

	Gain	Lead	Lead & Gain	Lag & Gain
K(s)	0.2615	$\left(\frac{2.615(s+4)}{s+40}\right)$	$\left(\frac{7.2171(s+4)}{s+40}\right)$	$\left(\frac{0.2156(s+0.34983)}{s+0.034983}\right)$
Phase Margin	40 degrees	72 degrees	40 degrees	40 degrees
0dB Gain Frequency	4.0031 rad/sec	5.7165 rad/sec	13.8325 rad/sec	3.4983 rad/sec
Kv	5.23	5.23	14.43	43.12

- Gain compensators set the phase margin. Phase margin determines the resonance, damping ratio, percent overshoot, etc. as per the second order approximations.
- Lead compensators increase the phase margin
- Lead & Gain compensators speed up the system and increase the error constant (decreasing tracking error)
- Lag compensators greatly increase the error constant, but slightly slow down the system.