

EE447 Course Notes

Blake Hannaford

September 13, 2012

(C) Copyright 2012, Blake Hannaford, all rights reserved.

Contents

1	Introduction, Review	3
1.1	Problem Statement and Learning Objectives	3
1.2	LODE	3
1.2.1	Basic definition	3
1.2.2	Solution of First Order LODE	3
1.3	Laplace Transform Review	4
1.4	Partial Fraction Expansion	6
1.5	Linearization	10
2	Translational Dynamical Systems	15
2.1	Problem Statement and Learning Objectives	15
2.2	System Elements	15
2.2.1	Displacements and Derivatives	16
2.2.2	Forces	16
2.2.3	Mechanical Network Schematic Diagram	16
2.3	Equations of Motion	17
2.3.1	Parallel and Series Combinations	19
2.3.2	Multiple Masses and EOMs	20
2.4	Conversion to Transfer Function	21
2.5	Examples	22
3	Rotational Dynamical Systems	27
3.1	Problem Statement and Learning Objectives	27
3.2	System Elements & constitutive relations.	27
3.2.1	Torque	27
3.2.2	Elements of Rotational Dynamical Systems	28
3.3	Equations of Motion	28
3.4	Gears	30
3.4.1	Gear Kinematic Relationships	30
3.5	Rotary to Linear Motion	36
4	Transient Response of 2nd Order Systems and Frequency Response	41
4.1	Problem Statement and Learning Objectives	41
4.2	The basic 2nd order dynamical system	41
4.2.1	Pole Location and Step Response	44
4.3	Frequency Response	45
4.3.1	Magnitude	49
4.3.2	Phase	50
4.3.3	Decibels	51
4.3.4	Bode Plot Sketching	52
4.3.5	Combining Magnitude Plots	55
4.3.6	Bode Asymptotic Phase Plot	59
4.3.7	Poles or zeros at the origin	62
4.3.8	Complex Conjugate Poles	64
4.3.9	Complex Conjugate Zeros	66

5 Feedback	69
5.1 Problem Statement and Learning Objectives	69
5.2 Block Diagram Transformations	69
5.2.1 Signals vs. Energy Flows	69
5.2.2 Block Diagrams	69
5.2.3 Transformations	70
5.3 Closed Loop Negative Feedback Gain	71
5.4 Sensitivity Analysis	73
5.5 Disturbance Rejection	75
5.5.1 Disturbance Rejection in the Frequency Domain	77
5.5.2 Location of Disturbance	78
5.6 Stability	80
5.6.1 Calculation of Roots	80
5.7 Stability in the Frequency Domain	81
5.7.1 Gain and Phase Margins	83
6 Root Locus	85
6.1 Problem Statement and Learning Objectives	85
6.2 Introduction to Root Locus	85
6.3 Root Locus Examples	86
6.4 Root Locus Steps	88
6.5 Root Locus FAQ	88
6.6 Hand Root Locus Examples	89
7 Closed Loop Control	93
7.1 Problem Statement and Learning Objectives	93
7.2 System Type and Steady State Error	93
7.2.1 Steady State Error Derivation	94
7.2.2 Steady State Error Examples	95
7.2.3 Steady State Error Summary	96
7.3 Time Domain Performance of 2nd Order Systems	98
7.3.1 Transient Performance Specifications	98
7.3.2 S-plane Regions	99
7.3.3 S-plane Performance Region Examples	101
7.4 PID Controller	104
7.4.1 Closed Loop Design Problem	104
7.4.2 PID Control	104
7.4.3 Basics	105
7.4.4 Simulation of PID controllers	105
7.4.5 Control Effort	106
7.5 Manual Design of PID controller	107
7.5.1 Solution method	108
7.5.2 Deriving K_P, K_I, K_D from controller zeros	108
8 Search and Optimization with Scilab	111
8.1 Problem Statement and Learning Objectives	111
8.2 Design with Scilab	111
8.2.1 Polynomials in Scilab	111
8.2.2 Overview	111
8.2.3 Performance Functions	112
8.2.4 Weights	112
8.2.5 Gain Space Searching and Optimization	113
8.2.6 Range Saturation	114
8.3 Using the Scilab packages	114
8.3.1 optivis.sce	115
8.4 Solving Design Problems	115
8.5 Description of Software Operation	116
8.6 Example Design	117

9 Conversion to Discrete Time: Tustin's Method	121
9.1 Problem Statement and Learning Objectives	121
9.2 Overview	121
9.3 Discrete Time and Z transform review/intro	122
9.3.1 Discrete and Continuous Comparison Table	123
9.3.2 Sampling Theorem	123
9.4 Tustin's Method	123
9.4.1 Tustin's method example 1	123
9.4.2 Tustin's method example 2	125
9.4.3 Conversion by Computer	126
9.4.4 Conversion of discrete transfer function to digital filter	127
9.5 Code Example	128
9.6 Limitations and properties	129

EE447 Autumn 2012

Lecture Schedule

wk.	Date	Topic	Reading
1	25-Sep-2012	Intro. / LODE and Partial Fraction Expansion Review	Chapt 1.
1		LODE and Partial Fraction Expansion (ICP)	
2	02-Oct-2012	Translational Dynamic Systems	Chapt 2.
2		Translational Dynamic Systems (ICP)	
3	09-Oct-2012	Rotational Dynamic Systems	Chapt 3.
3		Rotational Dynamic Systems (ICP)	
4	16-Oct-2012	Bode Diagram Sketching and response of 2nd Order Systems	Chapt 4.
4		Mid Term I	
5	23-Oct-2012	Bode Diagram Sketching and response of 2nd Order Systems (ICP)	
5		Feedback	Chapt 5.
6	30-Oct-2012	Feedback (ICP)	
6		Closed Loop Design: Root Locus	Chapt 6.
7	06-Nov-2012	Closed Loop Design: Root Locus (ICP)	
7		Mid Term II	
8	13-Nov-2012	Closed Loop Design: Closed Loop Control Concepts	Chapt 7.
8		Closed Loop Design: Closed Loop Control Concepts (ICP)	
9	20-Nov-2012	Closed Loop Design: Search and optimization	Chapt 8.
9		Thanksgiving	
10	27-Nov-2012	Closed Loop Design: Search and optimization (ICP)	
10		Tustin's Method	Chapt 9.

Chapter 1

Introduction, Review

1.1 Problem Statement and Learning Objectives

This week we will review some mathematical basics.

5 1.2 LODE

1.2.1 Basic definition

A Linear Ordinary Differential Equation (LODE) is an equation of the form

$$f(t) = \sum_{i=0}^N a_i \frac{d^i}{dt^i} x(t)$$

The highest derivative, N , is referred to as the *order* of the LODE. For example,

$$f(t) = 0.273 \frac{d^3}{dt^3} x(t) + 6.73 \frac{d^2}{dt^2} x(t) - 0.001 \frac{d}{dt} x(t)$$

is a third-order LODE. Often we introduce some shorthand by omitting the time dependence “ (t) ”, and by
10 using the dot notation for time derivatives:

$$\dot{x} = \frac{d}{dt} x(t) \quad \ddot{x} = \frac{d^2}{dt^2} x(t)$$

So that the example above would be written

$$f(t) = 0.273 \frac{d^3}{dt^3} x + 6.73 \ddot{x} - 0.001 \dot{x}$$

(you can use three dots for $\frac{d^3}{dt^3}$ if you wish.)

1.2.2 Solution of First Order LODE

A very basic first order LODE is

$$\dot{x} + Px = 0$$

15 If we guess the solution is

$$x(t) = e^{pt} \quad \dot{x}(t) = pe^{pt}$$

we can easily check it by plugging in to the original LODE:

$$pe^{pt} + Pe^{pt} = 0 \rightarrow pe^{pt} = -Pe^{pt}$$

giving

$$p = -P$$

$f(t)$ for $t \geq 0$	$\mathcal{L}(f)$
$1U(t)$	$\frac{1}{s}$
e^{at}	$\frac{1}{s-a}$
t^n	$\frac{n!}{s^{n+1}} (n = 0, 1, \dots)$
$\sin at$	$\frac{a}{s^2 + a^2}$
$\cos at$	$\frac{s}{s^2 + a^2}$
$\dot{f}(t)$	$s\mathcal{L}(f) - f(0)$
$\ddot{f}(t)$	$s^2\mathcal{L}(f) - sf(0) - \dot{f}(0)$

(Where $\dot{f}(t) = \frac{df(t)}{dt}$, $\ddot{f}(t) = \frac{d^2f(t)}{dt^2}$)

Table 1.1: Table of some commonly used Laplace Transform pairs.

so the solution is

$$x(t) = e^{-Pt}$$

Because of the partial fraction expansion (covered below), this covers a very wide variety of real world LODEs.

1.3 Laplace Transform Review

- ⁵ We will extensively use Laplace Transforms to manipulate the differential equations of control systems. As a review, the Laplace transform is

$$\mathcal{L}\{f(t)\} = F(s) = \int_0^\infty e^{-st} f(t) dt$$

where s is a complex variable, $s = \sigma + j\omega$. Technically this is the “one sided” Laplace Transform because the integral extends only to positive t . The inverse of the Laplace Transform is

$$f(t) = \mathcal{L}^{-1}F(s) = \frac{1}{2\pi j} \lim_{T \rightarrow \infty} \int_{\sigma-jT}^{\sigma+jT} e^{st} ds$$

- For this course, we will not need to evaluate these integrals, because all of the LODEs we study will result in just a few terms which have been worked out long ago and are widely available in tables (Table 1.1).

When using this table we need to keep in mind a few limitations and assumptions:

- We are only considering $t > 0$. One way to do this is to consider functions to be multiplied by the Unit Step function, $u(t)$

$$u(t) = \begin{cases} 0 & t < 0 \\ 1 & t \geq 0 \end{cases}$$

- ¹⁵ When using the differentiation relationship (last two lines of Table 1.1) we must be careful about the initial conditions ($f(0)$, $\dot{f}(0)$). Most often, we assume that all initial conditions are zero, but it is your responsibility to verify that this assumption applies to a given problem.

Example 1.1

Find the Laplace Transform of the following time functions: in all cases, assume the function is 0 for $t < 0$ and that initial conditions are zero

$$\mathcal{L}\{55e^{-1.2t}\}$$

Consulting the table and using the linearity property of the Laplace Transform integral:

$$\mathcal{L}\{55e^{-1.2t}\} = \frac{55}{s + 1.2}$$

$$\mathcal{L}\{3.2 \sin(7.6t)\} = \frac{7.6}{s^2 + 57.76}$$

$$\mathcal{L}\{2.6\ddot{x} - 3.52\dot{x} + 120x\} = X(s)(2.6s^2 - 3.52s + 120)$$

Note that we have assumed that $x(0) = \dot{x}(0) = \ddot{x}(0) = 0$. Although this seems restrictive, we often consider physical systems starting from rest and this situation is well modeled by such zero initial conditions.

Example 1.2

Find the inverse Laplace transform of the following functions

$$\mathcal{L}^{-1}\left\{\frac{10}{s + 3.7}\right\}$$

Consulting the table and using the linearity property of the Laplace Transform integral:

$$\mathcal{L}^{-1}\left\{\frac{10}{s + 3.7}\right\} = 10e^{-3.7t}$$

$$\mathcal{L}^{-1}\left\{\frac{144s}{s^2 + 144}\right\} = 144 \cos(12t)$$

$$\mathcal{L}^{-1}\{X(s)(s^3 + as^2 + bs + c)\} = \frac{d^3}{dt^3}x + a\ddot{x} + b\dot{x} + cx$$

Here we have implicitly assumed zero initial conditions.

⁵ Let's apply the Laplace Transform to the initial LODE above. First, we will modify the LODE to include a "Forcing Function" on the right hand side. A forcing function is typically a physical input to the system such as an applied voltage or force.

$$\dot{x} + Px = f(t)$$

Assuming zero initial conditions and taking the Laplace Transform of both sides (see the second to last line of Table 1.1).

$$sX(s) + PX(s) = F(s)$$

$$X(s)(s + P) = F(s)$$

$$\frac{X(s)}{F(s)} = \frac{1}{(s + P)}$$

Here we have derived a ratio called the “Transfer Function” between position over the forcing function. Our solution to the LODE was

$$x(t) = e^{pt} = e^{-Pt}$$

If we rewrite this $x(t) = e^{pt}$, then our transfer function becomes

$$\frac{X(s)}{F(s)} = \frac{1}{(s - p)}$$

⁵ We call p the “pole” of this transfer function because the transfer function goes to infinity when $s = p$.

1.4 Partial Fraction Expansion

A very useful property of polynomials is the partial fraction expansion:

$$\frac{\prod_j (s - z_j)}{\prod_i (s - p_i)} = \sum_i \frac{A_i}{(s - p_i)}$$

In the form on the left, we have terms called *zeros* on the top because any time $s = z_i$ the ratio is zero. We also have terms called *poles* in the denominator, because any time $s = p_i$, the ratio is infinite.

¹⁰ The partial fraction expansion is very useful for ratios of polynomials in s (such as the left side above) which we will encounter frequently because such a ratio becomes a series of terms which have an easy inverse Laplace Transform

$$\frac{A_i}{(s - p_i)} \leftrightarrow A_i e^{-p_i t}$$

For example,

$$G(s) = \frac{s + 5}{s^3 + 6s^2 + 35s + 400}$$

does not have an obvious inverse Laplace transform. However if we can factor it to get

$$G(s) = \frac{(s + 5)}{(s + 2)(s + 4)(s + 50)}$$

¹⁵ then we can use the Partial Fraction expansion to get it into the form

$$G(s) = \frac{A_1}{(s + 2)} + \frac{A_2}{(s + 4)} + \frac{A_3}{(s + 50)}$$

then we can immediately write

$$g(t) = A_1 e^{-2t} + A_2 e^{-4t} + A_3 e^{-50t}$$

It is not obvious that the partial fraction expansion is always possible but we will derive a class of cases and then perform some examples.

Assume

$$G(s) = \frac{(s - z)}{(s - p_1)(s - p_2)(s - p_3)} = \frac{A_1}{(s - p_1)} + \frac{A_2}{(s - p_2)} + \frac{A_3}{(s - p_3)}$$

²⁰ Now we multiply through by $(s - p_1)$:

$$G(s) = \frac{(s - p_1)(s - z)}{(s - p_1)(s - p_2)(s - p_3)} = \frac{A_1(s - p_1)}{(s - p_1)} + \frac{A_2(s - p_1)}{(s - p_2)} + \frac{A_3(s - p_1)}{(s - p_3)}$$

Now we do two things: 1) cancel terms where possible, 2) solve for the special case $s = p_1$

1)

$$\frac{(s - z)}{(s - p_2)(s - p_3)} = \frac{A_1}{1} + \frac{A_2(s - p_1)}{(s - p_2)} + \frac{A_3(s - p_1)}{(s - p_3)}$$

2)

$$\frac{(s+z)}{(s-p_2)(s-p_3)} = A_1$$

We have just solved A_1 . If we multiply through by each denominator in turn, we can get all the A_i .

Example 1.3

Expand

$$G(s) = \frac{50(s+1)}{(s+0.1)(s+14)(s+567)}$$

by the Partial Fraction Expansion.

$$\begin{aligned} G(s) &= \frac{A_1}{(s+0.1)} + \frac{A_2}{(s+14)} + \frac{A_3}{(s+567)} \\ A_1 &= \left. \frac{(s+0.1)50(s+1)}{(s+0.1)(s+14)(s+567)} \right|_{s=-0.1} = \frac{50(0.9)}{13.9 \times 566.9} = 0.00571 \\ A_2 &= \left. \frac{(s+14)50(s+1)}{(s+0.1)(s+14)(s+567)} \right|_{s=-14} = \frac{50(-13)}{-13.9 \times 553} = 0.0846 \\ A_3 &= \left. \frac{(s+567)50(s+1)}{(s+0.1)(s+14)(s+567)} \right|_{s=-567} = \frac{50(-566)}{-566.9 \times -553} = -0.090 \\ G(s) &= \frac{0.00571}{(s+0.1)} + \frac{0.0846}{(s+14)} + \frac{-0.090}{(s+567)} \end{aligned}$$

- ⁵ There are modifications to these methods for the case where a pole appears twice (for example $\frac{1}{(s+4)^2}$) and where poles are complex conjugates, working out the A_i can be more involved, but we will not need these cases.

Example 1.4

Use the Partial Fraction Expansion to find the inverse Laplace Transform of

$$G(s) = \frac{27}{(s+50)(s+3000)}$$

$$\begin{aligned} A_1 &= \left. \frac{(s+3000)27}{(s+50)(s+3000)} \right|_{s=-3000} = \frac{27}{-2950} = -0.00915 \\ A_2 &= \left. \frac{(s+50)27}{(s+50)(s+3000)} \right|_{s=-50} = \frac{27}{2950} = 0.00915 \\ g(t) &= 0.00915(e^{-50t} - e^{-3000t}) \end{aligned}$$

- ¹⁰ In the case that poles are complex conjugates¹, there are further simplifications possible after the Partial Fraction Expansion.

¹Recall that complex poles always come in complex conjugate pairs.

Example 1.5

Find

$$\mathcal{L}^{-1}\{G(s) = \frac{(s+5)}{(s+6)(s+2+j)(s+2-j)}\}$$

Using the techniques above we can get:

$$A_1 = -1/17 = -0.059 \quad A_2 = 0.029 + 0.38j \quad A_3 = 0.029 - 0.38j$$

(note that it is not necessary to compute A_3 because it will always be the case for complex conjugate poles that $A_3 = A_2^*$.)

$$G(s) = \frac{-0.059}{(s+6)} + \frac{0.029 + 0.38j}{(s+2+j)} + \frac{0.029 - 0.38j}{(s+2-j)}$$

Applying the inverse transform to each term:

$$g(t) = -0.059e^{-6t} + (0.029 + 0.38j)e^{(-2-j)t} + (0.029 - 0.38j)e^{(-2+j)t}$$

First, let's approximate

$$0.029 \pm 0.38j \approx 0.38e^{\pm j\pi/2}$$

by 1) ignoring the small real part and 2) converting to Magnitude-Angle form.

$$g(t) = -0.059e^{-6t} + 0.38e^{j\pi/2}e^{(-2-j)t} + 0.38e^{-j\pi/2}e^{(-2+j)t}$$

$$g(t) = -0.059e^{-6t} + 0.38e^{-2t} \left(e^{j(-t+\pi/2)} + e^{-j(-t+\pi/2)} \right)$$

Now we use Euler's famous equality

$$e^{j\theta} = \cos(\theta) + j \sin(\theta)$$

as follows:

$$g(t) = -0.059e^{-6t} + 0.38e^{-2t} (\cos(-t + \pi/2) + j \sin(-t + \pi/2) + \cos(-t + \pi/2) - j \sin(-t + \pi/2))$$

Since $\cos(\theta) = -\cos(\theta)$, and $\cos(\theta - /pi/2) = \sin(\theta)$

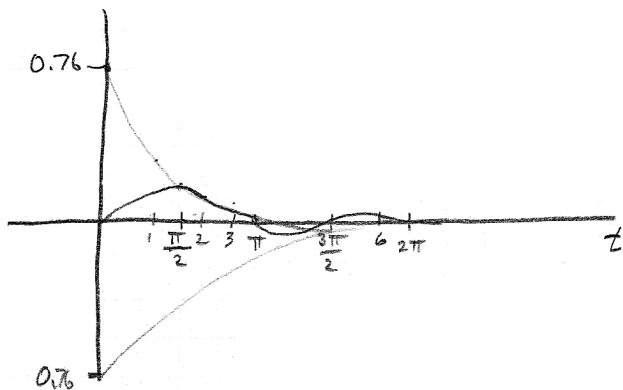
$$g(t) = -0.059e^{-6t} + 0.38e^{-2t} (2 \cos(t - \pi/2))$$

$$g(t) = -0.059e^{-6t} + 0.76e^{-2t} (\cos(t - \pi/2))$$

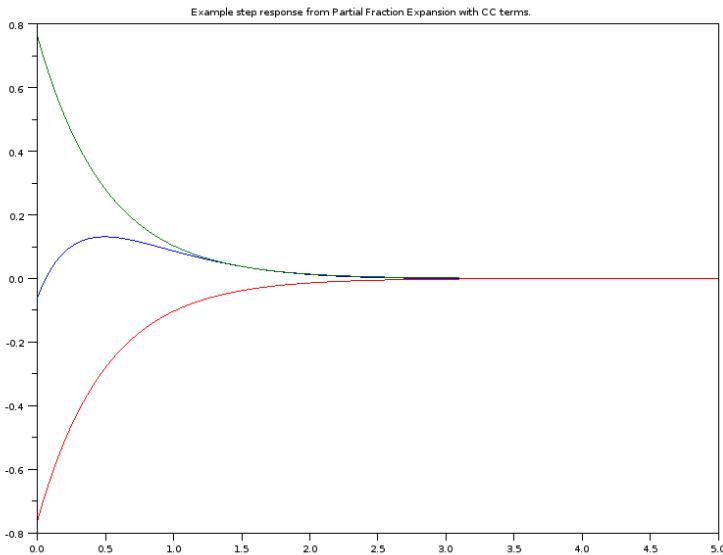
$$g(t) = -0.059e^{-6t} + 0.76e^{-2t} (\sin(t))$$

Example 1.5 cont.

By throwing a few values of t into the calculator, we can generate a sketch of the second term is shown below:



For a more accurate plot, let's use the computer (including both terms):



Another situation comes when a pole is repeated (i.e. $\frac{1}{(s+2)^2}$). In this case the trick we use for the partial fraction expansion no longer works! But instead we can still solve for A_i by differentiating the partial fraction

expansion.

Example 1.6

Apply the Partial Fraction Expansion to

$$G(s) = \frac{(s+5)}{s^2(s+3)}$$

(noting that there is a repeated root in the denominator (repeated pole). We start by setting up the problem with two terms for the repeated pole:

$$G(s) = \frac{A_1}{s^2} + \frac{A_2}{s} + \frac{A_3}{(s+3)}$$

A_1 :

$$\left. \frac{s^2(s+5)}{s^2(s+3)} \right|_{s=0} = \frac{s^2 A_1}{s^2} + \frac{s^2 A_2}{s} + \frac{s^2 A_3}{(s+3)}$$

giving

$$A_1 = 5/3$$

A_3 is also straightforward, giving $A_3 = 2/9$. But working through A_2 we find:

$$\left. \frac{s(s+5)}{s^2(s+3)} \right|_{s=0} = \frac{s A_1}{s^2} + A_2 + \frac{s A_3}{s^2(s+3)}$$

We now have the problem that we cannot cancel the s^2 in the denominator of several terms (which we need to do!). Instead differentiate the A_1 expression with respect to s :

$$\frac{d}{ds} \frac{(s+5)}{(s+2)} = 0 + A_2 + \frac{d}{ds} \frac{s^2}{(s+3)} A_3$$

$$\frac{-2}{(s+3)^2} = A_2 + \frac{s(s+6)}{(s+3)^2} A_3$$

evaluating at $s = 0$,

$$A_2 = -2/9$$

Note that we have used:

$$\frac{d}{dx} \frac{(x+a)}{(x+b)} = \frac{1}{(x+b)} - \frac{(x+a)}{(x+b)^2}$$

and

$$\frac{d}{dx} \frac{x^2}{(x+a)} = \frac{2x}{(x+a)} - \frac{x^2}{(x+a)^2}$$

This gets even more unwieldy when the repeated pole is non-zero but fortunately we rarely need this technique or can fall back on numerical methods.

1.5 Linearization

- 5 Laplace Transforms can only be used on linear equations. In most of this course we focus on linear differential equations but often real world applications involve non-linear functions. All is not lost if we can usefully approximate our non-linear function with a linear one. Our approach to linearization is to model the original function with a straight line.

Consider a nonlinear function, $f(x)$. The linearized version is always constructed about a specific operating point, x_0 .

$$\hat{f}(x) = f(x_0) + \frac{d}{dx}f(x_0)(x - x_0)$$

$\hat{f}(x)$ is a linear function of x which is most accurate in the neighborhood of x_0 . It is also the first two terms of a Taylor series. It is very helpful to visualize this process graphically by plotting the linearized function on top of the original function.

Example 1.7

Consider the nonlinear function

$$f_1(x) = 0.4x^2 - 0.1x^3 + 3 \sin(x)$$

and linearize twice, once about $x = -6$, and again about $x = 1$.

First evaluate $f_1(x)$ for the two linearization points:

$$f_1(-6) = 36.84 \quad f_1(1) = 2.824$$

Then let's get the derivative:

$$\dot{f}(x) = 0.8x - 0.3x^2 + 3 \cos(x)$$

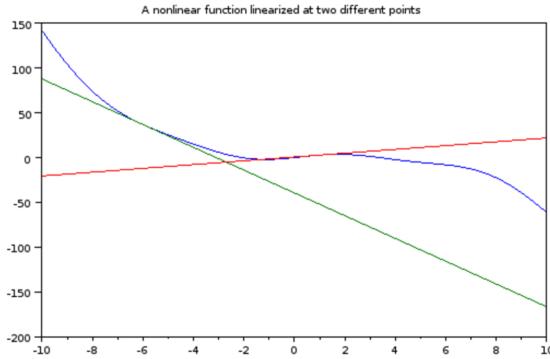
and evaluate it at the two points:

$$\dot{f}(-6) = -12.72 \quad \dot{f}(1) = 2.121$$

Now we get:

$$\hat{f}_{1a} = 36.84 - 12.72(x + 6) = -39.48 - 12.72x \quad \hat{f}_{1b} = 2.824 + 2.121(x - 1) = 0.703 + 2.121x$$

Plotting using Scilab:



The blue line is $f(x)$, the green line is $\hat{f}_{1a}(x)$ and the red line is $\hat{f}_{1b}(x)$. The linear approximations are reasonably accurate in the neighborhood of their operating points, but become very bad as we move away. The size of the “neighborhood” for which a degree of accuracy can be obtained depends on the function and the selected operating point.

Some functions are really easy to linearize.

Example 1.8

Linearize

$$f_2(x) = 0.723x^3 - 4.37x^2 + 67x$$

about the point $x = 0$

$$f_2(0) = 0$$

$$\dot{f}(x) = 2.169x^2 - 8.74x + 67$$

$$\dot{f}_2(0) = 67$$

$$\hat{f}_2(x) = 67x$$

Note that we are simply taking the linear term of $f_2(x)$. However we only could get away with this because 1) we linearized about $x = 0$ and 2) the function was a polynomial.

Example 1.9

Linearize $f_2(x)$ from the previous example about $x = 5.7$

$$f_2(5.7) = 373.81$$

$$\dot{f}_2(5.7) = 87.6521$$

$$\hat{f}_{2a} = 373.81 + 87.6521(x - 5.7)$$

$$\hat{f}_{2a} = -125.81 + 87.6521x$$

This is quite different from $\hat{f}_2(x)$.

When the equation to be linearized is a differential equation, we introduce a slightly strange notion: taking

the derivative with respect to a derivative:

Example 1.10

Linearize the differential equation

$$f_3(t) = \ddot{x}(t) - 0.42\dot{x}(t) + 0.01\dot{x}(t)^3 + 16x(t)$$

about $\dot{x}(t) = 0$.

$$f_3(t)|_{\dot{x}=0} = \ddot{x}(t) + 16x(t)$$

$$\frac{d}{d\dot{x}(t)} f_3(t) = 0 - 0.42 + 0.03\dot{x}(t)^2 + 0$$

for $\dot{x}(t) = 0$,

$$\left. \frac{d}{d\dot{x}(t)} f_3(t) \right|_{\dot{x}=0} = -0.42$$

The linearized differential equation is then

$$\hat{f}_3(t) = \ddot{x}(t) + 16x(t) - 0.42\dot{x}(t)$$

re-writing in traditional LODE form and dropping the (t) for convenience:

$$\hat{f}_3(t) = \ddot{x} - 0.42\dot{x} + 16x$$

Chapter 2

Translational Dynamical Systems

2.1 Problem Statement and Learning Objectives

- System Elements & constitutive relations.
- Equations of Motion
- Conversion to Transfer Function

2.2 System Elements

Translation refers to motion in a straight line. We will first consider systems which only contains elements moving along a single direction. Sometimes it is useful to think of this direction as a set of different but parallel axes, but this distinction does not change the physics. We only consider such systems which operate in an inertial frame such as the surface of the earth (to a good approximation at least) or in a vehicle moving at constant speed and direction.

We will analyze systems consisting of

- **Mass** The property of matter which resists acceleration, is acted on by gravity, and which stores kinetic energy.
- **Stiffness** The property of matter which resists displacement, and which stores potential energy.
- **Damping** The property of matter or interactions of matter which converts motion to heat.

Some properties of the various elements are summarized Table 2.1.

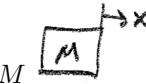
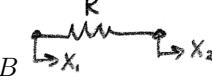
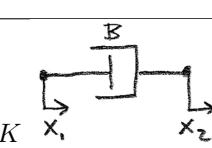
Name	Physical Realization	Symbol	Equation	Notes
Inertia	Point Mass		$f(t) = M\ddot{x}$	\ddot{x} is with respect to the inertial frame.
Stiffness	Massless Spring		$f(t) = Kx$	f is same on both sides. Assume zero rest length
Damping	Shock Absorber		$f(t) = B\dot{x}$	This is a linear model for friction.

Table 2.1:

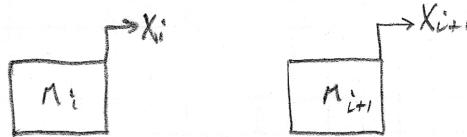


Figure 2.1: Each mass has an associated displacement.

2.2.1 Displacements and Derivatives

We shall analyze the state and the motion of translational systems in terms of Position, $x(t)$, Velocity, $\dot{x}(t)$, and Acceleration, $\ddot{x}(t)$. Where each dot represents a time derivative:

$$\dot{x} = \frac{d}{dt}x(t) \quad \ddot{x} = \frac{d^2}{dt^2}x(t)$$

Often we will omit the time dependence, i.e. $\dot{x} = \dot{x}(t)$.

⁵ 2.2.2 Forces

Each system element generates forces according to well known physical laws:

- Mass: $F = m\ddot{x}$.
- Stiffness: $F = K(x - x_0)$
- Damping: $F = B(\dot{x} - \dot{x}_0)$

¹⁰ In the case of stiffness and damping, the force is generated by the *difference* of two displacements or velocities. In the case of Mass undergoing translational motion in an inertial frame, the force is generated only by accelerations with respect to the inertial frame which

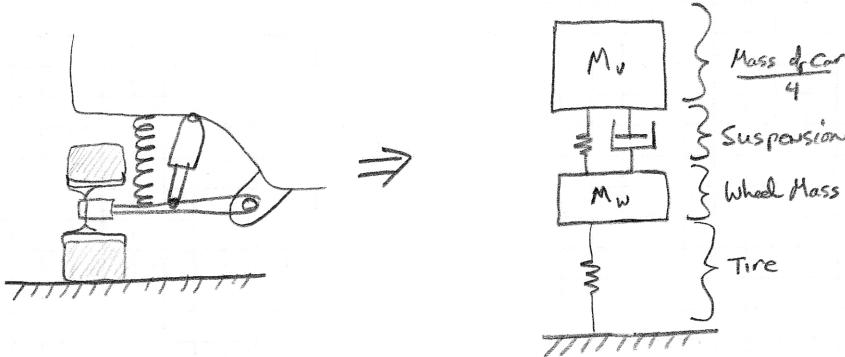
2.2.3 Mechanical Network Schematic Diagram

We must usually simplify the mechanical system into a purely translational one to apply the analysis of this chapter. To do so, we identify the mass for each moving part and draw it as a box labeled M_i (Figure 2.1). What each box actually represents is a point mass. Each point mass has a displacement, x_i which indicates its location along the axis of linear motion.

¹⁵ Springs and dampers are then connected between the moving parts. Alternatively, one end of a spring or damper may be connected to ground (a point at which $x = \dot{x} = \ddot{x} = 0$).

Example 2.1

Convert the tire, wheel, and suspension elements of a typical car to a linear mass-spring-damper model.



In this conversion we have used lots of knowledge about cars including the following facts:

- Tires are elastic and filled with a low mass material (air) and thus could plausibly be approximated by a spring.
- The weight of the wheel and tire can be combined into a mass.
- The suspension spring goes between the suspension beam and the car's body.
- Cars have four wheels so the mass of the body should be approximated by 1/4 of the car's total mass.
- The shock absorber is a damper which connects between the suspension arm and the car body (in parallel with the spring)
- The suspension arm is long enough compared to the tire's motion such that we can approximate the tire's motion as a straight vertical line.

None of this knowledge is required to excel at control systems design with one exception, *The control system designer must have enough knowledge of the application system or access to enough model validation data to make sure that the simplified model is good enough for all application requirements.* To the extent that these “facts” are true, our model is accurate, and to the extent that this model is an oversimplification, our model will not work.

“It can scarcely be denied that the supreme goal of all theory is to make the irreducible basic elements as simple and as few as possible without having to surrender the adequate representation of a single datum of experience.”

Albert Einstein

2.3 Equations of Motion

D'Alembert's Principle equates the famous inertial force $F = m\ddot{x}$ to all the other forces acting on a body. In a form that we will use:

$$M\ddot{x} + \sum_j B_j(\dot{x} - \dot{x}_j) + \sum_k K_k(x - x_k) = F \quad (2.1)$$

where there are several viscous elements connected between the mass M and other masses indicated by j , and there are several elastic elements connected to some other masses indicated by k .

Equation 2.1 is referred to as an Equation of Motion (EOM).

F indicates external forces imposed on the mass from sources other than springs and dampers in the system such as an actuator. If the translational system is vertical, the force of gravity would be one such force, $F = Mg$.

- The signs in equations of motion can be tricky. There is really no “correct” sign for each term, because
- 5 the equation is valid if you multiply both signs by -1 . However if we stick with the following rules, we can write equations of motion in a consistent way so that we can easily keep signs straight:

- The positive term in each subtraction associated with B and K must be the displacement of the mass for which the current EOM is being written.

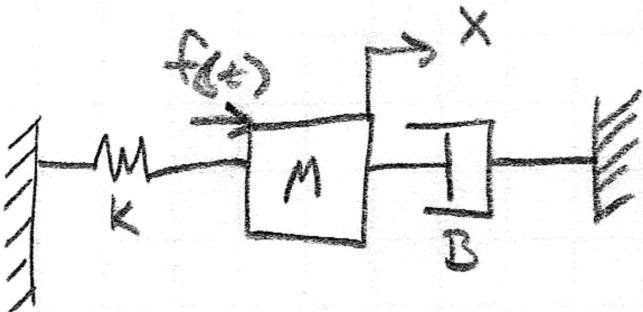
10

- Keep all position sign conventions consistent. For example, all displacements positive “to the left” or positive “facing up”.
- Keep the sign convention of the external applied force the same as for the displacements and keep it alone on the right-hand-side.

When there are multiple masses in the system, we write an EOM for each mass:

$$M_i \ddot{x}_i + \sum_j B_j (\dot{x}_i - \dot{x}_j) + \sum_k K_k (x_i - x_k) = F_i \quad (2.2)$$

Example 2.2



writing the EOM for this one mass:

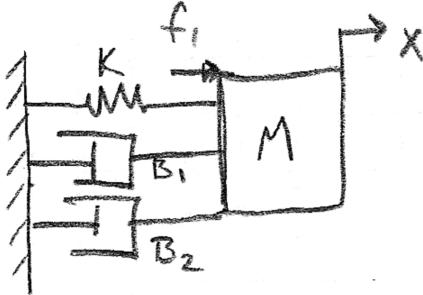
$$M\ddot{x} + B(\dot{x} - 0) + K(x - 0) = f(t)$$

The spring and damper are each connected to ground ($x = 0$, $\dot{x} = 0$ respectively) and so we have shown -0 for completeness. Normally when we have springs and dampers grounded we skip that and the EOM is simply:

$$M\ddot{x} + B\dot{x} + Kx = f(t)$$

Example 2.3

Here we have still a single mass, but multiple dampers are connected. Note that it makes no difference if the “ground” symbol is located to the left or right of the mass, it still represents $x = 0, \dot{x} = 0$.



EOM:

$$M\ddot{x} + B_1(\dot{x} - 0) + B_2(\dot{x} - 0) + K(x - 0) = f(t)$$

Simplifying

$$M\ddot{x} + (B_1 + B_2)\dot{x} + Kx = f(t)$$

2.3.1 Parallel and Series Combinations

We were able to simplify the EOM in Example 2.3 in a way which added the two dampers together. If you think about a simple modification to Example 2.3, you can see that the same could be done with springs as well. The general principle is that springs and dampers combine (like capacitors in electric circuits) (Figure 2.2) as follows:

Springs and dampers in *parallel* can be combined by addition.

Springs and dampers in *series* can be combined like parallel resistors:

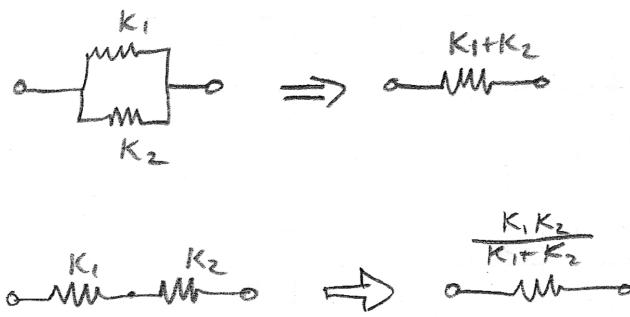


Figure 2.2: Springs in parallel and series can be combined (like capacitors).

¹⁰ *Proof:* Consider two springs in series. The force, f is the same throughout all elements of a serial chain, and both springs independently obey Hooke’s Law:

$$f = K_1\Delta x_1 = K_2\Delta x_2$$

The total change in length due to the applied force, f , is

$$\Delta x = \Delta x_1 + \Delta x_2$$

$$\Delta x = \frac{f}{K_1} + \frac{f}{K_2}$$

$$K_T = \frac{f}{\Delta x} = \frac{1}{1/K_1 + 1/K_2} = \frac{K_1 K_2}{K_1 + K_2} \quad (2.3)$$

An almost identical proof can be made for series connected dampers.

However, Mass, M is different (Figure 2.3) because of the unique property that the force on a mass depends on the acceleration only with respect to the inertial frame:

$$f = m\ddot{x}, \quad \text{NOT} \quad f = m(\ddot{x}_i - \ddot{x}_j)$$

Thus



Figure 2.3: Add two masses which ever way they are combined.

⁵ What about the case where a spring and damper are connected in series (Figure 2.4)?

Using a similar analysis based on the fact that f is the same in both elements of a serial chain:

$$f = K(x_1 - x_2) = B(\dot{x}_2 - \dot{x}_3)$$

The difference here is that we have a new unknown x_2 . This new unknown requires a new EOM however $m_2 = 0$. The equivalent system is thus:

2.3.2 Multiple Masses and EOMs

- ¹⁰ When there are multiple independent masses (who's displacements, x_i are not the same, Figure ??) then we need a separate EOM for each mass.

In a general system with multiple masses, dampers or springs can be connected between any two of the

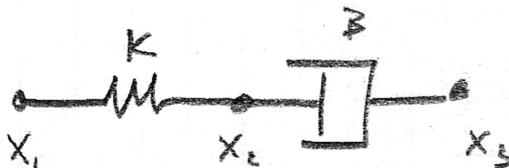
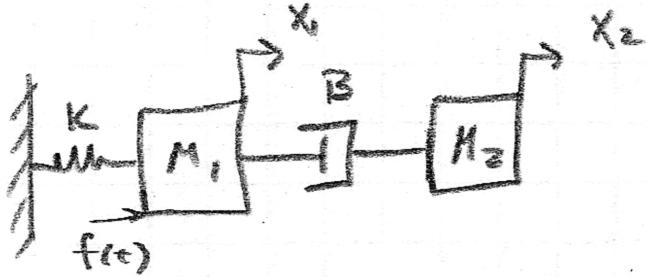


Figure 2.4: If a spring and damper are connected in series, a new EOM must be constructed for the node in between (x_2).

masses. This is why we used the subscripts and the subtractions in Equation 2.2.

Example 2.4


By applying Equation 2.2 to each mass,

$$(M_1) \quad M_1 \ddot{x}_1 + B(\dot{x}_1 - \dot{x}_2) + Kx_1 = f(t)$$

$$(M_2) \quad M_2 \ddot{x}_2 + B(\dot{x}_2 - \dot{x}_1) = 0$$

Note that each EOM for mass i always begins with $m_i \ddot{x}_i$ and that in the B, K terms, x_i is always taken positive.

2.4 Conversion to Transfer Function

- ⁵ EOMs are *Linear Ordinary Differential Equations*, LODEs. As such, we can easily apply the Laplace Transform. Using the EOMs of Example 2.4,

$$M_1 X_1(s)s^2 + BX_1(s)s - BX_2(s)s + KX_1(s) = F(s)$$

Note that we have assumed zero initial conditions. What does this assumption mean? Mathematically it means

$$x_i(t=0) = 0, \quad \dot{x}_i(t=0) = 0, \quad \ddot{x}_i(t=0) = 0$$

and physically this corresponds to the system being at rest and having no kinetic or potential energy.

- ¹⁰ We will use the Laplace transform to solve for *Transfer Functions*. Transfer functions are ratios between the Laplace Transforms of two physical variables. Examples:

$$\frac{X_2(s)}{F(s)} \quad \frac{X_1(s)}{X_2(s)} \quad \text{etc.}$$

Often we need to analyze a system when we know its input (say $X(s)$) but do not know its output (say $Y(s)$). If we can obtain the transfer function

$$G(s) = \frac{Y(s)}{X(s)}$$

then we can get the Laplace transform of the output by

$$Y(s) = G(s)X(s)$$

¹⁵ The transfer function is obtained by algebraically manipulating the Laplace transform of one or more

EOMs.

Example 2.5

For the system of Example 2.3, let's find the transfer function

$$G(s) = \frac{X(s)}{F(s)}$$

Using this transfer function we can compute the displacement as a function of the input force. Starting with the EOM

$$M\ddot{x} + (B_1 + B_2)\dot{x} + Kx = f(t)$$

First take the Laplace Transform:

$$(LT) \quad MX(s)s^2 + (B_1 + B_2)X(s)s + KX(s) = F(s)$$

then we factor out $X(s)$ from each term giving

$$X(s)(Ms^2 + (B_1 + B_2)s + K) = F(s)$$

dividing through we get the transfer function

$$G(s) = \frac{X(s)}{F(s)} = \frac{1}{Ms^2 + (B_1 + B_2)s + K}$$

- We will find it useful to *normalize* each transfer function before further analysis. This means that we will manipulate each polynomial in s so that the coefficient of highest power of s is 1. This is accomplished by just dividing through by the coefficient of the highest power of s as a final step.

Example 2.6

Normalize the transfer function of Example 2.5.

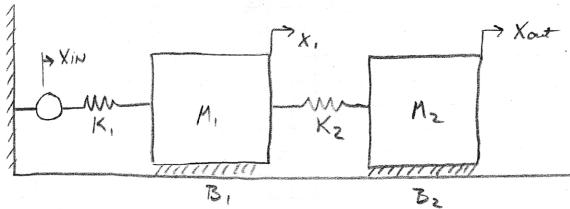
Dividing through top and bottom by M ,

$$G(s) = \frac{X(s)}{F(s)} = \frac{1/M}{s^2 + \frac{B_1+B_2}{M}s + K/M}$$

2.5 Examples

Example 2.7

Two sliding masses.



(Note that in this diagram, we used hatching between the mass and ground to indicate damping (\$B_1, B_2\$). This symbol is commonly used because it visually suggests sliding friction.

Find

$$G(s) = \frac{X_{out}(s)}{X_{in}(s)}$$

EOM:

$$M_1: M_1 \ddot{x}_1 + B_1(\dot{x}_1 - 0) + K_1(x_1 - x_m) + K_2(x_1 - x_{out}) = 0$$

$$M_2: M_2 \ddot{x}_{out} + B_2(\dot{x}_{out} - 0) + K_2(x_{out} - x_1) = 0$$

Laplace + Collect

$$M_1: X_1(s)(M_1 s^2 + B_1 s + K_1 + K_2) + X_{out}(s)(-K_2) + X_{in}(s)(-K_1) = 0$$

$$M_2: X_{out}(s)(M_2 s^2 + B_2 s + K_2) + X_1(s)(-K_2) = 0$$

Eliminate \$X_1(s)\$

$$\rightarrow X_1(s) = \frac{(M_2 s^2 + B_2 s + K_2) X_{out}(s)}{K_2}$$

$$\rightarrow X_{out}\left(\frac{1}{K_2}\right) \left[M_1 M_2 s^4 + M_1 B_2 s^3 + M_1 K_2 s^2 + M_1 B_1 s^3 + B_1 B_2 s^2 + (K_1 + K_2) M_2 s^2 + (K_1 + K_2) B_2 s + (K_1 + K_2) K_2 \right]$$

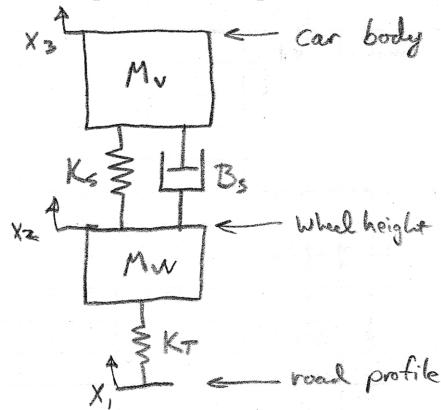
$$+ X_{out}(-K_2) = K_1 X_{in}(s)$$

$$\frac{X_{out}(s)}{X_{in}(s)} = \frac{K_1 K_2}{M_1 M_2 s^4 + (M_1 B_2 + M_1 B_1) s^3 + (M_1 K_2 + B_1 B_2 + (K_1 + K_2) M_2) s^2 + (B_1 K_2 + (K_1 + K_2) B_2) s + (K_1 + K_2) K_2 - K_2^2}$$

$$= \frac{\frac{K_1 K_2}{M_1 M_2}}{s^4 + \left(\frac{M_1 B_2 + M_1 B_1}{M_1 M_2}\right) s^3 + \left(\frac{K_2}{M_2} + \frac{B_1 B_2}{M_1 M_2} + \frac{(K_1 + K_2)}{M_1}\right) s^2 + \left(\frac{B_1 K_2 + (K_1 + K_2) B_2}{M_1 M_2}\right) s + \frac{K_1 K_2}{M_1 M_2}}$$

Example 2.8

Car Suspension Example

Find the transfer function from road ($x_1(s)$) to body ($x_3(s)$).

EOMs

$$M_w: M_w \ddot{x}_2 + B_s(\dot{x}_2 - \dot{x}_3) + K_s(x_2 - x_3) + K_T(x_2 - x_1) = 0$$

$$M_v: M_v \ddot{x}_3 + B_s(\dot{x}_3 - \dot{x}_2) + K_s(x_3 - x_2) = 0$$

Collect variables $x_3(s), x_2(s), x_1(s)$

$$M_w: M_w \ddot{x}_2 + B_s \dot{x}_2 + (K_T + K_s)(x_2) + B_s(-\dot{x}_3) + K_s(x_3) + K_T(-x_1) = 0$$

$$M_v: M_v \ddot{x}_3 + B_s \dot{x}_3 - B_s \dot{x}_2 + K_s(x_3 - x_2) = 0$$

Laplace Transform

$$M_w: X_2(s) (M_w s^2 + B_s s + K_T + K_s) + X_3(s) (-B_s s - K_s) + X_1(s) (K_T) = 0$$

$$M_v: X_3(s) (M_v s^2 + B_s s + K_s) + X_2(s) (-B_s s - K_s) = 0$$

Sub + eliminate $X_2(s)$

$$X_2(s) = X_3(s) \frac{(M_v s^2 + B_s s + K_s)}{B_s s + K_s}$$

$$\begin{aligned} X_3(s) \left(\frac{1}{K_s} \right) & (M_v M_w s^4 + M_w B_s s^3 + M_w K_s s^2 + M_v B_s s^3 + B_s^2 s^2 + B_s K_s s \\ & + (K_T + K_s) M_v s^2 + (K_T + K_s) B_s s + (K_T + K_s) K_s - B_s K_s - K_s^2) \\ & = K_T X_1(s) (B_s s + K_s) \end{aligned}$$

$$\begin{aligned} \frac{X_3(s)}{X_1(s)} & (M_v M_w s^4 + (M_v + M_w) B_s s^3 + (M_w K_s + B_s^2 + M_v (K_T + K_s)) s^2 \\ & + (B_s K_s + (K_T + K_s) B_s) s + (K_T + K_s) K_s - B_s K_s s - K_s^2) \\ & = \frac{B_s K_s s + K_s K_T}{K_T} \end{aligned}$$

Example 2.8 cont.

$$\frac{X_3(s)}{X_1(s)} = \frac{K_T B_s K_S s + K_S K_T}{M_v M_w s^4 + (M_v + M_w) B_s s^3 + (M_w K_S + B_s^2 + M_v (K_T + K_S)) s^2 + B_s K_S s + K_S K_T}$$

Normalize

$$= \frac{\frac{K_T B_s K_S s + K_S K_T}{M_v M_w}}{s^4 + \left(\frac{(M_v + M_w)}{M_v M_w} B_s\right) s^3 + \left(\frac{K_S}{M_v} + \frac{B_s^2}{M_v M_w} + \frac{K_T + K_S}{M_w}\right) s^2 + \frac{B_s K_S}{M_v M_w} s + \frac{K_S K_T}{M_v M_w}}$$

Chapter 3

Rotational Dynamical Systems

3.1 Problem Statement and Learning Objectives

- System Elements & constitutive relations.
- 5 • Equations of Motion
- Gears
- Conversion to Transfer Function

3.2 System Elements & constitutive relations.

Rotation is a different type of motion than translation and it makes subtle differences in dynamic analysis.
10 One of the most prominent difference is that if a body is rotating, every point in the body has a different velocity and acceleration. This complex situation can be considerably simplified by assuming a single axis of rotation, and representing a body by its *inertia* instead of its mass. The axis of rotation is a line along which points in a rigid body do not move when it is rotated about the axis.

15 Computation of the inertia of a rigid body is beyond the scope of this book, but it is a quantity which can be measured by rotational tests, or calculated from information such as a CAD model.

3.2.1 Torque

20 *Torque* (also called *moment*) is a vector quantity relating a force and an associated *moment arm* through which the force acts to rotate a body around an axis. The simplest case is a force which is perpendicular to both the axis of rotation and a radius connecting the axis and the point through which the force is acting on the rigid body (Figure 3.1, Left). In this case, the magnitude of the torque is

$$|\tau| = |r||F|$$

and the full magnitude and direction of the torque vector will be obtained by the right hand rule

$$\tau = r \times F$$

(where \times indicates the vector cross product.)

If the force vector is not applied at a right angle (Figure 3.1, Right), it must be resolved into perpendicular and radial components, F_p, F_r , and then the torque magnitude is

$$|\tau| = |r||F_p|$$

25 the full torque vector can still be obtained by the vector cross product above. When the axis of rotation is fixed, for example by a shaft mounted in bearings, then only the component of the torque vector which is parallel to the axis causes rotation about the axis.

In most of the problems we will study however, we will assume that a torque value is a known or measured quantity and not worry about the radius or moment arm. In a very common control system application, a
30 DC electric motor is applied to a shaft and the torque is simply proportional to the current

$$\tau(t) = K_m i(t)$$



Figure 3.1: An applied force F generates a torque if it acts through a point having a radius, r from the axis of rotation. Left: force is applied perpendicular to the moment arm. Right: force is applied in a general direction. (see text).

Name	Physical Realization	Symbol	Equation	Notes
Inertia	Flywheel		$\tau(t) = J\ddot{\theta}$	
Stiffness	Coil Spring		$\tau(t) = K\theta$	τ is same on both sides. Assume zero rest length
Damping	Fan, rotary damper, friction		$\tau(t) = B\dot{\theta}$	This is a linear model for friction.

Table 3.1:

3.2.2 Elements of Rotational Dynamical Systems

We will analyze systems consisting of

- **Inertia** The property of a rigid body which resists angular acceleration, and which stores kinetic energy.
- **Stiffness** The property of a rigid body which resists angular displacement, and which stores potential energy.⁵
- **Damping** The property of a rigid body which resists change in angular displacement and which converts motion to heat.

Some properties of the various elements are summarized Table 3.1.

3.3 Equations of Motion

- 10 Similarly to translational motion (see Equation 2.1), there is an Equation of Motion (EOM) for each inertia in the system:

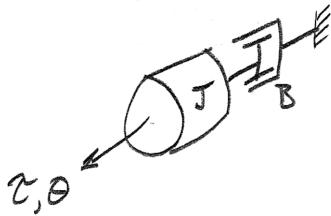
$$Jt\ddot{\theta} + \sum_j B_j(\dot{\theta} - \dot{\theta}_j) + \sum_k K(\theta - \theta_k)$$

The use of this EOM is similar to that of translational dynamical systems as illustrated in the following

examples

Example 3.1

Find the equation or equations of motion for the following system



There is one inertia (J) so there is only one EOM:

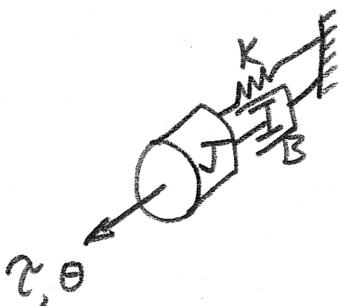
$$J\ddot{\theta} + B(\dot{\theta} - 0) = \tau(t)$$

or

$$J\ddot{\theta} + B\dot{\theta} = \tau(t)$$

Example 3.2

Find the equation or equations of motion for the following system



There is still only one EOM but it has an additional spring element:

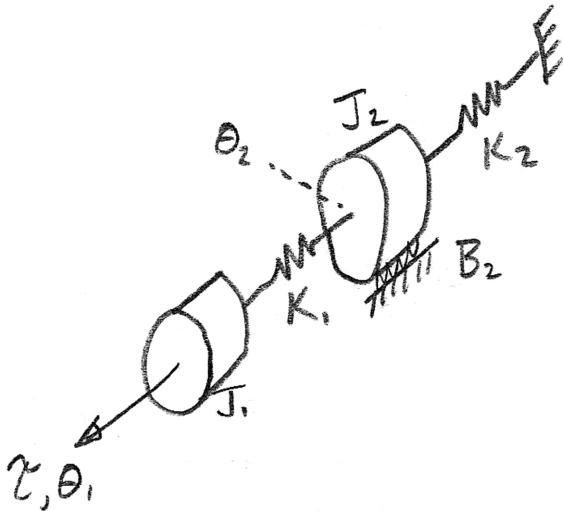
$$J\ddot{\theta} + B(\dot{\theta} - 0) + K(\theta - 0) = \tau(t)$$

or

$$J\ddot{\theta} + B\dot{\theta} + K\theta = \tau(t)$$

Example 3.3

Find the equation or equations of motion for the following system



This system has two masses. Although they appear to be on the same axis, they are separated by a spring and thus they can have different displacements depending on the deflection of the spring. As a result we have two EOM:

$$\begin{aligned} J_1 \ddot{\theta}_1 + K_1(\theta_1 - \theta_2) &= \tau(t) \\ J_2 \ddot{\theta}_2 + K_1(\theta_2 - \theta_1) + K_2\theta_2 + B_2\dot{\theta}_2 &= 0 \end{aligned}$$

These usually need to be solved simultaneously as with translational systems.

Once the OEMs are available, transfer functions can be derived in the same way as with translational systems.

⁵ 3.4 Gears



Figure 3.2: Meshing spur gears (<http://www.emersonindustrial.com/>).

3.4.1 Gear Kinematic Relationships

A common system element in rotary systems is gears. The corresponding element in translational systems, levers, seem to appear less often in control systems.

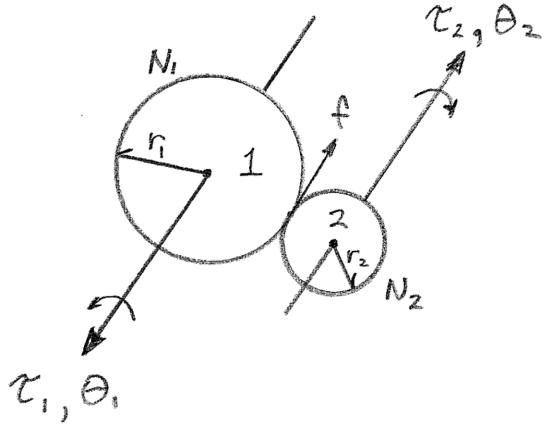


Figure 3.3: Two meshed gears.

Consider two meshed gears, gear 1 and gear 2 (Figure 3.3). Each gear has N_i teeth. The size of each tooth is $2\pi r_i/N_i$. The number of teeth which pass when a gear is rotated by θ_i is $N_i \frac{\theta_i}{2\pi}$. Since the teeth must be the same size for the gears to mesh, we can write

$$\frac{N_1 \theta_1}{2\pi} = \frac{N_2 \theta_2}{2\pi}$$

or

$$\frac{\theta_1}{\theta_2} = \frac{N_2}{N_1}$$

⁵ differentiating we also have

$$\frac{\dot{\theta}_1}{\dot{\theta}_2} = \frac{N_2}{N_1} \quad \frac{\ddot{\theta}_1}{\ddot{\theta}_2} = \frac{N_2}{N_1}$$

Commonly we define $n = N_1/N_2$. Thus

$$\dot{\theta}_2 = n\dot{\theta}_1$$

etc.

There is a force exerted by one tooth on the other in the tangential direction, f (Figure 3.3). Since it is tangential, we can relate it easily to the torques:

$$\tau_1 = r_1 f \quad \tau_2 = r_2 f$$

¹⁰ This gives

$$\begin{aligned} \tau_1 &= \frac{r_1}{r_2} \tau_2 = n \tau_2 \\ \tau_2 &= \frac{1}{n} \tau_1 \end{aligned}$$

Simplification of Geared Systems

We can use the properties of gear transmission of rotation and torque to simplify the process of writing EOM.

Consider a damper driven by a set of gears (Figure 3.4)

¹⁵ We have

$$\tau_2 = B\dot{\theta}_2$$

Using the relationships above we have

$$\frac{1}{n} \tau_1 = Bn\dot{\theta}_1$$

or

$$\tau_1 = Bn^2\dot{\theta}_1$$

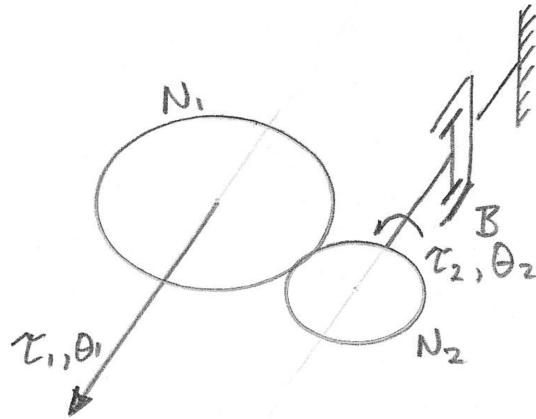


Figure 3.4: A viscous load (damper) driven by a set of gears.

Suppose the system “beyond” the gears had some mass and spring in addition to the damper of Figure 3.4. The argument above would be very similar:

We have

$$\tau_2 = J\ddot{\theta}_2 + B\dot{\theta}_2 + K\theta_2$$

Using the relationships above we have

$$\frac{1}{n}\tau_1 = Jn\ddot{\theta}_1 + Bn\dot{\theta}_1 + Kn\theta_1$$

⁵ or

$$\tau_1 = Jn^2\ddot{\theta}_1 + Bn^2\dot{\theta}_1 + Kn^2\theta_1$$

Let

$$\hat{J} = n^2 J \quad \hat{B} = n^2 B \quad \hat{K} = n^2 K$$

The EOM becomes

$$\tau_1 = \hat{J}\ddot{\theta}_1 + \hat{B}\dot{\theta}_1 + \hat{K}\theta_1$$

This is the EOM of a simpler system (Figure 3.5).

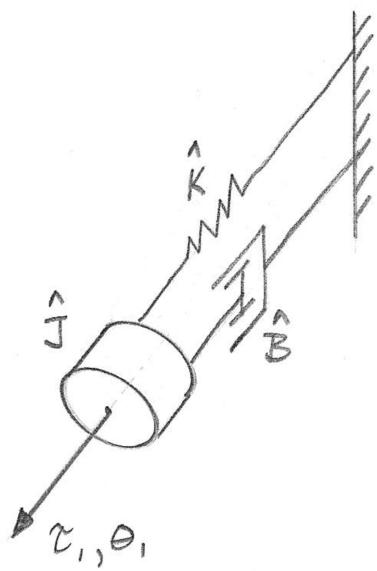
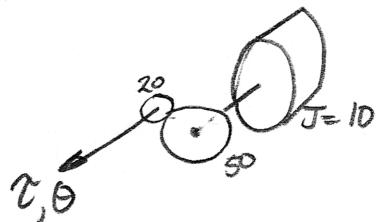


Figure 3.5: Simplified equivalent system of the system in Figure 3.4

Example 3.4

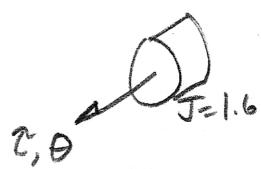
Transform the following geared system into an equivalent non/geared system and write the EOM.



$$n = \frac{20}{50} = 0.4$$

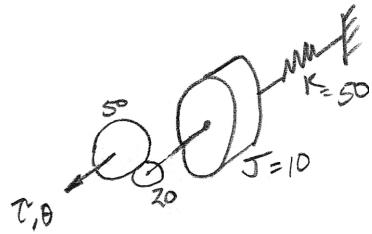
$$\hat{J} = 0.4^2 J = 0.16 \times 10 = 1.6$$

$$\tau = 1.6\ddot{\theta}$$



Example 3.5

Transform the following system into an equivalent system without gears

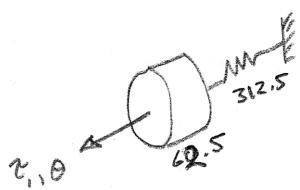


$$n = \frac{50}{20} = 2.5$$

$$\hat{J} = 2.5^2 \times 10 = 62.5$$

$$\hat{K} = 2.5^2 \times 50 = 312.5$$

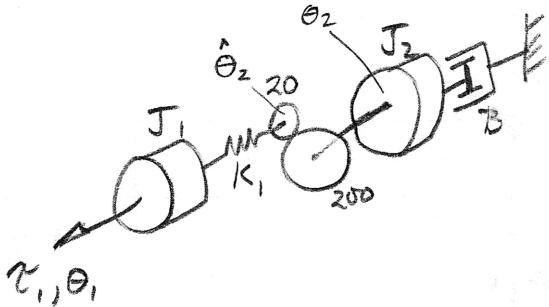
$$\tau = 62.5\ddot{\theta} + 312.5\theta$$



Example 3.6

Transform the following system with two rotational inertias and gears to eliminate the gears, and then write and solve EOMs to get the transfer function

$$G(s) = \frac{\theta_2(s)}{\tau(s)}$$



First, develop the transformations (by n^2) to change J_2 and B_2 so as to eliminate the gear set:

$$n = \frac{20}{200} = 0.1, \quad n^2 = 0.01$$

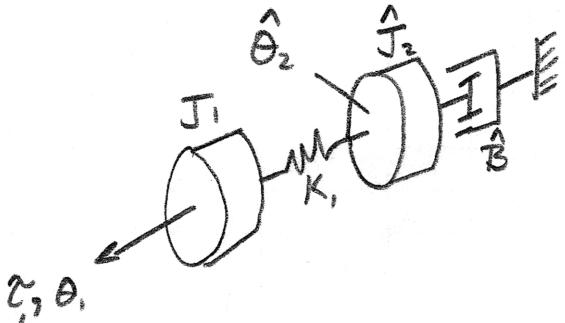
$$\hat{J}_2 = 0.01J_2 \quad \hat{B} = 0.01B$$

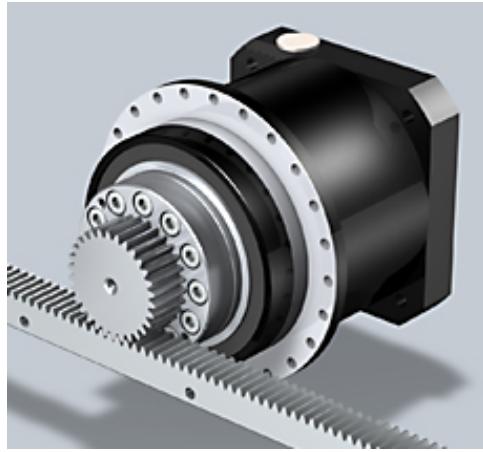
Also, the displacement of the second inertia is changed by

$$\hat{\theta}_2 = \frac{1}{n}\theta_2 = 10\theta_2$$

Note that the displacement θ_2 is transformed differently from the elements J_2, B_2 according to the derivations in Section 3.4.1. Also note that $\hat{\theta}_2$ is not the same as θ_1 because the spring K_1 can have an arbitrary deformation.

The transformed system is



Figure 3.6: Rack and Pinion drive system (Stober, www.stober.com)**Example 3.6 cont.**

Solving, using the techniques in Chapter 2:

EDM

$$\begin{aligned} J_1: \quad J_1 \ddot{\theta}_1 + K(\theta_1 - \hat{\theta}_2) &= \tau_{1(t)} \\ \hat{J}_2: \quad \hat{J}_2 \ddot{\hat{\theta}}_2 + K(\hat{\theta}_2 - \theta_1) + \hat{B}(\dot{\hat{\theta}}_2) &= 0 \end{aligned}$$

Laplace

$$\begin{aligned} J_1: \quad \Theta_1(s)(J_1 s^2 + K) + \hat{\Theta}_2(s)(-K) &= \tilde{\tau}_{1(s)} \\ \hat{J}_2: \quad \hat{\Theta}_2(s)(\hat{J}_2 s^2 + \hat{B}s + K) + \Theta_1(s)(-K) &= 0 \end{aligned}$$

Elim $\Theta_1(s)$

$$\begin{aligned} \Theta_1(s) &= \frac{1}{K} \hat{\Theta}_2(s) (\hat{J}_2 s^2 + \hat{B}s + K) \\ \hat{\Theta}_2(s) \left(J_1 \hat{J}_2 s^4 + J_1 \hat{B} s^3 + J_1 K s^2 + \hat{J}_2 K s^2 + K \hat{B} s + K^2 \right) - \hat{\Theta}_2(s) \frac{(-K)}{K} &= \tilde{\tau}_{1(s)} \end{aligned}$$

$$\hat{\Theta}_2(s) = \frac{K}{J_1 \hat{J}_2 s^4 + J_1 \hat{B} s^3 + (J_1 + \hat{J}_2) K s^2 + K \hat{B} s + K^2}$$

$$\frac{\hat{\Theta}_2(s)}{\tilde{\tau}_{1(s)}} = \frac{\frac{K}{J_1 \hat{J}_2}}{s^4 + \frac{\hat{B} s^3}{\hat{J}_2} + \frac{(J_1 + \hat{J}_2) K}{J_1 \hat{J}_2} s^2 + \frac{K \hat{B}}{J_1 \hat{J}_2} s}$$

converts $\hat{\theta}_2 \rightarrow \theta_2$

3.5 Rotary to Linear Motion

Sometimes the second gear in a chain is straightened out to $r_2 = \infty$. The case of infinite radius corresponds to what is called a *rack* - a set of gear teeth arrayed in a straight line. The gear which meshes with a rack is

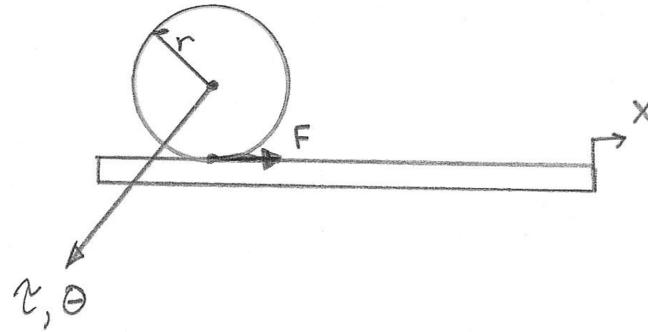


Figure 3.7: Rack and Pinion gear system converts rotary to linear motion and force to torque (and vice versa).

called a *pinion* (Figure 3.6). Such systems contain a combination of rotating and translating elements and they can be analyzed by careful application of the principles developed in this and the previous chapters.

Consider the rack and pinion shown in Figure 3.7. Assume the gear can rotate about its fixed axis and the rack is free to slide back and forth in the x direction. The force applied by the rack to the gear must be

$$F = \tau/r$$

- ⁵ because of the tangential contact constraint. The displacements are related by

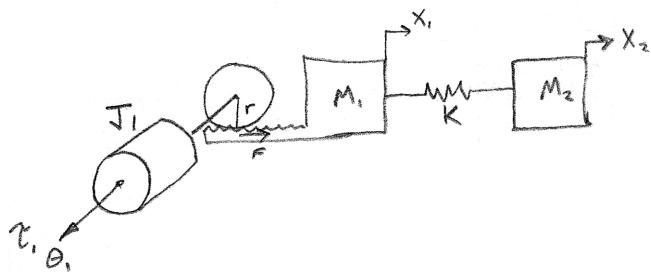
$$x = r\theta$$

by the basic geometry of circles.

In a combined system we write translational EOM(s) for the sliding components and rotational EOM(s) for the rotating components, but by substituting the relationships above, we can transform one of the EOMs so that both are in terms of rotary (or translational) variables.

Example 3.7

For the system below, find $\frac{\theta_1(s)}{\tau_1(s)}$



The initial EOMs are

$$J_1\ddot{\theta}_1 + rF = \tau_1$$

$$M_1\ddot{x}_1 + K(x_1 - x_2) = F$$

$$M_2\ddot{x}_2 + K(x_2 - x_1) = 0$$

$$\text{Combine 1 and 2: } J_1\ddot{\theta}_1 + rM_1\ddot{x}_1 + rKx_1 - rKx_2 = \tau_1$$

use

$$x_1 = r\theta \quad J_1\ddot{\theta}_1 + r^2M_1\ddot{\theta}_1 + r^2K\theta_1 - r^2K\theta_2 = \tau_1 \quad (1)$$

Convert to torque:

$$rM_2\ddot{x}_2 + rK(x_2 - x_1) = 0$$

$$x_1 = r\theta$$

$$r^2M_2\ddot{\theta}_2 + r^2K(\theta_2 - \theta_1) = 0 \quad (2)$$

$$\text{LT } (1) + (2), \text{ let } \hat{J}_2 = r^2M_2, \hat{K} = r^2K, \tilde{\tau} = \tau_1, \hat{\theta} = \theta_1 + \hat{\theta}$$

$$(1) \quad \Theta_1(s)(\tilde{\tau}s^2 + \hat{K}) - \Theta_2 \hat{K} = \tilde{\tau}_1(s)$$

$$(2) \quad \Theta_2(s)(\hat{J}_2 s^2 + \hat{K}) - \Theta_1 \hat{K} = 0$$

Example 3.7 cont.

Solve: eliminate θ_2 .

$$\textcircled{2} \Rightarrow \theta_2(s) = \frac{\hat{\theta}_1 K}{\hat{J}_2 s^2 + \hat{K}}$$

sub:

$$\theta_1(s)(\hat{J}s^2 + \hat{K}) - \frac{\hat{\theta}_1 K^2}{\hat{J}_2 s^2 + \hat{K}} = \tau_1(s)$$

$$\theta_1(s)(\hat{J}\hat{J}_2 s^4 + \hat{K}\hat{J}s^2 + \hat{K}\hat{J}_2 s^2 + \cancel{\hat{K}^2}) - \cancel{\hat{\theta}_1 \hat{K}^2} = \tau_1(s)(\hat{J}_2 s^2 + \hat{K})$$

$$\begin{aligned} \frac{\theta_1(s)}{\tau_1(s)} &= \frac{\hat{J}_2 s^2 + \hat{K}}{\hat{J}\hat{J}_2 s^4 + \hat{K}(\hat{J} + \hat{J}_2)s^2} \\ &= \frac{1}{\hat{J}} \frac{s^2 + \hat{K}/\hat{J}_2}{s^4 + \frac{(\hat{J} + \hat{J}_2)\hat{K}}{\hat{J}\hat{J}_2}s^2} \quad // \end{aligned}$$

$$\begin{aligned} \hat{K} &= r^2 K \\ \hat{J}_2 &= r^2 M_2 \\ \hat{J} &= r^2 M_1 + J_1 \end{aligned}$$

Chapter 4

Transient Response of 2nd Order Systems and Frequency Response

- This chapter starts with fairly detailed analysis of second order linear systems and the concepts of magnitude and frequency of steady state sinusoidal response. Computer techniques easily create highly accurate frequency response plots. We then develop techniques for hand drawing reasonably accurate frequency responses. While the hand techniques initially seem rather involved, after performing a few practice problems (and checking the results on the computer) they can be executed very rapidly with low effort.

4.1 Problem Statement and Learning Objectives

4.2 The basic 2nd order dynamical system

A dynamical system is said to be “second order” if the highest power of s in its denominator is 2. An example of such a system is

$$G(s) = \frac{1}{(s+a)(s+b)} = \frac{1}{s^2 + (a+b)s + ab}$$

$-a$ and $-b$ are roots of the polynomial in the denominator. When $s = -a$, the denominator = 0 and

$$G(-a) = G(-b) = \infty$$

Because $G(s)$ goes up to infinity at these locations, a, b , in the complex plane, we call $-a$ and $-b$ “poles”

of the transfer function $G(s)$.

Example 4.1

Find the poles of the transfer function and use the partial fraction expansion and the computer to evaluate the step response.

$$G(s) = \frac{1}{(s^2 + 13s + 30)}$$

By either simple factoring or use of the quadratic formula,

$$G(s) = \frac{1}{(s + 3)(s + 10)}$$

Thus the poles of $G(s)$ are $s = \{-3, -10\}$.

The partial fraction expansion is

$$G(s) = A_1/(s + 3) + A_2/(s + 10)$$

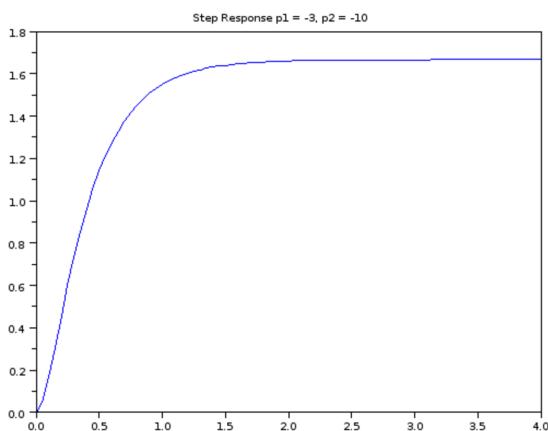
Solving for A_1, A_2 ,

$$A_1 = \quad A_2 =$$

Multiplying by the step input ($1/s$) and applying the inverse transform for $\frac{A}{(s+a)}$,

$$g(t) = 1.0 - A_1 e^{-3t} + A_2 e^{-10t}$$

Plotting the step response by computer:



Example 4.2

Find the poles of the transfer function,

$$G(s) = \frac{400}{12.7s^2 + 304.8s + 2641.6}$$

and plot the step response by computer.

Applying the quadratic formula to get the roots of the denominator:

$$\{a, b\} = \frac{-304.8 \pm \sqrt{304.8^2 - 4 \times 12.7 \times 2641.6}}{2 \times 12.7}$$

$$\{a, b\} = \{(-12 + j8), (-12 - j8)\}$$

In this case the poles are complex numbers.

Note that these numbers were kind of messy. This illustrates a good reason to normalize the denominator polynomial. Lets do the problem again but with a normalized denominator:

$$G(s) = \frac{3.15}{s^2 - 24s + 208}$$

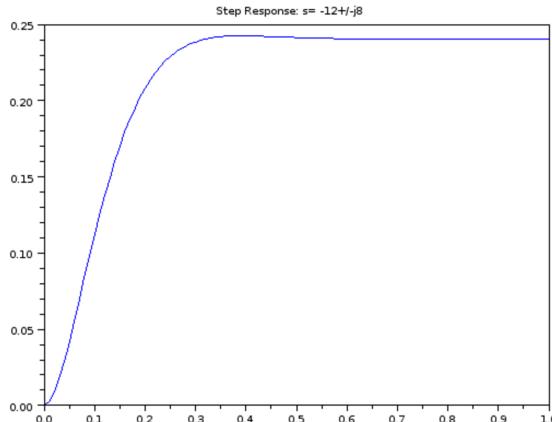
Applying the quadratic formula to get the roots of the denominator:

$$\{a, b\} = \frac{24 \pm \sqrt{24^2 - 4 \times 208}}{2}$$

This is simplified because the s^2 coefficient is one. The result is the same.

$$\{a, b\} = \{(-12 + j8), (-12 - j8)\}$$

Plotting the step response by computer:



There are lots of ways we could express the simple polynomial in the denominator of $G(s)$ and we'll play around with a few:

$$G(s) = \frac{1}{s^s + us + v} \quad (u = a + b, \quad v = ab)$$

$$G(s) = \frac{1}{s^2 + 2wxs + x^2} \quad (x = \sqrt{v}, \quad w = \frac{u}{2\sqrt{v}})$$

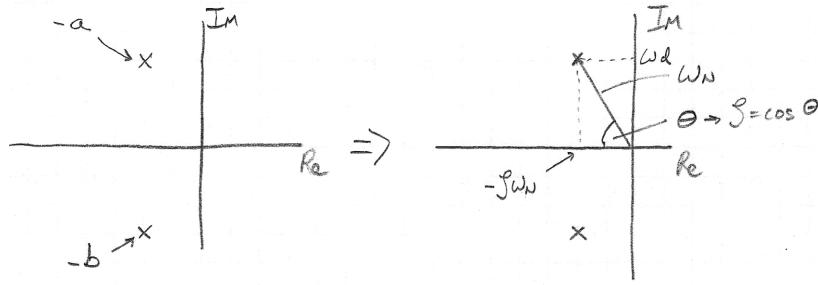


Figure 4.1: Complex conjugate poles in the complex plane can be represented in Cartesian Coordinates ($r + jm$) or polar coordinates $\{\zeta, \omega_n\}$

Finally, substituting some Greek letters which are classical in dynamics:

$$G(s) = \frac{1}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

$$\zeta(\text{"zeta"}) = x = \sqrt{ab}, \quad \omega_n(\text{"omega - n"}) = \frac{u}{2\sqrt{v}} = \frac{a+b}{2\sqrt{ab}}$$

So far this is just playing around with notation. The point of this is that ζ and ω_n , called the *damping ratio* and *natural frequency* respectively, give important insights not available from the poles (a, b) themselves.

In general the poles are complex, but if so, we know that they must be complex conjugates of each other. In other words,

$$\operatorname{Re}(a) = \operatorname{Re}(b) \quad \operatorname{Im}(a) = -\operatorname{Im}(b)$$

Using the properties of complex numbers, and the fact above, it is simple to show

$$\omega_n = \sqrt{\operatorname{Re}(a)^2 + \operatorname{Im}(a)^2} = |a| = |b|$$

$$\zeta = -\operatorname{Re}(a)/\omega_n$$

If we plot these points on the complex plane (Figure 4.1) we can see why these parameters give a different view from the poles.

4.2.1 Pole Location and Step Response

The location of poles in the complex plane determines the characteristics of the dynamic response to system inputs. One input which we care about a lot is the step function

$$u(t) = \begin{cases} 0 & t < 0 \\ 1 & t \geq 0 \end{cases}$$

The Laplace transform of the step input is $U(s) = \frac{1}{s}$.

The response to a step input determines what the system will do when we “change our minds” about what the output should be. For example, you walk into a room and turn the thermostat from 50° to 68° or you press “Resume” on your cruise control.

When we take a 2nd order transfer function of the form $G(s) = \frac{M}{(s+a)(s+b)}$, and multiply it by the step input, we get the output

$$Y(s) = \frac{1}{s} \frac{M}{(s+a)(s+b)}$$

we can expand it using the partial fraction expansion (Section 1.4) into the form

$$Y(s) = \frac{1}{s} + \frac{A_1}{(s+a)} + \frac{A_2}{(s+b)}$$

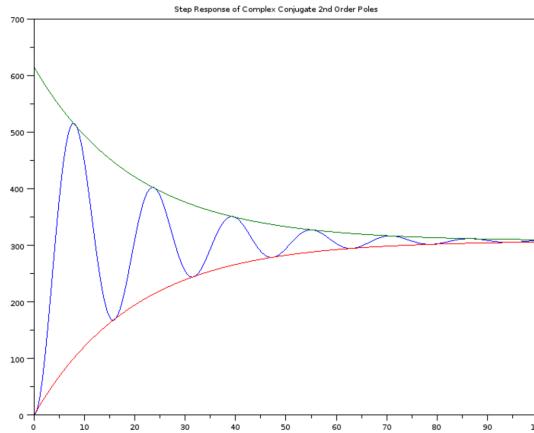


Figure 4.2: Step response of a typical system ($\omega_n = 0.4, \zeta = .124$) with complex conjugate 2nd order poles. The envelopes (red and blue) are also shown.

and further that the inverse transform of each term in the partial fraction expansion is

$$y(t) = 1 + A_1 e^{at} + A_2 e^{bt} \quad (t \geq 0) \quad (4.1)$$

It can be shown that for a 2nd order system with a pair of complex conjugate poles, $\{a, b\}$, this solution takes the form

$$y(t) = 1 - e^{-\zeta\omega_n t} (\cos(\omega_d t + \phi))$$

In this response, the exponential term *multiplies* the sinusoid term. Since the sinusoid cannot exceed the

- range $-1 \leq \sin(\omega t) \leq 1$, the response will be bounded by the exponential and we will call it the “envelope” since it “contains” the sinusoidal part of the response. Let’s look at some examples of this function. For $\omega_n = 0.4$, and $\zeta = .124$ (corresponding to the poles: $s = -0.05 \pm 0.4j$), we get the step response shown in Figure 4.2

- Now, let’s place these responses according to the location of their complex poles. Figure 4.3 represents the top half of the complex plane. Each plot is the response of a 2nd order system having one of its complex conjugate poles in its rough location. Note that the last two columns have positive real parts to their poles.

- Looking at Figure 4.3 there are two key patterns. First, the frequency of the sinusoid increases as the imaginary part of the pole increases. Second the envelope grows with time when the real part is positive, and shrinks over time when the real part is negative. The larger the magnitude of the real part, the faster the envelope changes.

4.3 Frequency Response

Another important way to analyze systems is in the frequency domain. In particular, we are interested in the steady state response when the system is driven by sinusoids of differing frequencies.

- Drawing frequency response plots by hand is still an important skill for control engineers. Hand drawing skill enables much deeper engineering insight into the behavior of systems and enables design work in a meeting or at the white-board. A hand sketch done prior to a computer plot gives you confidence that you entered everything correctly to generate the plot. Of course, our emphasis in hand drawing is on a balance favoring quick results which accurately plot the major qualitative features of the system. When detailed accuracy is required (later in the engineering cycle) we rely on the computer.

$$x(t) = \sin(\omega t) \Leftrightarrow X(s) = \frac{\omega}{s^2 + \omega^2}$$

- When a system is driven by a sinusoidal input, the output is derived by multiplying the Laplace transform of the sinusoid with the transfer function,

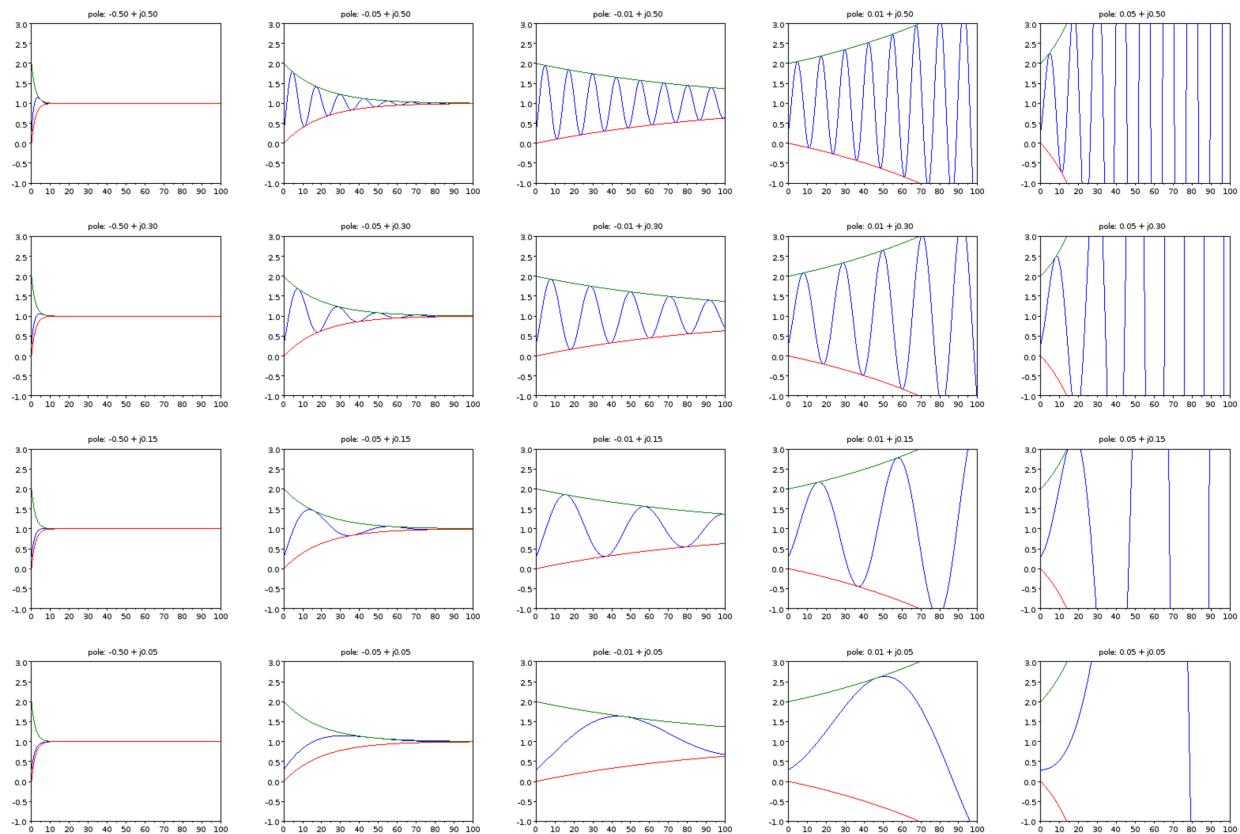


Figure 4.3: Step responses of 2nd order systems with the pole locations labeled above each plot. The array of sub-plots represents part of the upper half of the complex plane.

$$Y(s) = \frac{\omega}{s^2 + \omega^2} G(s)$$

The pole corresponding to the sinusoidal input is the root of $s^2 + \omega^2$ which is $s = j\omega$. Since the magnitude of $\sin(\omega t)$ is always 1 (i.e. does not vary with frequency, ω), the key quantity of interest is the magnitude of the transfer function, $|G(j\omega)|$ (which does vary with frequency). If the amplitude of the input sinusoid changes from

$$\sin(\omega t) \rightarrow A \sin(\omega t)$$

- 5 The frequency response can simply be scaled by A due to the linearity property.

$$|Y(j\omega)| = |G(j\omega)| \rightarrow |Y(j\omega)| = |AG(j\omega)| = A|G(j\omega)|$$

Thus we can focus on $|G(j\omega)|$ and get the response for any amplitude or frequency sinusoid.

We can show that the steady state output is also a sinusoid using the partial fraction expansion as we did above with the step response. Suppose

$$Y(s) = \frac{\omega}{s^2 + \omega^2} \frac{M}{(s + p_1)(s + p_2)(s + p_3)}$$

Then the partial fraction expansion would be

$$Y(s) = \frac{A_0}{s^2 + \omega^2} + \frac{A_1}{(s + p_1)} + \frac{A_2}{(s + p_2)} + \frac{A_3}{s + p_3}$$

- 10 The last three terms each transform into exponentials like Equation 4.1. We assume that the real part of each pole is negative so that the exponentials decay with time. We can thus neglect those terms since we are focused only on the steady state solution:

$$Y(s) = \frac{A_0}{s^2 + \omega^2}$$

$$y(t) = A/\omega \sin(\omega t + \phi)$$

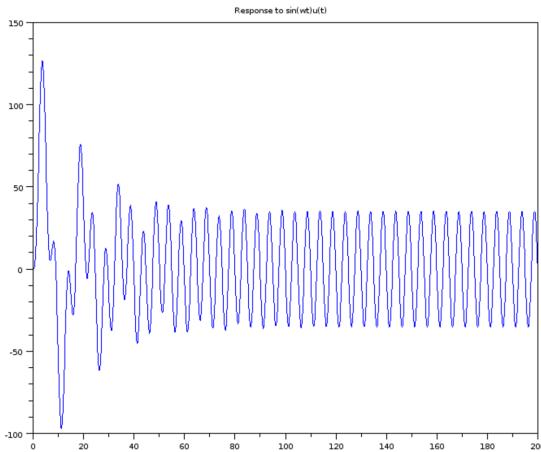
Where A and ϕ are quantities to be determined. This section is about efficient ways to determine how A and ϕ change as a function of ω .

Example 4.3

Obtain the sinusoidal steady state response of the following system

$$G(s) = \frac{50}{(s + 0.05 + j0.4)(s + 0.05 - j0.4)}$$

for $\omega = 1.25$ (the units of ω are radians per second).



15

There is a substantial transient response, but for $t > 80$ or so we see the steady state response.

Example 4.4

The system

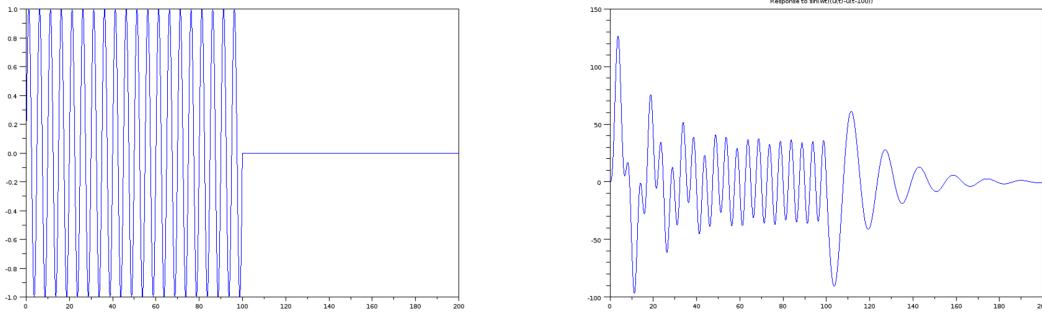
$$G(s) = \frac{50}{(s + 0.05 + j0.4)(s + 0.05 - j0.4)}$$

is driven by an interrupted sinusoidal function

$$x(t) = \sin(1.25t)(u(t) - u(t - 100))$$

Recall that $u(t)$, the unit step function, is required for single sided Laplace Transform analysis with zero initial conditions. The second term, $-u(t - 100)$, when combined with $u(t)$, “Turns off” the sinusoid at $t = 100$ because for $t > 100$, $u(t) - u(t - 100) = 0$. Numerically solving this system on the computer gives a response (below) which changes amplitude dramatically at both the turn ON transient ($t = 0$) and the turn OFF transient ($t = 100$), but settles to a constant sinusoidal output (the *steady state* response) for $80 < t < 100$. Note that if the sinusoid continued forever instead of shutting off at $t = 100$, the steady state response would also continue forever.

It is also worth noting that the frequency of the response changes when the input turns off. This is because the steady state response is a “forced” response (i.e. of the same mathematical form as the input), while the turn off transient is a “natural” response, i.e. determined by the ω_n, ζ of the system.



Left: Input signal. Right: System response includes transients both when the sinusoid turns ON ($t = 0$) and when it turns OFF ($t = 100$). Eventually the ON transient dies out to a steady state response ($75 < t < 100$).

Frequency response analysis ignores the transient response (both ON and OFF type) and focuses entirely on the forced, steady state response. Since the steady state response is always of the same mathematical form

- as the input, we need only concern ourselves with differences in amplitude and phase (between the input and output sinusoid).

Each root of the denominator of a transfer function is called a pole. Each root of the numerator is called a zero. Each pole and zero is a real or complex number which affects how the system response to both transient and steady state inputs.

10 4.3.1 Magnitude

As we focus on steady state sinusoidal response, we will concern ourselves only with the magnitude and phase of the response at each frequency.

In summary, the first task of frequency response analysis of a system described by $G(s)$ is to compute $G(j\omega)$ over some frequency range of interest, $\omega_{min} < \omega < \omega_{max}$. $|G(j\omega)|$ is computed by 1) plugging in

$s = j\omega$ and 2) evaluating the magnitude of each pole and zero term, and 3) combining the terms.

Example 4.5

Compute the magnitude of

$$G_1(j\omega) = \frac{10^5(s + 12.7)}{(s + 0.1)(s + 10)(s + 5000)}$$

for $\omega = 100$ rad/sec. Express the magnitude in dB.

Plugging in

$$|G_1(j100)| = \frac{10^5(12.7 + j100)}{(0.1 + j100)(10 + j100)(5000 + j100)}$$

Evaluating the magnitude of each term:

$$|G_1(j100)| = \frac{10^5(\sqrt{12.7^2 + 100^2})}{(\sqrt{0.1^2 + 100^2})(\sqrt{10^2 + 100^2})(\sqrt{5000^2 + 100^2})}$$

$$|G_1(j100)| = \frac{10^5(100.8)}{(100.000005)(100.5)(5001)}$$

Combining

$$|G_1(j100)| = 0.20056$$

Some observations about this computation follow:

1. The computations have been carried out to excessive precision. Practical control plants are rarely known to even 1% accuracy.
2. Another reason for excessive precision above is that we did not neglect any terms. In practice, a term like $|0.1 + j100|$ can be instantly replaced with 100 since we know the 0.1^2 is going to be insignificant. (Don't do this when the real and imaginary parts are close in magnitude!)
3. In an application where highly accurate numerical values must be obtained, we can use computer software. In modern engineering, hand calculations (even with a calculator) should only aim at quick approximate results.

4.3.2 Phase

⁵ Phase shift between the input and the output is represented by the angle of the complex transfer function evaluated at frequency ω . Phase is computed by a similar procedure to magnitude: 1) plugging in $s = j\omega$

and 2) evaluating the phase of each pole and zero term, and 3) combining the terms.

Example 4.6

Compute the phase angle of

$$G_1(j\omega) = \frac{10^5(s + 12.7)}{(s + 0.1)(s + 10)(s + 5000)}$$

for $\omega = 100$ rad/sec. Express the angle in degrees.

Plugging in

$$\angle G_1(j100) = \angle \frac{10^5(12.7 + j100)}{(0.1 + j100)(10 + j100)(5000 + j100)}$$

Evaluating the angle of each term:

$$\angle G_1(j100) = \frac{\tan^{-1}(0/10^5)(\tan^{-1}(100/12.7))}{(\tan^{-1}(100/0.1))(\tan^{-1}(100/10))(\tan^{-1}(100/5000))}$$

$$\angle G_1(j100) = \frac{0^\circ + 82.76^\circ}{89.94^\circ + 84.30^\circ + 1.146^\circ}$$

Combining

$$\angle G_1(j100) = 82.76^\circ - 89.94^\circ - 84.30^\circ - 1.146^\circ = -92.626^\circ$$

The observations of Example 4.5 apply almost exactly to the phase computation as well.

4.3.3 Decibels

- ⁵ Decibels are a logarithmic unit which are widely used for the analysis of frequency response. If we have a quantity, x , then

$$dB(x) = 20 \log_{10}(x)$$

where $dB(x)$ represents the decibel representation of x .

Because decibels are logarithmic units, we will make frequent use of the following properties (which are easily proved using basic properties of logarithms)

$$dB(ab) = dB(a) + dB(b)$$

¹⁰

$$dB(a/b) = dB(a) - dB(b)$$

$$dB(\sqrt{a}) = \frac{dB(a)}{2}$$

etc.

Example 4.7

Convert the following quantities to dB .

$$dB(1000) = 20 \log(1000) = 20 * 3 = 60dB$$

$$dB(6000) = 20 \log(1000 \times 6) = dB(1000) + dB(6) = 60 + 15.6 = 76.6dB$$

$$dB(X/100) \quad (\text{where } X = 40dB) = 40 - 20 \log(100) = 40 - 40 = 0dB$$

Example 4.8

Taking into account the points raised in Example 4.5, Let's redo the magnitude computation and get a quick approximate answer:

Plugging in

$$|G_1(j100)| = \frac{10^5(12.7 + j100)}{(0.1 + j100)(10 + j100)(5000 + j100)}$$

Quickly evaluating the magnitude of each term by neglecting the small parts :

$$|G_1(j100)| = \frac{10^5(100)}{(100)(100)(5000)}$$

Converting to dB

$$|G_1(j100)| = 100dB + 40dB - 2 * 40dB - 60dB - 20\log(5) = -14dB$$

Example 4.9

Convert the magnitude calculation of Example 4.5 to dB and compare it with Exercise 4.8.
Converting to dB :

$$dB(2.0056) = -14dB$$

which is the same result.

4.3.4 Bode Plot Sketching

The Bode asymptotic magnitude plot (named after famous Bell Labs engineer Hendrik Bode, 1905-1982) is a log-linear plot of magnitude vs. frequency, and is usually used with the Bode asymptotic phase plot which is a linear-linear plot of phase vs. frequency. Bode's key contribution was to understand that the key properties of $|G(j\omega)|$ and $\angle G(j\omega)$ can be obtained by sketching straight line asymptotes which are easily identified from

the transfer function. In the 1930s, this replaced hours of tedious hand calculation. Today, we get an accurate Bode plot from the computer in seconds, but the asymptotic hand sketch has two key remaining insights:

1. We can quickly estimate frequency response on the back of an envelope or at the white-board during group work.
2. We gain key insights about which parameters of the transfer function are responsible for which features of the frequency response.

Bode Asymptotic Magnitude Plot (BAMP)

Single Pole We start by looking at a simple transfer function consisting of one pole:

$$G(s) = \frac{1}{(s + a)}$$

In all frequency response analysis we assume that $Re(p) < 0$. For now we assume $Im(p) = 0$. We consider $s = j\omega$ and there are three values of ω which are relevant.

- 20 1. $\omega \ll |a|$
2. $\omega = |a|$
3. $\omega \gg |a|$

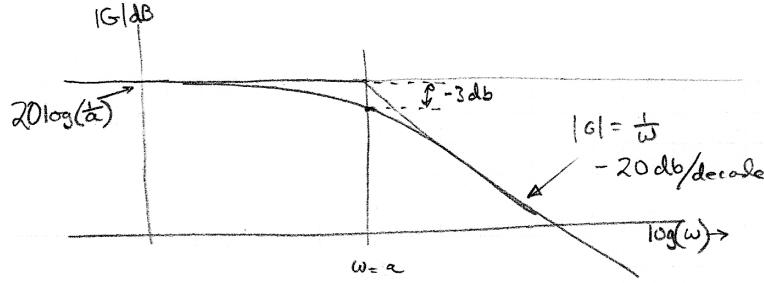


Figure 4.4: Bode Magnitude Plot of a single pole.

Ranges 1 and 3 are “asymptotic” because they become more and more true as $|\omega| \rightarrow \infty$. Value 2 is an exact value because we can easily compute an “anchor point” for the graph. For each region, as we plug in $s = j\omega$, we can approximate $|G(j\omega)|$ as

$$1. |G(j\omega)| \approx 1/a$$

$$5. |G(j\omega)| = \left| \frac{1}{a+ja} \right| = \frac{1/a}{\sqrt{2}}$$

$$3. |G(j\omega)| \approx \left| \frac{1}{j\omega} \right| = \frac{1}{\omega}$$

Importantly, the Bode Magnitude plot is logarithmic in the Magnitude and we express Magnitude in dB . Therefore we can re-write the approximations as

$$1. |G(j\omega)| \approx -dB(a)$$

$$10. |G(j\omega)| = \left| dB\left(\frac{1}{a+ja}\right) \right| = -dB(a) - 3dB$$

$$3. |G(j\omega)| \approx \left| dB\left(\frac{1}{j\omega}\right) \right| = -dB(\omega)$$

If we plot this we get Figure 4.4.

It's important to note a couple of things.

1. The low frequency asymptote is horizontal because it is constant with respect to ω ($-dB(a)$).
2. The high frequency asymptote intersects the low frequency asymptote at $\omega = a$, $|G(ja)| = -dB(a)$.
3. The actual curve is smooth and intersects the point $\{a, -dB(a) - 3dB\}$ in accordance with the calculation for $\omega \approx a$.
4. As omega gets greater than a , the magnitude drops with frequency according to

$$M = -dB(\omega)$$

In log-log coordinates, $-dB(\omega)$ is a straight line with a slope of -1 . When we say “slope of -1 ”, we mean the magnitude drops a factor of 10 for every factor of 10 increase in ω . We express this slope as $\frac{-20\text{dB}}{\text{decade}}$. The term ”decade” refers to an order of magnitude change of frequency.

Single Zero Now we consider the case of a system represented by

$$G(s) = \frac{(s + a)}{1}$$

As before, we assume $\text{Re}(p) < 0$. For now we assume $\text{Im}(p) = 0$. The same three ranges of ω are relevant:

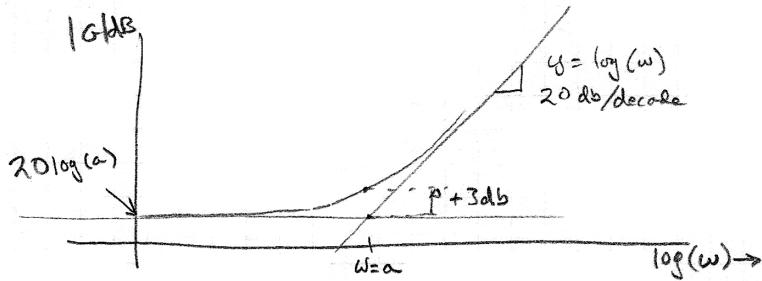


Figure 4.5: Bode Magnitude Plot of a single zero.

1. $\omega \ll |a|$
2. $\omega \approx |a|$
3. $\omega \gg |a|$

For each region, as we plug in $s = j\omega$, we can approximate $|G(j\omega)|$ as

- 5
 1. $|G(j\omega)| \approx a$
 2. $|G(j\omega)| \approx |a + ja| = \sqrt{2}$
 3. $|G(j\omega)| \approx |j\omega| = \omega$

Again, the Bode Magnitude plot is logarithmic in the Magnitude and we express Magnitude in dB . Therefore we can re-write the approximations as

- 10
 1. $|G(j\omega)| \approx dB(a)$
 2. $|G(j\omega)| \approx \left| dB\left(\frac{1}{a+ja}\right) \right| = +3dB$
 3. $|G(j\omega)| \approx \left| dB\left(\frac{1}{j\omega}\right) \right| = dB(\omega)$

If we plot this we get Figure 4.5.

The zero plot is identical to the pole plot except inverted (reflected around the line $|G(j\omega)| = 0dB$). The slope of the high-frequency asymptote is now $+1$ or $+20dB/\text{decade}$.

15 Note that these graphs are generic for any value of a . If they are multiplied by any different amplitude, A , then they can be decomposed as follows:

$$|G_A(s)| = \left| \frac{A}{(s+a)} \right| = |A| \left| \frac{1}{(s+a)} \right| = dB(A) + dB\left(\left| \frac{1}{(s+a)} \right| \right)$$

In other words they are shifted up or down by $dB(A)$.

Example 4.10

Plot the Bode Asymptotic Magnitude Plot for the following single-pole transfer function:

$$G_2 = \frac{2000}{(s + 200.0)}$$

As above we can decompose this into

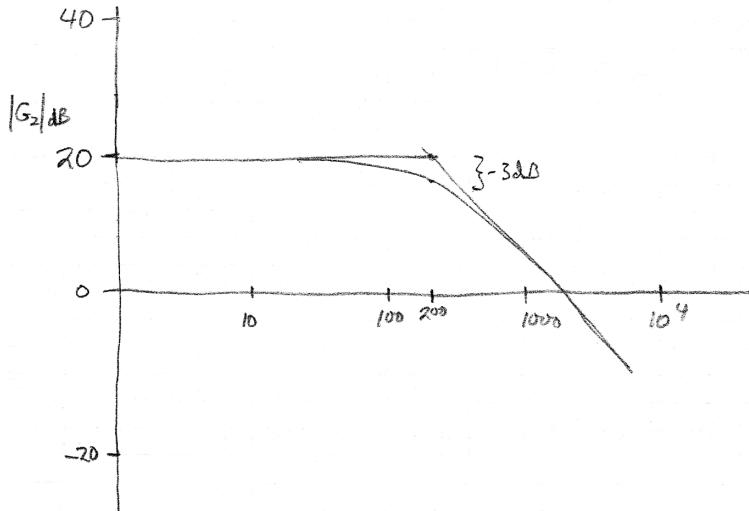
$$|G_2| = dB(2000) + dB(|1/(s + 200)|)$$

Thus the Bode plot of Figure 4.4 directly applies as long as we add $dB(2000) = 66dB$ and we have

1. The intersection of the two asymptotes is at $\omega = 200$.
2. The low frequency (horizontal) asymptote is at

$$|G_2(\omega)| = 66dB + (-46dB) = 20dB$$

Where $66dB$ comes from the factor of 2000, and $-46dB$ comes from $20 \log(1/a)$. Drawing the asymptotes and drawing the smooth curve through $20dB - 3dB$,



4.3.5 Combining Magnitude Plots

- 5 Consider the more realistic transfer function which has one zero and two poles:

$$G_3(s) = \frac{(s + b)}{(s + a)(s + c)}$$

To make a Bode Asymptotic Magnitude plot of this more interesting function, we recognize that it is the product of two poles and one zero:

$$G_d(s) = \frac{1}{(s + b)} \frac{(s + a)}{1} \frac{1}{(s + c)}$$

and since we are plotting in a dB scale that

$$dB(G_3(s)) = dB\left(\frac{1}{(s+b)}\right) + dB\left(\frac{(s+a)}{1}\right) + dB\left(\frac{1}{(s+c)}\right)$$

In other words, we can just add the three Bode plots together. This is a valid way to do it but is still a bit time consuming because four total plots have to be made. To find a simpler way let's constrain the first asymptotic frequency range slightly so that it is below the lowest pole, i.e.

$$\omega << \min(a, b, c)$$

⁵ For this case

$$|G_3(\omega)| = dB(a) - dB(b) - dB(c)$$

at this point we know where the low frequency (horizontal) asymptote intersects the dB axis. An important way to look at the basic plots of Figures 4.4 and 4.5 is that they are horizontal for $\omega < a$ and sloped (either down for poles or up for zeros) for $\omega > a$. Thus, the quickest way to draw the Bode Asymptotic Magnitude plot is to start from the horizontal asymptote and then to add in a component of slope as ω gets to each pole or zero.
¹⁰

Example 4.11

Hand draw the Bode Asymptotic Magnitude Plot for

$$G_4(s) = \frac{(s + 0.1)}{(s + 2)(s + 25)}$$

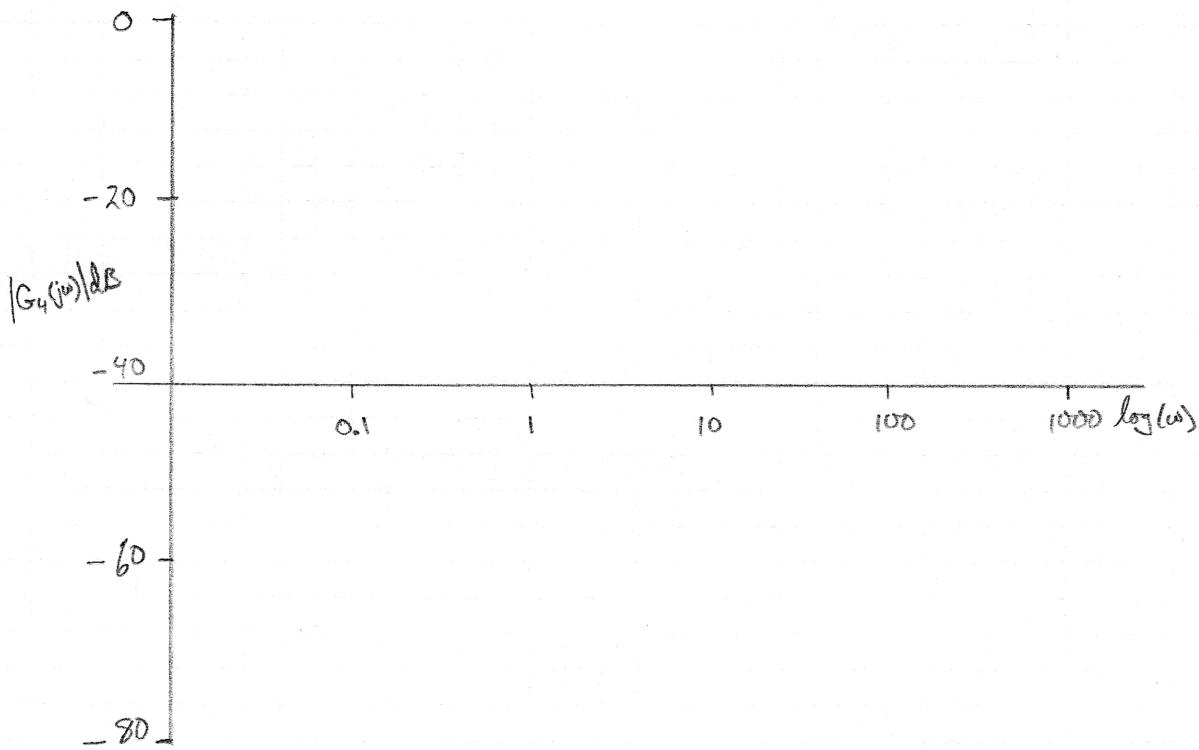
for the frequency range 0.01 — 1000 rad/sec.

Follow these steps:

- 1) Compute the magnitude for $\omega \ll \min(0.1, 2, 25)$ i.e. $\omega \ll 0.1$.

$$|G_4(s)| = dB(0.1) - dB(2) - dB(25) = -20dB - 6dB - 28dB = -54dB$$

- 2) Draw your log-log axes.



Notes about the axes:

1. It is *important* to make the size of a factor of 10 the same on both horizontal and vertical axes. Verify that this property holds for the above axes (keeping in mind that $20dB = 20\log(10)$).
2. It can be tricky to choose the ranges of dB and ω to plot.

For ω start with $\omega_{min} = 0.1 \min(p_i, z_i)$ (where p_i and z_i are your poles and zeros) and $\omega_{max} = 10 \max(p_i, z_i)$.

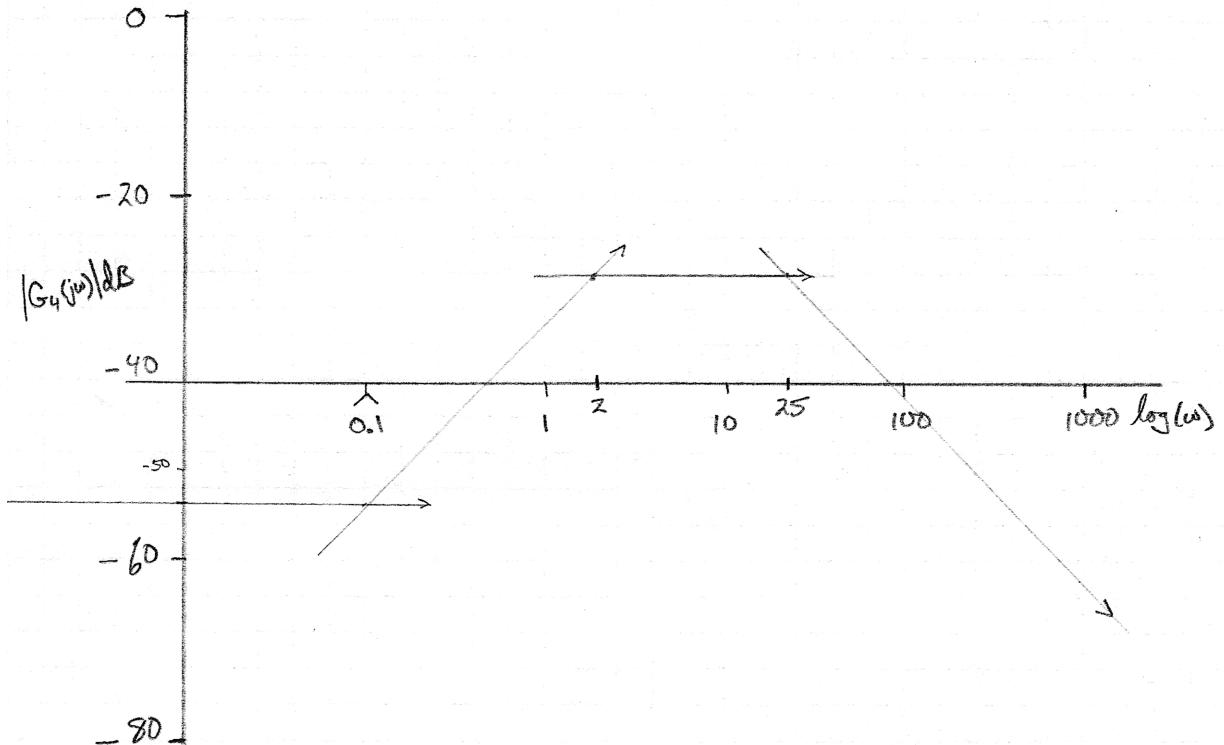
For dB range, try to guess if your plot will slope upward (zeros first) or downward (zeros at higher frequencies than poles). Then place your value obtained in step one near the bottom or top of the range respectively.

Example 4.11 cont.

3) Mark the zero and pole frequencies on the $\log(\omega)$ scale. Don't forget to take the log of ω before you mark it down.

4) Starting from the left, draw the low frequency horizontal asymptote (at $-54dB$) to the left through $\omega = 0.1$. At that frequency, the slope changes to positive due to the zero so draw an asymptote intersecting the horizontal at $\omega = 0.1$ and extending up at $+45^\circ$.

Next at $\omega = 2$, the first pole becomes active. This pole contributes a negative slope and thus cancels the positive slope. Draw a new horizontal asymptote (slope = 0) intersecting the upward line at $\omega = 2$. Finally, at $\omega = 25$, we have two negative slopes due to the two poles and still the one positive slope from the zero. The net result is -1 slope for frequencies above 25.

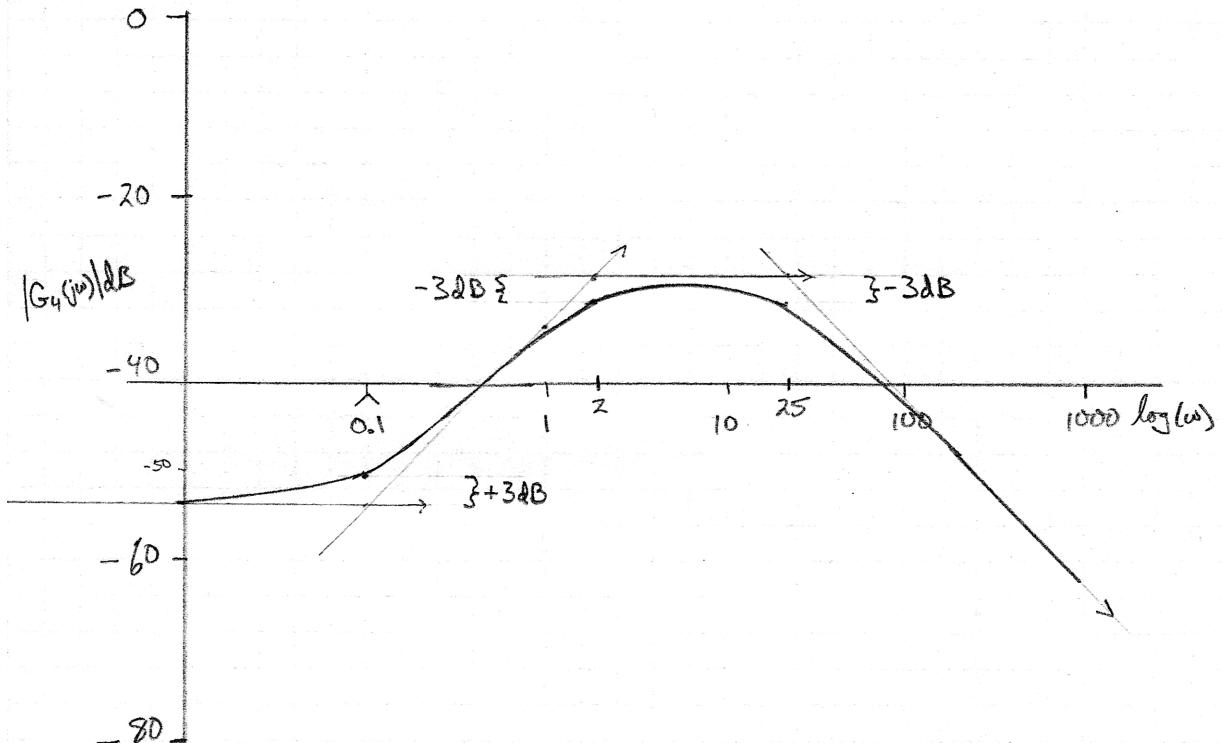


5) Mark corrections: $-3dB$ at $\omega = p_i$, $+3dB$ at $\omega = z_i$

6) Draw a smooth curve through each $3dB$ point. For a pole, start at $p_i/10$, draw a smooth curve through the $-3dB$ correction, and merge smoothly back into the asymptote at $10p_i$. Some artistic skill helps here. When poles or zeros are closer than a factor of 10 apart, you have to blend the smooth curves. See the range $2.0 < \omega < 25$ in the final plot below.

Example 4.11 cont.

The final plot looks like:



4.3.6 Bode Asymptotic Phase Plot

We will derive the Bode Asymptotic Phase Plot the same way as the magnitude: by considering three values of ω relative to the pole or zero.

Single Pole Again, we start by looking at a simple transfer function consisting of one pole:

$$G(s) = \frac{1}{(s + a)}$$

In all frequency response analysis we assume that $\text{Re}(p) < 0$. For now we assume $\text{Im}(p) = 0$. We consider $s = j\omega$ and there are three values of ω which are relevant.

- 1. $\omega \ll |a|$
- 2. $\omega = |a|$
- 3. $\omega \gg |a|$

For each region, as we plug in $s = j\omega$, we can approximate $|G(j\omega)|$ as

- 1. $\angle G(j\omega) \approx 0$
- 2. $\angle G(j\omega) = \angle \frac{1}{a+j\omega} = -45^\circ$
- 3. $\angle G(j\omega) \approx \angle \frac{1}{j\omega} = -90^\circ$

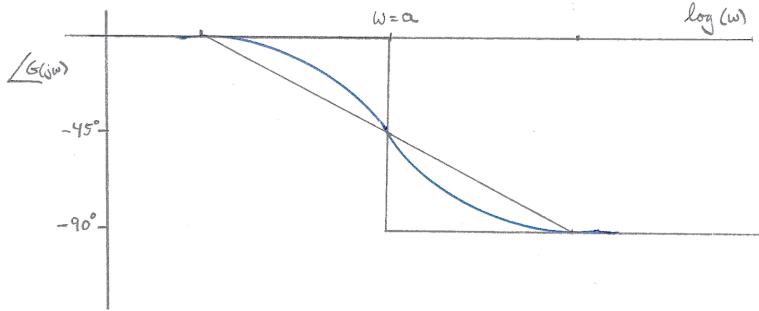


Figure 4.6: Bode Phase Plot of a single pole. Three approximations of increasing accuracy are given. 1) Straight line (step) asymptotes. 2) linear approximation between $0.1a < \omega < 10a$, 3) Smooth curve passing through 45° at $\omega = a$.

The Bode phase plot uses a linear vertical axis for the phase angle (but uses the same $\log(\omega)$ horizontal axis).

If we plot this we get Figure 4.6. The figure shows three increasingly accurate approximations to the true phase curve. Based on the asymptotic approximations above, we get asymptotes which look like a step function which changes from 0° to -90° as ω increases past a . A better approximation is a linear relationship passing through the points

$$\{\omega = 0.1a, \phi = 0^\circ\}, \{\omega = a, \phi = -45^\circ\}, \{\omega = 10a, \phi = -90^\circ\}$$

Finally, by making a smooth curve first above, and then below the linear approximation we can get quite close to a numerically accurate phase curve. In manual plotting, the intent is not high numerical accuracy, just quick insight. For precise phase curves, the computer is better.

- ¹⁰ **Single Zero** By very similar arguments, you can show that a zero such as

$$G(s) = (s + a)$$

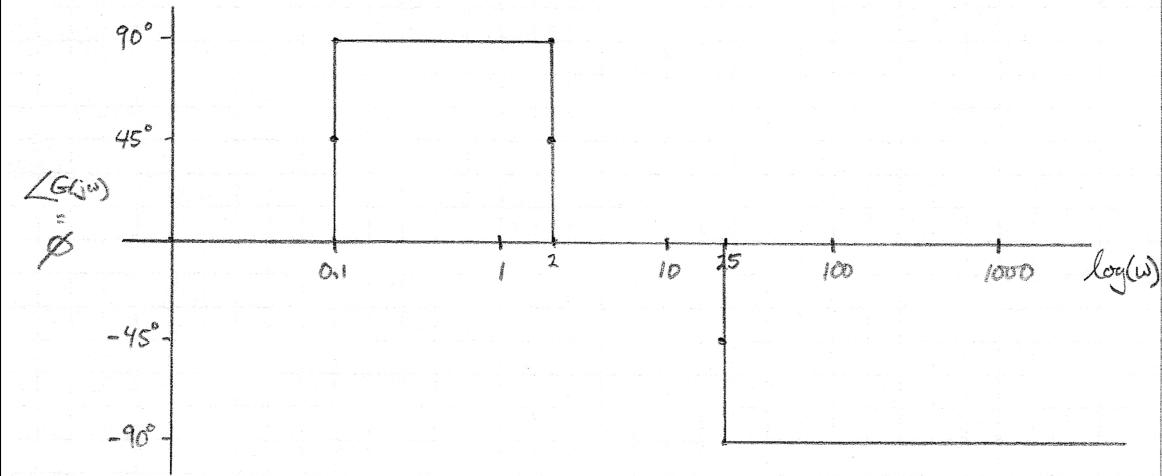
Contributes the same type of phase curve except flipped above the 0° horizontal axis.

Combining Phase Curves Just as we can add the Bode Asymptotic Magnitude plots of several poles and zeros (due to the $\log()$ nature of dB), we can add asymptotic phase curves from the different poles and zeros of a transfer function because the angles of two complex numbers add together when you multiply them.

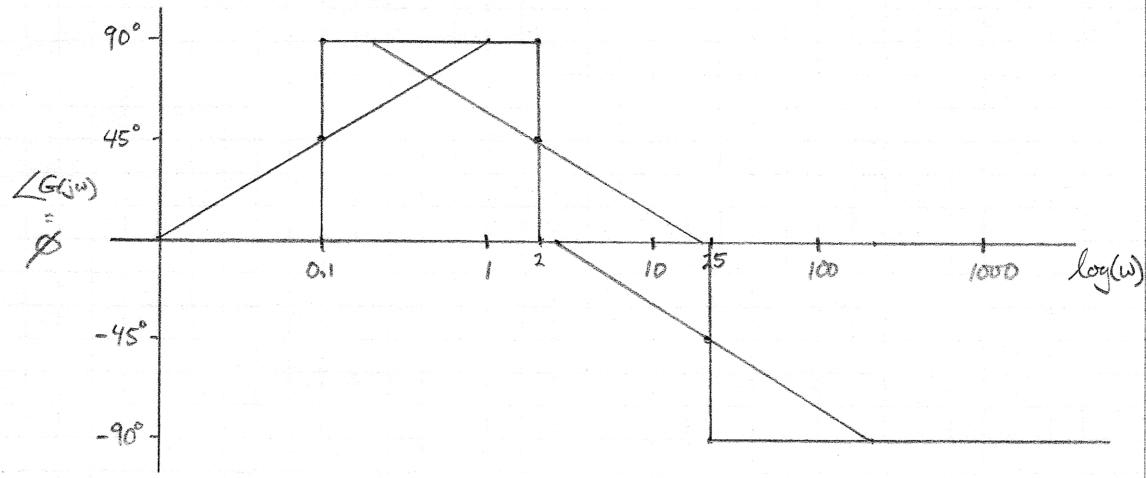
Example 4.12

Draw the Bode Asymptotic Phase Plot for the system of Example 4.11.

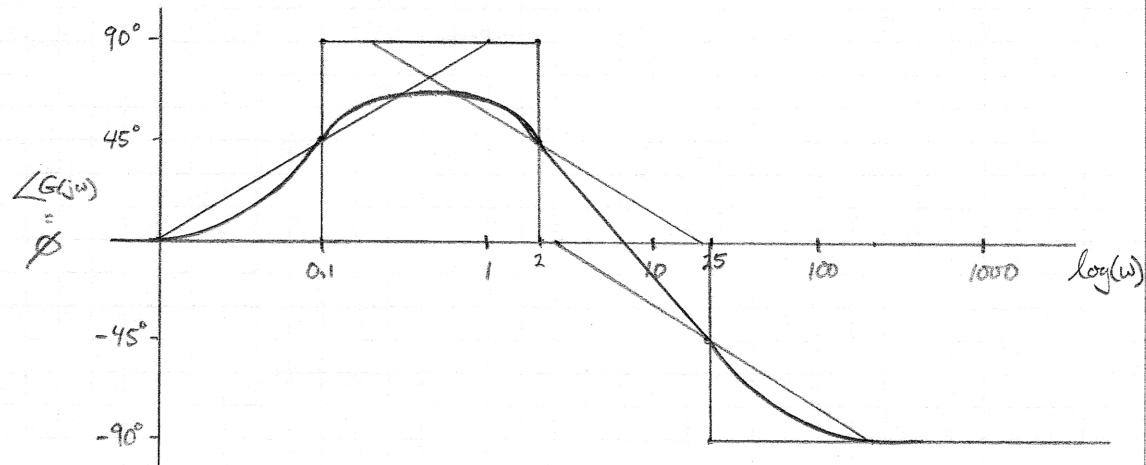
For $\omega < 0.1$ the two poles and the zero each contributes 0° . The zero will begin first and contribute $+90^\circ$, then, as ω increases, the poles will each contribute -90° . Starting from the left at $\phi = 0^\circ$, and drawing only the “step function” asymptotes,



Adding in the linear approximations



and finally, using a bit of artistic license, the smooth curves can be drawn in:



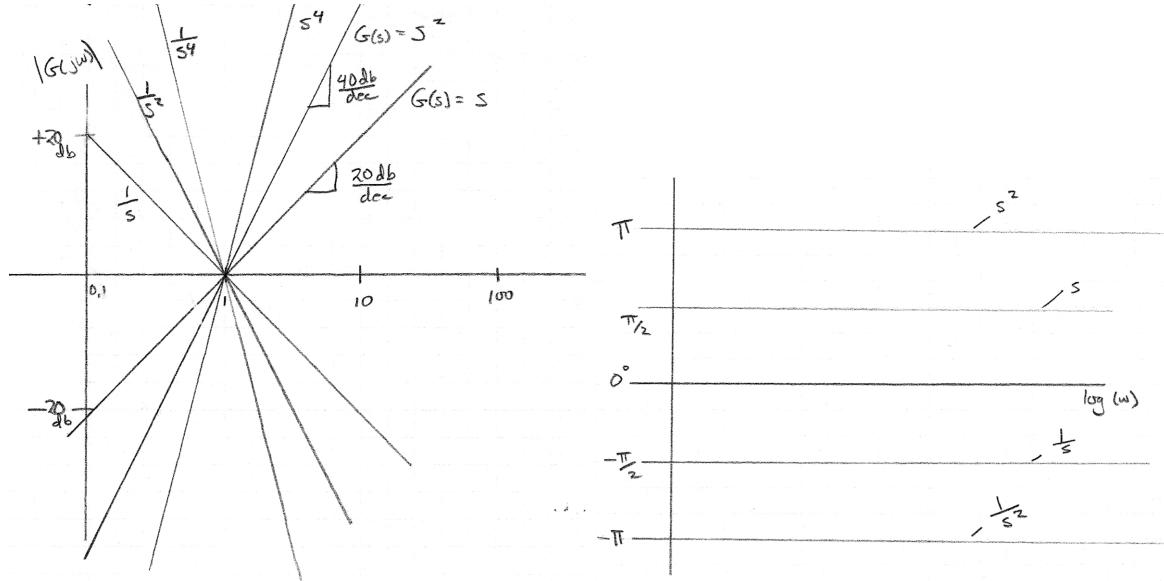


Figure 4.7: Bode Magnitude and Phase Plots of $G(s) = s^n$. Left: The magnitude plots all pass through the point $\{1, 0\}$. Right: Each term of s^n contributes $n \times 90^\circ$.

4.3.7 Poles or zeros at the origin

It is slightly trickier to draw the Bode plots when there are one or more poles at $s = 0$ because the first asymptotic approximation $\omega \ll a$ does not apply. In this case, there is a multiplicative term $1/s^n$, where n is negative for poles at $s = 0$ and positive for zeros at $s = 0$. For each value of n , the slope changes but in all cases

$$|s^n| = 1 \quad \text{for } s = j1$$

Figure 4.7 (Left) shows several Bode Magnitude plots of s^n . Each plot is a straight line passing through the point, $\{0 \text{ dB}, \omega = 1\}$.

When a transfer function containing various poles and zeros also has poles or zeros at $s = 0$, sketching the Bode Asymptotic Magnitude Plot is slightly different because it does not begin with a horizontal asymptote at small values of ω ($\omega \ll \min(p_i, z_i)$).

For transfer functions containing poles or zeros at the origin, we need to choose a specific frequency (often $\omega = 1$) at which to evaluate the magnitude of the transfer function. We also compute the slope of the low-frequency asymptote at $\omega \approx 0$ by looking at the exponent of the pole or zero at zero. The phase contribution of poles or zeros at the origin (Figure 4.7, Right) is easier because it is just a constant 90° for each power of s in the numerator and -90° for each power of s in the denominator.

Example 4.13

Sketch the BAMP of the following transfer function

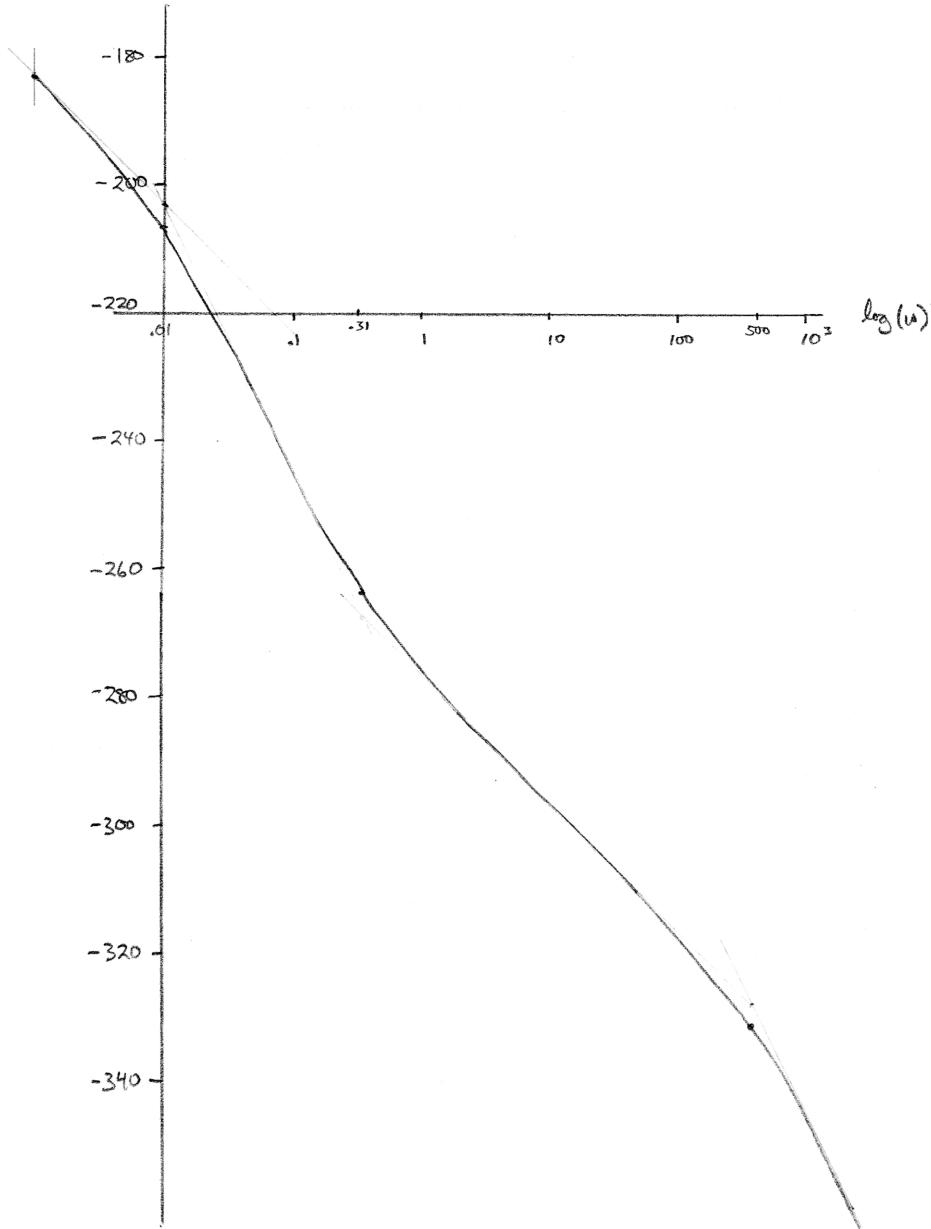
$$G(s) = \frac{(s + 0.31)}{s(s + 10^{-2})(s + 500)}$$

First, we note that there is a single pole at $s = 0$. In order to find the vertical range of the plot, we need to compute the magnitude at some frequency below the non-zero poles and zeros. Since the smallest feature is $s = 0.01$, we choose $s = 10^{-3}$. Computing the magnitude (to one or two significant figures)

$$|G(j0.001)| \approx \frac{0.31}{10^{-3} \times 1.1 \times 10^{-2} \times 500}$$

$$|G(j0.001)| \approx -10dB - 60dB - 39dB - 74dB = -183dB$$

The slope at this frequency is going to be $-20dB/\text{decade}$ (Figure 4.7, Left) because the non-zero poles and zero contribute this slope at $\omega = 10^{-3}$. The magnitude will go down from there as the poles become active. Plotting from the point $\{0.001, -183dB\}$ with slope $-20dB/\text{dec}$, and continuing with increasing ω we get



Since this transfer function has 3 poles and only one zero, its magnitude drops sharply with frequency.

4.3.8 Complex Conjugate Poles

The BAMPs above were restricted to real-valued poles and zeros. In this section we consider the BAMP of transfer functions having at least one pair of complex conjugate poles:

$$p_i = a \pm jb$$

- 5 As before, we are only interested in the case where $a < 0$ so that the response is stable (does not grow with time). A more typical system has a mixture of real and complex poles such as

$$G(s) = \frac{(s+5)}{(s+0.1)(s+1+3j)(s+1-3j)}$$

we will see that it is more convenient to do the BAMP as well as the phase plot when the complex conjugate poles are represented in polar form, that is in terms of ω_n and ζ . For the example above,

$$G(s) = \frac{(s+5)}{(s+0.1)(s^2 + 2s + 10)}$$

Note that using the standard polar form for the second order pole we have

$$G(s) = \frac{(s+5)}{(s+0.1)(s^2 + 2\zeta\omega_n s + \omega_n^2)}$$

- 10 where $\omega_n = \sqrt{10} = 3.1$ and $\zeta = -Re(-1+3j)/\omega_n = \frac{1}{\sqrt{10}}$.

Let

$$P_2(s) = \frac{1}{s^2 + 2\zeta\omega_n s + \omega_n^2}$$

The key idea is to analyze the asymptotes of the 2nd order pole, $P_2(s)$, as we did above, and also to consider the region $\omega = \omega_n$.

$$1. \omega \ll \omega_n$$

$$15 \quad 2. \omega = \omega_n$$

$$3. \omega \gg \omega_n$$

which give us for the magnitude:

$$1. |P_2(j\omega)| \approx \frac{1}{\omega_n^2}$$

$$2. |P_2(j\omega_n)| = \frac{1}{|j^2\omega_n^2 + 2\zeta j\omega_n^2 + \omega_n^2|} \text{ (We'll consider the significance of this for plotting below.)}$$

$$20 \quad 3. |P_2(j\omega)| \approx \frac{1}{\omega^2}$$

For case 1,

$$dB\left(\left|\frac{1}{\omega_n^2}\right|\right) = -2dB(\omega_n) \quad (\text{a constant})$$

For case 2,

$$|P_2(j\omega_n)| = \frac{1}{2\zeta\omega_n^2} \quad (\text{also a constant})$$

in dB

$$dB\left(\left|\frac{1}{2\zeta\omega_n^2}\right|\right) = -dB(2) - dB(\zeta) - 2dB(\omega_n)$$

For case 3,

$$dB(|1/\omega^2|) = -2dB(\omega)$$

Case 1 and case 3 correspond exactly to the system

$$G(s) = \frac{1}{(s + \omega_n)(s + \omega_n)}$$

For case 2, The key is to realize that $0 < \zeta < 1$ and therefore $-dB(\zeta) > 0$. The $\omega \gg \omega_n$ asymptote (3) slopes down at a -2 (-40dB/dec) slope. Consider case (2) for different values of ζ . For $\zeta = 1$, ($db(\zeta) = 0$),

$$dB(|P_2(j\omega_n)|) = -dB(2) - dB(\zeta) - dB(\omega_n^2) = -6dB - 2dB(\omega_n)$$

- 5 This is exactly like two real poles:

$$P_2(s) = \frac{1}{(s + \omega_n)(s + \omega_n)}$$

for $\zeta = 0.001$

$$dB(|P_2(j\omega_n)|) = -dB(2) - dB(\zeta) - dB(\omega_n^2) = -6dB - (-60)db - 2dB(\omega_n) = 54db - 2dB(\omega_n)$$

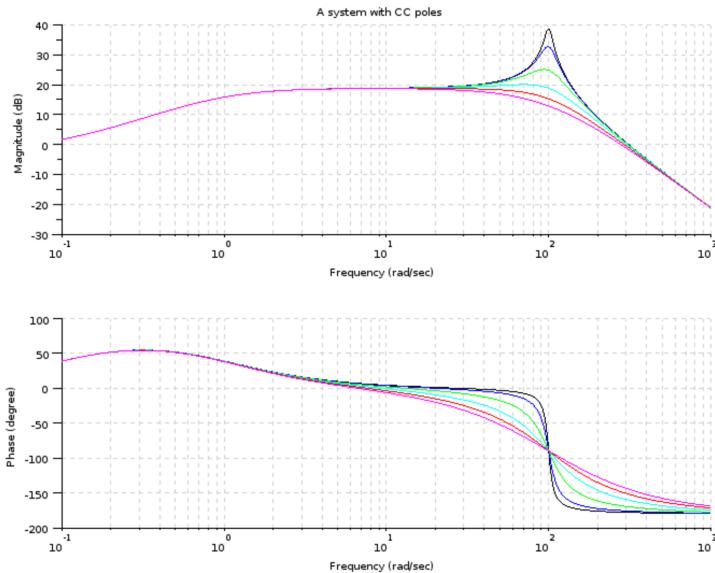
The magnitude is increased by 54dB. Thus, while the complex conjugate pole pair has two asymptotes which act just like two real poles at ω_n , the behavior at $\omega \approx \omega_n$ depends on ζ .

Example 4.14

Use the computer to plot the BAMP of

$$G(s) = \frac{8.7 \times 10^4(s + 0.1)}{(s + 1.0)(s^2 + 2\zeta 100s + 10^4)}$$

For $\zeta = \{0.05, 0.1, 0.25, 0.5, 0.75, 0.99\}$. Make all the plots superimposed on the same axes.



Example 4.14 cont.

Observations:

1. As ζ approaches zero (dark blue), the magnitude plot has a sharper and sharper peak near $\omega = \omega_n$.
2. As ζ approaches 1.0 (magenta), the magnitude plot smoothly curves through the point 6dB below the intersection of the two straight line asymptotes.
3. The high frequency asymptote has a slope of $-40dB/\text{decade}$.
4. The Phase plot goes from 0° to -90° as ω increases beyond ω_n .
5. As ζ approaches zero, the phase plot has a sharper transition $0^\circ \rightarrow -90^\circ$.
6. As ζ approaches 1.0, the phase plot acts like two real poles at $\omega = \omega_n$.
7. When plotting the magnitude peak by hand, just make the height roughly according to the table below (taken from these examples)

ζ	Peak (dB)
.001	+54dB
.05	+20dB
0.1	+12dB
.25	+6dB
.5	0dB
0	-6dB

8. Notice also that the location of the peak shifts a bit lower than ω_n when $\zeta \rightarrow 1$.

4.3.9 Complex Conjugate Zeros

A system can also have complex conjugate zeros. For example

$$G(s) = \frac{s^2 + 40s + 40,000}{(s + 0.1)(s + 1000)^2}$$

- ⁵ This system with three poles and two zeros has complex conjugate zeros at

$$z_1 = -20 + j199 \quad z_2 = -20 - j199$$

Zeros can also be expressed in terms of ω_n and ζ which in this case are

$$\omega_n = |z_i| = \sqrt{-20^2 + 199^2} = 200.0 \quad \zeta = -\text{Re}(z_i)/\omega_n = -(-20)/200 = -0.1$$

The frequency response of a complex conjugate pair of zeros is the inverse of a complex conjugate pair of poles. Instead of a peak, there is a dip in the magnitude response and instead of a phase change of -180° ,

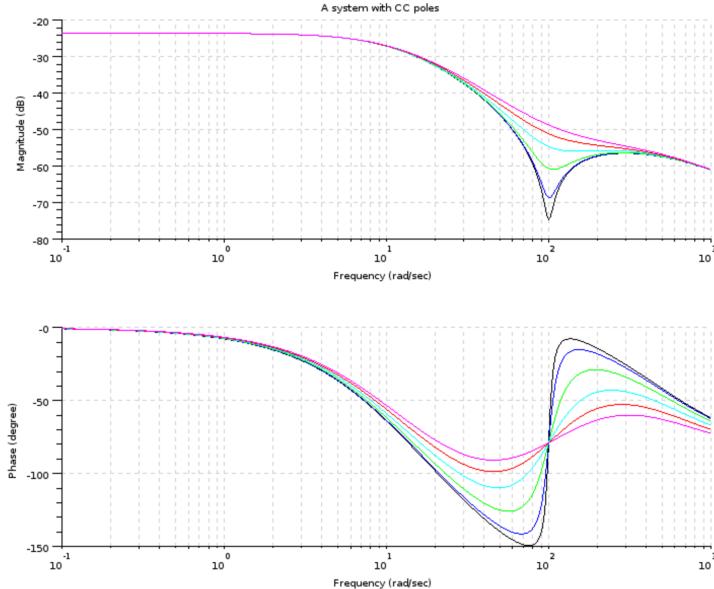
there is a phase change of $+180^\circ$.

Example 4.15

Use the computer to plot the BAMP of

$$G(s) = \frac{(s^2 + 2\zeta 100s + 10^4)}{(s + 10)(s + 30)(s + 500)}$$

In this transfer function we have included 3 poles, all at frequencies above the CC zeros ($\omega_n = 0.5$) For $\zeta = \{0.05, 0.1, 0.25, 0.5, 0.75, 0.99\}$. Choose a frequency range which shows all the features of $G(s)$ (all the poles and zeros). Make all the plots superimposed on the same axes.



Chapter 5

Feedback

5.1 Problem Statement and Learning Objectives

5.2 Block Diagram Transformations

5.2.1 Signals vs. Energy Flows

A key distinction in understanding systems is that between *signals* and *energy flows*.

- A *signal* is a variable which carries information but does not directly regulate the exchange of energy.

For example, a voltage, $V(t)$ which varies with time, carries some information. This voltage may be applied to a system component as an input but we will only call this a *signal* if the corresponding input current is zero (or negligible). In this case the input power is always zero.

- An *energy flow* is the case where a non-zero power flows into or out of the connection.

Example 5.1

Classify the following situations into signals vs. energy flows:

1. A voltage $V_1(t)$ is connected to a resistor of 50Ω .

$V_1(t)$ is part of an energy flow since the current in general is non zero.

2. A voltage $V_2(t)$ is connected to an amplifier with a high impedance input ($i_{in} \approx 0$).

$V_2(t)$ can be considered a signal since there is no power flow ($P = V_2(t) \times i_2(t) = 0$).

3. A force, $f_3(t)$ is applied to a translational dynamical system consisting of inertia, damping, and spring.

$f_3(t)$ is part of an energy flow since the velocity, $\dot{\theta}_3(t)$ is non zero, and therefore the mechanical power, $P = f(t) \times \dot{x} = 0$.

While it is possible to simulate energy flows using signals, and block diagrams, this case will not be considered further.

5.2.2 Block Diagrams

The basic idea of a block diagram is familiar to most people but there are a few subtleties. A block maps one signal to another. A single block (Figure 5.1), has an equivalent equation

$$Y(s) = G(s)X(s) \quad \text{or} \quad G(s) = \frac{Y(s)}{X(s)} \quad (5.1)$$



Figure 5.1: A single block,

In other words, the block is a graphical representation of multiplication of a Laplace Transform of a signal by the Laplace Transform of a transfer function.

Example 5.2

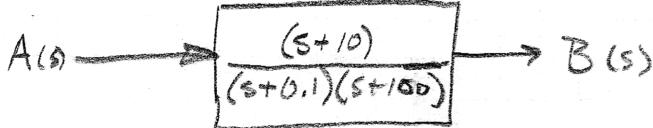
Find the expression for $Y(s)$



$$Y(s) = 5X(s)$$

Example 5.3

Find the expression for $B(s)$ and $b(t)$



$$B(s) = \frac{A(s)(s+10)}{(s+0.1)(s+100)}$$

$$b(t) = \mathcal{L}^{-1} \left\{ \frac{A(s)(s+10)}{(s+0.1)(s+100)} \right\}$$

5

One consequence of these definitions, is that there is no influence of the output of a block on its input. Put another way, there is no “loading” of an output by any number of subsequent inputs.

5.2.3 Transformations

- 10 When blocks are combined into block diagrams, the definitions above can easily be applied to figure out the meaning of the particular combination. Connections include (Figure 5.2):
- Series: the output of the first block is connected to the input of a second block.
 - Parallel: The output is the sum of the outputs of two blocks with the same input.
- Some transformations are slightly less obvious, but arise easily from Equation 5.1 as well as the properties of linearity.

The simple relationships

$$A(s)(x(s) + y(s)) \Leftrightarrow A(s)x(s) + A(s)y(s)$$

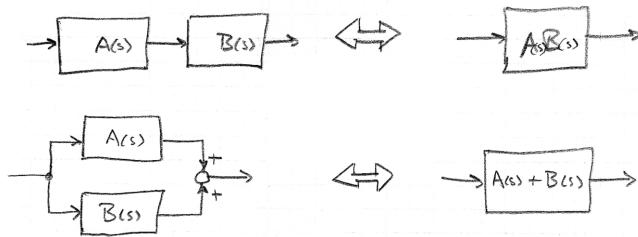


Figure 5.2: Series and parallel connections of blocks.

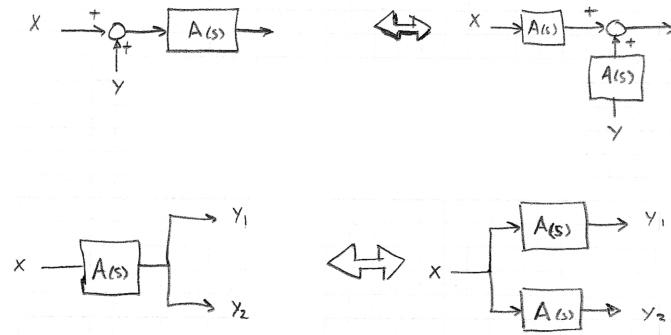


Figure 5.3: Block diagram transformations.

and

$$y_1(s) = A(s)x(s), y_2(s) = y_1(s) \Leftrightarrow y_1(s) = A(s)x(s), y_2(s) = A(s)x(s)$$

Can be used to manipulate block diagrams as shown in Figure 5.3

5.3 Closed Loop Negative Feedback Gain

One block diagram has supreme importance in control systems design (Figure 5.4). This is called the “closed loop negative feedback system.” As implied by its name, the connections of the diagram form a loop, the loop contains a minus sign, and the output is “fed back” to be subtracted from the input.

Even though this diagram is fairly simple, it is slightly more subtle to figure out an equivalent single block. In other words, can we figure out an expression for $Y(s)/X(s)$ from the block diagram of Figure 5.4? The key is identifying the output of the summation and giving it the name, $E(s)$, which stand for error. This term is called “error” because it is the difference between input and output. For example, if the closed loop negative feedback system were used to model a temperature control system, and the input was 68 degrees but the output (room temperature) was 72 degrees, then (with the frequently used assumption that $H = 1$)

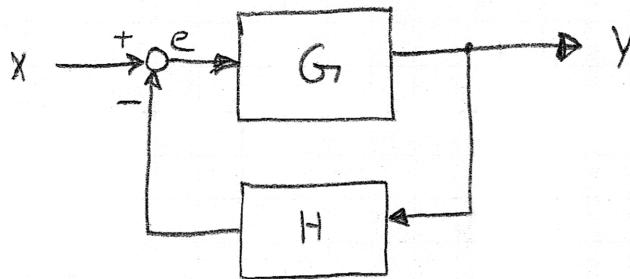


Figure 5.4: The closed loop negative feedback system.

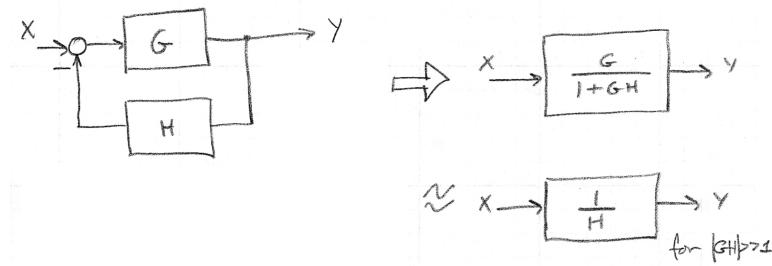


Figure 5.5: Equation 5.2 expressed as a block diagram transformation.

the error would be -4 degrees. Thus

$$E(s) = X(s) - Y(s)H(s)$$

Using block diagram relationships and dropping the (s) for convenience

$$Y = GE = G(X - YH)$$

$$Y = GX - GHY$$

$$Y(1 + GH) = GX$$

5

$$\frac{Y}{X} = \frac{G}{(1 + GH)} \quad (5.2)$$

This expression is called the closed loop transfer function. It was discovered by H.S. Black of Bell Labs in 1927.

A common application of Figure 5.4 is a feedback control system in which $G(s)$ represents a combination of a controller and a plant. The controller (typically implemented today with a microcontroller and associated I/O devices) generates a command signal to the plant which is the system to be controlled. The feedback element H is usually some kind of sensor which measures the output such as a temperature sensor or tachometer. In many control systems $H = 1$ since the objective is eliminating error between the desired output (X) and the actual output (Y).

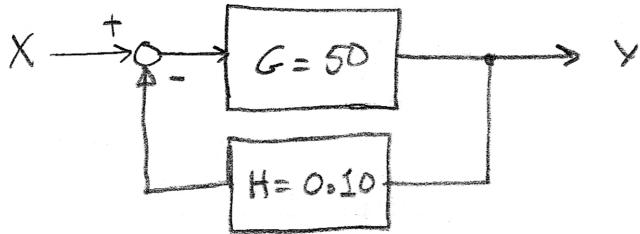
An important case is when $|GH| >> 1$. Applying this to Equation 5.2,

$$\frac{Y}{X} \approx 1/H$$

15 The quantity $GH(s)$ is called the *loop gain*. In more complex block diagrams, the loop gain is the product of all blocks around the loop. Expressed as a block diagram transformation, HS Black's equation (Eqn 5.2) is shown in Figure 5.5.

Example 5.4

For the following system,



Find $\frac{Y}{X}$

$$\frac{Y}{X} = \frac{G}{1 + GH} = \frac{50}{1 + 0.1 \times 50} = 8.33$$

What if $G = \{100, 500, 10^5\}$?

G	Y/X	GH
100	$\frac{100}{1+0.1\times 100} = 9.09$	$10 > 1$
500	$\frac{50}{51} = 9.80$	$50 > 1$
10^5	$\frac{10^5}{1+10^4} = 9.9999$	$10^4 >> 1$

As GH gets larger in magnitude, Y/X gets closer and closer to $1/H$.

In this example, $H < 1$. While $H = 1$ is more typical for control systems, the situation here is very desirable for design of amplifiers such as audio amplifiers (HS Black's application).

As we will see in detail in the next sections, the behavior of closed loop negative feedback systems when $GH \gg 1$ has major engineering advantages including:

- ⁵ • Reduced sensitivity to parameter variations.
- Ability to reject external disturbances.

5.4 Sensitivity Analysis

The performance of a system depends on all of its parts, but which parameters are most important in determining performance? Sensitivity analysis is a way to answer that question. Often a low precision

¹⁰ component costs more than a high precision version of the same component. If the sensitivity of performance to a parameter is low, then use of a low precision component should have a small effect on performance and cost can be saved. Conversely, if sensitivity of performance to a different parameter is high, then a variation of its parameter value will make a big impact on performance which might justify the additional cost of a precision component.

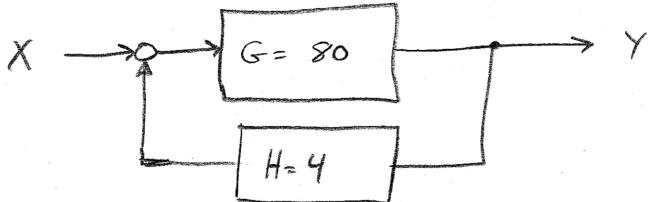
¹⁵ We will call some measure of system performance, P . If a system has multiple performance measures, we use P_i to designate one of them. The parameters of a model of the system will be p_i . With these definitions, we define Sensitivity of performance measure i to parameter j , about the current values, p_{j0}, P_{i0} , as

$$S_{ij} = \frac{\Delta P_i}{\Delta p_j} \frac{p_{j0}}{P_{i0}}$$

This is like a derivative, but it is normalized by the values of the parameter and performance measure. Qualitatively, sensitivity can be thought of as

$$S_{ij} = \frac{\% \text{ change in performance}}{\% \text{ change in parameter}}$$

Although sensitivity can be derived analytically, we will concentrate here on using a numerical method.

Example 5.5


One aspect of performance is the gain or magnitude ratio, $|Y/X|$. Find the sensitivity of $P_i = |Y/X|$ to the parameter G . In other words, compute sensitivity for

$$P_i = \left| \frac{Y}{X} \right| \quad p_j = G$$

Choose $\Delta = 10\%$. We'll tabulate values of G , H , and P_i in order to compute S_{ij} .

G	H	P_i
80	4	$\frac{80}{1+4\times80} = \frac{80}{321} = 0.24922$
88	4	$\frac{88}{1+4\times88} = 0.24929$

Now we compute ΔP_i by subtracting the numerical results (note that we need to use 6 significant figures to get a non-zero result).

$$\Delta P_i = 0.24929 - 0.24922 = 0.00007$$

Therefore,

$$S_{ij} = \frac{7 \times 10^{-5} / 0.24922}{8/80} = 2.81 \times 10^{-3} = 0.3\%$$

Since sensitivity values are normalized by the nominal values of parameter and performance, we can judge them on an absolute scale where $S_{ij} = 100\%$ indicates strong sensitivity. In this case we can see that sensitivity of closed loop gain to G is small.

Example 5.6

Find the sensitivity of the system of Example 5.5 to H

G	H	P_i
80	4	$\frac{80}{1+4\times80} = 0.24922$
80	4.4	$\frac{80}{1+4.4\times80} = 0.22662$

$$\Delta P_i = 0.22662 - 0.24922 = -0.02259$$

Therefore,

$$S_{ij} = \frac{-0.02259}{0.24922 / 0.4 / 4} = -0.906 = -91\%$$

This is a high degree of sensitivity. A negative value for S_{ij} means that the performance goes down when the parameter goes up.

The important point of the previous two examples is that performance of the closed loop negative feedback system depends strongly on H but weakly on G (especially as $|GH| \gg 1$).

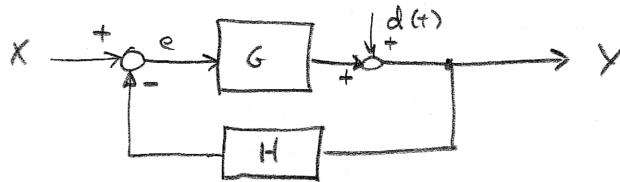


Figure 5.6: A closed loop system with a disturbance.

5.5 Disturbance Rejection

Another important aspect of control system performance is rejection of external disturbances. External disturbances are unwanted inputs injected from the environment into a system. *Disturbance rejection* is the amount by which a disturbance input to the system is reduced at the system output.

Example 5.7

Give two examples of disturbances in control systems. Identify the inputs and outputs and explain what is the disturbance signal.

- 1) Consider an automatic pilot on a commercial aircraft. The input to the automatic pilot is the desired heading in degrees relative to North. The output of the system is the plane's actual heading, for example as sensed by a compass. Gusts of wind which blow the plane off its heading constitute an external disturbance.
- 2) Consider the temperature control system for a refrigerator. The input is the desired temperature (such as a constant value of 38 deg F.). The system output is the actual temperature inside the refrigerator. When the door is opened there is an input of warm air which increases the air temperature. This increase in temperature is a disturbance.

The block diagram of Figure 5.6 is a representation of a closed loop negative feedback system with a disturbance, $d(t)$. Let's calculate the output, Y .

$$Y = D + EG$$

$$Y = D + G(X - YH)$$

$$Y(1 + GH) = D + GX$$

$$Y = \frac{D}{(1 + GH)} + \frac{G}{(1 + GH)}X \quad (5.3)$$

The output thus consists of two components, one due to the disturbance, D , and one due to the input, X . Note what happens however when $GH \gg 1$. In that case

$$Y \approx D/GH + X \frac{1}{H}$$

The disturbance input is reduced by the loop gain, GH .

15 Disturbance Examples There are many phenomena which can be treated as disturbances in analysis of a control system and thus reduced by Equation 5.3. Some frequently encountered disturbances include:

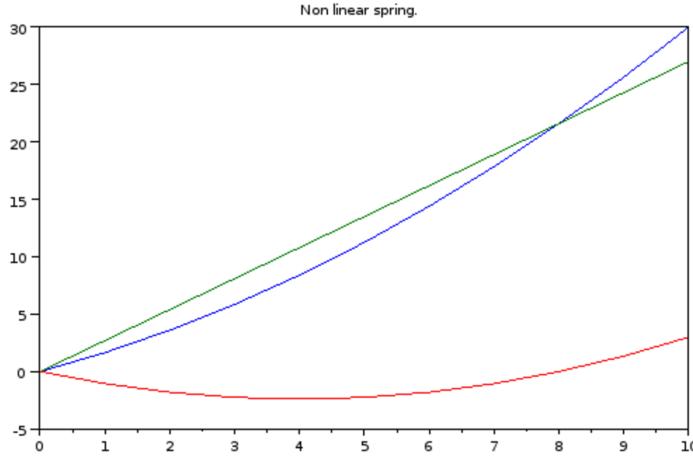
- Electrical Noise (additive)
- Unmodeled mechanical effects such as non-linear friction, or effects of temperature on mechanical parameters.
- Parameter value changes
- Vibrations
- Unmodeled flexibility or mechanical resonance.

Example 5.8

Illustrate how a non-linear spring could be broken down into a linear spring plus a disturbance. Suppose our spring obeys

$$f(x) = Kx + 0.1Kx^2$$

Notice that this can be broken down into a linear part (Kx) and nonlinear part ($0.1Kx^2$). One approach might be simply to separate the linear from the non linear terms above. As shown in the plot below, a different split gives a higher stiffness to the linear term in such a way as to make a smaller non-linear term.



This computer plot shows the nonlinear spring given by the equation above and $K = 1.5$ and also (in green) a linear fit:

$$f(x) = 2.7x$$

The difference

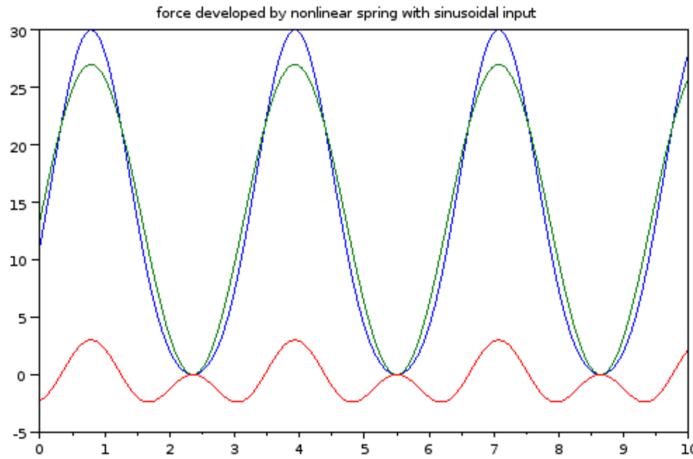
$$f_{NL} = Kx + 0.1Kx^2 - 2.7x$$

is shown in red.

Suppose the system goes through a trajectory,

$$x(t) = 5 + 5 \sin(5t)$$

Then this nonlinear spring would generate the following forces:

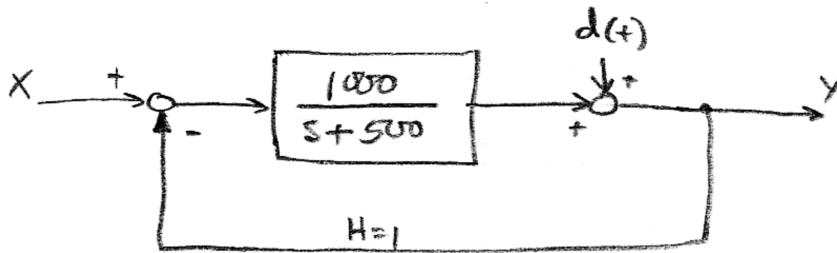


Here the sinusoidal force output is nonlinear (blue) but can be broken down into a linear part (green) and a non-linear part (red).

The point of this example, is that sometimes we can treat a nonlinear system as a linear one and the non-linear behavior can be thought of as a disturbance (which is then attenuated by Equation 5.3).

5.5.1 Disturbance Rejection in the Frequency Domain

Example 5.9



Find

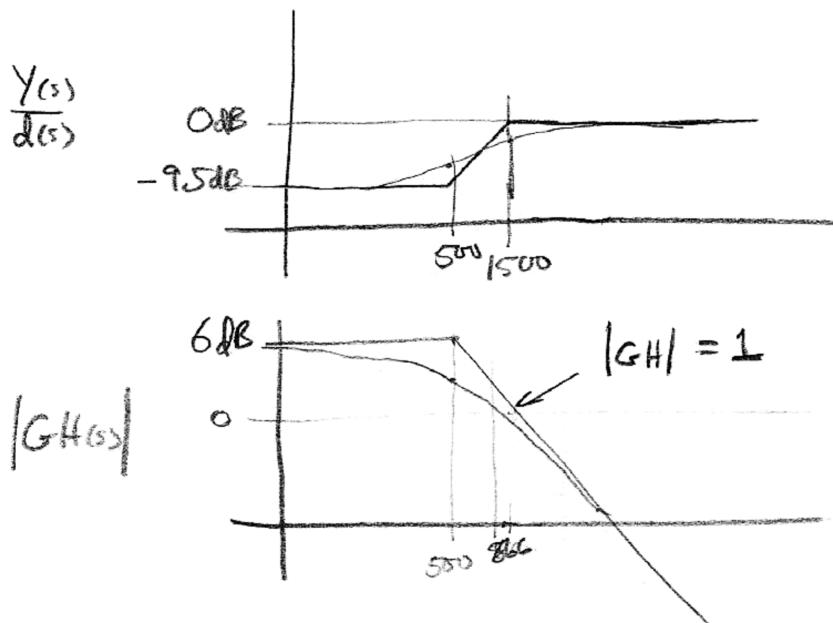
$$\frac{Y(s)}{D(s)} \quad \text{for} \quad X(s) = 0$$

and sketch the BAMP of $GH(j\omega)$ and $\frac{Y(j\omega)}{D(j\omega)}$.

$$\frac{Y(s)}{D(s)} = \frac{1}{1 + GH} = \frac{s + 500}{s + 500 + 1000} = \frac{(s + 500)}{(s + 1500)}$$

and

$$GH(s) = \frac{1000}{(s + 500)}$$



The disturbance rejection is $-9.5dB$ for frequencies below 500 rad/sec (not counting the $+3dB$ correction at 500 r/s). There is no more disturbance rejection above about 1500 .

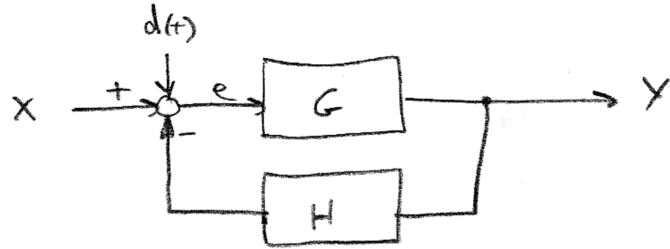


Figure 5.7: A closed loop negative feedback control system with a disturbance injected at the input.

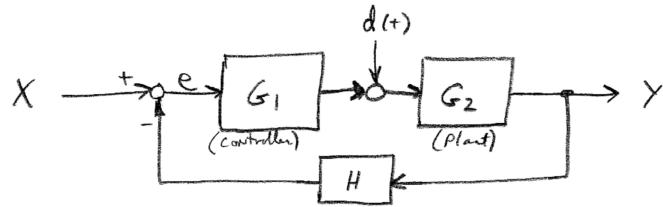


Figure 5.8: A closed loop negative feedback control system with a disturbance injected between the controller (G_1) and the plant G_2 .

5.5.2 Location of Disturbance

Disturbances can enter the control loop at different locations besides summing with the output. First consider the disturbance injected into the error computation (Figure 5.7).

$$Y = GE = G(X + D - YH)$$

$$Y(1 + GH) = GX + GD$$

$$Y = \frac{G}{1 + GH}X + \frac{G}{1 + GH}D = \frac{G}{1 + GH}(X + D)$$

In this case the disturbance and the input are treated exactly the same. There is no disturbance rejection at all. In retrospect this makes sense since it would be impossible for the controller to distinguish between the disturbance and the desired input.

- Now we consider a case in which “ G ” is split into two systems and the disturbance is injected between the two (Figure 5.8). As mentioned above, this is an important case where G consists of a controller coupled to a “plant” such as an industrial machine or a vehicle.

This time we have

$$Y = G_2(D + G_1E)$$

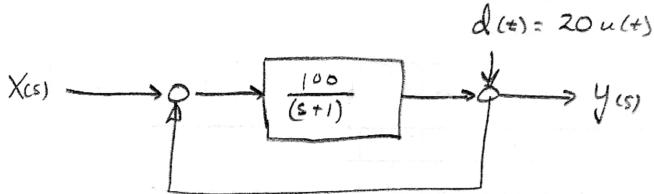
$$= G_2(D + G_1(X - YH))$$

$$Y = G_2D + G_1G_2X - YG_1G_2H$$

$$Y(1 + G_1G_2H) = G_2D + G_1G_2X$$

$$Y = \frac{G_2D}{1 + G_1G_2H} + \frac{G_1G_2X}{1 + G_1G_2H}$$

Considering the case of a large loop gain, $G_1G_2H \gg 1$, we have the situation where the disturbance is reduced by G_1H , which is less disturbance rejection than reduction by G_1G_2H .

Example 5.10


As a final example, for the system above, find $Y(s)$ and $y(t) = L^{-1}\{Y(s)\}$ for $x(t) = 0$, $X(s) = 0$, $d(t) = 20u(t)$.

$$Y(s) = \frac{G}{1+GH}X(s) + \frac{1}{1+GH}D(s)$$

Since $X(s) = 0$, and the Laplace transform of $20u(t)$ is $20/s$,

$$Y(s) = \frac{1}{1 + \frac{100}{(s+1)}} D(s) = \frac{(s+1)}{(s+101)} 20/s$$

Expanding this with partial fractions

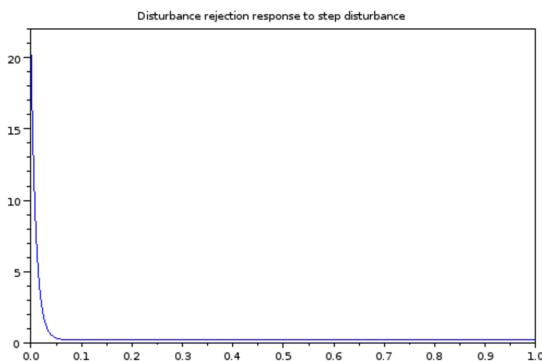
$$\frac{20(s+1)}{s(s+101)} = \frac{A_1}{s} + \frac{A_2}{(s+101)}$$

$$A_1 = \left. \frac{20(s+1)}{(s+101)} \right|_{s=0} = \frac{20}{101} \approx 0.2$$

$$A_2 = \left. \frac{20(s+1)}{s} \right|_{s=-101} = \frac{-2000}{-101} \approx 20$$

Applying the inverse transform,

$$y(t) = 0.2u(t) + 20e^{-101t}$$



The disturbance is reduced by about a factor of 100! Note however that the second term is a transient arising from the step input of the disturbance. Although this transient is over very quickly (e^{-101t}) it has a significant amplitude (20). Disturbance rejection cannot react instantly!

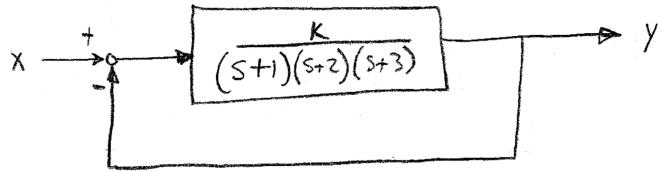


Figure 5.9: A closed loop negative feedback system with stable poles in the feed-forward path. K is a positive real constant, a gain.

5.6 Stability

We've seen many systems like that of Example 5.10, which contain transient solutions with exponential terms. As long as the coefficient of time in the exponential is negative (as in that example). Because of the negative term, all the exponentials fade out to zero with time.

- 5 On the other hand, if the time coefficient is positive, even for only one of the exponential terms arising from the partial fraction expansion, the output will grow exponentially without limit. In almost all practical systems this is unacceptable and undesirable. The time coefficient on the exponential terms comes from the real part of poles. As long as the poles have a negative real part (i.e. they are in the left half of the complex plane).

10 5.6.1 Calculation of Roots

At first glance it seems stability is trivial to assess. If any poles have positive real parts, the system is unstable.

The tricky part comes from closed loop negative feedback systems. Consider the system of Figure 5.9. In our terminology,

$$G(s) = \frac{K}{(s+1)(s+2)(s+3)}$$

- 15 where K is a positive real constant (we refer to terms like this as a *gain*) and $H = 1$. Clearly $G(s)$ is stable since the poles are $s = \{-1, -2, -3\}$.

The difficulty arises when we consider the closed loop gain

$$\frac{Y(s)}{X(s)} = \frac{\frac{K}{(s+1)(s+2)(s+3)}}{1 + \frac{K}{(s+1)(s+2)(s+3)}}$$

Multiplying through by the poles of $G(s)$ we get

$$= \frac{K}{(s+1)(s+2)(s+3) + K}$$

- 20 The denominator $(s+1)(s+2)(s+3) + K$ is called the characteristic equation and it has different roots than the poles of $G(s)$. For our closed loop transfer function, the poles are solutions to

$$s^3 + 6s^2 + 11s + 6 + K = 0$$

Below are the roots of this characteristic polynomial, solved by computer for various values of K :

- K=0.0
- 3. - 2. - 1.
K=2.0
- 3.5213797, - 1.2393101 - 0.8578736i, - 1.2393101 + 0.8578736i
25 K=4.0
- 3.7963219, - 1.101839 - 1.1916708i, - 1.101839 + 1.1916708i
K=6.0
- 4. - 1. - 1.4142136i, - 1. + 1.4142136i
30 K=8.0

– 4.1663127 K=10.0 – 4.3089073, K=20.0 – 4.8371387, K=30.0 – 5.214468, K=40.0 – 5.5173935, K=50.0 – 5.7744943, K=60.0 – 6., K=70.0 – 6.202156, K=80.0 – 6.386221, K=90.0 – 6.5557795, K=100.0 – 6.7133977,	– 0.9168436 – 1.587351i – 0.8455463 – 1.731557i, – 0.5814307 – 2.2443299i, – 0.5814307 – 2.2443299i – 0.3927660 – 2.5979998i, – 0.2413032 – 2.8773326i – 0.1127528 – 3.1120902i, 1.665D-15 – 3.3166248i 0.1010780 – 3.4990837i, 0.1931105 – 3.6645875i 0.2778897 – 3.8165881i 0.3566988 – 3.9575356i	– 0.9168436 + 1.587351i – 0.8455463 + 1.731557i – 0.5814307 + 2.2443299i – 0.3927660 + 2.5979998i – 0.2413032 + 2.8773326i – 0.1127528 + 3.1120902i 1.665D-15 + 3.3166248i 0.1010780 + 3.4990837i 0.1931105 + 3.6645875i 0.2778897 + 3.8165881i 0.3566988 + 3.9575356i
--	--	--

Notice two main points:

- Even at $K = 2.0$ two of the poles become complex conjugates (CC) whereas they were all real for $K = 0$.
- When $K = 60$ the real part of the CC poles is essentially zero.
- The real part of the CC poles becomes positive for $K > 60$.

From these observations we can conclude that the system is stable for gains below 60 but unstable for gains above that. While this is a simple analysis to perform with the computer, in the early days of control engineering, predicting stability of a closed loop system (based on the models of individual components) was a major challenge because of the lack of practical manual methods for solving polynomials above order 2. In response, some clever manual techniques were developed for stability analysis and some of those are still important today, especially during design.

5.7 Stability in the Frequency Domain

- The following is a very basic derivation of closed loop stability. It applies only to systems whose blocks have only poles with negative real parts or poles at the origin.

Consider the closed loop system of Figure 5.10, Left. Recall that any function of s has a complex value which in turn has an angle and magnitude. Suppose we look at the steady state sinusoidal domain ($s = j\omega$, see Section 4.3) and further suppose that for some ω ,

$$\angle A(j\omega) = 180^\circ$$

- This would have the effect of changing the negative sign on the feedback loop.

Suppose on the other hand that A is real (i.e. $\angle A = 0$) but we add a term of $H = -1$ to the feedback loop (Figure 5.10, Right). In both of these cases the total angle around the loop (the angle of the loop gain) is 180° . Analyzing the closed loop gain

$$Y = A(X - (-Y)) = AX + AY$$

$$\frac{Y}{X} = \frac{A}{1 - A}$$

- for the case $A = 1$,

$$\frac{Y}{X} \rightarrow \infty$$

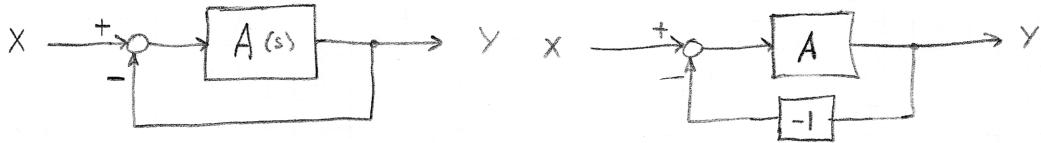


Figure 5.10: If the phase angle of $A(s)$ is 180° , the gain around the loop becomes positive.

Thus, if the loop gain is $M(s) = C(s)P(s)H(s)$, a condition on the loop gain for instability is

$$|M(j\omega)| = 1, \quad \angle M(j\omega) = 180^\circ$$

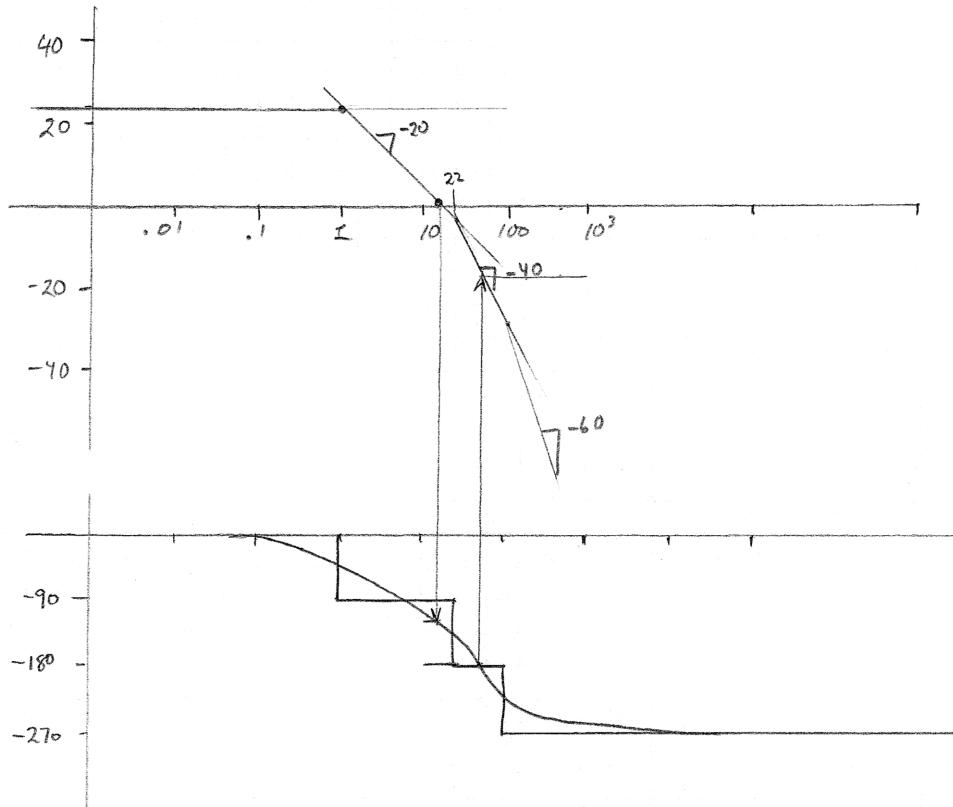
Using the BAMP we should be able to detect this combination.

Example 5.11

A system has the closed loop transfer function

$$G(s) = \frac{3.9 \times 10^4}{(s+1)(s+22)(s+100)}$$

Use the Bode magnitude and phase plots to check the closed loop stability. Drawing the Bode plots,



We first identify the point where the BAMP crosses $0dB$ (magnitude = 1), then we find the phase angle by reading the phase plot at the same frequency (about 10.5 rad/sec). This is illustrated graphically by a vertical line dropping down from the $0dB$ axis to the phase plot. Reading that value we get a phase angle of about 120° . This is well short of 180° so the system is stable.

Another way to evaluate stability is to look at where the phase curve crosses 180° and evaluate gain. In this case we find that the phase curve crosses 180° at about $\omega = 60$. Going straight up from the phase to the magnitude curves, we find the magnitude at $\omega = 60$ is about $-16dB$, also indicating a stable system.

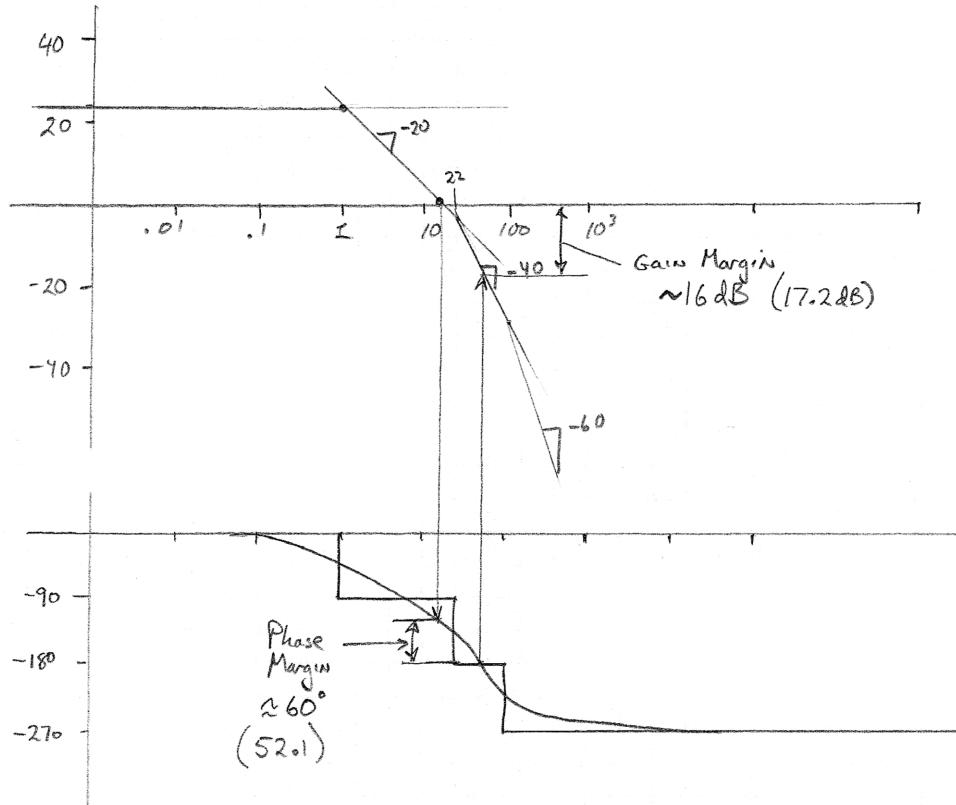
5.7.1 Gain and Phase Margins

In Example 5.11, when we looked at the frequency where magnitude was 1, we had a phase of 120° . The criterion for instability is 180° so we have a *margin* of 60° before stability is lost. Similarly, when we look at the frequency where angle was 180° , we got a magnitude of $-16dB$ so our margin is $16dB$ before gain magnitude is 1. These margins are important because they are the degree of safety we have in the face of possible changes in magnitude or phase due to changes in parameters which might happen due to real-world factors such as wear and tear. We thus define

- **Gain Margin** at the frequency where $\angle CPH(s) = 180^\circ$, the difference in dB between the BAMP and $0dB$.
- **Phase Margin** at the frequency where $|CPH(s) = 1|$, the difference in degrees between the phase plot and 180° .

Example 5.12

For the system and Bode plots of Example 5.11, find the Gain and Phase Margins.



The gain and phase margins are simply added on to the Bode Magnitude and phase plots.

Chapter 6

Root Locus

6.1 Problem Statement and Learning Objectives

6.2 Introduction to Root Locus

⁵ At the end of Chapter 5, we examined roots of the characteristic polynomial of a closed loop negative feedback system (Figure 5.9) as a gain, K was varied in increasing values from zero. Root locus (invented by Walter Evans in 1948) is a graphical method to visualize the movement of poles of the closed loop system as gain is increased. Since the location of the poles determine the stability (Section ??) and step response dynamics (Section ??), the root locus is extremely useful for design.

¹⁰ Back in 1948 it was very labor intensive to compute roots of a polynomial, especially to do it over and over for every value of K . Like the BAMP, the Root Locus plot can be sketched by hand (with practice it becomes quick) and significant insights are gained. As with BAMP, precise Root Locus diagrams are quickly obtained by computer so we can further speed the hand sketch by skipping some traditional steps which add significant labor but only refine details of the plot.

¹⁵ Some key points about the Root Locus computation are

1. Closed loop poles are not the same as the poles of the individual system blocks.

2. Closed loop poles predict the time response of closed loop system.

3. Closed loop poles predict the stability of the closed loop system.

4. The controller introduces parameter K .

²⁰ 5. The Root Locus diagram is a plot of how closed loop poles change with K .

6. We usually consider $0 \leq K \leq \infty$.

7. Scilab command: > `evans(sys)` .

8. Matlab command: > `rlocus(sys)` .

6.3 Root Locus Examples

Example 6.1

Use the computer to plot a Root Locus diagram for the system of Section 5.6.

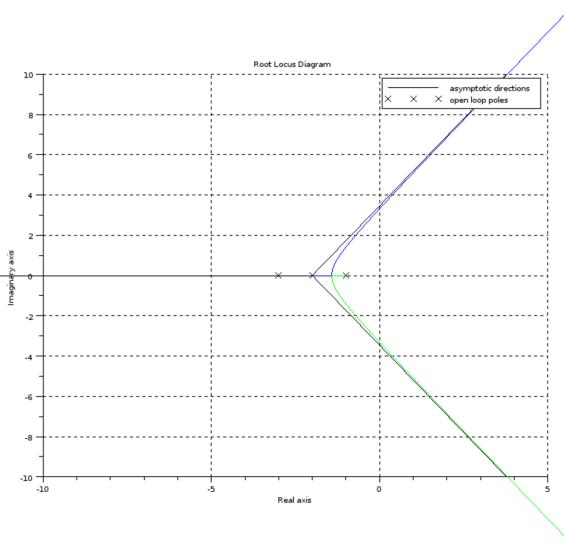
$$G(s) = \frac{K}{(s+1)(s+2)(s+3)}$$

Using Scilab, by default enter the system with $K = 1$:

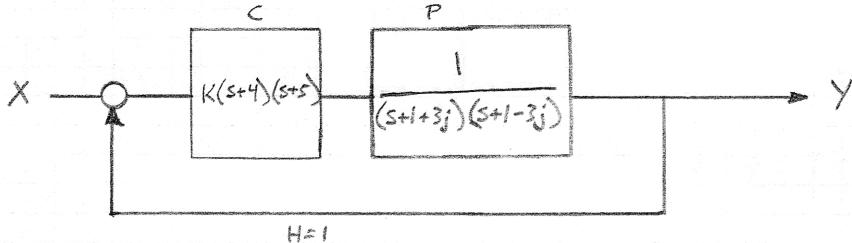
```
K = 1;
d = (s+1)*(s+2)*(s+3);

sys = syslin('c',K/d);

evans(sys);
title("Root Locus Diagram");
a=get("current_axes");//get the handle of the newly created axes
a.data_bounds=[-10,5, -10, 10];
a.grid=[1,1];
```



This plot shows first the three “open loop poles” as X's. A black, green, and blue line shows the path of the poles as $K \rightarrow \infty$. Notice that the blue and green lines cross the imaginary axis, at about $\pm 3.5j$, a result consistent with the computation of Section 5.6. The straight lines are asymptotes that the poles eventually follow, and the curved lines (in two cases) are the actual paths.

Example 6.2

Use the computer to plot a Root Locus diagram for the system above.

Here we have two blocks around the loop. C which represents a controller, and P which represents a plant. The closed loop system does not care how many blocks are in the loop, just the “loop gain” which is the product of all blocks around the loop.

$$G(s) = C(s)P(s) = \frac{K(s + 4)(s + 5)}{(s + 1 + 3j)(s + 1 - 3j)}$$

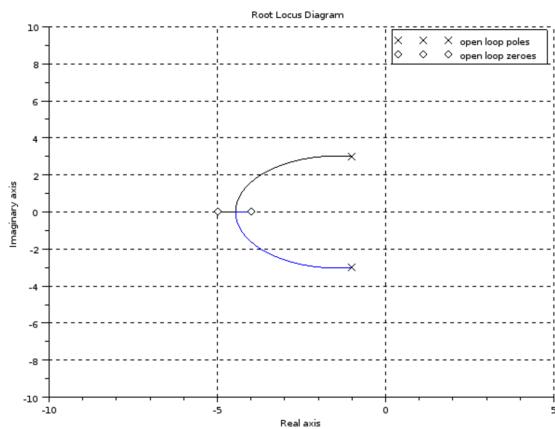
Using Scilab, by default enter the system with $K = 1$:

```

K = 1;
n = K*(s+4)*(s+5);
d = real((s+1+3*j)*(s+1-3*j));
sys = syslin('c',n/d);

evans(sys);
title("Root Locus Diagram");
a=get("current_axes");//get the handle of the newly created axes
a.data_bounds=[-10,5, -10, 10];
a.grid=[1,1];

```



This time the closed loop poles migrate toward the two zeros in the controller. First they join up at the real axis and then they split again and migrate along the real axis until they hit the poles.

6.4 Root Locus Steps

Evans figured out a set of rules, based on the mathematical properties of the closed loop characteristic polynomial, that allow us to sketch the Root Locus diagram quickly. Recall that the “loop gain” is the product of all transfer functions around the loop. For a simple system like that of Example 6.2, the loop gain is $KC(s)P(s)H(s)$ (where we have *assumed* that the controller has a constant gain term K and separated it out). Now we solve the closed loop gain using Equation 5.2.

$$\frac{Y(s)}{X(s)} = \frac{KC(s)P(s)}{1 + KC(s)P(s)H(s)}$$

Poles of this closed loop transfer function are values of s where its denominator is zero. In other words

$$1 + KC(s)P(s)H(s) = 0 \quad KC(s)P(s)H(s) = -1$$

Since the transfer functions are complex, we have

$$|KC(s)P(s)H(s)| = 1 \quad \text{and} \quad \angle KC(s)P(s)H(s) = 180^\circ$$

These two conditions are called the *magnitude condition* and the *angle condition* respectively. All points on the

- ¹⁰ Root Locus are poles of the CLTF for different values of K . Since K is positive and real, it always contributes 0° to the angle, and can be dropped from the angle condition. Thus all points on the Root Locus, for any value of K ($0 < K < \infty$), must observe both conditions.

The following Root Locus drawing rules derive from either the Magnitude Condition, the Angle Condition, or fundamental properties of polynomials.

¹⁵ **Root Locus (RL) Drawing Steps:**

1. RL Starts (when $K = 0$) at the roots of $CPH(s)$ so start out by plotting these poles and zeros (as x's and o's).
2. Find which parts of the real line contain parts of the RL. For each point on the real axis, if the total number of poles and zeros to right is ODD, that part is ON the RL.
- 20 3. The number of asymptotes (diverging branches which go out to infinity) is $n_p - n_z$, where n_p is the number of poles and n_z is the number of zeros.
4. If $n_p - n_z \neq 0$, the intercept of the asymptotes with the real line is:

$$\sigma_a = \frac{\sum \text{poles} - \sum \text{zeros}}{n_p - n_z}$$

5. The angles of the diverging branches are:

$$\frac{\pi(1 + 2m)}{n_p - n_z}$$

6. Poles which do NOT diverge circle back to zeros.

6.5 Root Locus FAQ

- ²⁵ This FAQ refers to the system of Section ?? where K is a real number > 0 .

Q: What is the point of the Root Locus?

A: The point is to predict how *closed loop* pole locations / performance will change, knowing only the *open loop* properties, as the gain constant, K , changes.

Question	Answer	Reason
Fact 1. What is true for any value of s on the root locus?	$\angle CPH(s) = \pi, 3\pi, 5\pi \dots$, "Angle Condition"	
Fact 2. What is k for a value of s on the RL?	$ CPH(s) = 1$ so $K = 1/ cPH(s) $. "Magnitude Condition"	
Rule 1. Where do branches of the RL go as $k \rightarrow \infty$?	From poles of $CPH(s)$ to zeros of $CPH(s)$ or they diverge to $ s = \infty$	M
Rule 2. How many branches diverge to $ s = \infty$?	$n_d = n_p - n_z$.	P
Rule 3. What angles do the asymptotes have?	$\theta_d = \frac{\pi(1+2m)}{n_p - n_z}$, $m = 0, 1, 2, 3 \dots$	A
Rule 4. Where do asymptotes intersect the real axis?	$\sigma_a = \frac{\Sigma \text{poles} - \Sigma \text{zeros}}{n_p - n_z}$	P
Rule 5. What parts of the real axis are ON the RL?	A segment is ON the RL if the number of real poles and zeros to the right is ODD.	A
Rule 6. At which point do branches leave or join the real axis?	at real values, s , where $\frac{d}{ds} \frac{P(s)}{Z(s)} = 0$	P,A
Rule 7. At what angle do branches depart from a complex pole? (or join a complex zero?)	$\theta_d = \pi - \Sigma \angle \text{poles} + \Sigma \angle \text{zeros}$	A

Notes:

A = "angle condition", M = "magnitude condition", P = theory of polynomials.

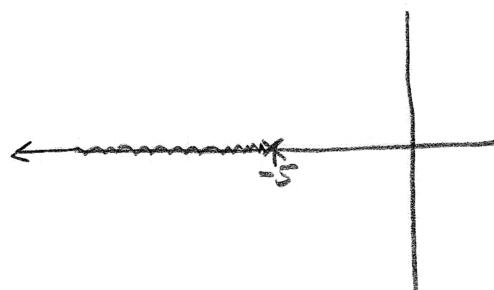
Rules 6 and 7 are from pre-computer days and no longer needed (in my opinion).

5 6.6 Hand Root Locus Examples

Example 6.3

$$G(s) = C(s)P(s) = \frac{K}{(s+5)}$$

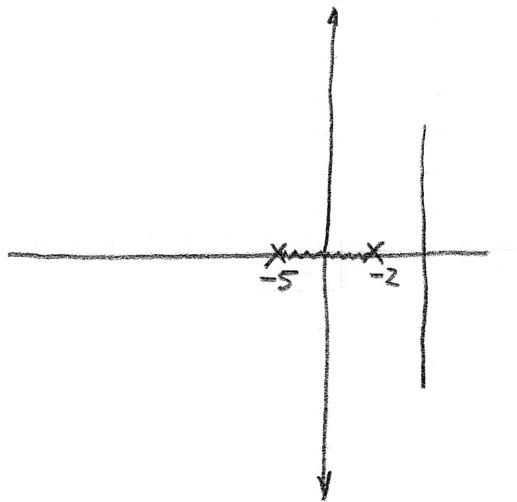
- 1) Plot the poles and zeros.
- 2) Real Line: where $Re(s) < -5$ there is one pole to the right. Therefore RL goes on real line for $x < -5$.
- 3) # of diverging asymptotes: $n_p - n_z = 1 - 0 = 1$
- 4) Angle of Asymptotes: $\frac{(2m-1)\pi}{n_p - n_z} = \{\pi\}$
- 5) Intercept of asymptotes: N/A (because π is parallel to the real axis).



Example 6.4

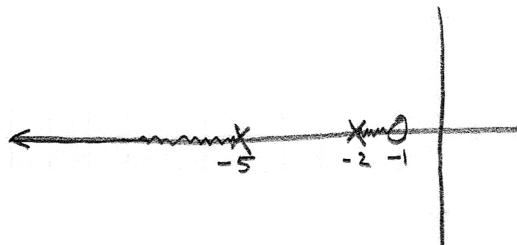
$$G(s) = C(s)P(s) = \frac{K}{(s+2)(s+5)}$$

- 1) Plot the poles and zeros.
- 2) Real Line: where $-5 < \text{Re}(s) < -2$ there is one pole to the right. Therefore RL goes on real line for $-5 < x < -2$.
- 3) # of diverging asymptotes: $n_p - n_z = 2 - 0 = 2$
- 4) Angle of Asymptotes: $\frac{(2m-1)\pi}{n_p - n_z} = \{\pi/2, 3\pi/2\}$
- 5) Intercept of asymptotes: $\frac{-2-5}{2} = -3.5$

**Example 6.5**

$$G(s) = C(s)P(s) = \frac{K(s+1)}{(s+2)(s+5)}$$

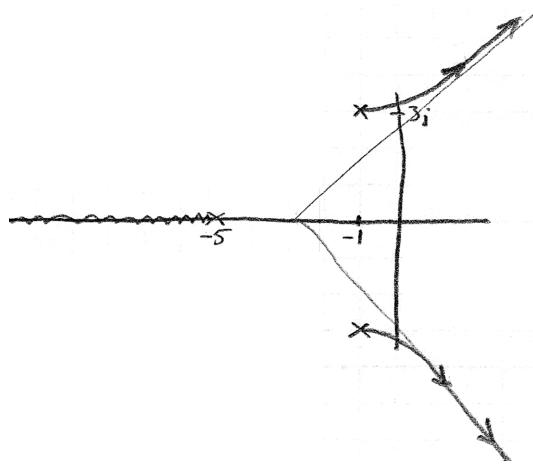
- 1) Plot the poles and zeros.
- 2) Real Line: where $-2 < \text{Re}(s) < -1$ AND $\text{Re}(s) < -5$, there is an odd number of poles to the right.
- 3) # of diverging asymptotes: $n_p - n_z = 2 - 1 = 1$
- 4) Angle of Asymptotes: $\frac{(2m-1)\pi}{n_p - n_z} = \{\pi\}$
- 5) Intercept of asymptotes: N/A (because π is parallel to the real axis).



Example 6.6

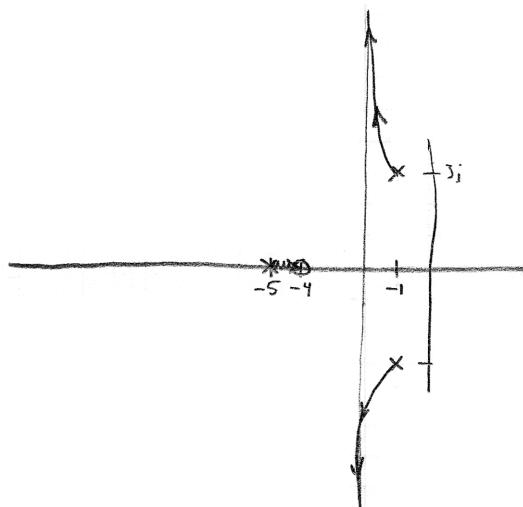
$$G(s) = C(s)P(s) = \frac{K}{(s+5)(s+1+3j)(s+1-3j)}$$

- 1) Plot the poles and zeros.
- 2) Real Line: where $\text{Re}(s) < -5$ there are three poles to the right. Therefore RL goes on real line for $x < -5$.
- 3) # of diverging asymptotes: $n_p - n_z = 3 - 0 = 3$
- 4) Angle of Asymptotes: $\frac{(2m-1)\pi}{n_p - n_z} = \left\{ \frac{\pi}{3}, \pi, \frac{5\pi}{3} \right\}$
- 5) Intercept of asymptotes: $\frac{-5-2}{3} = -2.33$.

**Example 6.7**

$$G(s) = C(s)P(s) = \frac{K(s+4)}{(s+5)(s+1+3j)(s+1-3j)}$$

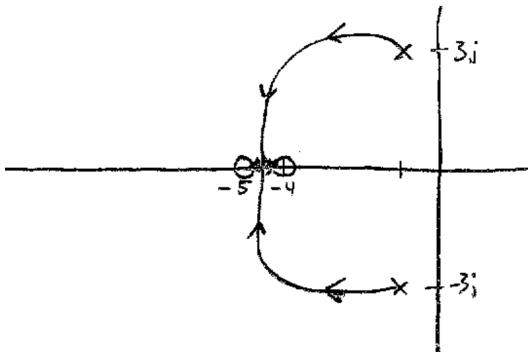
- 1) Plot the poles and zeros
- 2) Real Line: $-5 < \text{Re}(s) < -4$
- 3) # of diverging asymptotes: $n_p - n_z = 3 - 1 = 2$
- 4) Angle of Asymptotes: $\frac{(2m-1)\pi}{2} = \{\pi/2, 3\pi/2\}$
- 5) Intercept of asymptotes: $\frac{-5-2+4}{3-1} = -1.5$



Example 6.8

$$G(s) = C(s)P(s) = \frac{K(s+4)(s+5)}{(s+1+3j)(s+1-3j)}$$

- 1) Plot the poles and zeros
- 2) Real Line: $-5 < \text{Re}(s) < -4$
- 3) # of diverging asymptotes: $n_p - n_z = 0$
- 4) Angle of Asymptotes: N/A (because no asymptotes)
- 5) Intercept of asymptotes: N/A



(Compare to Example 6.2).

Chapter 7

Closed Loop Control

7.1 Problem Statement and Learning Objectives

7.2 System Type and Steady State Error

- 5 In this section, we will examine the “error” we have computed in control systems at the summing junction on the left side of the closed loop negative feedback system (Figure 7.1). Since in control systems, we most often consider $H(s) = 1$, the error, $E(s) = X(s) - H(s)Y(s)$ is a direct comparison between the input and the output. In some applications, (consider a medical device or a flight control system for an interplanetary spacecraft) it is critical to eliminate error. In others (consider a temperature control system for buildings) 10 an error of one or two percent might not matter.

It turns out that a key variable in studying the magnitude of error in a CLNFS is the number of poles at the origin in the loop gain ($C(s)P(s)H(s)$) of Figure 7.1). We call this number the *system type*. The error also critically depends on the kind of input. Some inputs are simply easier to track than others.

- System “type” is # of poles at origin: $\frac{1}{s^n}$ in the controller/plant

15 • Input can be
$$\begin{cases} \text{Step} & \frac{A}{s} \\ \text{Ramp} & \frac{A}{s^2} \\ \text{Parabola} & \frac{A}{s^3} \end{cases}$$

- If input is $\frac{A}{s^n}$, and your controller has *at least n* type, then there will be zero steady state error.

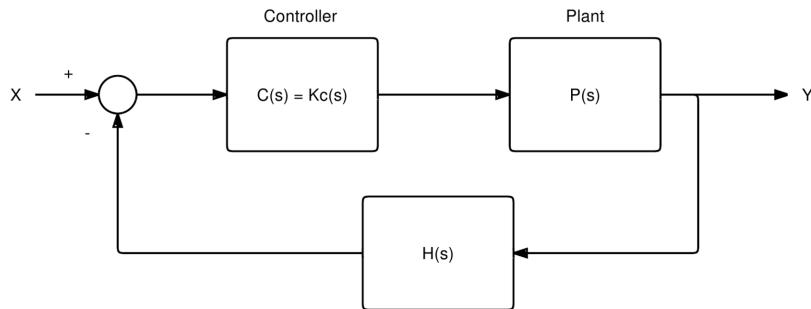


Figure 7.1: A basic closed loop control system.

Example 7.1

Find the system type for a CLNFS consisting of the following elements:

$$C(s) = \frac{500}{s(s+10)} \quad P(s) = \frac{(s+0.1)}{s(s^2+50s+1500)} \quad H(s) = 1$$

The combined system $CPH(s)$ has **two** poles at the origin so it is of **type 2**.

7.2.1 Steady State Error Derivation

The key to computation of steady state error is the Final Value Theorem of basic Laplace Transform theory:

$$\lim_{t \rightarrow \infty} f(t) = \lim_{s \rightarrow 0} sF(s) \quad (7.1)$$

- ⁵ The quantity on the left is the steady state error, after all transient terms have died out. The Final Value Theorem says we can find this final SSE by evaluating the limit on the right. However, this theorem only applies if the limit on the left actually exists. For example, if $f(t) = \sin(\omega t)$, then the limit does not exist. Looked at in the complex plain, the poles of $F(s)$ must be in the left half plane so that all transients die out.

Now let's apply the FVT to error.

$$E(s) = X(s) - C(s)P(s)H(s)E(s)$$

- ¹⁰ abbreviating $G(s) = C(s)P(s)$, and simplifying

$$E(s)(1 + GH) = X(s)$$

$$E(s) = \frac{X(s)}{1 + GH} \quad (7.2)$$

Let's apply this result to a specific system where:

$$C = 10 \quad P = \frac{50}{s+10} \quad H = 1 \quad X(s) = \frac{A}{s}$$

Note that we have chosen a specific input (step function with amplitude A), for this analysis.

Using the FVT of Equation 7.1, Applying Equation 7.2,

$$E(s) = \frac{A/s}{(1 + 500/(s+10))} = \frac{A(s+10)}{s(s+10+500)} = \frac{A(s+10)}{s^2 + 510s}$$

Applying the FVT:

$$\lim_{t \rightarrow \infty} e(t) = \lim_{s \rightarrow 0} s \frac{A(s+10)}{s^2 + 510s} = \lim_{s \rightarrow 0} \frac{A(s+10)}{s + 510} = \frac{10A}{510}$$

$$\lim_{t \rightarrow \infty} e(t) = 0.02A$$

- ¹⁵ In other words the Steady State Error (SSE) is 2%.

$$\begin{aligned}
 C = \frac{10}{s} &= \rightarrow [10] \rightarrow [\frac{1}{s}] \rightarrow \\
 &= \rightarrow [10 \int_0^t dt] \rightarrow
 \end{aligned}$$

Figure 7.2: A simple controller which has a single pole at the origin can be called an *Integral controller*

7.2.2 Steady State Error Examples

The following examples illustrate some key properties of steady state error.

Example 7.2

$$C = 10 \quad P = \frac{50}{s+10} \quad H = 1 \quad x(t) = Bt \quad X(s) = \frac{B}{s^2}$$

This is the same as above, but the input is now a ramp.

$$E(s) = \frac{B/s^2}{(1 + 500/(s+10))} = \frac{B(s+10)}{s^2(s+10+500)} = \frac{B(s+10)}{s^3 + 510s^2}$$

Applying the FVT:

$$\lim_{t \rightarrow \infty} e(t) = \lim_{s \rightarrow 0} s \frac{B(s+10)}{s^3 + 510s^2} = \lim_{s \rightarrow 0} \frac{B(s+10)}{s^2 + 510s} = \infty$$

The system is the same, but with a ramp input the steady state error is infinite!

Example 7.3

$$C = \frac{10}{s} \quad P = \frac{50}{s+10} \quad H = 1 \quad x(t) = Au(t) \quad X(s) = \frac{A}{s}$$

This is the same as the first example, but the controller, $C(s)$, now adds a pole at the origin.

$$E(s) = \frac{A/s}{(1 + 500/(s(s+10)))} = \frac{A(s^2 + 10s)}{s(s^2 + 10s + 500)}$$

Applying the FVT:

$$\lim_{t \rightarrow \infty} e(t) = \lim_{s \rightarrow 0} \frac{A(s^2 + 10s)}{s^2 + 10s + 500} = 0$$

The plant was the same, but by adding a pole at the origin to the controller, we have eliminated SSE for step input to zero. Since this is such a nice result it is worth a look at this new controller Figure 7.2. This controller can be implemented by building an integrator. Two ways to implement an integrator are an analog op-amp circuit with a feedback capacitor, and a software loop in a microcontroller which sums the difference between input and output.

Example 7.4

$$C = \frac{10}{s} \quad P = \frac{50}{s+10} \quad H = 1 \quad x(t) = Bt \quad X(s) = \frac{B}{s^2}$$

This is the same as Example 7.3, but the input is now a ramp.

$$E(s) = \frac{B(s^2 + 10s)}{s^2(s^2 + 10s + 500)}$$

Applying the FVT:

$$\begin{aligned} \text{SSE} &= \lim_{s \rightarrow 0} \frac{B(s^2 + 10s)}{s^3 + 10s^2 + 500} = \lim_{s \rightarrow 0} \frac{B(s + 10)}{s^2 + 10s + 500} \\ &= \frac{10}{500} B = 0.02B \end{aligned}$$

With the new controller, we have changed the SSE for ramp input from ∞ to 2%! Our controller has increased the system type by one and this made a big difference on SSE with the ramp input.

7.2.3 Steady State Error Summary

We've seen examples of how changing the system type or changing the input can make a big difference in the amount of steady state error. You might even notice a pattern in the examples above relating the "input type" (the power of s in the Laplace transform of the input signal) and the system type to the nature of the SSE. To see this relationship, let's take a closer look at Example 7.4.

Writing out the limit again without canceling any terms,

$$\text{SSE} = \lim_{s \rightarrow 0} \frac{sBs(s+10)}{s^2(s(s+10)+500)}$$

We have two s 's on the top. One comes from the final value theorem, and the second one from the denominator of $C(s)$. On the bottom, we have s^2 , which comes from the ramp input. The FVT term and the $C(s)$ denominator term combine to cancel the s^2 from the ramp input. Thus, if the input is

$$X(s) = \frac{A}{s^n}$$

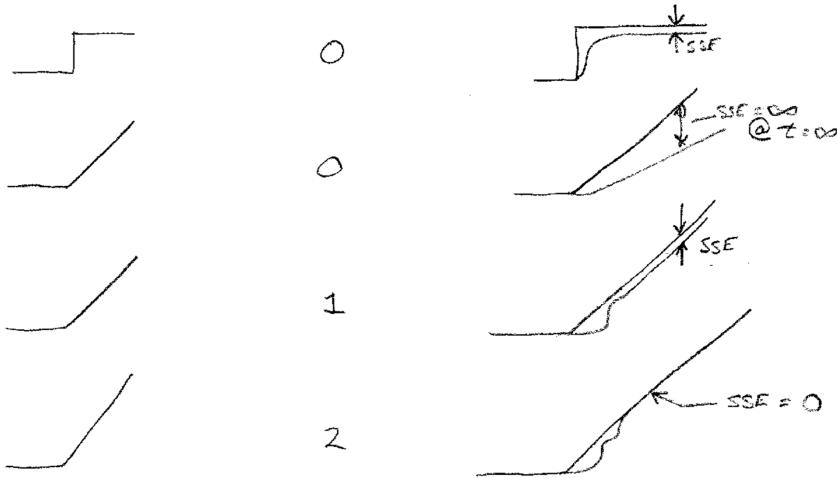


Figure 7.3: Qualitative Illustrations of different SSE and transient responses.

Type	$C(s)P(s)$	Step ($n = 1$)	Ramp ($n = 2$)	Parabola ($n = 3$)
0	$K \dots$	finite	∞	∞
1	$\frac{K}{s} \dots$	0	finite	∞
2	$\frac{K}{s^2} \dots$	0	0	finite

Table 7.1: SSE vs. System Type and Input Type

then we need at least $n - 1$ poles at the origin in the combined controller and plant (again assuming $H = 1$).

Example 7.5

Now we'll consider a general system with a gain factor, K , and n poles at the origin in the controller, as well as a general input,

$$C = \frac{K}{s^n} \quad P = \frac{50}{s + 10} \quad H = 1 \quad x(t) = Bt \quad X(s) = \frac{D}{s^m}$$

$$\text{SSE} = \lim_{s \rightarrow 0} \frac{s}{s^m} \frac{D}{\left(1 + \frac{K}{s^n} \frac{50}{(s+10)}\right)}$$

$$\lim_{s \rightarrow 0} \frac{1}{s^{m-1}} \frac{Ds^N(s+10)}{(s^n(s+10) + 50K)} = \lim_{s \rightarrow 0} \frac{Ds^n(s+10)}{s^{n+m} + 10s^{n+m-1} + 50Ks^{m-1}}$$

For this limit to be finite as $s \rightarrow 0$, we need to have no remaining powers of s in the denominator after cancellation. Thus if

$$n > m - 1$$

the error is zero. If $n = m - 1$ we have after cancellation

$$\text{SSE} = \lim_{s \rightarrow 0} \frac{Ds^n(s+10)}{s^{2n-1} + 10s^{2n-2} + 50ks^n} = \frac{10D}{50K}$$

All of the relationships illustrated by these examples can be summed up in Table 7.1. It is worth remembering that SSE only applies after transients due to the non-zero poles are over. Such transients are illustrated for some typical situations in Figure 7.3

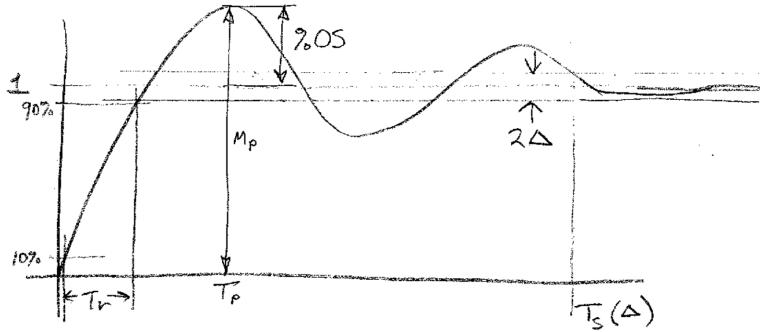


Figure 7.4: A step response with labels for percent overshoot (%OS) and settling time, T_s .

7.3 Time Domain Performance of 2nd Order Systems

In this section we will describe some ways to measure the performance of system response. While with steady state error we focused on the response after the transients died out, here we will focus on the transient characteristics of the step response in particular.

5 7.3.1 Transient Performance Specifications

The transient performance of second order systems (systems having 2 poles) is fairly easy to characterize. We develop intuition about the relationship of pole locations to time response from these second order systems. Although practical systems are usually higher order and we use computer techniques to fully understand their dynamics, the relationship between time domain performance and 2nd order poles is important to learn.

- 10 Looking at a typical step response (Figure 7.4), the most basic measures are 1) the time it takes to settle down, and 2) the amount the response overshoots the target before it settles down.

If the input step has amplitude A , *Settling time*, T_s is defined as the time it takes for the transient to enter a window around the final value such that

$$0.98A < y(t) < 1.02A \quad \forall t > T_s$$

- 15 In other words, T_s is the *last* time $y(t)$ goes into a window of $\pm 2\%$ around the final value. We know that the sinusoidal component of the output response is bounded by the exponential envelope (Figure 4.2) and that the exponential envelope is

$$\text{env}(t) = e^{\sigma t}$$

where σ is the real part of the pole. for the output response to be 2% of the amplitude at $t = T_s$, we need

$$e^{\sigma T_s} = 0.02$$

In other words,

$$T_s = \frac{\ln 0.02}{\sigma} \approx \frac{-4}{\sigma}$$

Example 7.6

Find the approximate settling time, T_s for the following system:

$$G(s) = \frac{10^7}{(s + 4.7 + 10j)(s + 4.7 - 10j)}$$

The poles are $s = -4.7 \pm 10j$. Therefore,

$$T_s \approx \frac{-4}{-4.7} = 0.85\text{sec}$$

%OS	ζ	θ
10%	0.587	54°
5%	0.695	46°
2%	0.777	39°
1%	0.829	34°

Table 7.2: Table of numerically computed values of percent overshoot vs. damping ratio (ζ).

Overshoot is expressed as a percentage of the input amplitude. In some applications a brief transient overshoot is acceptable. In other applications, (think of a elevator controller) overshoot is unacceptable¹.

The analytical calculation of overshoot from second order systems is somewhat involved. A few computational results gives the following lookup table in which overshoot depends on the damping ratio, ζ (Chapter 4), and the angle formed by the complex conjugate poles ($\theta = \cos^{-1}(\zeta)$.)

7.3.2 S-plane Regions

The performance specifications, T_s and %OS, correspond to constraints on where the poles can be located. Since $T_s = -4/\sigma$, the specification requires that the poles lie somewhere along a vertical line at $\sigma = -4/T_s$ (recall that σ is the real part of the poles). Similarly, using Table 7.2, we can see that an exact overshoot

¹There are other techniques called Trajectory Generators, beyond the scope of this chapter, which can be used to eliminate overshoot.

specification requires that the poles lie on a line from the origin at the angle determined by the lookup table.

Example 7.7

A 2nd order system has the following transient step response specifications.

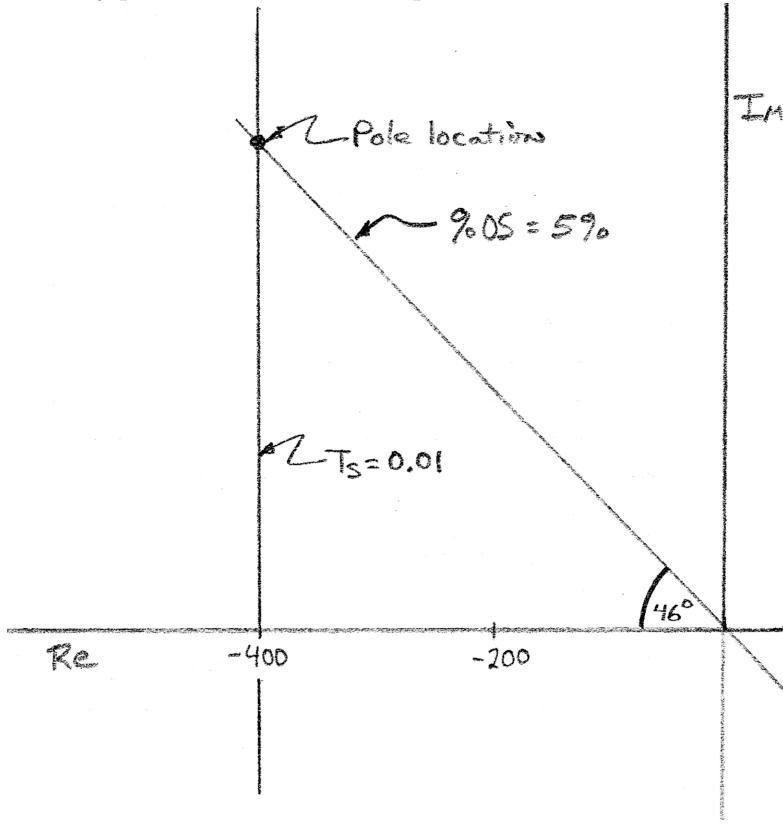
$$T_s = 0.01 \text{ sec} \quad \%OS = 5\%$$

Find where the poles must be located.

We know the real part of the poles must be

$$\sigma = -4/0.01 = -400$$

but at the same time, based on Table 7.2, the angle the pole makes with the negative real axis must be 46° . The only point which meets both specs is the intersection of these two lines.



- Sometimes specifications are expressed in terms of inequalities. For example, if the specification is $T_s \leq 0.25$, then any pole location to the left of $\sigma = -0.25$ meets the specification. Two inequality specifications generate a region in the s -plane which the poles must be in to meet the specifications (Figure 7.5).

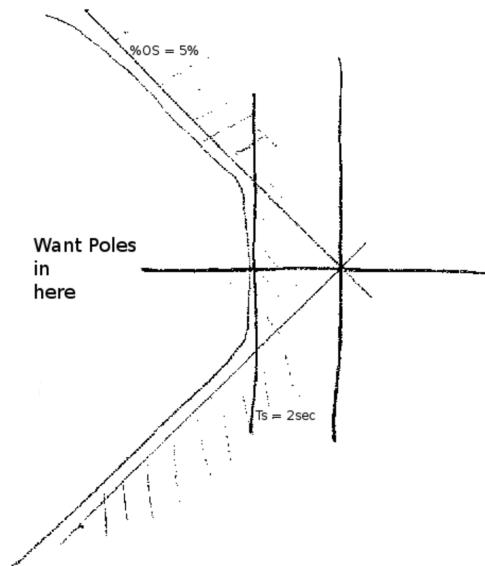


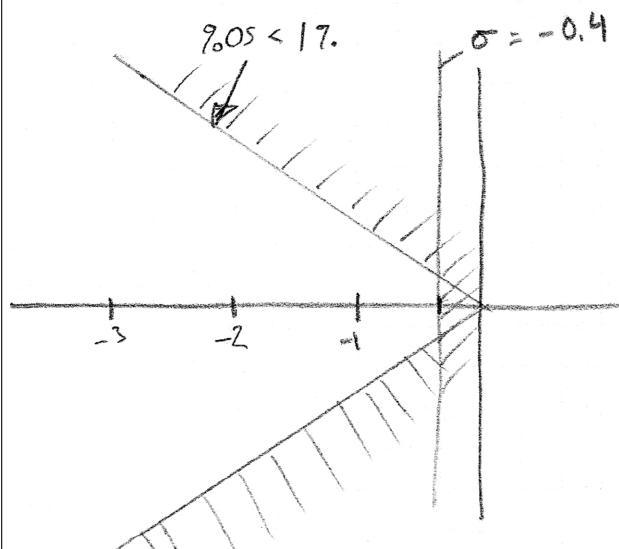
Figure 7.5: Any poles in the region shown will meet or exceed the specs: $\%OS = 5\%$ and $T_S \leq 2\text{ms}$.

7.3.3 S-plane Performance Region Examples

Example 7.8

Find the s -plane region in which poles of a 2nd order system meet the following specifications:

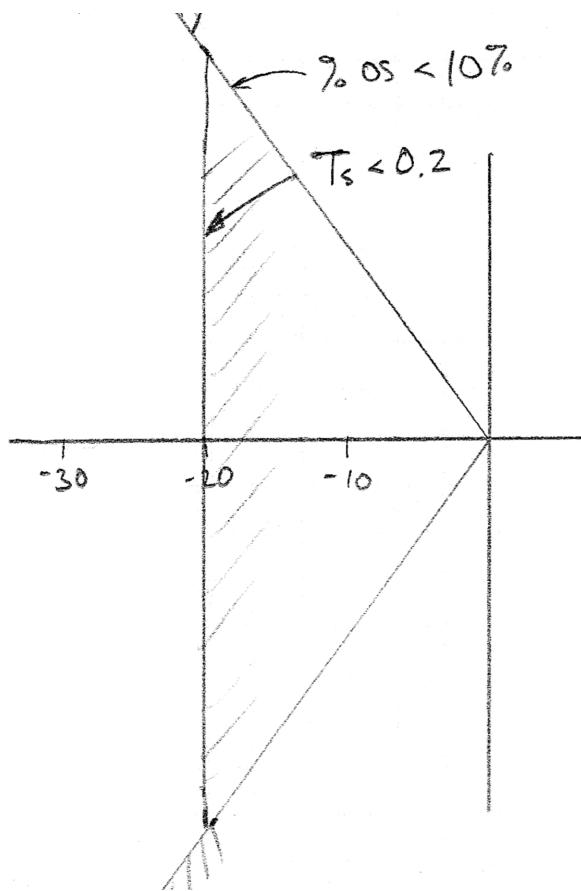
$$T_s < 10\text{sec} \quad \%OS < 1\%$$



Example 7.9

Find the s -plane region in which poles of a 2nd order system meet the following specifications:

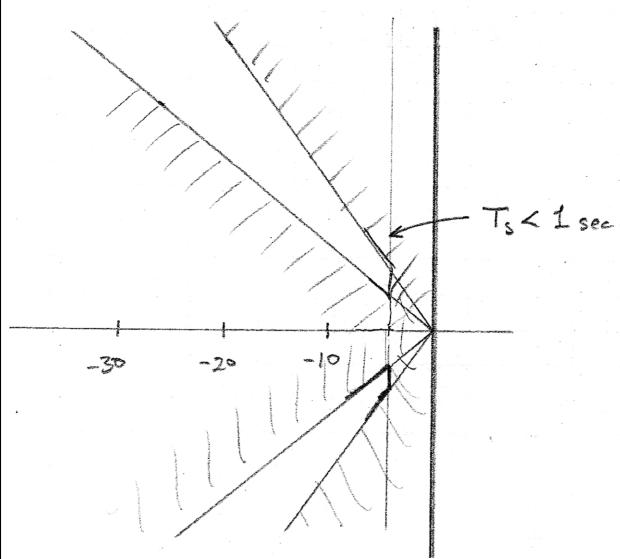
$$T_s < 0.2 \text{ sec} \quad \%OS < 10\%$$



Example 7.10

Find the s -plane region in which poles of a 2nd order system meet the following specifications:

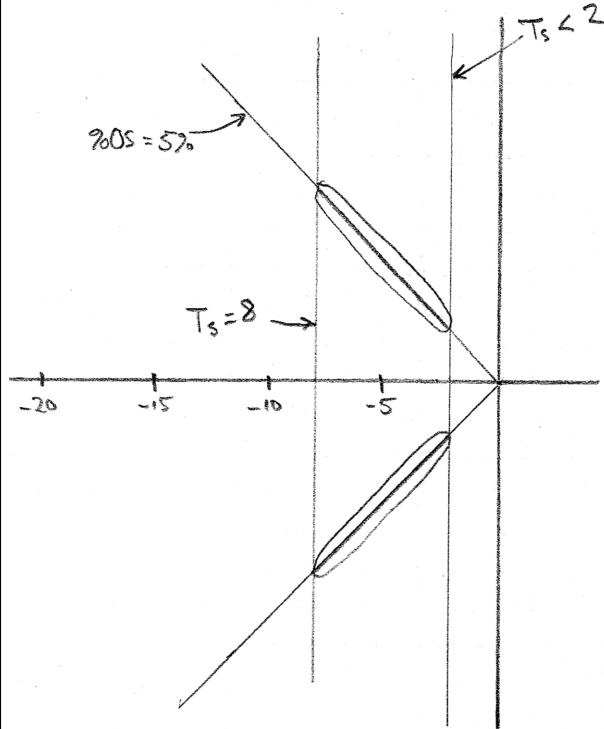
$$T_s < 1 \text{ sec} \quad 2\% < \%OS < 10\%$$



Example 7.11

Find the s -plane region in which poles of a 2nd order system meet the following specifications:

$$0.5 < T_s < 2.0 \text{ sec} \quad \%OS = 5\%$$



7.4 PID Controller

7.4.1 Closed Loop Design Problem

- Plant: $P(s)$
- Controller: $C(s)$
- Closed Loop Response: $\frac{C(s)P(s)}{1+C(s)P(s)}$
- **Design problem:** Specify Controller/Compensator $C(s)$ to improve step response compared to open loop system, $P(s)$.

7.4.2 PID Control

- Most common controller in industry BY FAR

$$\bullet C(s) = \frac{Ks + K_D s^2 + K_i}{s} = \frac{K_D(s^2 + \frac{K}{K_D}s + \frac{K_i}{K_D})}{s} = \frac{K_D(s+z_1)(s+z_2)}{s}$$

- 0, 1, or 2 zeros, for designer to place, plus one pole at $s = 0$

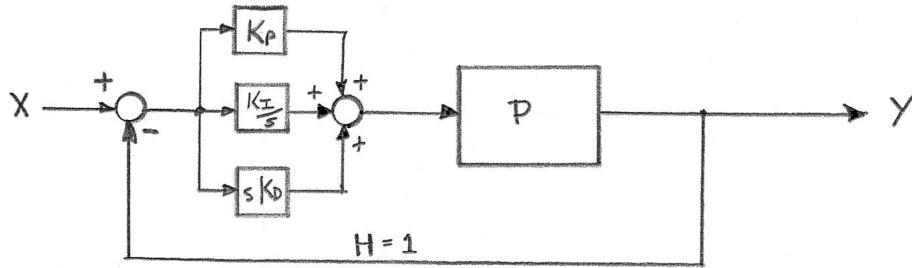


Figure 7.6: The PID controller.

- Zeros: Help stability, improve transient response
- Pole at origin: improve steady state error.

7.4.3 Basics

The PID controller is

$$C_{PID}(s) = K_P + \frac{K_I}{s} + K_D s = \frac{K_P s + K_I + K_D s^2}{s} = \frac{K_D(s^2 + \frac{K_P}{K_D}s + \frac{K_I}{K_D})}{s} \quad (7.3)$$

⁵ K_P is the proportional gain. Looking at the second term above, you can see that K_P directly multiplies the error (the input to the controller). If the other gains were zero, the control output, u , will be linearly proportional to error.

$$u(t) = K_P e(t)$$

¹⁰ K_I is the integral gain. Looking at the second term above we see that K_I appears multiplied by $1/s$, the integral operator. If the other gains were zero, the control output, u , would be K_I times the time integral of the error:

$$u(t) = K_I \int_0^t e(t) dt$$

K_D is the derivative gain. Looking at the second term above we see that K_D appears multiplied by s , the derivative operator. If the other gains were zero, the control output, u , would be K_D times the time derivative of the error:

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

In fact, using the inverse Laplace transform we can write:

$$u(t) = K_P e(t) + K_I \int_0^t e(t) dt + K_D \frac{d}{dt} e(t)$$

15 7.4.4 Simulation of PID controllers

Looking again at equation (7.3), we can see from the far right hand side that the PID controller has two zeros and one pole. Such a system cannot be physically realized and is called *improper*. Often this condition is ignored because the plant has enough poles to make the overall forward path ($C(s)P(s)$) proper. However, what if you would like to make and sell a PID controller box (Figure ??).

²⁰ We also have this problem because we define the controller in our script as a separate system and Scilab cannot simulate an improper system. To fix this we simply add another pole to the PID controller at a high enough frequency so that it does not affect our response. This immediately raises two questions

1. Why doesn't a high frequency pole affect the system?

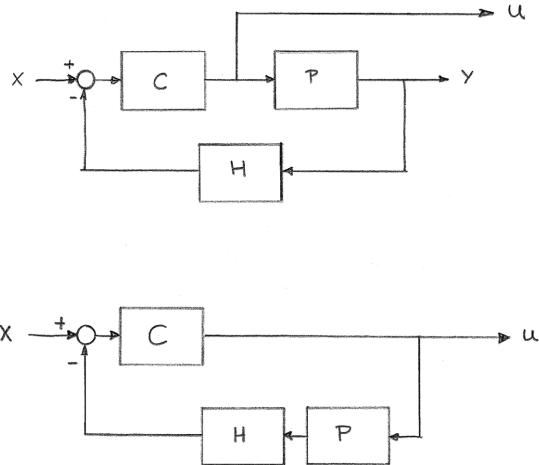


Figure 7.7: Control effort signal $u(t)$ or $U(s)$ comes from the output of the controller. We can easily get U by simulating the bottom system.

2. How high is “high enough”?

For question one, consider the Bode plot of the basic one-pole system

$$P(s) = \rho/(s + \rho)$$

for $\omega \ll \rho$, $P(j\omega) = 1$. Thus for frequencies substantially below ρ , $P(s)$ has no effect.

For question two, let us set ρ 10 times higher than the highest pole or zero of our system. Technically we should know the zeros of the PID controller to do this, but if we *assume* that the zeros of a good controller will be in the neighborhood of the poles of the plant, then 10 times greater than the highest frequency plant pole/zero will also be far from the PID controller zeros.

$$\rho = 10 \times p_{z_{max}}$$

where $p_{z_{max}}$ is the highest pole or zero in the system.

Thus we will add a pole, ρ , to the PID controller in our simulations as follows:

$$C_{PID2}(s) = \frac{\rho(K_P s + K_I + K_D s^2)}{s(s + \rho)} = \frac{\rho K_D (s^2 + \frac{K_P}{K_D} s + \frac{K_I}{K_D})}{s(s + \rho)}$$

Now C_{PID2} is proper because it has the same number of poles and zeros, two. Thus, Scilab can simulate it and it is also physically realizable.

7.4.5 Control Effort

Control effort is the level of output needed by the controller to achieve the step response. All other things being equal, a controller which achieves the specs with lower control effort is better. Often there is a limited maximum effort that a given system can output. For example, a DC motor has a maximum torque that it is capable of. In this case, it is meaningless to have a settling time that is very fast if that requires 10 times more torque than the motor has available. Often, controller gains can be found to meet **any** T_S and $\%OS$ spec if control effort limits are ignored.

Computing control effort is easy. Consider the system below which has a controller, plant and feedback.

The top system looks conventional, except we have brought out the control effort signal. In the second system we have simply rearranged the blocks without changing any connections. However we can now see this as a new feedback system having feedforward path C and feedback PH . Giving the traditional name U to the control output,

$$\frac{U(s)}{X(s)} = \frac{C(s)}{1 + C(s)P(s)H(s)}$$

If we have a limit on our actuator, for example,

$$\tau_{max} = 1.5NM$$

then an appropriate measure of performance would be the maximum value of $u(t)$: does it exceed $1.5NM$? . On the other hand, if we are concerned with total energy consumption, an appropriate measure might be

$$\int_0^{T_{max}} u^2(t)dt$$

where T_{max} defines a time window that makes sense for our application.

⁵ 7.5 Manual Design of PID controller

In this section we will design PID control gains through a “manual” method. Specifically we will find the values of K_P, K_I, K_D for a given plant and a set of performance specifications, including control effort. Although this method uses the computer to speed Root Locus plotting, it is designed to give a starting point for more accurate design methods discussed below in which we will rely on computerized search-based optimization to design controller parameters.

Although the optimization search is effective and can take more performance criteria into account, the search can go significantly faster with an initial starting guess. Unfortunately there is no “typical” range of PID parameter values which could be used as a standard starting range.

Although manual design of PID controllers is possible (see most of the textbooks), there is no reason that we cannot use the computer to speed the manual calculation. In particular, we will use the root locus command to guide our PID design into roughly the neighborhood where we expect solutions to reside.

Example 7.12

We have a large industrial machine with the following plant model:

$$P(s) = \frac{(s+1)}{(s+2.0)(s+0.7+0.2j)(s+0.7-0.2j)}, \quad H = 1$$

Our desired performance specifications are:

$$T_{SD} = 3.33\text{sec} \quad \%OS = 10\% \quad SSE_D = 0$$

It is decreed that we shall use a PID controller, but we have no initial values for K_P, K_I, K_D . We will be using the root locus method to analyze performance of PID controllers. Because we need to plot zeros and poles of the controller, the most useful form of the PID controller is:

$$C(s) = \frac{K(s+z_1)(s+z_2)}{s}$$

Problem Statement: Find values of K, z_1, z_2 or equivalently K_P, K_I, K_D that can be reasonably expected to meet the specifications.

7.5.1 Solution method

1. Make the assumption that the two complex conjugate poles closest to the origin are dominant (false!).
2. Draw the s-plane and draw the region(s) corresponding to the performance specifications.
3. Plot the poles and zeros of the plant on the s-plane.
4. Place two zeros in places such that they will “pull” the root locus through the target region^a. Use Scilab to plot the Root Locus (“`evans(sys)`” command)
5. If the root locus goes through the target region, use the mouse to find the value of K at the target location.
6. Use the chosen z_1, z_2 and K values to derive K_P, K_I, K_D (see Section 7.5.2)
7. If the root locus does NOT go through the target region, go back to step 4 and try again.

^aAssuming the plant poles do not already meet the specs(!)

7.5.2 Deriving K_P, K_I, K_D from controller zeros

There are several forms of the PID controller including:

$$C(s) = K_D \left(s^2 + \frac{K_P}{K_D} s + \frac{K_I}{K_D} \right) \frac{1}{s}$$

⁵ Let

$$PC(s) = \left(s^2 + \frac{K_P}{K_D} s + \frac{K_I}{K_D} \right)$$

Suppose that zero locations we like are

$$z_i = (s + 1.7 \pm 0.5j)$$

and that the result of step 5 above yields $K_D = 1.85$. Multiplying out gives

$$PC(s) = (s + 1.7 + 0.5j)(s + 1.7 - 0.5j) = s^2 + 3.4s + 3.14$$

Therefore,

$$C(s) = 1.85PC(s)\frac{1}{s} = 1.85(s^2 + 3.4s + 3.14)\frac{1}{s}$$

Giving us

$$K_P = 3.14 \times 1.85 = 6.29 \quad K_I = 3.4 \times 1.85 = 5.81 \quad K_D = 1.85$$

This gives us a rough design which should be a good start for the computer optimization process.

Chapter 8

Search and Optimization with Scilab

8.1 Problem Statement and Learning Objectives

8.2 Design with Scilab

8.2.1 Polynomials in Scilab

First, we have to define the letter 's' as a symbolic variable:

```
--> s = poly(0,'s');
```

Now, polynomials are entered as an expression, for example:

$$X = s^4 + 62s^3 - 5742s^2 + 689s + 27.2$$

```
--> x = s^4 + 62*s^3 + 5742*s^2 + 689*s + 27.2
```

Note that you must use * to designate multiplication.

OR, if you know the roots you could enter:

```
--> X = (s+5)*(s+27)*(s+2-6*j)*(s+2+6*j)
```

However, I have had bad luck with using complex roots explicitly as above. If such an expression causes

errors, manually multiply out the complex roots so as to get:

```
--> X = (s+5)*(s+27)*(s^2+4*s+40)
```

8.2.2 Overview

We will use a set of Scilab scripts to search the three dimensional space defined by

$$K_P, K_I, K_D$$

for the "best" design.

Performance Criteria

We will look at the most common performance criteria, and one additional criterion: control effort

- Settling Time, T_S
- Percent Overshoot, $\%OS$
- Steady State Error, SSE .
- Control Effort, cu .

We supply a set of Scilab functions which can analyze a step response and determine T_S and $\%OS$. The functions are

- `ts = settletime(t,y)`
- `o = overshoot(t,y)`
- `sse`
- `Cu`

⁵ Steady state error is approximated by the final value in the step response, and we compute maximum control effort by simulating the system again using the equation in Section 7.4.5.

8.2.3 Performance Functions

The scilab function `stepperf.sce` evaluates the step response of a given controller with your plant and computes the performance measures described below.

¹⁰ **Settling Time:** T_S

Settling time (T_S) is the time from the start of step input until the response stays within 2% of its final value (not necessarily the *desired* final value if SSE is non zero).

Percent Overshoot: POS

POS is the percentage by which the step response exceeds its final value.

¹⁵ **Steady State Error:** SSE

SSE is the difference between the final value of the step response and the amplitude of the input step (usually 1.0).

Actuator Effort: U_{MAX}

²⁰ Actuator effort (same as control effort) is the amount of output from your actuator which is given to the plant. A controller design must achieve its step response specs without requiring excessive output from its actuator. By including actuator effort into the cost function, we make sure that the controller we design is practical. Our measure of actuator effort that we use will be the peak value during the simulation time, $\max u(t)$.

8.2.4 Weights

²⁵ We need some kind of way to combine the different performance measures into a single 'cost'. For example, what if we get really good settling time, but horrible overshoot? We will use a combined score as follows:

$$S = w_1 \times T_S + w_2 \times \%OS + w_3 \times SSE + w_4 \times \max(u(t))$$

where w_i are weights which add up to 1.

Now the question is which weights to choose? Since our approach will simulate so many controllers and this is fairly time consuming, we are going to keep track of several weighting schemes simultaneously:

³⁰

Scheme Name	w_1	w_2	w_3	w_4
Settling Time	1	0	0	0
Overshoot	0	1	0	0
Steady State Error	0	0	1	0
Control Effort	0	0	0	1
Balanced	0.25	0.25	0.25	0.25

We will also normalize each type of error against our desired value so that the units of each measure do not distort the weights. Specifically

$$S = w_1 \times T_S/T_{Sd} + w_2 \times \%OS/\%OS_d + w_3 \times SSE/SSE_d + w_4 \times \max(u(t))/u_{max}$$

u_{max} (`cu_max` in the code) must be set by our actuator specification. Alternatively, if the actuator is not specified yet, we can experiment to see what kind of actuator is necessary for a give set of specs.

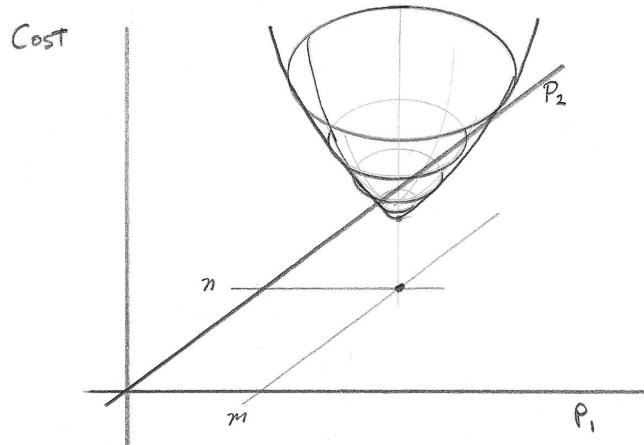


Figure 8.1: Idealized cost function which has a minimum (optimum) at $P_1 = m, P_2 = n$.

8.2.5 Gain Space Searching and Optimization

Our strategy will be to

1. Choose initial values for the three gains K_P, K_I, K_D .
2. Define a range of each value to search.
3. Define how many discrete values to search for each gain, $Nvals$
4. Try every combination of values and find those which produce the “best” step response.

As an example, consider a controller having two parameters, P_1, P_2 . For each point $\{P_1 = m, P_2 = n\}$, there is a certain step response and a certain resulting cost. In the figure below, the two values, P_1, P_2 form a plane, and we can represent the cost at each point in the third axis. In the illustration, the point $\{m, n\}$ represents the lowest value of cost over the whole plane. A simple function like a parabola can usually be easily optimized, however the cost function for step responses is not so simple, and is not known analytically.

In our PID control design problem, the three parameters could be thought of as forming a 3 dimensional space. Each controller is a single point in that space.

The simplest optimization method is to discretize the parameters and search all of the possible combinations.

When the space of all parameter values gets very large, it can be too computationally expensive to try all the possible points in parameter space. In this case special algorithms are used or mathematical assumptions are made to speed the process. In our PID control design however we have only a 3 dimensional parameter space and simulation of step responses is sufficiently fast that we can do the brute-force exhaustive search in a reasonable time:

```

20 for Kp = kmin:dk:kmax,
    for Ki = kimin:dki:kimax,
        for Kd = kdmin:dkd:kdmax,
            *** simulation and optimization code here
25
        end;
    end;
end;
```

The algorithm will loop through a set of values for each of the three gains and keep track of which one produced the highest performance by each of several measures.

Search Range We will define our search range in terms of the center value and a multiplicative range r . If our nominal value is K_0 , then

$$K_{min} = K_0 / \sqrt{r} \quad K_{max} = \sqrt{r} \times K_0$$

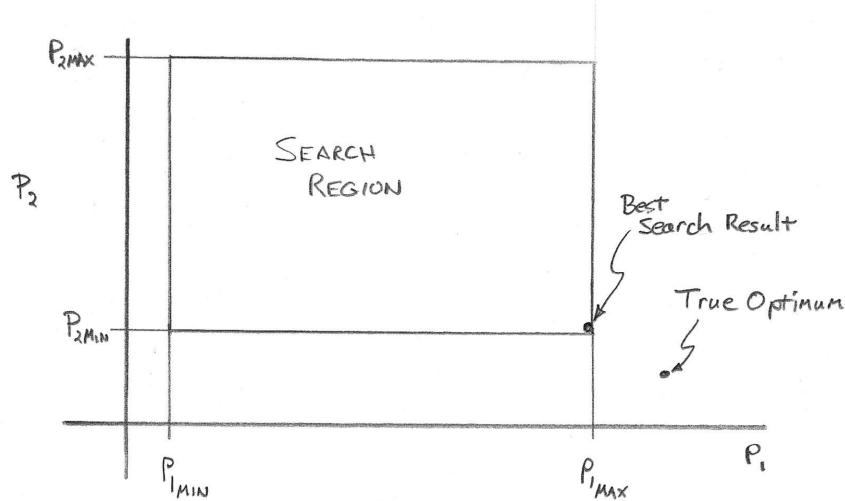
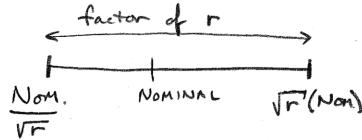


Figure 8.2: Search range does not contain the true optimum of the function and finds a minimum in one corner.

With this scheme,

$$K_{max} = r K_{min}$$

This method is illustrated below with respect to the nominal (center) value.



Other search range methods are possible but note that this approach will never generate negative gain values (which are not allowed for PID controllers anyway).

It can be tricky to know a good initial value for the gains K_p, K_i, K_D . Depending on the problem they can range from much less than 1 to hundreds. Start your search over a wide range and then narrow it down on subsequent searches.

Search size Next we choose how many discrete values we will try within the search range for each of the three gains, $Nvals$. The number of simulations we must run is then $Nvals^3$. My computer can do about 2000 simulations per minute in Scilab.

8.2.6 Range Saturation

One way this scheme can fail is if an optimum exists outside the range of parameters that you specify. In this case, the algorithm is likely to find a value at the extreme of its search range. If the algorithm reports a value at the extreme of its range, this fact is announced for you in the output and it is then a good idea to run the simulation again, centering on the extreme of the output range. The algorithm will indicate that its best value was found at the edge of its “box” by printing, for example, $k_p \text{ min}$. This would mean that the value of K_p which yielded the best value was at the edge of the search space.

Figure 8.2 shows a two-dimensional example in which the search has saturated its range at P_{1max} and P_{2min} . The actual best design is outside the search range and the search only found the closest point it could. Clearly we should move the search range to the lower right and run again. Note that we have made an assumption about the cost function in doing this, what is that assumption?

8.3 Using the Scilab packages

I have supplied three Scilab files:

- **setup.sce** This file contains code to initialize the simulation. In here you define your system, define your specifications (requirements), and your initial values and search ranges. Make a new copy of this, with a new name, for each problem you work on.
- **stepperf.sce** This file contains functions to evaluate T_S and %OS.
- **optigain2.sce** This file searches for the best design according to different weight schemes and saves the best ones. All “best” designs are plotted at the end of the search. This function takes on the order of minutes to complete.

8.3.1 optivis.sce

An experiential script is available to visualize the search space to make it easier to narrow your search.

10 optivis.sce can be substituted for **optigain2.sce** in your startup script to experiment with this feature.
optivis.sce

- Visualize the search space
- help to locate good start point(?)
- still experimental

15 8.4 Solving Design Problems

Here is the procedure to use these tools to solve a design problem.

First, collect your information:

- System model (in the form of a Scilab “syslin()”). (Know the poles and zeros of your plant)
- Required step response specs: % Overshoot, Settling Time (2%), SSE (usually 0).
- Actuator Effort normalization constant. If you don’t care about actuator effort, set the constant to a really huge number.

Then follow these steps:

1. Copy the file **setup.sce** to a new file such as **setup_problem5.sce**.
2. open the new file in a text editor
3. Set the simulation time where it says **tmax = .** This is how long the step response will be simulated and it should be about 5 times your desired settling time.
4. Enter the transfer function of your system (not your PID controller) under the comment **//plant transfer function**.
5. Identify the highest frequency pole or zero in your plant. Multiply it by 10 and set the **pp** variable to that value. This is the controller normalization pole, ρ .
6. Edit the desired performance specs below their comment. Note that 5% overshoot should be entered 1.05.
7. Enter **nvals** This is the number of values which will be tried of each parameter. Note that the total search time will be proportional to **nvals**³ so keep this below 10 until you get a feel for how long the searches take.
8. Save your file.
9. Within scilab type **exec('setup_problem5.sce')**; and this will initiate the search.

```

optigain2.sce (/home/blake/Courses/447/Notes/ClosedLoopDesign2/Scilab_files/optigain2.sce) - SciNotes
Untitled 1 optigain2.sce

1 //optigain.sce
2 //
3 //... search for optimal PID gains.
4 //
5
6 exec('stepperf.sce');
7
8 t = 0:dt:tmax;
9
10 start_time = getdate("s");
11
12 /////////////////////////////////
13 //
14 //-----function which evaluates controller performance, PID version
15 //
16 [ts, po, ss, cu, y] = costPID(plant,Kp,Ki,Kd)
17 ... //define controller
18
19 ... maxval = 10; //max allowable value of step response (cheap stability test)
20 ... pctl = pp*(Kd*s^2 + Kp*s + Ki)/(s^(s+pp));
21 ... ctl = syslin('c',pctl);
22 ... //H=1 feedback
23 ... sys = ctl*plant /. H; ... // H is feedback
24 ... ctl_effort = ctl /. plant; ... //get actuator output effort
25 ... y = csim(ones(t),t,sys); ... //get closed-loop
26 ... u = csim(ones(t), t, ctl_effort);
27 ... //pause;
28 ... if (max(y) > maxval | min(y) < -maxval) then
29 ... ts = 999; po = 999;
30 ... else
31 ... ts = settlingtime(t,y);
32 ... po = overshoot(t,y);
33 ... ss = ss+1;

```

Figure 8.3: Code listing from the scilab editor (SciNotes) showing how syntax highlighting makes the code easier to read.

8.5 Description of Software Operation

Please refer to the code listings for each script file. You may want to open them in the Scilab editor so you can see good syntax highlighting (Figure 8.3).

setup.sce After clearing the graphics windows and clearing all Scilab variables, we begin by defining the

- 5 plant transfer function. T_{max} and dt are set here because they depend on the details of the problem. For example, if the expected T_S is 0.1sec, we do not need to simulate for 10 minutes. Typically, set this to about 4-5 times the T_S specification. Set dt to have at least 100 simulation steps during the $0 - T_{max}$ interval.

Do not use complex roots in specifying the transfer function. Instead multiply out these terms to clear complex numbers out. For example, for the terms below, please use the form on the right hand side:

$$(s + 4 + 4j)(s + 4 - 4j) = s^2 + 8s + 16$$

- 10 Next we enter our specifications for T_S , %OS, SSE. We also enter our maximum actuator output.

Finally, we have to specify what range of parameter space, and how many values we should search. The more values ($Nvals$) that we select, the more accurate our result. However computation time is proportional to $Nvals^3$.

- 15 Finally we enter the center of the search range for each of the three gains, K_P , K_I , K_D and the scale range factor (r) described in Section 8.2.5.

optigain.sce This code starts by setting up the time vector and storing the starting time of the simulation run (in seconds).

- 20 The function $[ts, po, cu, y] = \text{costPID}(\text{plant}, K_p, K_i, K_d)$ accepts the plant and the three PID controller parameters, and returns the settling time, ts , the percent overshoot, po , the max control effort, cu , and the step response, y . This will be used to evaluate how good each controller is in the search. Also costPID approximately determines whether or not the system is “stable”. This determination is made by checking whether the step response exceeds a threshold called maxval . Maxval should be adjusted for different problems since a “large” output value depends on the application.

- 25 Now we set up weight schemes so that we can store the “best” controller by several different definitions. Each weight scheme has a name such as “Overshoot”.

After we initialize the storage and set up the limits for each parameter, we start searching in the “Main Loop”. Three **for** loops iterate through the 3D parameter space. For each point, we have values of K_p , K_i , K_d to test with $\text{costPID}()$. Sometimes the combination of gains we have picked results in an unstable system.

If `costPID` returned that the system was unstable, an asterisk is printed and the evaluation of performance is skipped. Otherwise, normalized error terms are computed and then weighted into the various scores. The best controller is saved for each different weighted score.

Finally, the results are printed and plotted for the user.

⁵ **stepperf.sce** These functions compute the step response and two performance measures from the step response, settling time, and overshoot. They are called by `optigain.sce`.

8.6 Example Design

(This problem is Example 9.5 from Nice, page 483). The output of this example was generated before the controll effort computation was added to `optigain.sce`.

¹⁰ **Problem:** Design a PID controller for a system where the plant is:

$$P(s) = \frac{(s+8)}{(s+3)(s+6)(s+10)}$$

Step response must have

$$T_S = 0.55(\text{sec}) \quad \%OS = 20\% \quad SSE = 0$$

Solution Procedure

Edit `setup.sce` to input the plant and the performance specifications above. Probably you should rename `setup.sce` so that you can keep this problem around. Set the initial values of $K_P = K_I = K_D = 1$ (note ¹⁵ K_1, K_2, K_3 are used in the code instead of K_P, K_I, K_D). Set `Nvals = 10`. (the code actually searches 11 values each).

Search 1

From the Scilab console, enter --> `exec('setup.sce', -1)` This runs the script and the -1 suppresses junky output.

²⁰ From the Scilab console, enter --> `exec('optigain.sce', -1)`. This performs the search of $11^3 = 1331$ systems having different values of the three gains and finds the combination of gains that gives the best step response for each weight scheme.

At the end of the search the four best step responses are plotted automatically and the gains are reported on the console.

²⁵ The results are pretty horrible! This is because we have just picked arbitrary initial gains. Let's focus on the "Balanced" result (which looks the best anyway). On the console we get:

```
[      Balanced] Kp: 3.2  Ki: 0.3  Kd:  2.024
Overshoot: 16.1 percent  Settling Time:  2.23
Search boundary reached:  Kp max Ki min
```

³⁰ Note the last line "Search boundary reached." This means that the best value is on at least one edge of the search space. This implies we need to change the range to look outside.

Search 2 Let's re-search starting with the current best "Balanced" values. Edit `setup.sce` so that the initial values are:

$$K_P = 3.2 \quad K_I = 0.3 \quad K_D = 2.024$$

Let's also search over a larger range: `scale_range = 100`.

³⁵ This time we get

```
[      Balanced] Kp: 32.0  Ki: 1.2  Kd: 16.232
Overshoot: 6.5 percent  Settling Time:  0.55
Search boundary reached:  Kp max
```

Note we are still "maxing out" on K_P .

⁴⁰ **Search 3** Let's start over with the new "best" gain values.

Next we get:

```
[      Balanced] Kp: 26.6  Ki: 3.7  Kd: 162.000
Overshoot: 8.9 percent  Settling Time:  0.55
Search boundary reached:  Kd max
```

Search 4 We need to get bigger on K_D , so let's search again at this value, but narrow the search range to 20.

This time we note that the “SSE” step response is the “nicest” looking even though it ignores %OS and T_S ! Let's continue to search around that optimum:

```
5 [      SSE] Kp: 1147.6 Ki: 16.5 Kd: 36.224
Overshoot: 7.2 percent Settling Time: 0.12
Search boundary reached: Kp max Ki max Kd min
```

Note there still seems to be room to go higher on K_P, K_I .

Search 5 Let's simply start again at these new values. Note that this re-centers the search space around the current optimum.

This time the “Overshoot” step response looks the best. The only problem is the SSE is about 5%. Note that the SSE result has a K_I value which is higher

```
15 [Overshoot = 1.20] Kp: 744.2 Ki: 45.7 Kd: 8.095
Overshoot: 20.0 percent Settling Time: 0.28
Search boundary reached: Kd min
```

```
[      SSE] Kp: 5132.2 Ki: 73.8 Kd: 23.474
Overshoot: 54.0 percent Settling Time: 0.30
Search boundary reached: Kp max Ki max
```

Search 6 Let's this time increase the K_I (because the SSE output improves SSI and has a bigger K_I). Lets start with

$K_p = 744 \quad K_i = 73.8 \quad K_d = 8.9$

Bingo!!

The “Balanced” response meets our specs quite well (Figure 8.4). The “Overshoot” design is pretty good as well.

Complete output:

```
30 [      Ts = 0.550] Kp: 482.5 Ki: 79.4 Kd: 14.915
Overshoot: 3.1 percent Settling Time: 0.55
```

```
[Overshoot = 1.20] Kp: 1114.6 Ki: 205.2 Kd: 14.915
Overshoot: 20.0 percent Settling Time: 0.23
```

```
35 [      SSE] Kp: 166.4 Ki: 299.5 Kd: 24.686
Overshoot: 0.0 percent Settling Time: 0.98
Search boundary reached: Kp min
```

```
40 [      Balanced] Kp: 798.5 Ki: 330.9 Kd: 5.143
Overshoot: 23.7 percent Settling Time: 0.55
Search boundary reached: Ki max Kd min
```

45 Search Time: 2 minutes. N = 1000

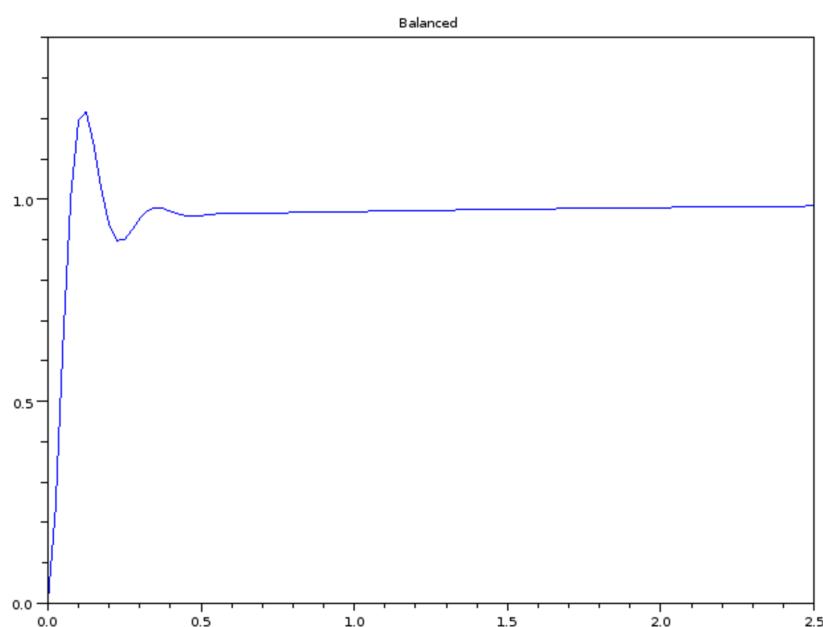


Figure 8.4: Simulation result of final design.

Chapter 9

Conversion to Discrete Time: Tustin's Method

9.1 Problem Statement and Learning Objectives

⁵ 9.2 Overview

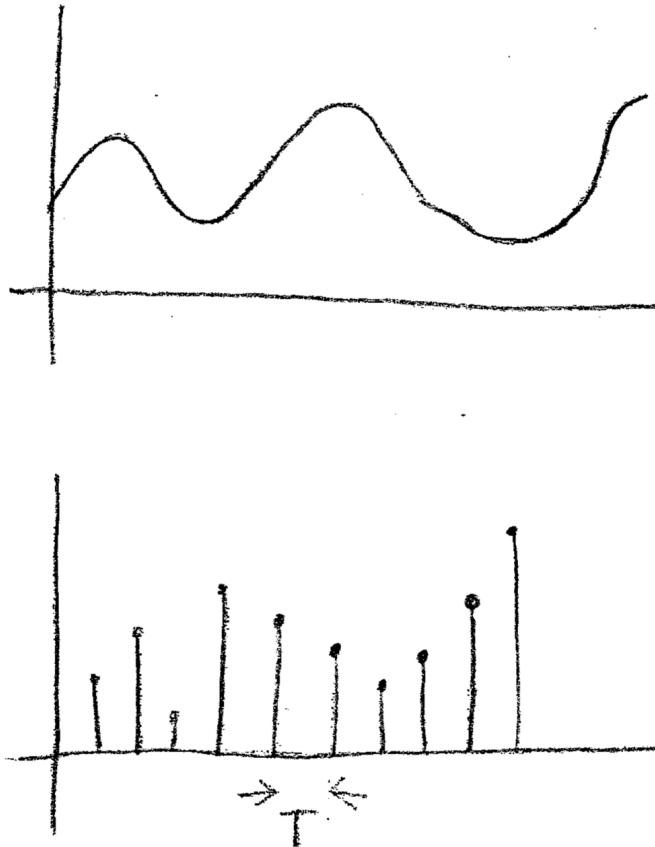
In these notes, we will describe a procedure to convert a continuous time transfer function (such as a controller you might design on the computer) into a line of code you could build into a software application.

In the following, it will help if you have been exposed to some discrete time signals and systems theory (the Z-Transform), but full knowledge of Z-transforms is not required to calculate this conversion.

¹⁰ Our procedure will boil down to the following process for converting your continuous time controller into computer code.

1. Model your system
2. Design your controller as in previous weeks (for example with sisotool)
3. Convert controller from continuous time to discrete time
- ¹⁵ 4. Convert your discrete time controller to a digital filter which can be easily coded.
5. Code and test your filter in the computer.

9.3 Discrete Time and Z transform review/intro



Period, T is the time between samples.

- A discrete time signal, also known as a “sampled data signal”, is a series of unit impulses, scaled by the signal values, and delayed by multiples of T . If there is a continuous signal, $x(t)$, the discrete time version of that signal is

$$x(n) = \sum_n \delta(t - nT) \times x(nT) \quad n = 0, 1, 2 \dots$$

The unilateral Z-Transform (discrete time analog of the Laplace Transform) is

$$X(z) = \sum_{n=0}^{n=\infty} x(n)z^{-n}$$

where z is a complex number (like s), usually represented $Ae^{j\psi}$.

The Z-Transform can be used like the Laplace transform to analyze systems expressed as digital filters.

9.3.1 Discrete and Continuous Comparison Table

Sampled World	Continuous World
$x(n) \quad n = 0, 1, 2 \dots$	$x(t)$
Digital Filter	Differential Equation
$x(n) = 4.2x(n-1) - 2.7x(n-2)$	$\ddot{x}(t) = 6.7\dot{x} - 3.2x + 10$
Digital Convolution:	Continuous Convolution:
$f(n) = \sum_{k=-\infty}^{k=\infty} h(k)x(k-n)$	$f(t) = \int_{\tau=-\infty}^{\tau=\infty} h(\tau)x(t-\tau)d\tau$
Z-Transform	Laplace Transform
Discrete Transfer Function	Continuous Transfer function
$G(z) = \frac{Y(z)}{X(z)}$	$G(s) = \frac{Y(s)}{X(s)}$
Stability: Outside Unit Circle	Stability: Left Half Plane

9.3.2 Sampling Theorem

A powerful theorem due to Nyquist and Claude Shannon, states that

If a continuous signal has bandwidth, B radians per second, and it is sampled (converted to a sampled signal) with a sampling interval $T \leq \frac{\pi}{B}$, then the continuous time signal *can be reconstructed perfectly* from the discrete time signal.

Equivalently,

If a continuous signal has bandwidth, b Hertz, and it is sampled (converted to a sampled signal) at a sampling rate $f_N \geq 2b$, then the continuous time signal *can be reconstructed perfectly* from the discrete time signal.

f_N is called the “Nyquist Rate”. Although this theorem is commonly applied to signals (i.e. sample some music and make it sound good when you play it back) we will use f_N to decide how to sample control systems.

9.4 Tustin's Method

- Suppose we have a continuous time system (i.e. a Laplace Transform transfer function), $H(s)$. Arnold Tustin developed the following way to derive a Z-transform transfer function which is a digital approximation to $H(s)$. His result is

$$H(z) = H(s)|_{s=\frac{2(z-1)}{T(z+1)}}$$

Where T is the sampling time.

In words, to generate a discrete version of $H(s)$, substitute $\frac{2(z-1)}{T(z+1)}$ for s in $H(s)$.

- Let's put these ideas to work on a control system such as that of Figure 9.1. We want to implement the control system in a computer. The Plant ($P(s)$) of course stays outside the computer system since it is NOT a simulation. A control system implemented by computer would thus look like the parts of Figure 9.2 which are inside the dashed line. Referring to Figure 9.2, an input $x(n)$ is provided, for example from a user interface, the actual system output is sensed and sampled and error ($e(n)$) is computed in the computer. The controller $c(n)$ is a digital filter creating the force output $f(n)$ which is applied to the plan by the output device (typically a digital-to-analog converter plus amplifier).

9.4.1 Tustin's method example 1

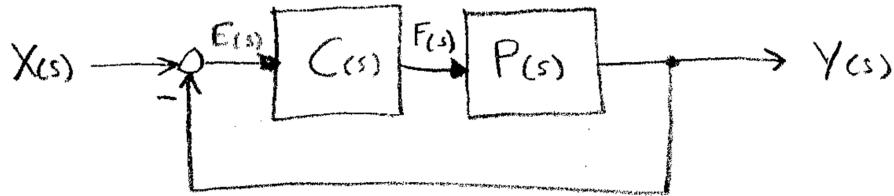


Figure 9.1: Closed loop continuous time control system. Assume that we have designed a transfer function $C(s)$ which gives us a satisfactory system response.

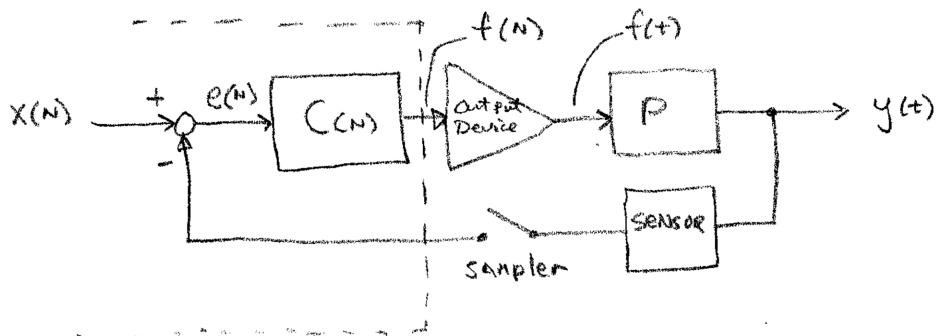


Figure 9.2: Closed loop control system using a computer, an output device, and a sensor to implement the controller. $C(n)$ refers to a discrete time implementation of $C(s)$. Parts inside dashed lines are implemented in a computer such as a microcontroller.

Example 9.1

Apply Tustin's method to convert $C(s)$ into $G(z)$ for $T=1$ sec. where

$$C(s) = \frac{F(s)}{E(s)} = \frac{50}{(s+10)}$$

Applying Tustin's method,

$$\begin{aligned} C(z) &= \frac{F(z)}{E(z)} = \frac{50}{\left(\frac{2(z-1)}{T(z+1)} + 10\right)} \\ C(z) &= \frac{50T(z+1)}{2(z-1) + 10T(z+1)} \end{aligned}$$

Applying $T = 1$,

$$\begin{aligned} C(z) &= \frac{50(z+1)}{2z - 2 + 10z + 10} = 50 \frac{(z+1)}{12z + 8} \\ C(z) &= \frac{50}{12} \frac{(z+1)}{(z + 0.6667)} \end{aligned}$$

As with continuous time transfer functions, we get a ratio of polynomials (this time in z) and we want to normalize them.

Note: For reasons we will see in Section 9.6, $T = 1$ would NOT be fast enough for this control system.

9.4.2 Tustin's method example 2

Example 9.2

Apply Tustin's method to convert $G(s)$ into $G(z)$ for $T=0.01$ sec. where

$$G(s) = \frac{10(s+4)}{(s+0.1)(s+100)}$$

$$G(s) = \frac{10(s+4)}{(s+0.1)(s+100)} = \frac{10(s+4)}{s^2 + 100.1s + 10}$$

$$G(z) = \frac{10\left(\frac{2(z-1)}{T(z+1)} + 4\right)}{\left(\frac{2(z-1)}{T(z+1)}\right)^2 + 100.1\left(\frac{2(z-1)}{T(z+1)}\right) + 10}$$

Let's multiply through by $T^2(z+1)^2$:

$$= \frac{10(2(z-1)T(z+1) + 4T^2(z+1)^2)}{4(z^2 - 2z + 1) + 100.1(2(z-1)T(z+1)) + 10T^2(z+1)^2}$$

Here's a couple of intermediate results we can plug in twice below:

$$2(z-1)T(z+1) = 0.02(z^2 - 1) \quad T^2(z+1)^2 = 10^{-4}(z^2 + 2z + 1)$$

$$\begin{aligned} G(z) &= \frac{10(0.02(z^2 - 1) + 4 \times 10^{-4}(z^2 + 2z + 1))}{4z^2 - 8z + 4 + 2.02(z^2 - 1) + 10^{-3}(z^2 + 2z + 1)} \\ &= \frac{0.204z^2 + 0.008z - 0.1960}{6.021z^2 - 8.002z + 1.981} \\ G(z) &= 0.339 \frac{z^2 + 0.03922z - 0.9608}{z^2 - 1.3290z + 0.3290} \end{aligned}$$

Note: For reasons we will see in Section 9.6, $T = 0.01$ would NOT be fast enough for this control system.

9.4.3 Conversion by Computer

MATLAB

- Matlab has a function called `c2d()` which converts continuous to discrete time systems. It has multiple methods it can use but one of them is Tustin's. The arguments to the `c2d` function are: the system, T , and the name of the method in string form:

Matlab computation:

```
>> g
10
Zero/pole/gain:
 10 (s+4)
-----
 (s+0.1) (s+100)
15
>> sz = c2d(g,0.01,'tustin')

Zero/pole/gain:
 0.033983 (z-0.9608) (z+1)
-----
 (z-0.999) (z-0.3333)
```

Sampling time: 0.01

>>

SCILAB

In Scilab we will apply Tustin's method more directly:

```

5  -->T = 0.01;

-->m = (2/T)*(z-1)/(z+1)
m =
10      - 200 + 200z
-----  

1 + z

-->pz = 10*(m+4)/((m+0.1)*(m+100))
15  pz =
2
- 0.0326503 + 0.0013327z + 0.0339830z
-----  

20
0.3330002 - 1.3323338z + z

```

You can verify that this result is the same as Matlab's.

Note that in this type of computation we should NOT round our results. An exact rule for how much precision we need in the coefficients of the digital filter is beyond the scope of the course. However you should to use at least 6 digits in the problems we will work on.

Now that we have our controller in Z-transform form, we need one more step before we can code it: we need to convert it to a digital filter.

9.4.4 Conversion of discrete transfer function to digital filter

30 First, we “unwrap” the transfer function so that it is now the Z transform of a digital filter. This conversion is actually a simple step. For example, using our first example above,

$$C(z) = \frac{F(z)}{E(z)} = \frac{50}{12} \frac{(z+1)}{(z+0.6667)}$$

becomes

$$F(z)(z+0.6667) = \frac{50}{12} E(z)(z+1)$$

or

$$zF(z) + 0.6667F(z) = 4.1667(zE(z) + E(z))$$

Now, there is an important property of the Z-transform:

$$Z\{x[n-k]\} = z^{-k} X(z)$$

35 In words, shifting a signal in the time domain by k samples is equivalent to multiplying by z^{-k} in the Z domain. Looking at it another way, we have a transform pair:

$$x[n+k] \leftrightarrow z^k X(z)$$

Returning to our example, and multiplying through by z^{-1} will prove useful so we get:

$$F(z) + 0.6667z^{-1}F(z) = 4.1667(E(z) + z^{-1}E(z))$$

and applying the delay property to our unwrapped transfer function (and using linearity) gives:

$$f(n) + 0.6667f(n-1) = 4.1667(e(n) + e(n-1))$$

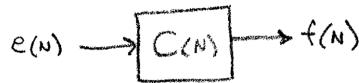
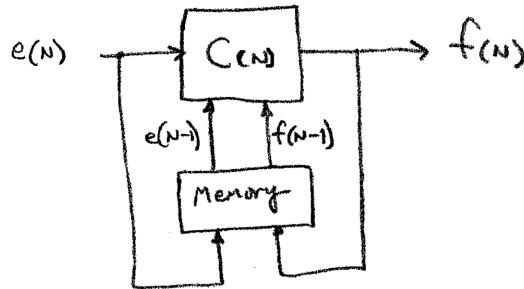


Figure 9.3: Just the controller part of the system.

Figure 9.4: A way to implement the example digital controller. The memory stores previous values for $e(n)$, $f(n)$.

Let's step back a bit and recall that our controller is a relationship between the sampled error in our control system ($e(n)$) and the sampled controller output ($f(n)$, Figure 9.3). Isolating $f(n)$,

$$f(n) = 4.1667(e(n) + e(n - 1)) - 0.6667f(n - 1)$$

This is essentially the line of computer code which defines our controller!
We could implement this equation with the block diagram of Figure 9.4.

⁵ 9.5 Code Example

Let's put this equation into some computer code. We will not worry about many details such as the operating system or exact computer syntax, and we will assume that functions are available to do the I/O for us.

```

1 /* EE447, U. of Washington. Example of basic digital control code */
2 double e, x, f;      // define our system loop variables
3 double ys;           // this will be our sensed y for feedback
4 double en1, fn1;     // these will be e(n-1) etc.
5 double T=1.0;        // sampling time in seconds.
6 double t, t1;        // variables for keeping track of time.
7 en1 = fn1 = 0.0;      // we have to start them with something!
8
9 // loop forever
10 while(1) {
11     t = get_current_time();
12     x = get_command_input(); // get our input from somewhere
13     ys = read_sensor();    // get feedback from our system
14     e = x-ys;              // compute error (H=1)
15
16     f = 4.1667 * (en+en1) - 0.6667*fn1; // compute controller output
17
18     output_to_plant(f); // send controller output to plant
19
20     en1 = e;             // store previous values of e, f
21     fn1 = f;
22
23     t1 = get_current_time();
24     wait(T-(t-t1));     // wait for next sample time
25 }

```

The code runs in an infinite loop (line 10) and executes the controller over and over. First, we note the current time and store it in t . We get the controller input, for example from a user interface or a trajectory generator, in line 12. Then we read a sensor to measure the actual output ys (line 13). After we compute the error, we are ready to compute the controller output (line 16) using the equation we derived above. Note that the coefficients in line 16 are specific to $T = 1.0$ and will be wrong if we arbitrarily change T in the code. After we put the controller output out to the plant (line 18) we store the current values of e , f in the previous values for use in the next cycle. Finally, we figure out how much time has elapsed between lines 11 and 23, and use that to derive the correct argument for a `wait(t)` function so that our timing is accurate.

9.6 Limitations and properties

35 Tustin's method creates a controller which only approximates the continuous time controller. However useful discrete time controllers can be made if the following properties and limitations are taken into account.

- If the continuous time TF is stable then the discrete time version will be stable.
- The DT controller will have the same “features” of its frequency response (number and frequency order of poles and zeros) as the CT controller.
- 40 • Frequencies of the poles and zeros will in general be shifted.
- For frequencies much less than the Nyquist rate (f_N), the approximation will be very accurate.

To make sure the discrete time controller is accurate, make sure that $T \gg 2\pi/pz$ where pz is the frequency of the highest pole or zero in the CT system (including both $C(s)$ and $P(s)$). For example, let

$$C_1(s) = \frac{500(s + 10)}{(s + 1)(s + 100)}$$

For this controller, $pz = 100$ so a suitable sampling frequency would be $f = 1000$ samples/second. Thus we 45 would convert this to discrete time using $T = 0.001$.

