

CONTROL SYSTEM LABORATORY EXPERIMENTS

ECE 486

Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign

TABLE OF CONTENTS

0	Introduction to MATLAB and Equipment	3
1	Simulation Using the Analog Computer	9
2	Digital Simulation	15
3	Digital Simulation of a Closed Loop System	23
4	Introduction to the DC Motor	27
5	PD Control: Analog Computer & Simulink Desktop realtime	41
6	Lead Controller Design	53

Lab Session 0

INTRODUCTION TO MATLAB AND EQUIPMENT

This exercise is intended to give you a brief introduction to MATLAB, the computation software package you will be using throughout the semester in homework and laboratory exercises, and also an introduction to the laboratory equipment.

∞ INTRODUCTION TO MATLAB ∞

These exercises are just to get you familiar with MATLAB; other commands will be introduced as needed. The documents section of the course website has links to several tutorials: <http://courses.engr.illinois.edu/ece486/>. There are helpful control system tutorials for MATLAB at



<http://www.engin.umich.edu/class/ctms/>

As needed, use the resources available, including MATLAB's help facility (try either `helpwin` or `help <command>` or `doc <command>`), for any **boldfaced** words in this exercise.

1. Using **tf**, construct the transfer function:

$$\frac{1}{(s+1)(s^2+0.5s+3)}$$

Plot the **step** response, add a **grid** and use **zoom** to find the steady-state value. Then **zoom out** to the original axis scale.

2. Type **edit** to create a new script file (**.m file*). M-files allow you to save a sequence of MATLAB commands to run later. The `F5` key or  icon (in MATLAB R2013a or newer) can be used to run these files. In addition, a **function** may be defined in its own *.m* files (you may explore differences between script file vs. function file in the documentation); this function may be used just like the built-in functions. The commands **cd** (“change directory”) together with **pwd** (“print working directory”) and **path** (or graphically click  under HOME tab in MATLAB toolbar to check current “path”) may be needed to tell MATLAB where your functions are.
3. At the top of this file, add a comment (a line starting with `%`) with your name and this lab number. Now **clear** all previously defined variables, clear the command window (**clc**), and **close all** previous plots.
4. Add another comment with the steady-state value found in the first part.

5. Re-create the transfer function in the m-file. This time, put the **step** response results into the variables y and t . Compute the peak value of the step response using **max**. **Plot** y and t .
6. Use **colon** to generate a set of times t_2 covering the range of the step response, with 0.1s spacing. Put a semicolon at the end of the line to suppress the printing of this array.
7. Compute $z = 0.33 * (1 - \exp(-0.5 * t_2))$. Set **hold on** for the previous plot and overlay a plot of z versus t_2 , this time using the dotted (':') line style.
8. Label the traces with **legend**, and the axes with **xlabel** and **ylabel**. Put your name as the **title** of the figure. Also put the Greek characters α , ζ , π , and ω in the title (use L^AT_EX format, i.e. ' $\backslash\alpha$ ', ' $\backslash\zeta$ ', ...). **Print** the figure in landscape **orientation**.
9. Print out your m-file and turn it in, along with the last plot (one per group).

TA \checkmark :

☪ INTRODUCTION TO LABORATORY EQUIPMENT ☪

Brief descriptions of the equipment are given, followed by a simple exercise for you to work with them.

I. EQUIPMENT DESCRIPTIONS

- Comdyna GP-6 [Analog Computer] (on table):
Described in Appendices A and B.
- Patch panel:
Allows for easy access to:
 - computer DAQ boards
 - HP power supply (described below)
 - internal voltage sources
 - internal power amplifier for DC motor
- DC motor (on table):
Basic brushed DC motor with an internal *tachometer* [tach] (measures angular velocity) attached, and you can also attach a *potentiometer* [pot] (measures angular position). There are 5 banana jacks integrated with the motor mount: (see Figure 1)
 - Orange: Tach +
 - Gray: Tach -
 - White: Motor chassis
 - Black: Motor -
 - Red: Motor +

- Computer:
Besides MATLAB, the computer has hardware and software for collecting and transmitting analog input and output data by using an internal Data Acquisition [DAQ] board. The DAQ is connected via a ribbon cable to the patch panel described above.
- Hewlett-Packard DC power supply [HP power supply]:
A well-regulated 100W power supply, internally connected to patch panel.
- Agilent oscilloscope [scope]:
A 2-channel oscilloscope that can display and perform measurements on DC (steady) and AC (time-varying) signals. It has lots of functionality; we will only use some of it. Connections are made using coaxial cables with a BNC connector (looks like your cable TV hookup) on one end.
- HP digital multimeter [DMM or DVM]:
This works basically the same as handheld multimeters. Using two cables with banana connector ends, you can measure the voltage, current, continuity, frequency or resistance in circuits.
- HP function/arbitrary waveform generator [signal generator]:
Generates periodic signals in a variety of shapes: sinusoids, square waves, triangular waves, sawtooth waves. You must specify the frequency, amplitude (peak-to-peak), and DC offset (how much the signal will be shifted up).

II. HARDWARE EXERCISE: OPEN-LOOP MOTOR VELOCITY CONTROL

Wire up the system as follows:

1. Connect the signal generator output to the amplifier input on the patch panel.
2. Connect the “Ref -” (black) and “Sig Gnd” (white) jacks to each other on the amplifier input on the patch panel.
3. Connect the amplifier output to the motor (use the gray 3-plug cable and follow color codes).
4. Connect the pot to the motor by loosening the table-mount thumbscrews and pushing the joints together. Then re-tighten the thumbscrews.
5. Apply power to the pot by connecting it to the +5V supply on the patch panel. Connect **HI** to the bottom jack (black) on the pot, and **LO** to the top jack (black also) on the pot.
6. Connect the pot outputs to the scope Channel 1. To do this, connect the red plug to the middle (white) jack and the black plug to the top (black) jack.
7. Connect the tach outputs to the scope Channel 2. To do this, **Tach +** (orange, on the motor mount) should receive the red plug, and **Tach -** (grey, also on the motor mount) should receive the black plug.

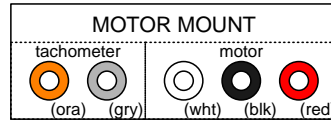


Figure 1. : Banana jacks on motor mount.

Now you have a reference input (signal generator), plant (amplifier and motor), sensors (pot and tach), and viewable output (scope screen). This setup is called an “open-loop” configuration, since the pot and tach data is not used to modify the input to the motor.

1. First, power on the patch panel (flip the switch) and the scope. Rotate the motor by hand and adjust the scope display settings until you get a rolling view of the pot data (Channel 1). (To get the scope into **ROLL** mode, press the Horizontal *Main/Delayed* button and select *Roll*).
2. Describe what you see. What is the output when the motor is still? When it is spinning? How does it change when the pot is powered by the +10V supply instead? Try it.
3. Now we will apply power to the motor. Turn on the signal generator and pick a signal type, amplitude, and offset. **Always keep the voltage below $\pm 5V$ to avoid burning out the motor.** Keep the frequency below 10 Hz or you will not be able to notice its effect. Power on the motor (flip the switch in the Amplifier part of the patch panel), and press the black button to run the motor. Look at both the pot data (Channel 1) and the tach data (Channel 2) on the scope in “roll” mode. Note that you can shift and scale the channels independently of each other.
4. Describe what you see. What kind of output does the tach provide? Why do you see a sawtooth from the pot, even with a DC signal? On the signal generator, play with different input waveforms, amplitudes, and offsets. (If you’re having trouble seeing the data on the scope, you can freeze the data by pressing the *Run/Stop* button.)
5. Finally, explore a major problem with open-loop velocity control. Ask the TA to connect a magnetic break to the system. What is the problem? How would you fix it?

If you wish to backup your data or use it outside of lab, you may save files to the U: drive. This drive maps to your ECE Active Directory. If this drive does not appear after you have logged in two or so times, contact ECE’s Computer and Technology Services (<http://www.ece.illinois.edu/cts>, obsolete, will be directed to a new URL, <http://it.engineering.illinois.edu>) to obtain an Active Directory account.

When you leave the lab each day, remember to *log off* from the PC and *shut down* the equipment and *then shut off* the bench power (the beige toggle switch lights up when it is off). Also *clean up* the area, put away wires, etc.

☞ REPORT ☞

1. On the Lab page of the class website, there is a column for *Report Templates*. Open the template for Lab 0 and follow the directions. You will modify a small MATLAB m-file to automatically calculate M_p , t_s and t_r .

Lab Session 1

SIMULATION USING THE ANALOG COMPUTER

The objective of this experiment is to explore second order systems. We will use an analog computer to simulate a basic second-order dynamic system. A thorough understanding of the response of such systems is essential for successful control system design.

🔔 PREPARATION 🔔

READINGS:

- ▷ Appendix A: The Analog Computer: Theory and Practice;
- ▷ Appendix B: The GP-6 Analog Computer;
- ▷ G. F. Franklin, *et al.*, *Feedback Control of Dynamic Systems*, 7th or 6th Ed., Sec. 3.8, 7.4.1 (3rd or 4th Ed., Sec. 2.6., 7.2.1 or 5th Ed., Sec. 3.6, 7.4.1).
- ▷ The PC will control the test instrument (an oscilloscope) through the software package VEE. Your TA will explain how to use VEE. The PC will also be used for data logging and processing. By default, the data will be stored on your **U:\ drive**.

PRELAB:

Using the symbols and instruction given in Appendix A, prepare an analog computer block diagram for the transfer function:

$$H(s) = \frac{Y(s)}{U(s)} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}.$$

We will simulate this on the analog computer using *normalized time*. This is a common technique for analog simulations; it allows us to speed up extremely slow processes or slow down those which complete in a flash. This allows us to perform useful analysis in the lab.

Steps:

- (a) Write the differential equation for the transfer function.

- (b) Convert the differential equation to normalized time using the transformation: $\frac{\tau}{\omega_n} = t$ as shown in the Appendix and Franklin, *et al.*, Sec 3.8.2, 7th Ed. (6th Ed., Sec 3.8.2; 5th Ed., Sec 3.6.23; 3rd or 4th Ed., Sec 2.6.4).

- Express $\frac{dy}{dt}$ and $\frac{d^2y}{dt^2}$ in terms of $\frac{dy}{d\tau}$ and $\frac{d^2y}{d\tau^2}$.
- Now convert your differential equation to one in terms of τ .

- (c) Solve for the highest derivative and integrate to solve for $y(\tau)$.

- (d) Use the “solving for the highest derivative” method on page vii of Appendix A to create an all-integrator block diagram for the differential equation. Assume zero initial conditions and that $u(\tau)$ is a 2V unit step.

Make sure your integrators have only gains of 1, 10, or 0.1, since these are the only possible values on the Comdyna GP-6. Also be sure to consider the *signs* of the gains. Arrange for easy adjustment of ζ from 0 to 2, using only one potentiometer (the GP-6 potentiometers have a range of 0 to 1). Using the Comdyna’s “+” reference (10V), generate a 2V step for the input signal.

- (e) Use one of the patch panel diagram layouts at the very end of this lab manual or on the lab website to show the connections needed to implement your circuit. Note that the “-” reference terminals are actually -10V; use the two ground terminals if 0V is needed.

▷ *Do not* use the current multiplier in Figure 1.1 for multiplying voltages.

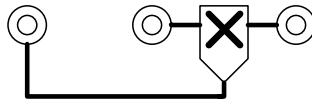


Figure 1.1. : Current Multiplier (do not use)

▷ Tie the output of each unused op-amp back into one of its input resistors. This causes negative feedback that reduces any random noise in the analog computer.

✱ LABORATORY EXERCISE ✱

*Do not turn on power on the patch panel until instructed to do so.
Never “hot wire” the system.*

First make the following connections:

1. Start VEE and load the file N:\labs\ECE486\486Lab1.vxe (*Note:* by default only *.vee files are shown, you need to choose “all files” from drop-down list to see *.vxe). When the program loads in the VEE environment the “Program Explorer” and “Properties” windows may also open. Close both of these windows to allow you to see the entire VEE program.

2. Connect wires for a second order system on the analog computer using your patch panel wiring diagram. *The appropriate wires are in the long drawer under your bench.*
3. Connect Channel 1 of the oscilloscope to y , the output of the analog computer. Connect the black lead to the analog computer ground.
4. Connect Channel 2 of the oscilloscope to any **OP** jack on the analog computer (this will be the trigger input). Connect the black lead to the analog computer ground.

Turn on power to the Oscilloscope.

5. In Agilent VEE, verify that the Scope parameters are set to the following:

$$\begin{array}{l}
 \text{Channel1} \left\{ \begin{array}{l} \text{V/div} \rightarrow 1\text{V} \\ \text{Offset} \rightarrow 0\text{V} \\ \text{Coupling} \rightarrow \text{DC} \\ \text{Band-Width Limit} \rightarrow \text{ON} \end{array} \right. \qquad \text{Channel2} \left\{ \begin{array}{l} \text{V/div} \rightarrow 5\text{V} \\ \text{Offset} \rightarrow 0\text{V} \\ \text{Coupling} \rightarrow \text{DC} \\ \text{Band-Width Limit} \rightarrow \text{ON} \end{array} \right. \\
 \\
 \text{Trigger} \left\{ \begin{array}{l} \text{Level} \rightarrow -6.0\text{V} \\ \text{Source} \rightarrow \text{CHANNEL2} \\ \text{Slope} \rightarrow \text{NEGATIVE} \\ \text{Noise Reject} \rightarrow \text{ON} \\ \text{High Freq. Reject} \rightarrow \text{ON} \end{array} \right. \qquad \text{Time} \left\{ \begin{array}{l} \text{Base} \rightarrow 2 \text{ S/div} \\ \text{Reference} \rightarrow \text{LEFT} \end{array} \right.
 \end{array}$$

6. Send these parameters to the scope by pressing the **Send New Scope Parameters** button in Agilent VEE. Whenever you make a change to the scope parameters you will need to press this button to send the information to the scope. *Note:* To change the path and filename of your data file in VEE, click on the *To File:* button in the To File box, type in the new path and click Ok.
7. For your first run set ζ to 2.0 by adjusting the appropriate potentiometer. To run the simulation on the analog computer first push the **IC** (initial condition) button to put the integrators in their initial state. Then the **OP** (operate) button to start the simulation. If you do not see the response starting to display on the oscilloscope the scope missed the trigger. Try again by switching back and forth between **IC** and **OP**. For your report plots, it will look nice if your responses all start at the same point in time. So before you press the **Collect Data from Scope** button make sure you have a response that starts one time division in from the edge.

Have the TA check your data.

8. For $H(s)$ as in the prelab, obtain step responses (zero initial conditions) for $\zeta = 2.0, 1.5, 1.0, 0.8, 0.7, 0.5, 0.3, 0.2$.
9. In MATLAB import the data by changing into the directory where the data is stored and simply typing the name of the file (without the “.m”). Each data set will be stored as a variable in order: $y1, y2$, etc. Each variable has

two columns—the first contains the time vector, and the second the voltage at each time instant. For example, to plot the y_1 data versus time, type `plot(y1(:,1),y1(:,2))`.

10. Display all responses (y vs. τ) on a single plot. As check points for your simulation, verify that these are close to those given in Franklin, *et al.*, Fig. 3.19(b) (Fig. 3.18(b) in 6th Ed., Fig. 3.16(b) in 5th Ed., Fig. 3.23(b) in 4th Ed., Fig. 3.12(b) in 3rd Ed.). Note that $\tau = \omega_n t$ is the scaled (normalized) time variable. *Tip*: cyclically use the ‘-’, ‘:’, ‘-.’, and ‘--’ line styles so the different traces are identifiable. Colors do not really work on the black-and-white laser printer. Print this plot for the report, and make sure everyone has a copy of the `lab1.m` file.

TA \checkmark :

☞ REPORT ☞

1. Include the plots obtained in lab. Compute estimated and observed values for M_p , t_r and t_s (5%) and compare; discuss how they vary with ζ . Also, print one sample response with relevant points for calculating M_p , t_r , and t_s marked. Use the (approximate) equations given below to estimate values:

$$M_p = \exp\left(\frac{-\pi\zeta}{\sqrt{1-\zeta^2}}\right) \quad \text{if } 0 < \zeta < 1 \quad (= 0 \text{ if } \zeta \geq 1 \text{ by def.})$$

$$t_r \approx \begin{cases} \frac{1.2 - 0.45\zeta + 2.6\zeta^2}{\omega_n} & \text{if } 0 < \zeta < 1.2 \\ \frac{4.7\zeta - 1.2}{\omega_n} & \text{if } 1.2 \leq \zeta < 3 \end{cases}$$

$$t_s \approx \begin{cases} -\frac{0.5}{\zeta\omega_n} \ln((1-\zeta^2)/400) & \text{if } 0 < \zeta \leq 0.69 \\ \frac{6.6\zeta - 1.6}{\omega_n} & \text{if } 0.69 < \zeta < 3 \end{cases}$$

Notes:

▷ These equations are different from those given in Franklin, *et al.* Rise times are 10%–90% as usual. Settling times are to within 5% (instead of 1%) because of noise present in the experimental data. **We will use the above formulas for all lab work.** However, check with your instructor before using them on homework or tests.

▷ Notice that t_r varies continuously with respect to ζ , but that is not the case for t_s . In fact, the use of two approximations represents only the most significant discontinuity in t_s .

▷ In this lab, since we used normalized time, we may use $\omega_n = 1$ rad/s.

2. How does ζ affect pole locations? Derive the equation. Include in your discussion the effects of pole locations on M_p , t_s and t_r for both underdamped and overdamped/critically damped second-order systems.
3. Investigate the effects of approximating an overdamped 2nd order system with a 1st order system. The approximation will be done by using a transfer function with only the pole that is closer to the origin:

$$H_1(s) = \frac{p_1 p_2}{(s + p_1)(s + p_2)} \Rightarrow H_2(s) = \frac{p_{\min}}{s + p_{\min}}$$

Consider this situation for $\zeta = 1.5$, $\zeta = 5$ and $\zeta = 40$. Explain the similarities and differences between the step responses of the 2nd order systems and their 1st order approximations. How does the value of ζ affect the accuracy of the approximation? *Three graphs in MATLAB would be very helpful!*

Lab Session 2

DIGITAL SIMULATION

Digital simulation packages are an essential part of control system design. Such packages are used routinely in control system analysis and design. This lab introduces SIMULINK, the digital simulation package sold with MATLAB. SIMULINK will be used to study the dynamic response of a second order system, in both block-diagram and state-space forms, together with the effects of an additional pole and an additional zero. The characteristic responses of these systems provide important insights into control system design.



☞ PREPARATION ☞

READINGS:

- ▷ Appendix A: See section IV-c “Representing Systems with Zeroes”;
- ▷ G. F. Franklin, *et al.*, *Feedback Control of Dynamic Systems*, 3rd, 4th, 5th, 6th, or 7th Ed., Sec. 3.5.

PRELAB

$$H_1(s) = \frac{Y(s)}{U(s)} = \frac{25}{s^2 + 6s + 25}$$

1. Develop an all-integrator block diagram for the transfer function $H_1(s)$. Do not normalize the time. What are ζ and ω_n ? What are the poles of the system? Include this work in your prelab.
2. In this step, you will simulate your all-integrator block diagram using SIMULINK:
 - (a) At the MATLAB command line, type `simulink`, or you can click SIMULINK library icon  under HOME tab in MATLAB toolbar to start it graphically.
 - (b) Click “Blank Model.” Once a blank SIMULINK file pops up, click the icon , this will open the SIMULINK library browser window.
 - (c) Open a new model by clicking on the New Model icon or by going to the File menu, under New, and selecting Model. A model window will open.
 - (d) Open the Continuous library by clicking on the *Continuous* tab in the left pane of library window.

- (e) Find the *Integrator* block in the library and drag-and-drop it into the model window using the left mouse button. Or you can search with keyword of block “integrator”.
- (f) Copy the *Integrator* block either by using the Copy and Paste commands or by dragging the *Integrator* within the model window with the right mouse button.
- (g) Open the Math library by clicking on the *Math Operations* icon in the library window, and drag a *Gain* block and a *Sum* block into the model window.
- (h) Open the *Sources* library and drag a *Step* block into the model window.
- (i) Open the *Sinks* library and drag a *To Workspace* block into the model window.
- (j) Copy the blocks as needed and connect the blocks so that the model resembles your all-integrator block diagram. You can change the direction of the blocks by right-clicking on the block and selecting the Format menu. To connect the blocks, drag the output port of one block (an outward-pointing arrow) to the input port of another (an inward-pointing arrow). Change the summer inputs by double-clicking on it and changing the signs in the “List of Signs” box. Experiment with the entry in this box.

Note: You can also connect two blocks in SIMULINK by clicking on the source block, holding *Ctrl*, and clicking on the target block.

- (k) Find and add a *To Workspace* block to the output of the first integrator. (Optionally, you can instead use “scope” block, the settings are similar to *To Workspace*: choose data type “array” and set variable name. The variable is two-dimensional. The first column of the variable stores time, the second column response data.)
- (l) Label the output of the first integrator “*y_dot*” by double-clicking on the line representing the output. Type the text in the box that appears. Click outside the text box to save the text.
- (m) Change the value of the *Gain* blocks by double-clicking on the blocks, then entering the appropriate values.
- (n) Double-click on the *Step* block. Set the parameters to:
 - ▷ Step Time: 0
 - ▷ Initial Value: 0
 - ▷ Final Value: 1
 - ▷ Sample Time: 0
- (o) Double-click on the *To Workspace* block connected to the output. Set the parameters to:
 - ▷ Variable name: *y*
 - ▷ Save format: Array
- (p) Set the parameters on the *To Workspace* block connected to *y_dot* to:

- ▷ Variable name: y_dot
 - ▷ Save format: Array
- (q) Set the simulation parameters, by clicking on the Simulation menu and choosing Configuration Parameters, to:
- ▷ Stop Time: 3 (chosen to capture the main response)
 - ▷ Type: Variable-step
 - ▷ Solver: ode45
 - ▷ Max Step Size: 0.001 (smaller max step sizes give smoother responses, but take longer to compute)
 - ▷ Click on *Data Import/Export* and uncheck the *Limit data points to last* box. (*Comment: Whenever you have to deal with data export in Simulink[®], always remember doing this. This option will be encountered numerous times especially in later labs.*)
- (r) Run the simulation by clicking on the Simulation menu and Start.
3. In the MATLAB workspace, you will find three variables: $tout$, y , and y_dot . SIMULINK always saves the time vector as $tout$. Plot the variables y and y_dot as two subplots in one figure (see **subplot**); use $tout$ as the time axis for both. Label appropriately. **Print** two copies of this graph; include one with your prelab and save the other for your lab report. In addition, **print** the SIMULINK block diagram (the model window) to turn in with your prelab.

🎯 LABORATORY EXERCISE 🎯

PART I: STATE SPACE MODEL OF $H_1(s)$

1. Open a new SIMULINK model.
2. Drag in a *State-Space* block from the Continuous library.
3. In the MATLAB workspace window, set the following variables:

$$\mathbf{A} = \begin{bmatrix} -6 & -25 \\ 1 & 0 \end{bmatrix}, \mathbf{B} = \begin{bmatrix} 25 \\ 0 \end{bmatrix}, \mathbf{C} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \mathbf{D} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

4. Double-click on the *State-Space* block, and set the \mathbf{A} , \mathbf{B} , \mathbf{C} , \mathbf{D} parameters to “A”, “B”, “C”, “D” respectively (without the quotes). This causes SIMULINK to look up the values in the MATLAB workspace when the model is run.
5. Open the Sources library and drag a *Step* block into the model window. Connect the *Step* block to the input of the *State-Space* block, and change the parameters to those used in the prelab.
6. Open the Signals Routing library and drag a *Demux* block into the model window. Connect the *State-Space* output to the *Demux* input.

7. Open the Sinks library to find the *To Workspace* block. Connect each output of the *Demux* to a separate *To Workspace* block. Set the variable name of the top *To Workspace* block to *y_dot*, and the bottom one to *y*. Make sure both *To Workspace* blocks are set to an Array type save format (as in the prelab).
8. Set the Simulation Parameters as in the prelab, and run the simulation.
9. Make a plot containing subplots of both *y* and *y_dot*. Label and print this for your report. Set *y1=y*, *y_dot1=y_dot*, and *tout1=tout* to save these responses for later.

PART II: EFFECTS OF AN EXTRA ZERO

We will now use the information from Appendix A section IV-c to introduce a zero into the all-integrator block diagram from the prelab. Sketch the block diagram by hand before implementing it in SIMULINK. This diagram will be similar to Figure 16 as shown in the appendix.

$$H_2(s) = \frac{25 \left(1 + \frac{s}{\alpha\zeta}\right)}{s^2 + 10\zeta s + 25}$$

1. Modify your all-integrator block diagram so that the zero in $H_2(s)$ is added. Remove the *y_dot* “To Workspace” block. In the block diagram, use variables like “a” or “zeta” for α and ζ ; this way you can easily set their values in the MATLAB workspace.
2. Notice that the zero is added at $s = -\alpha\zeta$, with ζ as found in the prelab. With this ζ , simulate the system with the real part of the zero in various regions. We will rename the variables after each simulation so they are not overwritten.
 - ▷ ten times further left than the poles (save as *y2a* and *tout2a*)
 - ▷ on the real axis directly between the poles (save as *y2b* and *tout2b*)
 - ▷ halfway between the poles and the origin (save as *y2c* and *tout2c*)
 - ▷ in the right-half plane, with $\alpha = -3$ (save as *y2d* and *tout2d*)
 - ▷ to see the trend, also try $\alpha = -30$ (save as *y2e* and *tout2e*)

Signal	zero location	α
<i>y2a</i>		
<i>y2b</i>		
<i>y2c</i>		
<i>y2d</i>		
<i>y2e</i>		

3. Overlay all these responses along with *y1* on a single graph and label appropriately. Print a copy for the report. Note that in general $tout2a \neq tout2c$; since we are using variable-step integration, MATLAB is free to choose which times it evaluates. Remember this; a common mistake is to use the wrong time data when plotting. TA \checkmark :

PART III: EFFECTS OF AN EXTRA POLE

$$H_3(s) = \frac{25}{\left(1 + \frac{s}{\alpha\zeta}\right) (s^2 + 10\zeta s + 25)}$$

1. Remove the zero from the model in Part II and add an extra pole instead, as in $H_3(s)$. Sketch the block diagram by hand before implementing it in SIMULINK.

2. The extra pole is added at $s = -\alpha\zeta$, with ζ as found in the prelab. With this ζ , simulate the system with the real part of the added pole:
- ▷ ten times further left than the poles (save as $y3a$ and $tout3a$)
 - ▷ on the real axis directly between the poles (save as $y3b$ and $tout3b$)
 - ▷ halfway between the poles and the origin (save as $y3c$ and $tout3c$)
 - ▷ We do not want a pole in the right half plane because that is unstable. Try it and see what happens.

Signal	pole location	α
$y3a$		
$y3b$		
$y3c$		
$y3d$		

3. Overlay all the stable extra pole responses on a single graph and label appropriately. Be sure to include the response $y1$ from the first portion of the lab. Print.

TA \checkmark :

✧ REPORT ✧

I: STATE SPACE MODEL OF $\mathbf{H}_1(s)$

1. Compare the plots of y_{dot} and y as obtained in Part I of the lab to the plots made in the prelab. Do they appear to come from the same system? Attach the plots to your report, making sure they are clearly labeled.

II: EFFECTS OF AN EXTRA ZERO

1. How are M_p , t_r , and t_s affected by the location of the zero? Answer this by measuring these values from the Part II plot and tabulating them. Include the values from Part I for comparison. Attach the plot to the end of your report.
2. A zero in the right half plane is called a non-minimum phase zero. What is interesting about the plot for this case?
3. Take $H_2(s)$, set ζ to the value found in the prelab, and separate the numerator terms to make a sum of two fractions. What does each term represent? As α changes, which term dominates? Be sure to consider the limits as α approaches 0 or ∞ . What happens when α is negative?

III: EFFECTS OF AN EXTRA POLE

1. How are M_p , t_r , and t_s affected by the location of the additional pole? Answer this by measuring these values from the Part III plot and tabulating them. Include the values from Part I for comparison. Attach the plot to the end of your report.
2. Take $H_3(s)$, set ζ to the value found in the prelab, do a partial fraction expansion to separate the extra pole from the original system, and then separate the numerator as before. The expansion should look something like:

$$\begin{aligned} H_3(s) &= \frac{25}{\left(1 + \frac{s}{\alpha\zeta}\right)(s^2 + 10\zeta s + 25)} \\ &= \frac{k_1}{1 + \frac{s}{\alpha\zeta}} + \frac{k_2 s + k_3}{s^2 + 10\zeta s + 25} \\ &= \frac{k_1}{1 + \frac{s}{\alpha\zeta}} + \frac{k_2 s}{s^2 + 10\zeta s + 25} + \frac{k_3}{s^2 + 10\zeta s + 25} \end{aligned}$$

Solve for k_1 , k_2 , and k_3 in terms of α . What does each of these three terms represent? How do k_1 , k_2 , and k_3 change as α changes? Be sure to consider the limits as α approaches 0 or ∞ .

Lab Session 3

DIGITAL SIMULATION OF A CLOSED LOOP SYSTEM

In this experiment SIMULINK is used to study a closed loop system for speed control of a magnetic tape drive such as a cassette player. You will compare different controllers for their effectiveness in simultaneously accomplishing disturbance rejection and a smooth step response.

PREPARATION

READINGS:

- ▷ G. F. Franklin, *et al.*, *Feedback Control of Dynamic Systems*, 3rd or 4th Ed., Sec. 4.1, 4.1.1. These pages are copied for you and attached at the end of this manual. In 7th or 6th Ed., see Sec. 4.2.2 for disturbance rejection.

PRELAB:

Modeling: A diagram of the system is shown below

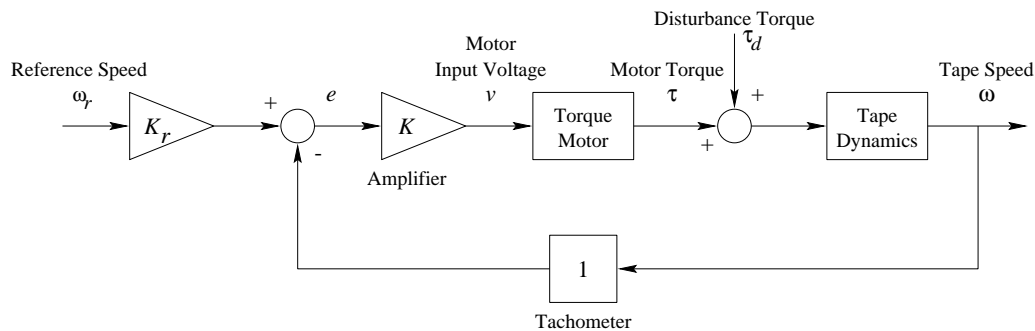


Figure 3.1. : Closed-Loop System with Unity Feedback.

The tape dynamics are modeled as a rotational mass-damper system:

$$J\dot{\omega}(t) + B\omega(t) = \tau(t) + \tau_d(t),$$

where $J = 0.25[\text{kg} \cdot \text{m}^2]$, $B = 3[\text{N} \cdot \text{m} \cdot \text{s}]$. The torque motor transfer function is determined by an open-loop test *with the motor detached from the system*. A one-volt step is applied (zero initial conditions), and the resulting motor torque is

$$\tau(t) = 5(1 - e^{-3t}), \quad t \geq 0.$$

The amplifier is a non-inverting gain, K . It is assumed that the tape speed sensor (tachometer) dynamics can be neglected.

Design: We want to have a closed loop system whose steady-state response from a disturbance is small, and whose step response from a reference input has low overshoot.

- Obtain an all-integrator block diagram for the system in figure 3.1. Show the intermediate variables (τ , v etc.). Note that the motor torque is the convolution of a step input with the motor's transfer function.
- For the closed-loop system, find the transfer functions $\Omega(s)/\Omega_r(s)$ (with $\tau_d = 0$) and $\Omega(s)/T_d(s)$ (with $\omega_r = 0$). Use numerical values where given. (*Hint:* remember to make your lead coefficient 1.)
- For what values of K is the closed loop system stable?
- Controller 1:** Suppose the disturbance torque is a unit step [N · m]. Find the range of K such that the steady state output in tape speed due to τ_d (use the transfer function you get above with $\omega_r = 0$) is ≤ 0.01 rad/s.
- Express the desired value of K_r , in terms of K , such that the steady-state output speed for a unit step in ω_r is 1. ($\tau_d = 0$)
- Using the minimum value of K that you found in (d), what are the values of the closed loop ζ and ω_n ? Consider the anticipated response $\omega(t)$ to a step at ω_r ($\tau_d = 0$). What will be the expected t_s , t_r , and M_p ? Use the equations on page 13.
- Controller 2:** Suppose, alternatively, that K is selected to produce a low overshoot ($\zeta \geq 0.75$). Find the range of K that meets this specification. If K is selected such that $\zeta = 0.75$, what will be the steady-state disturbance response to a unit step? What M_p do we achieve? Can we achieve both our desired disturbance rejection and our desired overshoot?
- Controller 3:** To improve the response, suppose we add *derivative feedback* (measured by an *accelerometer* in this case) as shown:

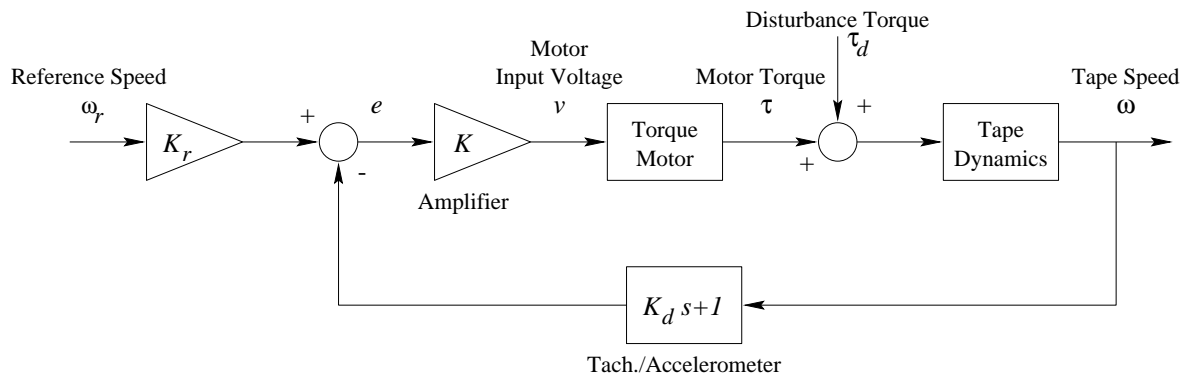


Figure 3.2. : Closed-Loop System with Derivative Feedback.

Compute again the new closed-loop transfer functions $\Omega(s)/\Omega_r(s)$ (with $\tau_d = 0$) and $\Omega(s)/T_d(s)$ (with $\omega_r = 0$). Give the ranges of K and $K \cdot K_d$ for which the closed-loop system is stable.

- (i) Select K and K_d such that the steady state output in tape speed due to τ_d (τ_d is a unit step as before) is 0.01 rad/s and $\zeta = 0.75$. What are the resulting M_p , t_r , and t_s ?

✂ LABORATORY EXERCISE ✂

1. Develop an all-integrator SIMULINK block-diagram for the three closed-loop control systems. It is a good simulation practice to retain explicitly in the simulation the various subsystems - actuator, plant, amplifiers, etc. Your diagram should do so, and you should label the inputs and outputs, ω , τ , v , e , of the subsystems. It is possible to make only one diagram, and simply use MATLAB variables in the SIMULINK blocks for the different cases. Do *not* use differentiators in your diagram. Instead, route the required states around the integrators. See appendix A, section IV-c for an example of this.
2. Obtain and print the following time responses for each of the three controllers:
 - $\omega(t)$ for the disturbance response to a unit step in τ_d ($\omega_r = 0$). For comparison, overlay and label the plots for the different controllers.
 - $\omega(t)$ for the reference response to a unit step in ω_r ($\tau_d = 0$). Remember to use the appropriate value of K_r for each controller. Overlay again for comparison. You should end up with six graphs on two plots.

Note: For comparison, in short, in this lab you end up with two figures: in figure (1), overlay *three* graphs of responses due to unit step disturbances; similarly in figure (2), overlay *three* graphs of responses due to unit step references.

TA \checkmark :

✂ REPORT ✂

1. Include your prelab calculated values and experimental data (time responses) obtained. Compare the values of M_p , t_r , and t_s as calculated in the prelab with the values obtained from MATLAB. Which controllers met the specifications (steady state disturbance response ≤ 0.01 rad/s and $\zeta \geq 0.75$)?
2. For the system in Figure 3.1, derive the relationship between the steady-state error ($e_{ss} = \omega_r - \omega$) and the natural frequency, ω_n . Consider the error as a function of both ω_r and τ_d , when both are step inputs. Since the system is linear, superposition allows the two components of e_{ss} to be calculated separately and then summed.
3. For controller 3, solve for ζ and ω_n in terms of the gains K and K_d . From these equations, sketch a plot of how the pole locations change as $K_d > 0$ increases in value.

Lab Session 4

INTRODUCTION TO THE DC MOTOR

Several basic features of control systems can be understood by carefully examining DC motor control as a case study. Also, the DC motor is an important component in a myriad of control systems. This DC motor experiment, the first of three, emphasizes system modeling in the time domain. The motor we will study is shown in Figure 4.1, and Figure 4.2 shows a functional block diagram.

Figure 4.3 describes the motor dynamics. This diagram will be used to develop the motor transfer function.

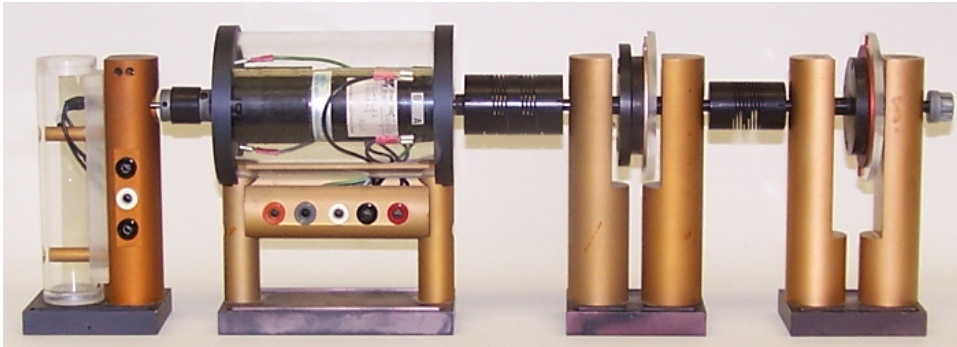


Figure 4.1. : The DC motor plant, showing the potentiometer (left), motor with integrated tachometer (center), and double flywheel (right).

System parameters:

Parameter	Symbol	Units
Potentiometer gain	K_{pot}	Volts/rad
Tachometer gain	K_{tach}	Volts · s/rad
Armature resistance	R_a	Ohms
Armature inductance	L_a	Henries
Back EMF constant	K_V	Volts · s/rad
Motor torque constant	K_τ	N · m/amp
Motor inertia	J_M	kg · m ²
Load (flywheel) inertia	J_L	kg · m ²
Total inertia (= $J_M + J_L$)	J	kg · m ²
Coulomb (static) friction	c	N · m
Viscous (sliding) friction	b	N · m · s/rad

∞ PREPARATION ∞

READINGS:

A basic description of the system components may be found in B. C. Kuo, *Automatic Control Systems*, 7th edition. (A pdf file is available on the lab web site. The book is also on reserve in the engineering library.) See Sections 4.5 and 4.6, which cover potentiometers, tachometers and DC motors.

We will follow Franklin *et al.* for some basic methods in system modeling in the time domain. See Franklin *et al.* 7th or 6th Ed. Sec. 3.7 (Sec. 3.7 in 3rd Ed., Sec. 3.8 in 4th Ed.) for an introduction to system identification using transient response data. These pages are copied for you and attached at the end of this manual.

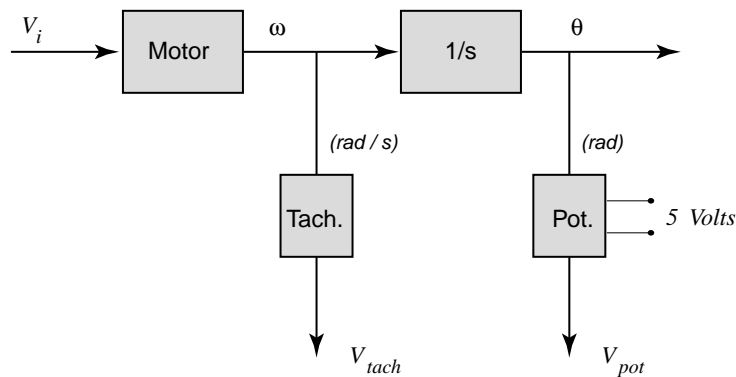


Figure 4.2. : Motor Block Diagram.

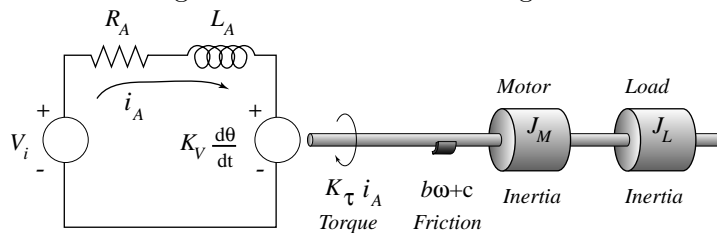


Figure 4.3. : Motor Schematic Diagram.

PRELAB:

- (a) The DC motor, *with the flywheel clamped* ($\omega = 0$), is described by the electrical schematic of Figure 4.4. The resistance R_s has been added to obtain voltage measurements, so that the current's time response can be viewed on an oscilloscope. Find $V_o(s)/V_i(s)$ in terms of R_a , R_s , and L_a . What is the *electrical time constant*, τ_e , for the motor?
- (b) The electrical parameters R_a and L_a can be determined from the step response of the motor with its flywheel clamped, assuming R_s is already known. Derive the time-domain step response of $V_o(t)$. Use this formula to show how τ_e can be

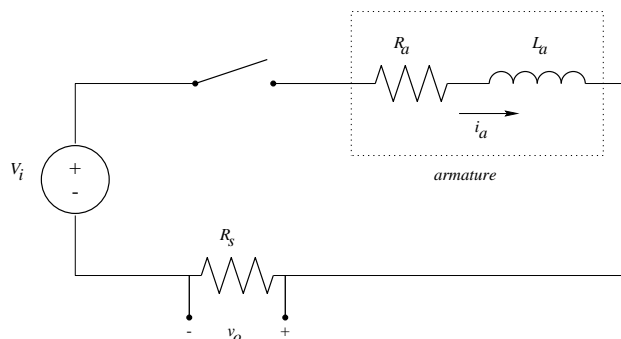


Figure 4.4. : Motor Electrical Diagram: Flywheel clamped. *Note:* The final model of the motor we will develop will not have R_s . This is only used for identification.

obtained from a linear fit of $\ln(V_{ss} - V_o(t))$ where V_{ss} is the steady-state output voltage. This will give a reliable estimate in spite of noise.

Load the MATLAB data in the file `prelab4.mat` from the website. The variable `Elect` (“Elect” is short for Electrical time constant data) contains the time and voltage data of a step response in columns 1 and 2, respectively (so `time=Elect(:,1)` and `voltage=Elect(:,2)`).

There are two ways to get a linear fit on this processed data. On the command line, use `polyfit` to obtain the coefficients; for example, `coeffs=polyfit(x,y,1)` fits the model $y \approx \text{coeffs}(1) \cdot x + \text{coeffs}(2)$. To get the fit graphically, plot the data; then go to the plot’s menu *Tools* → *Basic Fitting* and select the options *linear* and *show equation in figure window*.

Regardless of the method used to fit the data, be careful to get a fit on *linear* data. If V_{ss} is wrong, the plot will be curved instead of straight; vary the value of V_{ss} until the first part of the curve is linear. Now notice that there is a sharp bend in the curve around $t = 0.5$ ms; this marks where the circuit achieved steady-state and the inductance is no longer visible. Thus you should only fit the data between $t = 0$ and $t \approx 0.5$ ms (you can pick the exact range; *Note:* if you do not know how to find the exact range, plot the *whole* range of data then you can decide which part is linear); including the other data will corrupt your results. For example, you can have Matlab only plot the first 10 data points by typing `plot(Elect(1:10,1), Elect(1:10,2))`.

Obtain τ_e and L_a from this data, assuming that $R_s = 2.5\Omega$ and $R_a = 3.3\Omega$. Print out the log plot for your prelab. As a sanity check for your results, $L_a \approx 1\text{mH}$.

- (c) Now consider the DC motor with the rotor free to turn. The parameters K_V , K_τ ($= K_V$ in SI units), b , and c can be obtained from the steady-state response to a known step input V_i . Derive the steady-state equations and solve them to find these parameters from multiple trials.

Suppose the results from two experiments were as follows:

V_i	steady-state i_a	steady-state ω
5 Volts	0.249 A	77.0 rad/s
10 Volts	0.304 A	163 rad/s

Using $R_a = 3.3\Omega$ and $L_a = 1\text{mH}$, what are the values for $K_\tau = K_V$, b , and c ? Use the average value obtained for K_τ as the “correct” value.

- (d) The mechanical time constant, and hence the total moment of inertia, J , can be determined by opening the motor circuit ($i_a = 0$) while the motor is rotating and observing the resulting decay in speed. The situation is shown in Figure 4.5.

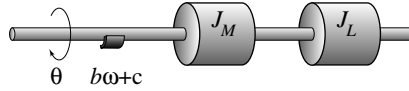


Figure 4.5. : Motor Mechanical Diagram: open electrical circuit.

With the motor spinning only in one direction, the Coulomb friction will not change sign, and we can consider it to be a constant (a **step function** in the Laplace Domain). The dynamics are then described by the differential equation:

$$J\dot{\omega} + b\omega + c = 0.$$

Find $\omega(t)$ in terms of J , b , c , and $\omega(0)$. Explain how to find J if the other parameters are known (similar to part (b)). Illustrate your method on the inertial data saved as the variable `Iner` in the `prelab4.mat` data file. `Iner` contains the time and angular velocity data of the motor in columns 1 and 2, respectively. Use the friction values found in part (c). Print and turn in any plots needed by your method.

- (e) If we assume the motor has a strictly positive angular velocity, we can get a linear model by disregarding the Coulomb friction. (i.e., here you can discard c term but remember to keep b term for viscous friction. We will address this again in lab 5.) Using the numerical values above, obtain the resulting second order transfer function, $\Omega(s)/V_i(s)$ for the motor. Then compute the poles of both the second-order transfer function and its first-order approximation. Use the pole location to discuss the difference between the original transfer function and when L_a is ignored. To observe the effects of inductance, create a first-order transfer function by completely ignoring the inductance (set $L_a = 0$). Use MATLAB to produce step responses and Bode plots for both cases (overlay both cases on the same plot so they can be compared more easily). Print these plots for your prelab. What changed when the inductance was ignored? Which frequency ranges are the same? Different? Does inductance cause significant effects?

✧ LABORATORY EXERCISE ✧

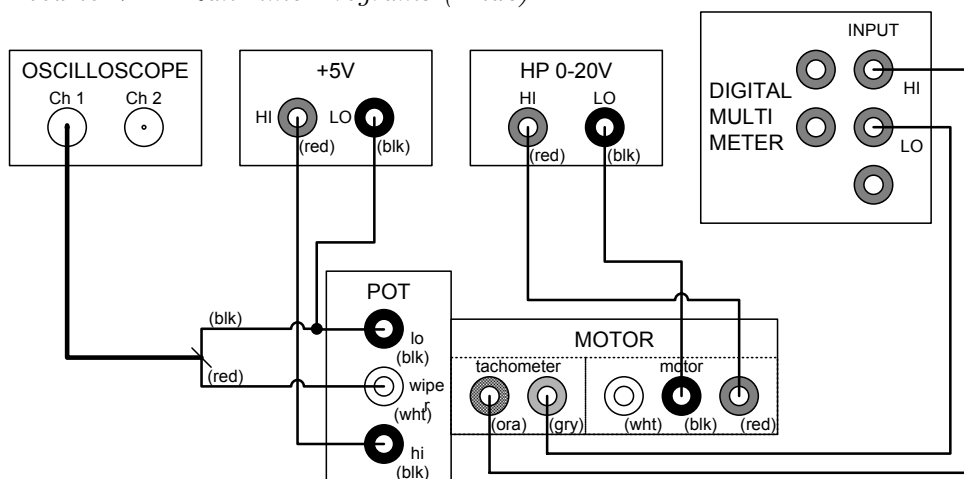
To identify the motor you will collect a large amount of data. In addition to the data obtained using VEE, data tables are provided in the lab text to help you collect all necessary measurements.

I. CALIBRATION OF THE TACHOMETER

The tachometer is a generator with an output voltage proportional to the angular velocity of the motor shaft. Here we use the potentiometer to measure the angular shaft position, θ , and obtain an accurate estimate of the velocity. This allows us to calibrate the tachometer *gain*, the ratio of voltage generated by the tachometer per unit angular velocity (rad/sec). We will not need to save any waveforms to disk with VEE.

1. Turn on the PC, the scope, DMM, and the HP programmable power supply.
2. Connect the potentiometer to the motor and firmly tighten the thumb screws at the base of the potentiometer mount.
3. Start VEE and open the project file at `N:\labs\ECE486\4861ab4.vxe`. You should see control panels for the Agilent scope and the HP 6632A power supply. By default, VEE will save your data in `U:\1ab4.m`. If you prefer a different file location, change the **To File** field at this time. Also, if you have to restart VEE in the middle of taking data, rename the saved file to avoid overwriting it when VEE collects new data.

*Note: Agilent VEE displays files of type *.vee by default. You must change this box to VEE RunTime Programs (*.vxe)*



4. Wire up the circuit as shown. Make sure the wires do not touch any spinning parts. Also, set the DMM to measure DC voltage.
5. Verify in VEE that the scope parameters are:

$$\begin{array}{l}
 \text{Channel1} \left\{ \begin{array}{l} \text{V/div} \rightarrow 1\text{V} \\ \text{Offset} \rightarrow 3\text{V} \\ \text{Coupling} \rightarrow \text{DC} \\ \text{Band-Width Limit} \rightarrow \text{ON} \end{array} \right. \quad \text{Channel2} \left\{ \begin{array}{l} \text{V/div} \rightarrow 1\text{V} \\ \text{Offset} \rightarrow 3\text{V} \\ \text{Coupling} \rightarrow \text{DC} \\ \text{Band-Width Limit} \rightarrow \text{ON} \end{array} \right. \\
 \\
 \text{Trigger} \left\{ \begin{array}{l} \text{Level} \rightarrow 1\text{V} \\ \text{Source} \rightarrow \text{CHANNEL1} \\ \text{Slope} \rightarrow \text{POSITIVE} \\ \text{Noise Reject} \rightarrow \text{ON} \\ \text{High Freq. Reject} \rightarrow \text{ON} \end{array} \right. \quad \text{Time} \left\{ \begin{array}{l} \text{Base} \rightarrow 20\text{m (20 ms) S/div} \\ \text{Reference} \rightarrow \text{CENTER} \end{array} \right.
 \end{array}$$

Then click the **Send New Scope Parameters** button in VEE. Finally press the **Measure Period of Channel1** button. This will setup the scope to automatically measure the period of the saw tooth waveform that will be produced by the potentiometer when the motor is spinning. You will see cursors on the scope screen measuring the period. The measured period is displayed in a small box at the bottom left of the scope screen.

6. Turn on the 5 V power supply connected to the potentiometer using the switch on the patch panel.
7. To initialize the power supply, in VEE click the **Reset** button. Next disable the programmable power supply's output and set **Set Voltage** to 5 V and **Set Current** to 2 A.
8. Turn the power supply's output on by using the **Output Enable** button in VEE. The motor should begin to spin and you should see a sawtooth waveform on the scope. This is the voltage output by the potentiometer as the motor spins.
9. Record the voltage output by the tachometer on the DMM in the table below. This is the voltage generated by the tachometer for the estimated angular velocity of the motor.
10. Also record the period from the scope window. Now disable the output of the power supply to save wear on the potentiometer.
11. Repeat this experiment twice more to complete the table. The ratio of the tachometer voltage and the angular velocity of the motor measured in radians/second is the tachometer gain:

$$K_{\text{tach}} = \frac{V_{\text{tach}}}{\omega}$$

	Measured		Calculated	
V_i	Δ_t	V_{tach}	ω	K_{tach}
5V				
10V				
15V				

Notes: Δ_t is the duration of one rotor revolution.

12. Use the average of the three observations as your estimate of the tachometer gain.

Average K_{tach} : _____

II. ARMATURE RESISTANCE AND BACK-EMF

Now you will collect the data needed to find R_a and K_V .

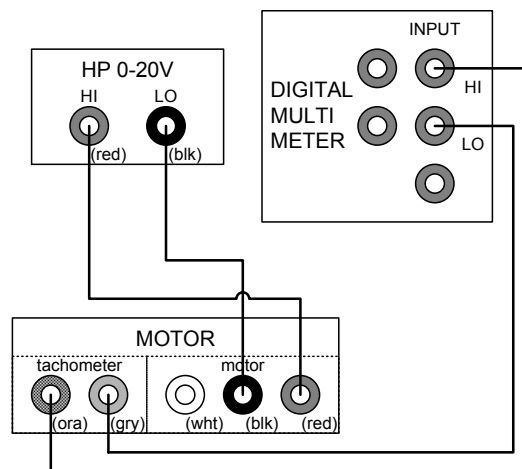


Figure 4.6. : Wiring Diagram for measuring V_i , i_a and V_{tach}

1. Remove the potentiometer from the motor assembly. Adjust your wiring to match figure 4.6.
2. Within VEE adjust **Set Voltage** to 5 V, and **Set Current** to 2 A.
3. Using the **Output Enable** switch on the power supply window turn on the power to the motor. For each voltage V_i applied across the motor, record the steady-state current i_a drawn by the motor shown on the power supply and V_{tach} , the voltage generated by the tachometer as shown on the DMM.
4. Repeat this experiment 7 times to again cover the voltage range 5V–12V. Now switch leads on the *HP* 0–20V panel to get negative voltages, and repeat the 8 measurements. **Your i_a values must now be recorded as negative because the motor voltage has been reversed.** This experiment is noisy, so you will probably be able to get only 2 digits of accuracy for i_a and 3 digits for V_{tach} . There is a button in VEE that will take several measurements and report the average. Use this to get more accuracy (To make it less likely to be dysfunctional, click **Reset** before you **Set Voltage** each time; if the average button does not work, you will hear a beep, then you can directly read current and voltage values rather than use the “average” button).

V_i	Measured		Calculated
	steady-state i_a	V_{tach}	ω
5V			
6V			
7V			
8V			
9V			
10V			
11V			
12V			
-5V			
-6V			
-7V			
-8V			
-9V			
-10V			
-11V			
-12V			

Note: calculate ω using the gain you found in the last section

5. In prelab, you solved for V_i in terms of i_a , ω , R_a and K_V . Enter the equation below.

$$V_i = \underline{\hspace{2cm}}$$

6. In MATLAB use the matrix division operator “\” (this “left divide” operator is actually not a “normal” division; it is internally using least square approximation in MATLAB if the problem is over-determined, like in this case, we have 16 equations but only two unknowns) to solve for R_a and K_V .

```
%[ ia, w]*[Ra; Kv] = Vi - where ia, w, Vi are column vectors
% to solve Ax = b, use the matrix division operator x = A\b.
%   A      x      = b
%[ ia, w]*[Ra; Kv] = Vi
x_val = ([ ia, w]\Vi)';
```

R_a : _____

K_V : _____

III. CONSTANT AND VISCOUS FRICTION COEFFICIENTS

Now you will solve for b and c . For various reasons, the frictional forces are often asymmetric; there is more resistance when the motor spins one way than the other. For this reason, we are using the friction model of Figure 4.7; the forward and reverse friction coefficients will be found independently.

1. Use two best-fit lines (one in each direction) to determine the friction coefficients.

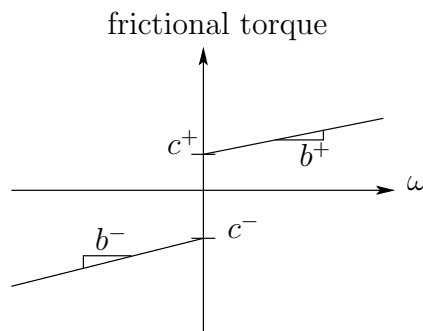


Figure 4.7. : Rotary friction model

b_{pos} : _____ c_{pos} : _____
 b_{neg} : _____ c_{neg} : _____

IV. ARMATURE INDUCTANCE

This section uses parts from the long drawer under the bench. The *Resistor Box* is a small blue box with banana jacks on the ends. Be careful to keep the resistor box on the green rubber mat so nothing shorts on the metal tabletop.

The armature inductance L_a is measured by observing a step-response to an input voltage while the rotor is locked. This is done by observing the voltage drop across a resistor R_s connected in series with the locked motor. The step-by-step instructions are given below:

1. In this section, we are working with very small resistances that could easily be affected by bad connectors. Use the DMM to check the resistance of all the wires you use; they should have no more than 0.05Ω of resistance. If they have more, then have the lab TA tighten the connectors with a screw driver.

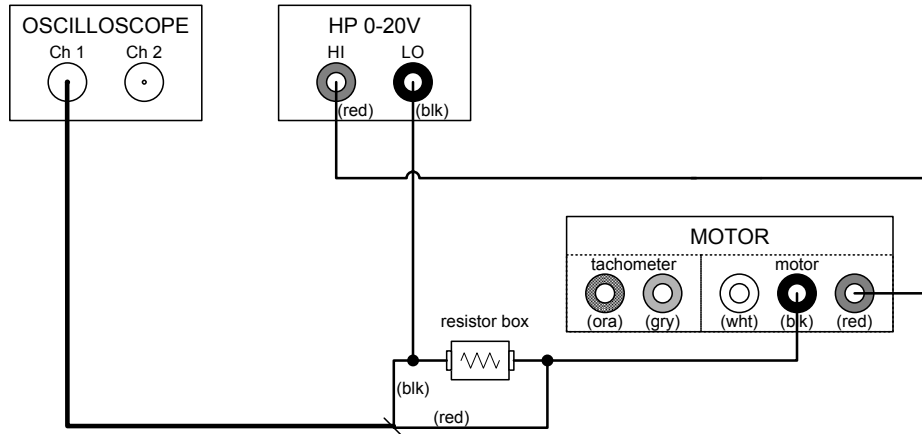
First we determine R_s :

2. Measure the resistance R_s of the resistor box using the DMM, it should be close to 2.5Ω .

Measured R_s : _____

To calculate L_a :

3. Disable the power supply's output by changing the **Output Enable** button to "off".
4. Attach the rotor-locking attachment to the motor, and tighten the thumb screws. There is less flex in the system if you **place the locking mechanism on the left** of the motor.
5. Wire up the circuit shown below in order to measure L_a . This corresponds to the electrical circuit of Figure 4.4.



6. Set the oscilloscope to the following:

Channel1	{	V/div → 0.5V Offset → 1.5V Coupling → DC Band-Width Limit → ON	Channel2	{	V/div → 0.5V Offset → 3V Coupling → DC Band-Width Limit → ON
Trigger	{	Level → 0.5V Source → CHANNEL1 Slope → POSITIVE Noise Reject → ON High Freq. Reject → ON	Time	{	Base → 0.2m (0.2 ms) S/div Reference → CENTER

and then press the **Send New Scope Parameters** button to initialize the scope.

7. In Agilent VEE set the programmable power supply's voltage to 5 V and its current limit to 2 A.
8. Turn on the power supply by selecting **Output Enable** *on* in VEE. Disconnect and reconnect the (red) power supply HI voltage from the bench panel. (Make sure the scope is in the Run mode, **Run/Stop** button green.) Do this a few times until you get a smoothly rising step response. It may help to **slide the plug in and out** instead of pulling it completely out (hence we are trying to create a “manual” step function). To get the cleanest signal, it may help if you pre-stress the rubber connectors by turning the flywheel in the direction the motor applies torque; this is not always required. Do not use the Output Enable button in VEE to switch power to the motor because the time constant $\tau_{\text{powersource}} \gg \tau_{\text{motor}}$.
9. When you have a clean waveform, press the VEE **Collect Data From Scope** button. Each time that you press **Collect Data From Scope**, the data from the scope will be stored in the vector “Elect j ”, where the integer j is incremented by a counter each time that you store new data.
10. While you are not collecting data, turn off the power using the **Output Enable** switch to keep the motor from overheating.

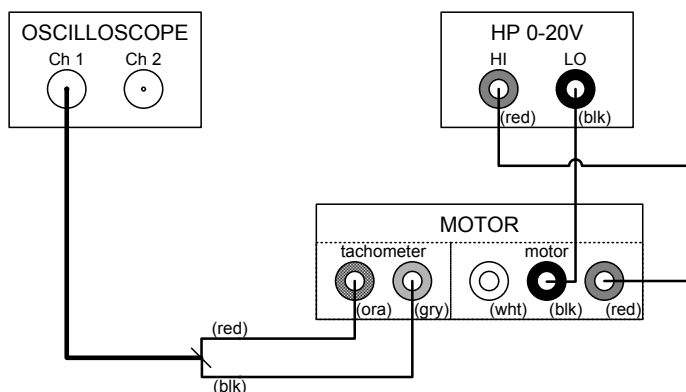
11. Repeat this experiment five more times to cover the voltage range 5V–10V in 1 Volt increments. If the 9V or 10V signal goes off the screen, change the V/div setting for the scope in VEE; this change will not cause problems when MATLAB loads the data, but data which is not shown by VEE will be truncated. Use the technique from the prelab to get estimates of L_a , then report the average value.

Measured L_a Values

1	2	3	4	5	6	Average L_a

V. ROTOR MOMENT OF INERTIA

The rotor moment of inertia can be inferred from the mechanical time constant τ_m , and the friction constants b and c . The mechanical time constant can be obtained by observing the decaying angular velocity when the current supply to the motor is cut off.



1. Remove the rotor-locking attachment and the resistor box. Change the wiring to match this diagram.
2. In VEE first click the **Clear** button to reset the counter to zero. Also in the **Data Name Formula** box change the name from “Elect” to “Iner”. (Iner is short for Inertial time constant data.)
3. Set the Oscilloscope to the following:

$$\begin{array}{l}
 \text{Channel1} \left\{ \begin{array}{l} \text{V/div} \rightarrow 1\text{V} \\ \text{Offset} \rightarrow 3.5\text{V} \\ \text{Coupling} \rightarrow \text{DC} \\ \text{Band-Width Limit} \rightarrow \text{ON} \end{array} \right. \\
 \\
 \text{Channel2} \left\{ \begin{array}{l} \text{V/div} \rightarrow 1\text{V} \\ \text{Offset} \rightarrow 3.5\text{V} \\ \text{Coupling} \rightarrow \text{DC} \\ \text{Band-Width Limit} \rightarrow \text{ON} \end{array} \right. \\
 \\
 \text{Trigger} \left\{ \begin{array}{l} \text{Level} \rightarrow 4\text{V} \\ \text{Source} \rightarrow \text{CHANNEL1} \\ \text{Slope} \rightarrow \text{NEGATIVE} \\ \text{Noise Reject} \rightarrow \text{ON} \\ \text{High Freq. Reject} \rightarrow \text{ON} \end{array} \right. \\
 \\
 \text{Time} \left\{ \begin{array}{l} \text{Base} \rightarrow 200\text{m (200 ms) S/div} \\ \text{Reference} \rightarrow \text{CENTER} \end{array} \right.
 \end{array}$$

Press the **Send New Scope Parameters** button to send these settings to the scope.

4. With the power supply's output disabled, adjust **Set Voltage** to 10 V in VEE. Keep **Set Current** at 2 A.
5. Using the **Output Enable** switch in the power supply's window turn on the power to the motor. Wait till the tachometer reading is roughly constant. Pull the HI (red) power supply cable from the patch panel.
6. When you have a waveform on the Oscilloscope, make sure the scope is in the **Stop** mode and then press the **Collect Data From Scope** pushbutton. The scope waveform is now stored in a coordinate format.
7. Using the method from the prelab, calculate the moment of inertia, J , from this curve.
 - ▷ **Remember:** The measured curve is $V_{\text{tach}}(t)$, not $\omega(t)$
8. Repeat this experiment five more times, incrementing the voltage 1 V each time to 15 V. You may need to adjust the trigger level, scale, and/or time base to keep the whole trace on the scope. You may need to switch to 2V/div for the 14V and 15V trials. TA ✓:

Measured J Values

1	2	3	4	5	6	Average J

🔮 REPORT 🔮

I: CALIBRATION OF TACHOMETER

1. Briefly explain the procedure for determining K_{tach} . Why is it important to know K_{tach} ?
2. Report the tachometer gain K_{tach} . Use the average value.

II: ARMATURE RESISTANCE AND BACK-EMF

1. Explain the procedure for obtaining the Armature Resistance, R_a , and the torque gain ($K_v = K_\tau$). Why can we ignore L_a ?
2. Report R_a and K_V .

III: CONSTANT AND VISCOUS FRICTION COEFFICIENT

1. Explain the procedure for obtaining the Coulomb and viscous friction coefficients. Include equations. Plot out the the friction torque ($= K_\tau I_a$) against ω in either direction. From these plots determine the values of the Coulomb and viscous frictions coefficients for both directions (c^+ , c^- , b^+ , and b^-).

IV: ARMATURE INDUCTANCE

1. Report R_s . Using the slope-method that you applied in the prelab, compute τ_e and L_a . (Average the six values—they should all be similar.) Include the responses (plots) obtained in the laboratory exercises, *carefully labeled*. Explain the process for measuring both parameters. Explain how holding the motor still with the rotor-locking attachment allows us to more easily measure L_a .

V: ROTOR MOMENT OF INERTIA

1. Using the method derived in part (d) of the prelab, compute J using your estimates of b and c . (Average the six values—they should all be similar.) Explain how to obtain J . Include equations. Also explain why we need to measure transient behavior rather than steady-state behavior to obtain J .

VI: CONSERVATION OF ENERGY

1. Ignoring losses such as friction, apply the conservation of energy law between the electrical and mechanical systems in figure 4.3 to show that K_V and K_τ are identical. *Hint*: electrical power is voltage·current and mechanical power is force·velocity (or torque·angular velocity). Use unit conversions to show that, in the SI system, the units for K_V and K_τ are equivalent.

PART VII: SYSTEM TRANSFER FUNCTION

1. Using the 2nd order transfer function $\Omega(s)/V_i(s)$ found in prelab part (e), plug in the numerical values found in this lab exercise to calculate the numerical transfer function, and find its poles and zeros. (Remember that Coulomb friction is being ignored, but viscous friction is still present.)
2. Repeat the previous question for the 1st order transfer function from prelab part e, and compare.

PART VIII: STEADY-STATE RESPONSE OF NONLINEAR SYSTEM

1. Including the *nonlinear* (Coulomb) friction term, what will be the steady-state angular-velocity for a 4V DC input voltage? Solve the steady-state equations algebraically.

Write down your system parameters to use later. You will probably need them before this lab report is returned.

Parameter	Value
K_{tach}	
R_a	
L_a	
K_V	
J	
c^+	
b^+	
c^-	
b^-	

Lab Session 5

PD CONTROL: ANALOG COMPUTER & SIMULINK DESKTOP REALTIME

This experiment continues the study of the DC motor begun in Lab 4. The model that you obtained in the last set of experiments was derived from transient response data. You will now design a proportional-derivative controller to do position control, and implement it on both the analog controller and digitally using Simulink Desktop Realtime. In each case, you will record and analyze the step response of the closed-loop system for a set of feedback gains. Furthermore, you will explore the effects of friction compensation.

🔗 PREPARATION 🔗

READINGS:

To prepare for the design in this lab, you should review block diagram manipulation in Franklin, *et al.*, 3rd, 4th, 6th or 7th Ed., Sec. 3.2.1. Also be comfortable with root locus design from lecture and also Chapter 5 in Franklin, *et al.*, 4th, 6th or 7th Ed., especially selecting gain from the root locus, Sec. 5.4, or 6th/7th Ed., Sec. 5.2.3 (Sec. 5.5 in 3rd Ed.).

PRELAB:

- (a) Using the results from your last experiment, obtain the time constant τ and the gain K in the motor transfer function $\Omega(s)/V_m(s) = K/(\tau s + 1)$.

We next consider the design of a closed loop position control system using PD control. The structure is shown in Figure 5.1. For the sake of concreteness, assume from now on that the block MOTOR has the transfer function

$$\Omega(s)/V_m(s) = \frac{18}{1 + 0.2s}.$$

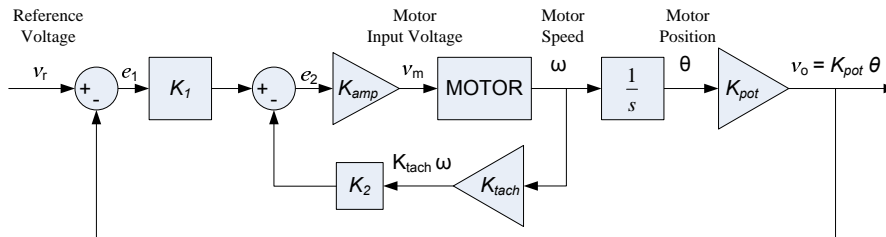


Figure 5.1. : A Feedback Control System for the DC Motor

In the lab, we will connect the system illustrated in Figure 5.2 to the analog computer set up shown in Figure 5.3, to create the closed loop system of Figure 5.1. We are then interested in the voltage from the potentiometer, the output of the system, and how it behaves as the reference input voltage v_r is varied. This reference input will be supplied by the HP waveform generator.

Note: In practice, the reference input would typically be an angle, not a voltage. One then must construct a device (such as a potentiometer) to translate the desired angular position to an input voltage v_r .

- (b) Suppose $K_{\text{amp}} = 2.4$, $K_{\text{pot}} = 10/2\pi \approx 1.6$ volt \cdot s/rad, and $K_{\text{tach}} = 0.03$ volt \cdot s/rad. Find the closed loop transfer function $V_o(s)/V_r(s)$ in terms of the potentiometer settings $P_1 (= K_1/10)$ and $P_2 (= K_2/10)$. Given this transfer function, compute the closed loop pole locations for the 3 sets of gains:

P_1	0.15	0.25	0.1
P_2	0	0.35	0.5

- (c) For the same control system considered above, find values of P_1 and P_2 such that overshoot is approximately 15% ($M_p \approx 0.15$) and the rise time is approximately 30 ms ($t_r \approx 30\text{ms}$). Use the formulas on page 13.
- (d) Show the two root locus plots for your P_1 and P_2 values. Using the root locus plots, show that your values of P_1 and P_2 meet the M_p and t_r requirements from the previous question. Use the command `rlocus()`. You will have to rewrite the denominator.
- (e) Draw a patch panel wiring diagram for the analog computer setup shown in Figure 5.3 (P_1 and P_2 are variable). (This diagram implements the PD control law.)

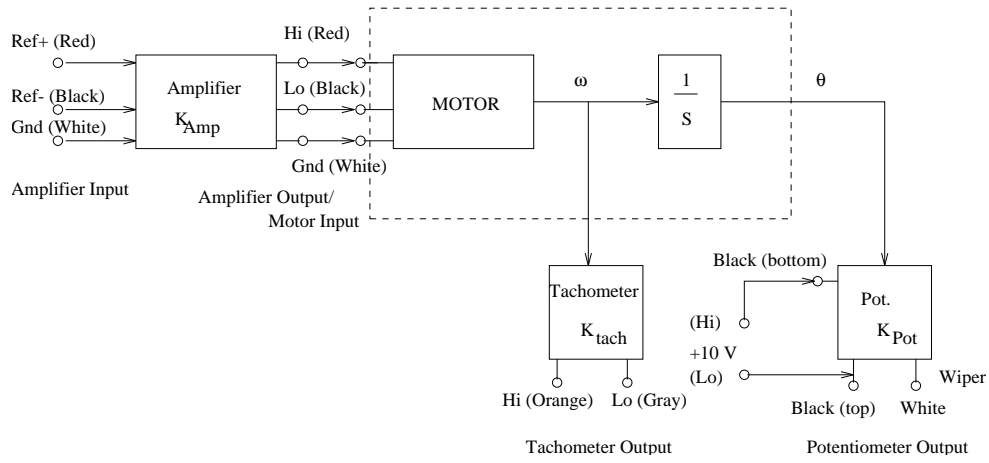


Figure 5.2. : Motor Block Diagram.

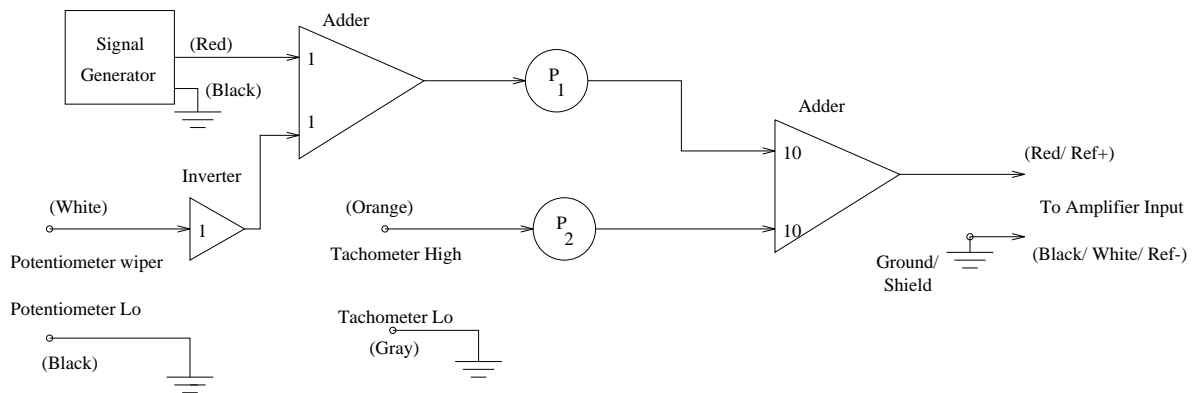


Figure 5.3. : Analog Computer Wiring Diagram.

🌀 LABORATORY EXERCISE 🌀

I. PD CONTROL USING ANALOG COMPUTER

To begin, wire up the analog computer according to your patch panel diagram from prelab part (d). Then follow these steps.

1. Make sure power to the patch panel is off.
2. Connect the potentiometer to the motor assembly.
3. Use the wiring diagram (Figure 5.4) to connect the analog computer to the other components.
4. In order to collect data for analysis in MATLAB, start VEE and open N:\labs\ECE486\486lab5.vxe. Remember to select *Files of type:* to be *.vxe. If desired change the data filename with the **To File** button and the “y” in the **Data Name Formula** box.
5. Turn on the oscilloscope and function generator; wait for them to start up.
6. In VEE, verify that the scope is set to:

$$\begin{array}{l}
 \text{Channel1} \left\{ \begin{array}{l} \text{V/div} \rightarrow 1\text{V} \\ \text{Offset} \rightarrow 5\text{V} \\ \text{Coupling} \rightarrow \text{DC} \\ \text{Band-Width Limit} \rightarrow \text{ON} \end{array} \right. \qquad \text{Channel2} \left\{ \begin{array}{l} \text{V/div} \rightarrow 1\text{V} \\ \text{Offset} \rightarrow 5\text{V} \\ \text{Coupling} \rightarrow \text{DC} \\ \text{Band-Width Limit} \rightarrow \text{ON} \end{array} \right. \\
 \\
 \text{Trigger} \left\{ \begin{array}{l} \text{Level} \rightarrow 4\text{V} \\ \text{Source} \rightarrow \text{CHANNEL2} \\ \text{Slope} \rightarrow \text{POSITIVE} \\ \text{Noise Reject} \rightarrow \text{ON} \\ \text{High Freq. Reject} \rightarrow \text{ON} \end{array} \right. \qquad \text{Time} \left\{ \begin{array}{l} \text{Base} \rightarrow 100\text{m (100 ms) S/div} \\ \text{Reference} \rightarrow \text{LEFT} \end{array} \right.
 \end{array}$$

Press the **Send New Scope Parameters** button to send these settings to the scope.

7. Also verify that the function generator is set to:
 - Type of Waveform \rightarrow SQUare.
 - Volts Peak to Peak \rightarrow 2V.
 - Offset \rightarrow 4V.
 - Frequency in Hertz \rightarrow 0.5Hz.

Press the **Send Function Generator Settings** button to send these settings to the function generator.

Since the scope is triggering on a 0.5 Hz signal, it will take a while to update; so be patient. If the scope never triggers, in VEE push the **Put Scope in Roll Mode for Debugging** button to switch to **ROLL** mode to see what is happening. After finding your triggering problem, in VEE press the **Send New Scope Parameters** button to set the scope back up.

Have your TA check your wiring and instrument settings before you proceed.

TA \checkmark :

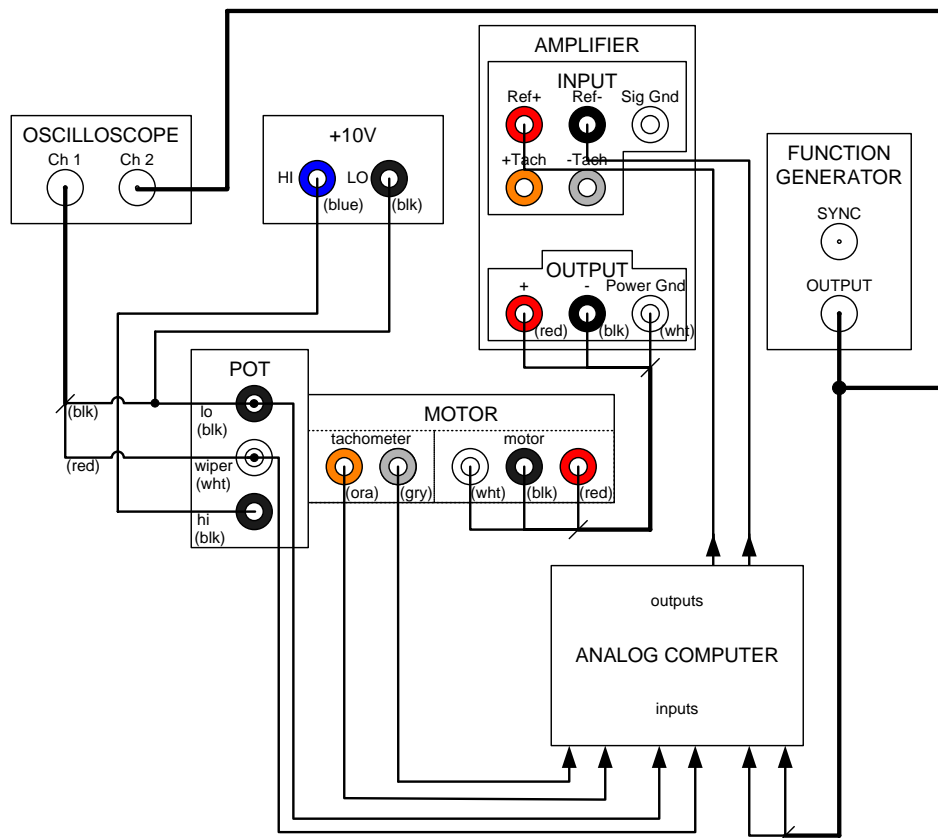


Figure 5.4. : Wiring diagram for Part I.

8. Connect the push button (in the long drawer under the bench) to the **Amp Inhibit** on the amplifier. Turn on both power switches on the patch panel.
9. Turn on the analog computer. In **Pot Set** mode, set the potentiometer levels P_1 and P_2 . The values of P_1 and P_2 for which the experiment should be repeated are:

P_1	0.15	0.25	0.1	prelab (c) value: _____
P_2	0	0.35	0.5	prelab (c) value: _____

10. Switch to **OPR** mode on the analog computer. Press the pushbutton and wait. Soon you should see a step response on the scope (it may take two changes of the input, based on the triggering settings). When you have a good trace on the scope and the scope is in the **Stop** mode, press **Collect Data from Scope** in VEE to display the waveform on the computer and save it to the file. Repeat these steps for the other values of P_1 and P_2 .
11. You can adjust P_1 and P_2 while the controller is running. Find the P_1 and P_2 that meet the specifications.
 Best Analog Computer P_1 : _____
 Best Analog Computer P_2 : _____
 Demonstrate to your TA how to reject disturbances and avoid overshoot using these two gains. TA \checkmark :
12. Turn off the patch panel, the analog computer, and the function generator. Close VEE.

II. PD CONTROL USING SIMULINK DESKTOP REALTIME (SLDRT)

Simulink Desktop Realtime (SLDRT) is a plug-in for MATLAB that allows you to design controllers graphically using SIMULINK; it auto-generates C code to implement your block diagram and communicate with an I/O board. We will use it to send analog signals out (the reference voltage and control signal) and collect data from the motor (V_{POT} and V_{TACH}). The I/O board is connected to the Analog I/O and Digital I/O sections of the patch panel.

To guard against possible network glitches, your SLDRT file **must be saved** on the **C:** drive of your station's computer before compiling. To save your model, you can move this folder back to the network drive (**U:**) at the end of lab. Each SLDRT file should be compiled in its own directory to prevent compilation problems. Note that files compiled on **C:** will not run on **U:** without recompiling, but they will run if returned to the **C:** drive first.

Tip: When running SLDRT, some settings need to be different from those used in SIMULINK. Specifically, use a **fixed step size** so that evaluations coincide with the sampled data and use **ODE1** to properly process Laplace transform ($\mathcal{L}(s)$) blocks.

1. Create a personal directory on the **C:** drive. Use something like **C:\NETID**, not **C:\Lab5**.

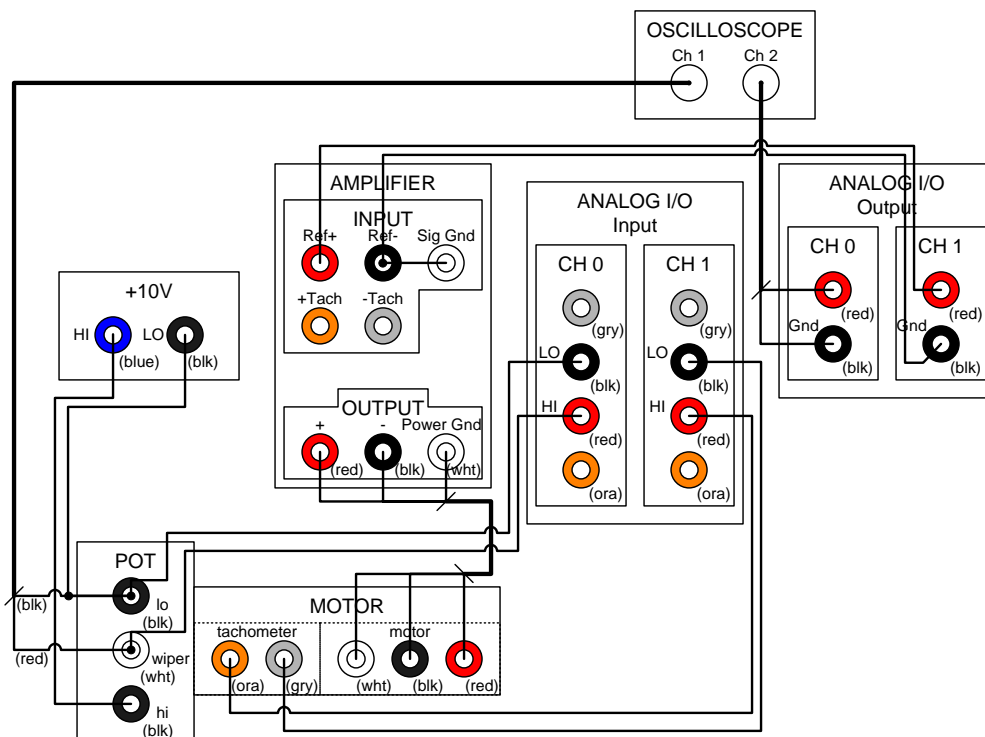



Figure 5.5. : Wiring diagram for Part II.

2. Change the *Current Directory* in MATLAB to your new directory on the **C:** drive.
3. Wire the circuit as shown in Figure 5.5.
4. At the MATLAB command prompt, type `rtwin486`. This opens a SIMULINK model that has the correct settings for SLDRT. Use “Save As” to save a copy in your directory on the **C:** drive (please *do not* change the original copy because it serves as a template).
5. You need to set the SLDRT sampling period. To do this, click on the *Simulation* menu of your menu, and select *Model Configuration Parameters*. In the *Solver* pane, change *Fixed-step size:* to 0.002; this gives a 500 Hz sample rate (if you hear a buzz from your motor assembly, then you need to increase *Fixed-step size* to a greater value, like 0.005, sample rates decrease to 200 Hz).
6. Open the SIMULINK Library Browser by typing `simulink` at the MATLAB prompt or by clicking on the library browser icon () in MATLAB toolbar.
7. To understand what portion of the control loop is accomplished in SIMULINK, refer to Figure 5.1. The inputs to the PC are $V_{pot} = v_o$ and $V_{tach} = K_{tach} \cdot \omega$, and the output from the PC is e_2 . Therefore everything between the inputs and the output will be done in software. You will also output v_r for display on the oscilloscope.

8. Create a signal v_r just like that in the previous section: a square wave from 3V to 5V with a frequency of 0.5 Hz. (Tip: Sum a **Constant** with a **Pulse Generator**)
9. Complete your model, using blocks from the SIMULINK Library Browser. As in Figure 5.5, V_{pot} is connected to ADC channel 0, V_{tach} is connected to ADC 1, v_r is connected to DAC 0, and e_2 is connected to DAC 1. Use **Slider** blocks instead of a gain blocks for K_1 and K_2 .
10. Use a **Mux** to observe v_r and V_{pot} simultaneously. Attach a **Scope** to the **Mux** output. Set up the scope to record data by double-clicking on the scope to open it. Then click on the *parameters* button (⚙). Select the *Logging* tab, click *Log data to workspace* and change the *Format* to **Array**.

Have your TA check your wiring and SIMULINK settings before you proceed.

TA ✓:

11. Connect the push button to the **Amp Inhibit** on the amplifier. Choose "SL-DRT" tab and click "Run" button, the Simulink model will run in realtime.
12. For the report, you will use MATLAB to view, plot, or manipulate the data to compute the overshoot, rise time, settling time, and steady-state error of the step response for 4 sets of gains, so collect data for each of these cases:

K_1	1.5	2.5	1	prelab (c) value: _____
K_2	0	3.5	5	prelab (c) value: _____

Note that these are the same values you experimented in the previous section. These are simply given in terms of K instead of P . Make sure to convert your prelab design values to K as well (they should be greater than 1, less than 10).

13. Find the K_1 and K_2 that meet the specifications. *Tip*: set the simulation stop time to a large value (999 or `inf`) and adjust the sliders while the system runs.
 Best SIMULINK K_1 : _____
 Best SIMULINK K_2 : _____
14. Close your SIMULINK model.

III. PD CONTROL WITH FRICTION COMPENSATION

Friction compensation is a common technique for alleviating the effects of friction, without changing the control model. It works by conceptually separating the motor into an ideal, frictionless part and treating friction as an external feedback. By adding friction compensation as in Figure 5.6, the effects of friction can be nearly canceled.

1. Copy the model `N:\labs\ECE486\lab5\lab5_friccomp.mdl` into your directory. Open it in MATLAB.
2. Notice the switches: these turn on/off the controller and friction compensation. Double-click a switch to "switch" it. *Note*: **They can be switched while the code is running!**

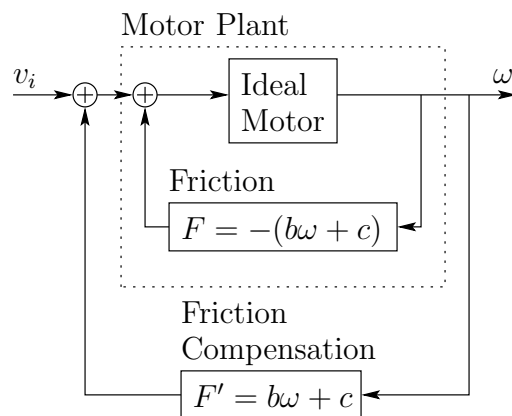


Figure 5.6. : Friction-compensated plant

3. Notice the gains K_1 and K_2 . They can also be changed while the code is running.
4. Double-click on the *Asymmetric Linear Friction* block. There is a list of friction parameters. The values of these parameters can also be changed while the code is running, and they take effect as soon as you hit **Apply**.

Note: the *Asymmetric Linear Friction* block may be obtained by typing **fricblocks** on the MATLAB command line.

5. Build and run this program in open-loop with friction compensation applied (i.e. no controller running) and observe the behavior when you apply an initial velocity to the motor (spin it by hand).

▷ Adjust the friction gains (double-click on the *Asymmetric Linear Friction* block) so that the motor seems to run “frictionless” in both directions. First adjust the Coulomb gains (c), with the viscous gains (b) set to 0. Start with the Coulomb values you found in lab 4. Adjust them so that the motor slowly oscillates when no force is applied.

▷ Next, increase the viscous gains so that the motor does not slow down too quickly when started at high speeds. The goal is not perfect compensation, just a substantial reduction in the observed friction. If the gains are too high, the motor will speed up due to the positive feedback.

The coefficients you find will be somewhat larger than those in lab 4. This is mostly due to the addition of the potentiometer to the system. Record these gains in Table 5.1, and demonstrate the running system to your TA. (*Comment:* As the foregoing in the previous paragraph, the (b) and (c) gains obtained are not perfect for friction compensation. But it can reduce observed friction. In terms of resulting transfer function $\frac{\Omega(s)}{V_m(s)} = \frac{K}{1+\tau_m s}$ in Figure 5.1, the damping caused by (b) term is reduced so that new responses would be expected to be faster due to less friction. Since you have a new MOTOR block transfer function here (with compensation turned on, hence less damping), what you really have to do in Step 6 below is to go back to prelab 5 and re-calculate the open-loop transfer function and re-design controllers K_1 and K_2 based on root locus method. However since lab time is limited and we want to make your life

easier, you can simply follow what is described in Step 6 and just make sure you collect responses that are “stable” so that you can calculate specifications of time domain responses, such as t_r , t_s , and M_p .)

TA ✓:

6. **Final Data Run.** Run this program in closed-loop (i.e. with the controller running) with friction compensation applied. Collect step response data for the same four sets of gains that you tested in the previous section. Set the simulation stop time to 5.0s, and change the *Variable name* (on the scope Parameters menu) for each set of gains. You may need to decrease viscous friction coefficients (b) (1st choice) or Coulomb friction coefficients (c) (2nd choice) by 20% if the system goes unstable; try the full values for each controller and only make this change as necessary.

Table 5.1. : Friction Values

	Lab 4	Lab 5 _{open-loop}	Lab 5 _{closed-loop}
b_{pos}			
c_{pos}			
b_{neg}			
c_{neg}			

◁ REPORT ▷

PART I: COMPARISON OF RESPONSES

1. Using K and τ from the motor model given on page 42, compute theoretical values of ζ , ω_n , M_p , t_r , and t_s for each value of P_1 and P_2 , for the system in Figure 5.1. Use the formulas on page 13.
2. Determine the three sets of experimental values of M_p , t_r , and t_s from the step response data in Sections I, II, and III.
3. Compare your results from section I (analog computer) with those from section II (SLDRT). Note any differences and characteristic similarities.
4. Compare your results from section II with those from section III. This represents the difference between simple PD control and PD control with one type of friction compensation applied. Discuss; especially the contribution of Coulomb friction.

PART II: PERFORMANCE COMPARED TO SPECIFICATIONS

1. How well did your design perform in the lab? Did your design meet the specifications laid out in the prelab? If not, suggest improvements to do so.
2. Explain how unmodeled plant dynamics might cause problems.

PART III: STEADY-STATE ERROR

1. Was the steady state error as expected in all cases?
2. What adjustments of the gains help to decrease the steady-state error?

PART IV: FRICTION COMPENSATION

1. What friction values did you get experimentally in section III part 5?
2. By how much did they need to be reduced to ensure stability? Compare them with your values for friction identified in Lab 4 section III.

Save your results to compare with the next experiment.

Lab Session 6

LEAD CONTROLLER DESIGN

In Lab Session 5, motor position and angular velocity measurements were used to construct a compensator. Frequently in practice, derivative measurements are not directly available, and differentiation of position measurements may not be advisable because of measurement noise. Lead compensators are a class of controllers that can be successfully used even when derivative information is not directly available. Lead compensators have the general form

$$G_c(s) = K_c \frac{\tau_z s + 1}{\tau_p s + 1},$$

with $\tau_p < \tau_z$. The pole provides high frequency gain roll-off that results in less amplification of high frequency measurement noise than is observed with a PD compensator. This is one significant advantage of the lead compensator. In this set of experiments, you will build upon the frequency domain viewpoint. In this lab you will directly use a Bode plot rather than a root locus to design the lead compensator G_c to meet the following objectives:

- Reasonable bandwidth
- Good transient response
- Good disturbance rejection to overcome motor friction

✧ PREPARATION ✧

READINGS:

From Franklin *et al.*, 3rd Ed., 4th Ed., 6th or 7th Ed.,

- ▷ Frequency Response and Bode plots Sec. 6.1
- ▷ Stability Margins Sec. 6.4
- ▷ Lead Compensation Sec. 6.7.2

PRELAB:

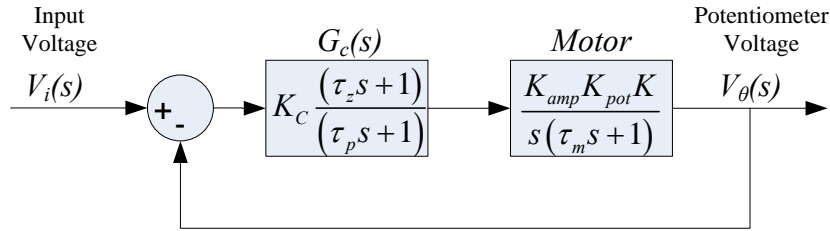


Figure 6.1. : DC Motor Lead Controller System.

- (a) Design an all-integrator block diagram to implement the lead compensator $G_c(s)$. Try to have one gain block per gain parameter (K_c , τ_z , and τ_p). No credit will be given for designs using derivative blocks.
- (b) Consider the motor controller of Figure 6.1 using the proportional control $G_c(s) = K_c$. Develop a Bode plot of the resulting loop transfer function

$$G_c(s)G(s) = \frac{V_\theta(s)}{V_i(s)} = K_c \frac{1}{s} \frac{K_{\text{pot}} K_{\text{amp}} K}{\tau_m s + 1}.$$

Use $K_{\text{pot}} = 10/(2\pi)$, $K = 18$, $\tau_m = 2/10$, $K_{\text{amp}} = 2.4$ and $K_c = 1$.

To obtain a short rise time, the closed loop bandwidth must be correspondingly large. A good rule of thumb for design is to approximate the closed-loop bandwidth by the open-loop crossover frequency since we typically have $\omega_c \leq \omega_{\text{bw}} \leq 2\omega_c$. We will use this heuristic to approximate the closed-loop bandwidth in our control designs. Using your Bode plot obtained above, determine the crossover frequency ω_c of the system when no compensation is used (i.e. $K_c = 1$), and find the phase margin. The MATLAB command **margin** may be used.

Note: ω_{bw} is not necessarily the same as ω_c ! The bandwidth is defined as in Figure 6.5 of Franklin. Its relationship to the damping ratio can be seen in Figure 6.3(a) of Franklin.

Now determine the necessary value of K_c to achieve $\omega_c = 55$ rad/s. Plot the magnitude and phase response of this compensated open loop transfer function, and determine the phase margin (PM) of your design. For a second order system, as the PM approaches 0° , the damping ratio ζ is reduced and the system response becomes more oscillatory. Do you notice a decrease in the phase margin as you increase the gain?

To improve the PM you will now design a lead compensator.

- (c) Obtain a Bode plot for the lead compensator with transfer function

$$G_c(s) = K_c \left(\frac{0.1s + 1}{0.01s + 1} \right).$$

You may take $K_c = 1$. Note that the phase plot has a positive bulge in the frequency range $3 \text{ rad/s} \leq \omega \leq 400 \text{ rad/s}$. When a lead compensator such as

this is cascaded with the motor, their phase responses add to increase the overall phase in this frequency band. In your design, you will need to choose the size and frequency of the center of this phase bulge.

(d) **High DC Gain compensator**

A lead compensator plus motor gives the loop transfer function:

$$G_c(s)G(s) = K_c \left(\frac{\tau_z s + 1}{\tau_p s + 1} \right) \left(\frac{K_{\text{pot}} K_{\text{amp}} K}{s(\tau_m s + 1)} \right)$$

As in Lab 5 we would like at most 15% overshoot ($M_p \leq 0.15$) and 30 ms rise time ($t_r \leq 30\text{ms}$). Use the formulas on page 13 to get the necessary ζ and ω_n . These pole location specifications for a second-order system can be translated to frequency specifications on an open-loop system—see the transfer functions on pages 348 through 353 in Franklin *et al* (pages 334 through 337 in 6th Ed.), especially equation 6.29 (if 4th Ed., see page 405) (page 378 in 3rd Ed.). Find the necessary PM and ω_c as though we had a second order system. This will serve as a starting point for your design.

Using frequency domain techniques (See page 364 in 7th Ed., page 349 in 6th Ed. or 367 in 5th Ed., page 418 in 4th Ed., and page 390 in 3rd Ed.), design G_c to obtain this cross-over frequency and phase margin. To avoid poor response or control saturation, keep the DC gain in the range $2 \leq K_c \leq 4$. To prevent noise amplification keep your poles and zeros closer to the origin than 500 rad/s. As before, we are interested in the magnitude and phase characteristics of the open-loop response to determine the closed-loop response, using the feedback configuration illustrated in Fig. 6.1.

Note that higher phase margin gives lower overshoot, and higher cross-over frequency gives lower rise time, so it is fine if your design exceeds the phase margin and cross-over frequency specifications.

Give the resulting closed loop system, and plot its step response. Check that you met the time-domain specifications and adjust your design if necessary. Document the choices you made for each iteration. Also obtain a root locus plot of your final design with K_c varied from 0 to ∞ , and on the root locus mark the closed loop pole locations for your design.

(e) **Low DC Gain compensator**

Consider a lead controller with transfer function

$$G_c(s) = 0.40 \frac{s/2.3 + 1}{s/92 + 1} \tag{6.1}$$

Verify that this controller meets the phase margin and bandwidth requirements. Plot the step response of the closed loop system, and verify the time domain specifications also. Note however, that the DC gain of this controller is only 0.40, which should be much smaller than the compensator that you designed. Theoretically both designs should produce a good step response, and since we have a type 1 system, we should obtain zero steady state error. However, this argument ignores the effects of disturbances: In practice we will find that a larger DC gain is required to overcome the effect of Coulomb friction.

🌀 LABORATORY EXERCISE 🌀

I. SIMULATION

In the following exercises, the lead compensators studied in the prelab will be implemented using SIMULINK and Simulink Desktop Realtime (SLDRT). But before implementing the controllers, you will first study one of the side effects of real world implementation: saturation.

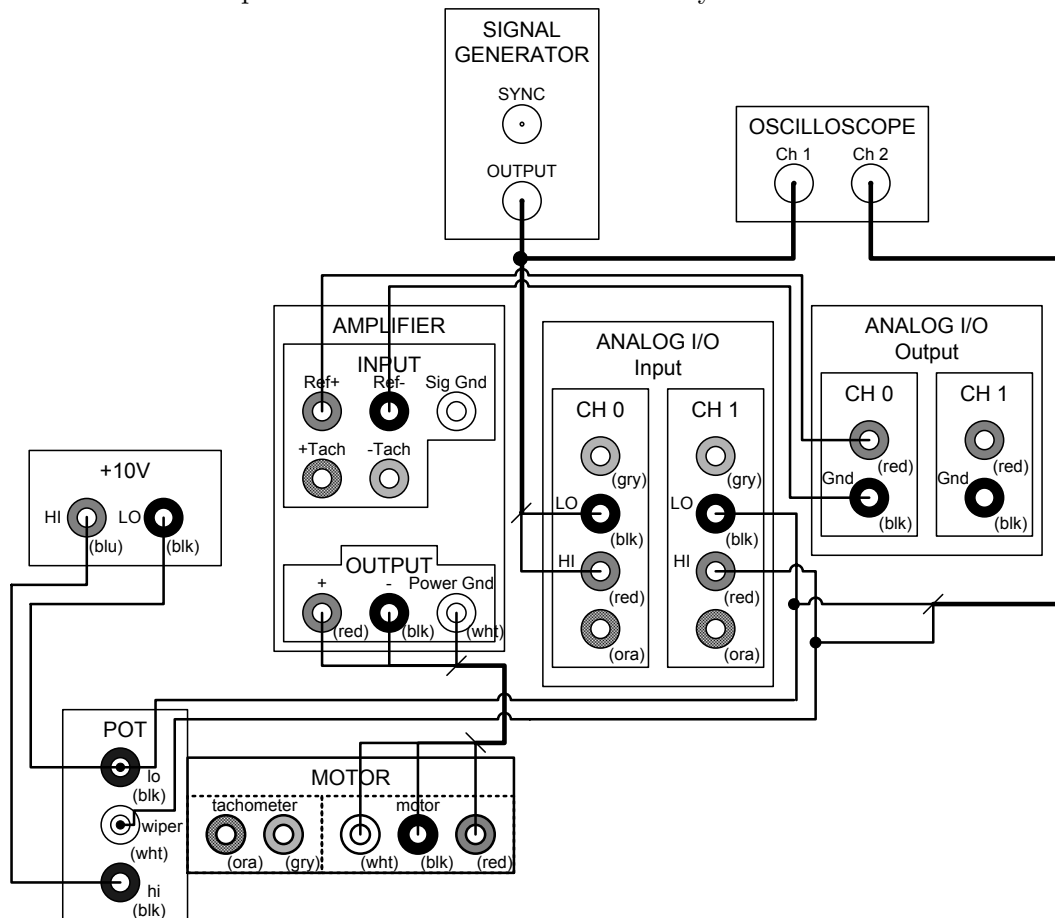
1. Generate a SIMULINK model of Figure 6.1; use the control block diagram and the gains in step (e) of the prelab. Use a **To Workspace** block to store the step response as “V”, and set the “Save Format” to **Array**. In the simulation parameters, use **ODE45** for the best results; to smooth out the curve, set **Max step size** to 0.01 or even 0.001. Also, change the stop time to shortly after the transient response finishes (a couple seconds).
2. Lead compensators often produce relatively large control signals; however, the DACs used in this lab saturate at $\pm 10V$. As a result, the control effort produced by SLDRT may not match the desired signal; and correspondingly the system response may not be as expected. In order to simulate this effect of the DAC, add a **Saturation** block between the lead controller and the motor plant.
3. First simulate the system with saturation disabled; set the **Upper limit** to **inf** and the **Lower limit** to **-inf**. Set the step input to 0.5V, and run the simulation. In the MATLAB workspace, set `V05=V; time05=tout;` to save the results. Next set the step to 5V, simulate, and save with `V5=V; time5=tout;`. Finally, set the step to 50V, simulate, and save with `V50=V; time50=tout;`.
4. Now simulate the system with the saturation modeling; set the limits to $\pm 10V$. Again simulate with 0.5V, 5V and 50V steps; this time, save the data as `V05sat`, `V5sat` and `V50sat`.
5. Make one plot overlaying `V05` and `V05sat`, another plot overlaying `V5` and `V5sat`, and another plot overlaying `V50` and `V50sat`. Print these plots for the report.
6. Save this model so you have it for the report. TA \checkmark :

II. STEP RESPONSE WITH LOW DC GAIN LEAD CONTROLLER

The objective of this exercise is to study the step response of the motor with the low DC gain controller from Equation 6.1 in part (e) of the prelab. Since this is a type 1 system, in the absence of disturbances the steady state error will be zero. In this experiment you will see that this compensator is not able to overcome the Coulomb friction of the motor. This will result in a large steady state error in the step response.

1. Make sure power to the patch panel is off. Turn on the scope and the signal generator.

2. Connect the potentiometer to the motor assembly.



3. Wire up the circuit shown above.

4. Connect the push button to the **Amp Inhibit** on the patch panel.

5. The waveform generator should have the following settings:

(a) Output impedance = Infinity.

- Press the **Shift** key, then **Enter** to activate the “MENUS” mode.
- Press the > key 3 times until the menu item displayed on the screen is “D: SYS MENU”.
- Press the ∨ key twice until the display reads “50 OHM”.
- Press the > key to change to “HIGH Z”.
- Press the **Enter** key to exit menu setup.


(b) Frequency = 200 mHz

(c) Amplitude = 1 V p-p

(d) Waveform = square

(e) DC offset = 2 V

6. Scope settings:

- (a) Channel 1 and Channel 2: Both should be On, and should be set on D.C. Coupling, 500mV/div. Place the ground lines () one division from the bottom of the screen
 - (b) **Main / Delayed** button set to **Main mode**.
 - (c) **Timebase** = 50 ms.
 - (d) **Trigger Mode** = Normal.
 - (e) **Trigger Edge Level** = 2 V.
 - (f) **Trigger Edge Source** = Ch 1
 - (g) **Waveform Acquire** → Acq. Mode → High Resolution
7. Follow the procedures from Lab 5 (remember load `rtwin486` and save it as your own copy in folder `C:\NETID`, then connect ports to corresponding signals) to create a SLDRT model that implements Figure 6.1. You may use a **Transfer Fcn** block for the controller. Set the gains to match the low-DC gain control from part (e) of the prelab. Adjust the configuration to use **ODE1** with a *Timestep* of 0.002. Set the *Stop time* to a large value (999 s).
 8. Turn on the patch panel power and amplifier.
 9. Press the push button to enable the motor amplifier. Explore the effects of disturbances on this controller. In between transitions of the square wave, physically rotate the motor shaft with your fingers and observe if the controller is able to bring the motor back to its previous position. Apply a disturbance torque with your fingers. Lightly press on the motor shaft to increase its friction. Record your observations. This control is rather loose because of its low DC gain.
 10. Press the push button until a good step response appears on the scope. Then release the push button and press **Stop** on the scope to store the display. If the trace disappears, pressing **Stop** again will recall it.
 11. Using the scope's built-in measurement functions, measure the output overshoot, rise time. As for settling time, and the steady state error, use scope cursor to manually measure them. Does the position of the flywheel before the step make a difference?
 M_p : _____ t_r : _____ t_s : _____ e_{ss} : _____
 12. Press **Run** on the scope to resume data capture.

III. STEP RESPONSE WITH HIGH DC GAIN COMPENSATION

In the previous exercise, you saw that the controller was weak and unable to overcome disturbances. We shall see in this exercise that a compensator with greater DC gain is useful in overcoming the effect of disturbances.

1. Retain all the connections that you had in the previous section.

2. Change the control gains to correspond to your design from prelab part (d). (A rebuild is required only if you change the frequency-domain blocks or signal paths.)
3. Explore the effects of disturbances on this controller. In between transitions of the square wave, physically rotate the motor shaft with your fingers and observe if the controller is able to bring the motor back to its previous position. Apply a disturbance torque with your fingers. Lightly press on the motor shaft to increase its friction. Record your observations. The control should feel much *tighter* than before, indicating a higher DC gain.
4. Observe the step response and measure the overshoot, rise time, settling time, and steady-state error.

M_p : _____ t_r : _____ t_s : _____ e_{ss} : _____

IV. FREQUENCY RESPONSE OF THE HIGH DC GAIN DESIGN


One benefit of frequency design is that the frequency response can generally be found experimentally, without the need for a mathematical system model. Design may be done directly on a Bode plot, or the plot may be used to extract poles and zeros and thus create a system model. To obtain the frequency response, input a sinusoidal control signal and measure the output. In this experiment, we use the Dynamic Signal Analyzer (DSA) to automate the process. After it has collected sufficient data, the DSA will numerically fit poles and zeros to the frequency data, thus creating a system model.

Because the potentiometer wraps every 2π radians, it would be extremely difficult to use with the DSA. Thus, we will obtain the velocity transfer function (6.2) using the tachometer and convert it to find the position transfer function (6.3).

$$\frac{V_{\text{tach}}(s)}{V_i(s)} = \frac{K_c(\tau_z s + 1)}{\tau_p s + 1} \frac{K K_{\text{tach}} K_{\text{amp}}}{\tau_m s + 1} \quad (6.2)$$

$$\frac{V_\theta(s)}{V_i(s)} = \frac{K_c(\tau_z s + 1)}{\tau_p s + 1} \frac{K}{\tau_m s + 1} \frac{K_{\text{pot}} K_{\text{amp}}}{s} \quad (\text{open-loop TF}) \quad (6.3)$$

Procedure:

- Turn off the motor amplifier and waveform generator. Turn on the DSA.
- While the DSA is booting (takes a while), its connection with the PC needs to be initialized. To do this, right click on the IO button () in the system tray (bottom right, next to the clock). When the menu comes up (takes about 20 seconds), click on **Agilent Connection Expert** and check that the DSA has been detected.
- On the scope, change the **Timebase** to 500 ms. Switch to **ROLL** mode (**Main/Delayed** button). The other scope settings are the same as before.
- Be sure the parameter settings are correct for the high DC gain controller.

- Disconnect the potentiometer from the motor assembly (not only the feedback loop in the simulink file has to be disabled but also the potentiometer on the left of the motor has to be *physically* detached).
- In the SIMULINK model, remove the feedback loop (disconnect ADC 1 and remove the sum block). Use the high DC gain controller from prelab part (d). Set the *Stop time* to a large value (999 or *inf*).

Wire the bench according to Figure 6.2. Do not disconnect the trigger cable attached to channel 1 of the DSA.

Have the TA check your wiring and SIMULINK model. TA √:

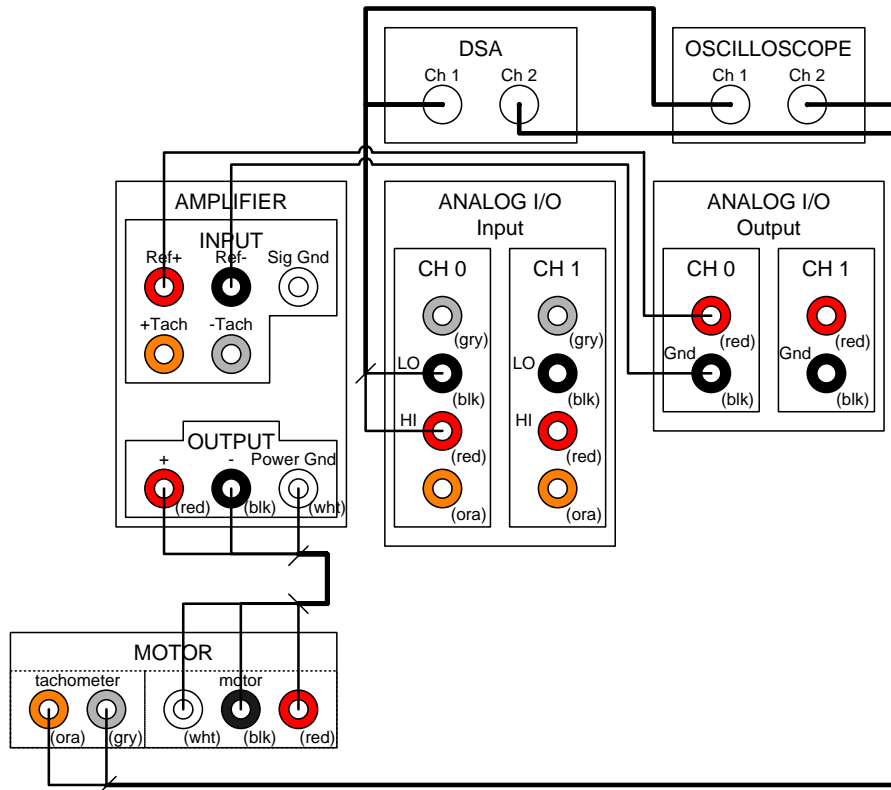


Figure 6.2. : Wiring diagram for frequency response.

We will use VEE to control the DSA. Open the file `N:\labs\ECE486\dsacontrol.vxe`. This will use the DSA to get samples of the frequency response from Channel 1 to Channel 2.

*Note: Agilent VEE displays files of type *.vee by default. You must change this box to VEE RunTime Programs (*.vxe)*

- Press the **Initialize** button in VEE.
- Build, Connect and Start the SLDRT system
- Turn on the amp
- Push the $\frac{1}{4}$ inch plug into the pushbutton jack.

- Press **Start Measurement** in VEE.
- Enter 0.5 for the start frequency, 20 for the stop frequency, 15 for the number of points, 0.2 for the (source) level, and 1 for the (source) offset voltage.

Watch the waveforms on the DSA and the scope. When the sine sweep is completed (takes about 3-5 minutes), you will save your data using VEE. The DSA displays “**Sweep Complete**” when finished.

- Pull out the $\frac{1}{4}$ inch plug from the pushbutton jack.
- Press **Collect Data**. The frequency response samples will be saved into file `U:\fresp.m`. The first column of this file is the frequency points in rad/s and the second and third columns are the real and imaginary parts of the frequency response.
- Press **Get Fit**. Enter the frequency range for the fit—usually the same as the sweep, unless the high frequency data looks corrupted (jagged or sharp turns). Enter 2 for the number of poles and 1 for the number of zeros. The DSA will then find the coefficients for (6.4).

$$\frac{K_{\text{DSA}}(\bar{s}[\text{Hz}] - z[\text{Hz}])}{(\bar{s}[\text{Hz}] - p_1[\text{Hz}])(\bar{s}[\text{Hz}] - p_2[\text{Hz}])} \quad (6.4)$$

with K_{DSA} , z , p_1 , and p_2 in Hertz as indicated.

K_{DSA} : _____ z : _____[Hz] p_1 : _____[Hz] p_2 : _____[Hz]

Notes: Since our analysis uses s [rad/s], unit conversion is required. The easiest way to do this correctly is to convert \bar{s} in-place as shown in (6.5); incorrect conversion can change the system gain by a multiple of 2π .

$$\frac{K_{\text{DSA}}\left(\frac{s[\text{rad/s}]}{2\pi} - z[\text{Hz}]\right)}{\left(\frac{s[\text{rad/s}]}{2\pi} - p_1[\text{Hz}]\right)\left(\frac{s[\text{rad/s}]}{2\pi} - p_2[\text{Hz}]\right)} \quad (6.5)$$

In addition to converting this transfer function, you will be using the raw frequency response data in the lab report. To load in the data, you might do something like this:

```
cd U;
fresp
omega=dsa(:,1);
data=dsa(:,2)+i*dsa(:,3);
```

Making a Bode plot involves using **subplot** to separate the magnitude and phase plots. Use **abs** in MATLAB to obtain the magnitude and **phase** to obtain the phase. Do not forget to scale the magnitude to dB (see **log10**) and the phase to degrees.

🔔 REPORT 🔔

I: SIMULATION

1. Discuss the effect of the saturation block in simulation. When was it significant or insignificant? Include the plots to support your statements.

II: MOTOR DC GAIN

1. Compute the transfer function of the system from the data obtained from the DSA. Then, compute the DC gain K of the motor (solve for K). Compare it with the value obtained in Lab 4 or PreLab 5a. *Note:* first, rewrite the transfer function from the DSA in radians. Also, remember that the DC gain obtained from the DSA includes the gains of the tachometer, amplifier, and compensator.

III: REAL AND SIMULATED RESPONSE OF THE LOW AND HIGH DC GAIN COMPENSATORS

1. Make a table comparing the overshoot, rise time, settling time and steady state error for the slow and fast lead compensators. It must include the actual values found in the lab and the values found from simulating the closed-loop system using the motor parameters found in Lab 4. Also redo the simulation using the gain (with K_c removed) and slow pole from the DSA fit for the motor plant parameters; include the values from these simulations as well. Discuss.
2. Comment on the two controllers, and compare them with the PD controller used in Lab 5. In each case, could you meet the controller design specs when implementing on the real motor?

IV: BODE PLOTS OF TRANSFER FUNCTION

1. Use MATLAB to make a Bode plot of the transfer function fit by the DSA and overlay (magnitude and phase) the frequency samples of $\frac{V_{tach}}{V_i}(s)$ obtained from the data in fresp.m (use a different line-type). Note the quality of the fit.
2. Obtain the frequency response of $\frac{V_\theta}{V_i}(s)$ from the poles and zeros fit by the DSA, using the relations between Equations 6.3 and 6.2, and make a Bode plot (they should match well). Overlay the actual samples after computing $\frac{V_\theta}{V_i}(s)$ from the data in fresp.m. Include this plot in your report. Find the phase margin and cross-over frequency ω_c for both curves. Do they meet the design specifications from the prelab part (d)? Discuss.

Hint: Converting data points from (6.2) to (6.3) is actually rather simple. Notice that the only differences are the gain and a pole. These may be compensated for by adjusting the magnitude and phase responses separately. Using your knowledge of how a pole or gain affects a Bode plot, you can scale or offset the data as needed.

3. Use this open-loop transfer-function-fit to compute the closed-loop transfer function and make a Bode plot with MATLAB. Compute samples of the closed loop response from the samples of $\frac{V_o}{V_i}(s)$ and overlay. What is the closed-loop bandwidth? Is it reasonable? Why?

Congratulations on finishing this lab manual!
Enjoy the final project.



The Analog Computer - Theory and Practice

I. Introduction

Many problems encountered in scientific and engineering studies require the solution of mathematical equations or sets of equations which describe the behavior of physical systems. For complex systems, the solution of the equations may be difficult or tedious by means of pencil and paper. An analog computer may be used to overcome this difficulty by providing the engineer with an electronic model which permits the rapid solution of mathematical equations. The electronic analog computer makes possible the building of an electrical model of a physical system in which voltages will behave with time in a similar way to the actual variables of interest in the actual network or system.

The analog computer is basically a set of building blocks, each able to perform specific mathematical operations, such as addition, subtraction, integration, multiplication, etc. on voltages. By constructing an appropriately connected group of such blocks, an electronic model can be created in which the voltages at the outputs of the various blocks obey the relationships given in the mathematical description of the physical system. As our interest is primarily in dynamic systems, the describing equations are usually differential equations with time as the independent variable. By applying appropriate initial conditions and forcing functions to the electrical model, its behavior will be the same as the original physical system. By using an oscilloscope or other device, the voltages at various points in the model can be measured and compared to time or each other.

The purpose of this tutorial is to introduce the student to a few methods of interconnecting basic computing units (this is known as analog programming) and to provide familiarity with the GP-6 analog computer. For a more extensive treatment on the use of analog computers, the student should refer to specialized texts on analog computers. Several are available in the Engineering Library.

II. Analog Computer Functions

We will be concerned with three basic operations:

- a. Addition
- b. Multiplication by a constant
- c. Integration

One computing unit which is capable of performing all three of these operations is a high gain D.C. amplifier known as the operational amplifier. The symbol most commonly used for the operational amplifier is:

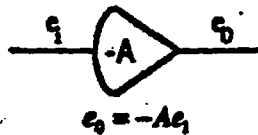


Fig. 1.

Typically, A is very large, and is usually assumed to approach infinity when the op-amp is an element of a larger circuit, as will be illustrated below.

a. Addition

Consider the operational amplifier interconnected with input and feedback resistors shown in Figure 2. Summing the currents at the node G (assuming that the D.C. amplifier does not draw any input current) we have

$$\frac{e_1 - e_1}{R_1} + \frac{e_2 - e_1}{R_2} + \frac{e_0 - e_1}{R_f} = 0$$

and, since $e_0 = -Ae_1$, $-e_1 = e_0/A$,

$$\frac{e_1}{R_1} + \frac{e_2}{R_2} + \frac{e_0}{R_f} = \left(\frac{e_0}{A}\right)\left(\frac{1}{R_1} + \frac{1}{R_2} + \frac{1}{R_f}\right) = 0$$

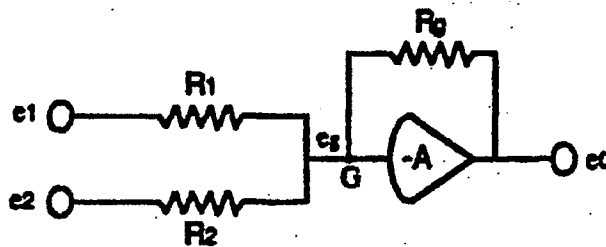


Fig. 2.

If A approaches infinity, the last term above may be dropped, and

$$-\left(\frac{e_1}{R_1} + \frac{e_2}{R_2}\right) = \frac{e_o}{R_f}$$

Thus, we have obtained the result that the input-output relation of this amplifier circuit is solely dependant on the ratio of feedback to input resistors.

To perform simple addition, we can let $R_1 = R_2 = R_f$; then

$$e_o = -(e_1 + e_2) \quad (\text{note the minus sign})$$

If $R_1 \neq R_2$ then we will perform a weighted addition. This can be generalized to more than two inputs.

b. Multiplication by a Constant

With only one input to the operational amplifier, the input-output equation is

$$-\left(\frac{R_f}{R_1}\right) e_1 = e_o$$

and we can perform multiplication by a negative constant of any magnitude by appropriately choosing R_1 and R_f .

If the multiplicative constant is less than one, there is a simpler way to design the multiplier using a potentiometer, as shown in Figure 3. Note that in the case of the potentiometer, there is no sign inversion.

$$e_o = \left(\frac{R_2}{R_2 + R_1}\right) e_1$$

or

$$e_o = A e_1, \quad 0 \leq A \leq 1$$

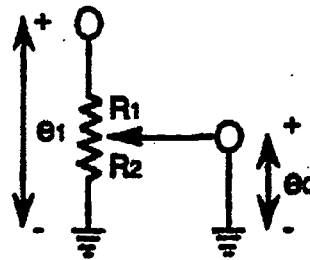


Fig. 3.

c. Integration with Respect to Time

If we connect the operational amplifier as shown in Figure 4, with a capacitor in the feedback path, the current equation at node G is

$$\frac{e_1 - e_2}{R_1} + \frac{e_2 - e_2}{R_2} + C \frac{d(e_0 - e_2)}{dt} = 0$$

Substituting $e_2 = -e_0/A$ into the last equation, we have

$$C \frac{d\left(e_0 - \frac{e_0}{A}\right)}{dt} = - \left[\frac{\left(e_1 - \frac{e_0}{A}\right)}{R_1} + \frac{\left(e_2 - \frac{e_0}{A}\right)}{R_2} \right]$$

Since A approaches infinity, the last equation becomes

$$C \frac{de_0}{dt} = - \left[\frac{e_1}{R_1} + \frac{e_2}{R_2} \right]$$

Integrating both sides with respect to t , we have

$$\int_0^t C \frac{de_0}{dt} dt = - \int_0^t \left[\frac{e_1}{R_1} + \frac{e_2}{R_2} \right] dt$$

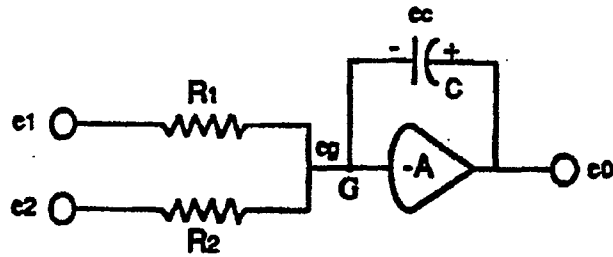


Fig. 4.

or,

$$e_0(t) = \left[-\frac{1}{R_1 C_0} \int e_1(t) dt - \frac{1}{R_2 C_0} \int e_2(t) dt \right] + e_c(0)$$

Thus we can integrate and sum using the same technique as in Figure 4. Since $e_0(0) = e_c(0) + e_g(0)$ and e_g is very small (less than 1 mV), $e_0(0) = e_c(0)$. The integration constant can therefore be physically obtained by applying a voltage to the capacitor of magnitude $e_0(0)$. Because of the nature of the charging circuit for this voltage a negative voltage must be applied to the initial condition input to obtain a positive initial condition of the same magnitude. This inversion has been adopted into the block diagram convention described below.

It should be noted that many other input-output relations can be obtained through use of the operational amplifier and other devices. (Multiplying two variables together, variable switching, nonlinear transfer functions, etc.)

III. Analog Programming Diagrams

To make drawing analog computer diagrams easier, the circuit schematic has been replaced by an appropriate symbol:

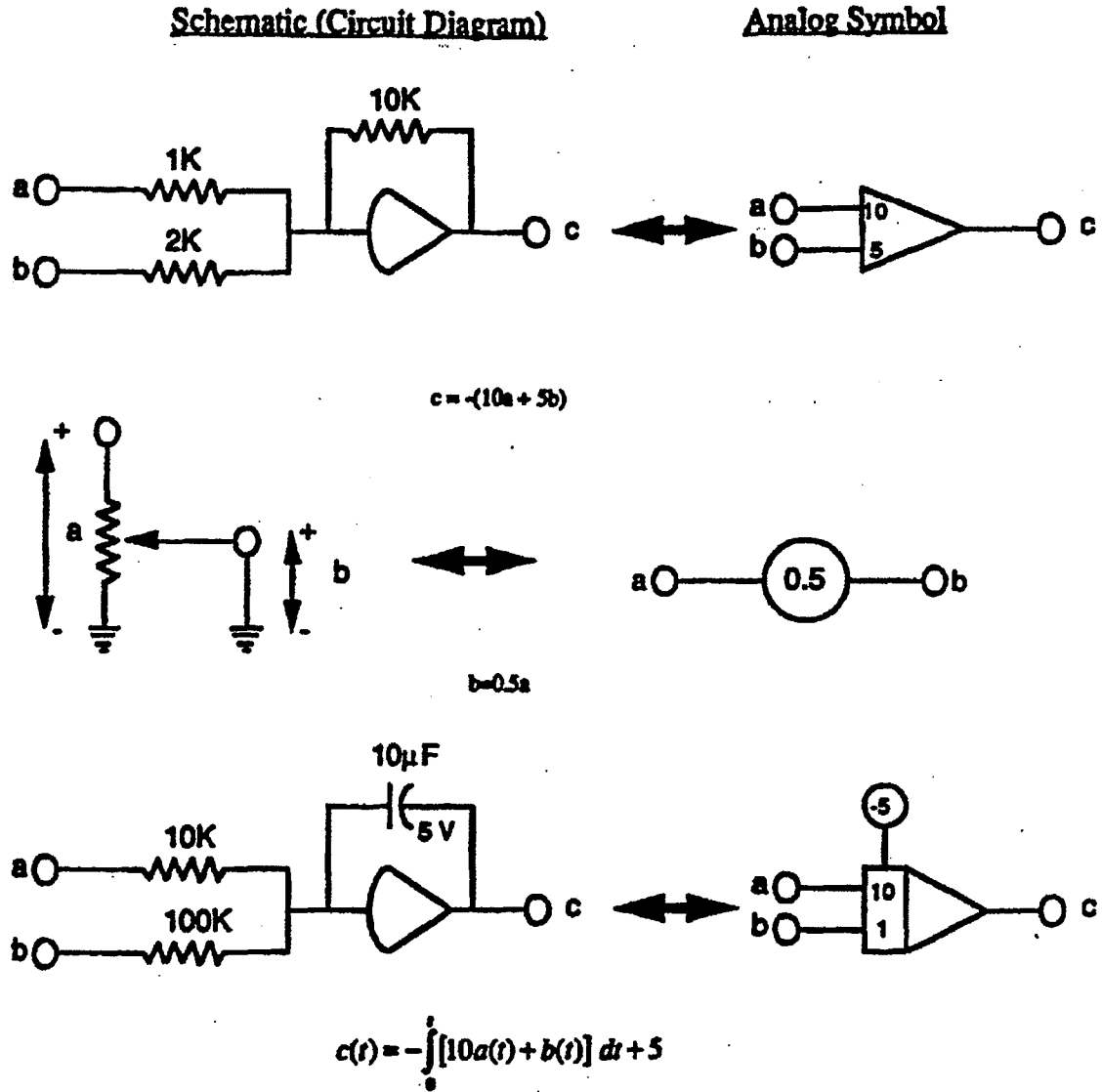


Fig. 5.

IV. Going from the Equation to the Block Diagram

While there are many different methods of implementing a computer diagram from a mathematical equation, we shall concentrate on the following two methods:

- a. Solving for the highest-order derivative from a single differential equation.
- b. State Variable Approach (solving several simultaneous first-order differential equations)

Part c will deal with a generalization of a) to the situation where there are derivatives of the input signal (zeros) in the transfer function.

a. Solving for the highest-order derivative

This method will be illustrated by the following problem:

Given:

$$\frac{d^2 y(t)}{dt^2} - 3y(t) = u(t) \quad \text{with } y(0) = 5 \text{ and } \frac{dy(0)}{dt} = 4$$

Problem: Draw a computer block diagram that will give the solution for (i.e. has the output) $y(t)$.

1. The first step is to solve for the highest-order derivative in the differential equation. Thus

$$\frac{d^2 y(t)}{dt^2} = u(t) + 3y(t)$$

Now, applying the right hand side of the last equation to the inputs of an integrator, we have the situation as shown in Figure 6. The output of the integrator will be $-(dy/dt)$ since

$$-\int [u(t) + 3y(t)] dt = -\int \frac{d^2 y(t)}{dt^2} dt = -\frac{dy(t)}{dt}$$

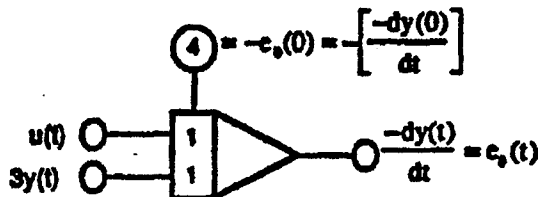


Fig. 6.

2. Continue integrating the output of this integrator until $y(t)$ is obtained (applying initial conditions to the appropriate integrators along the way).

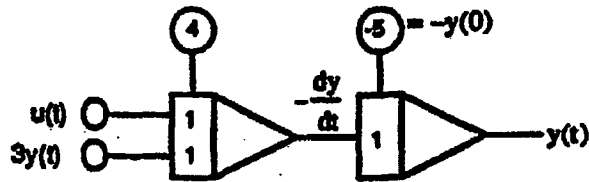


Fig. 7.

3. Form all interconnections necessary to complete the diagram. Figure 9 gives the complete computer block diagram after all necessary connections have been made.

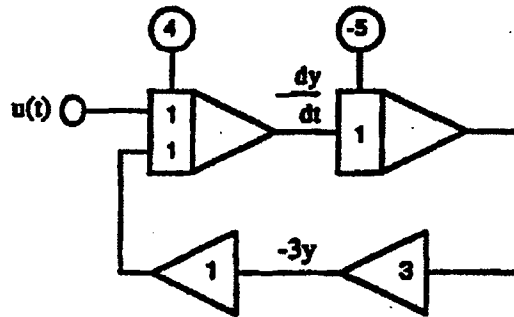


Fig. 8.

4. It is possible to simplify the diagram in Fig. 9 in order to obtain one with the lowest number of components. The important thing to keep in mind here is that if all the loop-gain products are not changed, the same transfer function from the input to the output will result. (This can be proven by Mason's rule) Figure 10 shows the simplified block diagram and the input source which supplies the unit step input.

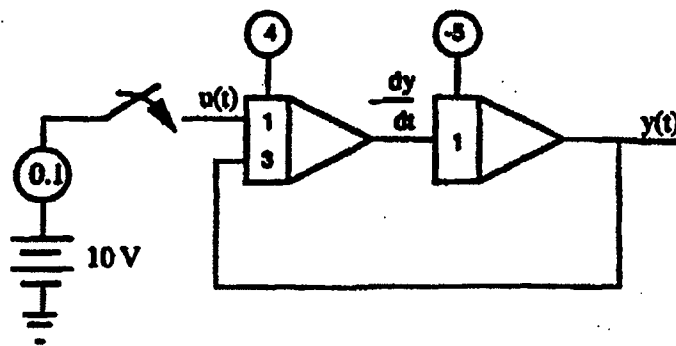


Fig. 9.

As another example, consider the differential equation:

$$\frac{d^2x(t)}{dt^2} + 1.5\frac{dx(t)}{dt} + 4x(t) = f(t)$$

with $x(0) = 4$ and $\frac{dx(0)}{dt} = -6$.

Following the method outlined above, the computer diagram show in Figure 10 may be obtained.

Verify this!

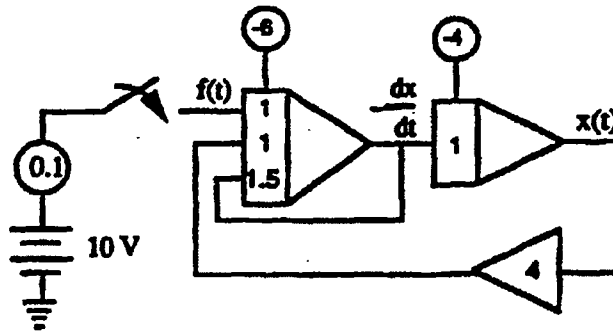


Fig. 10.

b. State variable approach to computer programming

Since state equations are first-order differential equations, computer implementation may be performed for each equation using the method described in the previous section.

Consider the state equations:

$$\frac{dx_1(t)}{dt} = 2x_1 - 3x_2 + f(t) \quad x_1(0) = 2$$

$$\frac{dx_2(t)}{dt} = 5x_1 + 2f(t) \quad x_2(0) = 3$$

$$y(t) = x_1 + 10x_2$$

Problem: Determine the output $y(t)$ when the input is a ramp function, $f(t) = tu(t)$

First, applying each of the right-hand side terms of the state equations to an integrator, we have the block diagram shown in Figure 11.

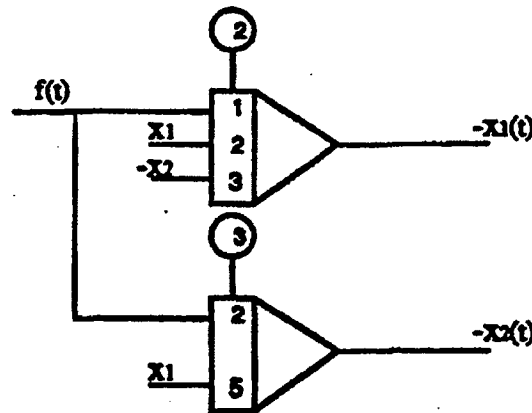


Fig. 11.

Next, we make the necessary interconnections to satisfy the state equations and the block diagram shown in Figure 12 results. Figure 13 gives the complete computer diagram with the input supply and the realization of the output equation.

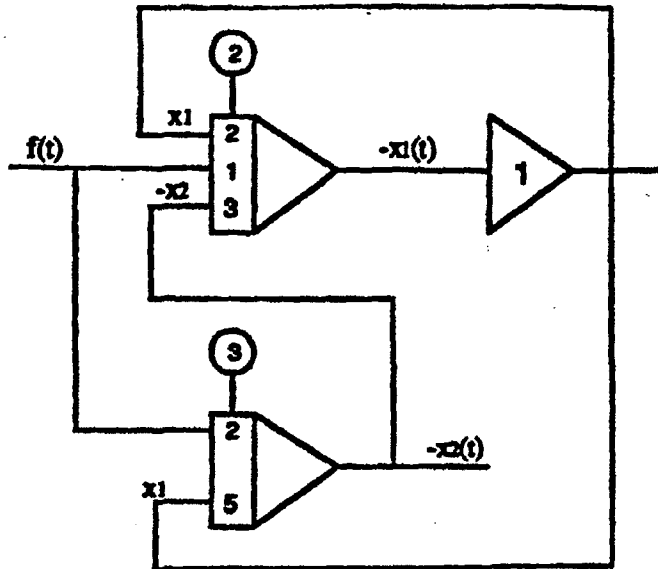


Fig. 12.

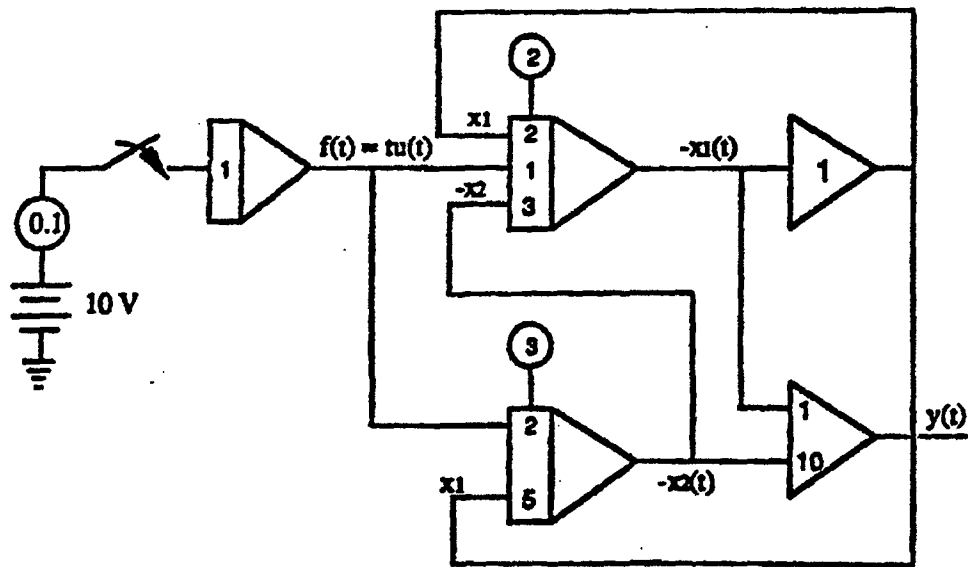


Fig. 13.

c. Representing Systems with Zeros

When a physical system has one or more zeros, the solving for the highest-derivative approach leads to difficulty when the designer actually tries to move from the equations to the block diagram. Consider a system with the transfer function

$$F(s) = \frac{Y(s)}{U(s)} = \frac{as+b}{cs+d}$$

We can represent this as two separate transfer functions:

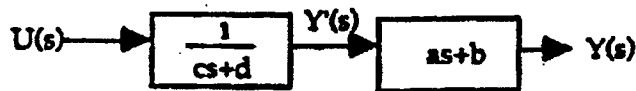


Fig. 14.

The transfer function from $U(s)$ to $Y(s)$ is in standard form, and we can easily reduce this to a block diagram, as follows

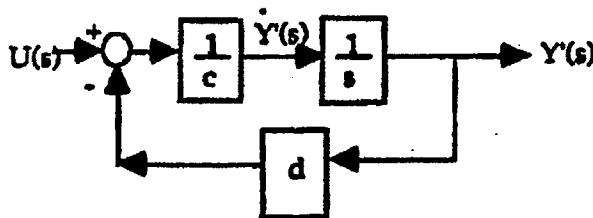


Fig. 15.

(Note that this is the traditional all-integrator block diagram. There is no inversion in the gain blocks or in the integrator. However, it is simple to convert the all-integrator block diagram to the analog computer diagram)

We have labeled the derivative of $Y(s)$. We now note that

$$Y(s) = (as + b) \dot{Y}(s) = a \dot{Y}(s) + b Y'(s)$$

Both Y' and Y' are measurable signals. We can represent the complete block diagram as

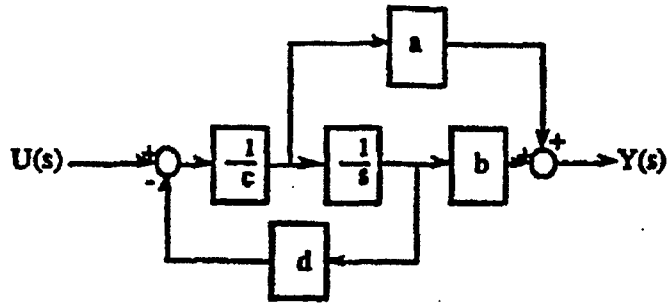


Fig. 16.

This technique can be extended to systems with any number of integrators, so long as the number of zeros do not exceed the number of poles. Transfer functions that have this characteristic are known as *proper* transfer functions.

Appendix B

The GP-6 Analog Computer: Characteristics and Operating Instructions

MODE CONTROL Push Buttons:

1. **Initial Condition:** Resets integrators to initial conditions
2. **Hold:** Stops integration at current position
3. **Operate:** Starts integrators operating
4. **Repetitive Operation:** Alternately resets and operates integrators

MODE SELECTOR Switch:

The MODE SELECTOR switch alternates between POT SET and OPR.

POT SET Mode

1. The inputs of the potentiometers are set to +10 Volts.
2. The input to the digital voltmeter (DVM) is connected to the output of the potentiometers.
3. The integrators are placed in their initial condition state.

OPR Mode

1. The inputs to the potentiometers are connected to the patch board.
2. The input to the DVM is connected to the rear terminal METER INPUT.
3. The integrators are controlled by push button selections.

Setting Coefficient Potentiometers:

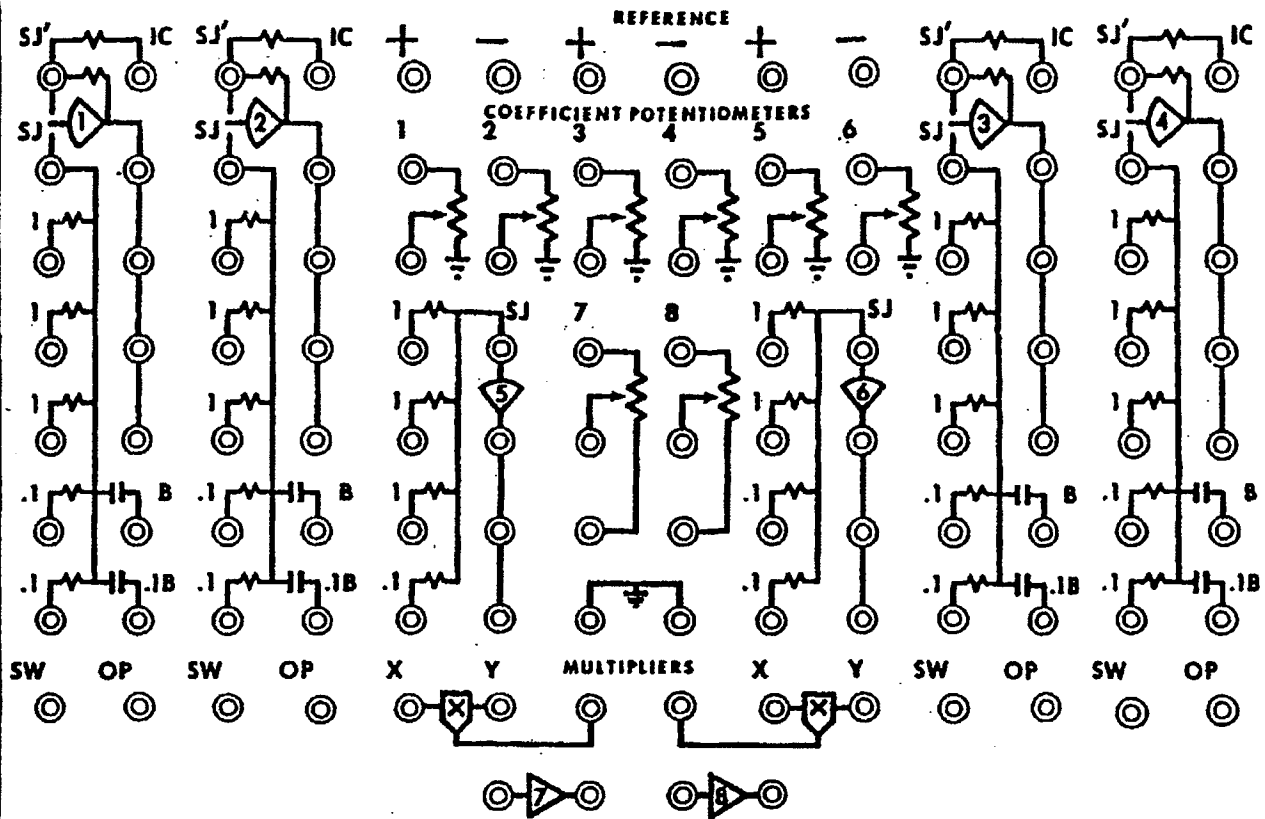
1. Complete all patching so settings are made under their operating loads.
2. Position the MODE SELECTOR switch to POT SET.
3. Position the Y/POT ADDRESS to the number of the pot to be set.
4. Adjust the pot knob until the desired setting is displayed.

If an amplifier overload alarm occurs when the POT SET mode is selected, remove the condition by patching the overloaded amplifier output directly to its summing junction. (Be sure to remove the patch cord prior to running the program.)

Measurement of amplifier outputs in the OPR modes:

1. Connect the Y OUTPUT rear terminal to the METER INPUT terminal
2. Position the MODE SELECTOR switch to the OPR position.
3. Position the Y/POT ADDRESS switch to the number of the amplifier output to be measured.











3. GP-6 PATCH PANEL



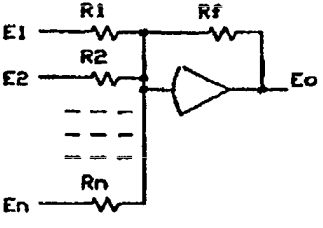

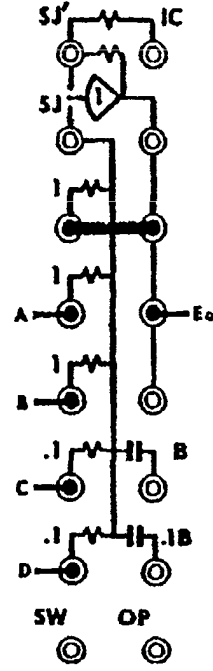
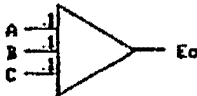
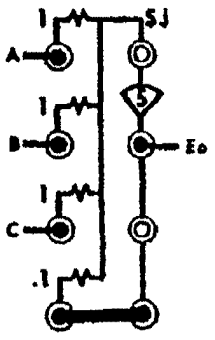

Patch panel graphics use standard analog computer programming symbols. Amplifiers 1 thru 4 are single ended, high gain amplifiers with electronic mode switches and summing resistor/integrating capacitor networks that may be programmed as summer/inverters, integrators, track/store and single pole, double throw electronic switch amplifiers. Amplifiers 4 and 5 are summer/inverters. Amplifiers 7 and 8 are inverters only. Potentiometers 1 thru 6 are grounded attenuators. Potentiometers 7 and 8 have their bottom ends open and may be used as either voltage dividers or attenuators. Multiplier networks produce current outputs for direct connection to amplifier summing junctions, and thus may be patched as multipliers, dividers, squarers and square root extractors.

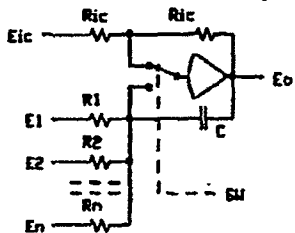
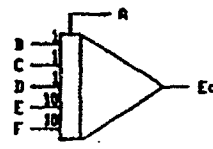
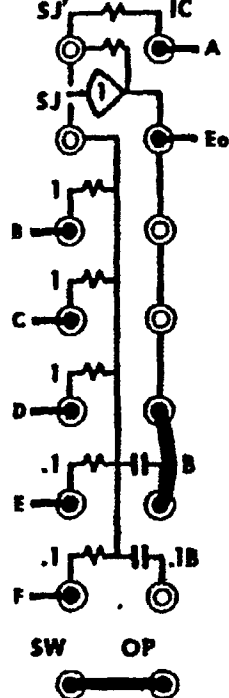



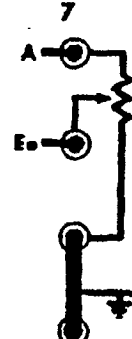
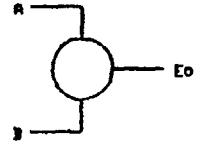
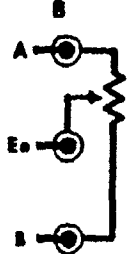
The following is an explanation of patch panel symbols:

<u>SYMBOL</u>	<u>COLOR CODE</u>	<u>DESCRIPTION</u>
+	Red	Positive reference, considered unity (1.000) for normalized programming. (Actual amplitude is 10 volts.)
-	Yellow	Negative reference.
		High gain operational amplifier.
		High gain operational amplifier with electronic switch.
		Inverter.
Y	Red	Amplifier output.
SJ	Gray	The summing junctions for amplifiers 1 - 6. (Active for amplifiers 1 - 4 when a logic "1" is patched to the SW switch control jack or when there is no switch control patching.)

<u>SYMBOL</u>	<u>COLOR CODE</u>	<u>DESCRIPTION</u>
SJ'	Gray	Alternate summing junction for amplifiers 1 - 4. (Conducting when a logic "0" is patched to the SW switch control jack.)
	Green	Standard input summing resistor, normalized as a unity value to simplify programming. (Actual resistance is 50K ohms.)
	Green	Summing resistor input one tenth the standard value. (Actual resistance is 5K ohms.)
	Green	One end of standard integrating capacitor, normalized so that the "1" resistor and B capacitor combine to produce a one second time constant, as referred to programming time scales. (Actual capacitance is 20 ufd for the slow and .05 ufd for the fast time scales.)
	Green	Integrating capacitor that has a value one tenth the standard B capacitor. (Actual capacitance is 2 ufd for the slow and .005 ufd for the fast time scales.)
	Green	Resistor network for the SJ' summing junction. Amplifier becomes an inverter when SJ' is conducting. Normally used for integrator initial conditions. May also be used as the feedback and input with the SJ summing network. (See Summer patching.)
	Yellow	Attenuator: bottom end grounded; top end input and wiper output brought out to the panel -- wiper indicated by the arrow.
	Yellow	Voltage divider: top and bottom end inputs and wiper brought out to the panel -- wiper indicated by the arrow.
	Black	System ground.
		Multiplier network symbol.
X	Brown	One of two multiplier inputs.
Y	Brown	One of two multiplier inputs.
	Gray	Multiplier output, a current proportional to the product of "X" and "Y;" normalized so that when connected to the summing junction of an operational amplifier with a "1" resistor feedback, and with reference patched as both inputs, the amplifier output equals reference.
SW	White	Electronic switch control. Logic 0 (ground or positive voltage,) SJ' conducts, SJ shuts off. Logic 1 (-5V or less,) SJ conducts and SJ' shuts off. Hold logic (-2V thru -3V) SJ' shuts off and the summer resistor network is disconnected electronically from the amplifier/capacitor feedback.
OP	White	Computer's operate bus, integrator mode logic from the central operate button control, patched to "SW" for normal integrator operation.

PATCH PANEL OPERATIONS

FUNCTION	OPERATION	PATCHING
<p>SUMMER (amplifiers 1 - 4)</p>	<p>Fundamental Summer Operation</p>  $E_o = -R_f(E_1/R_1 + E_2/R_2 \dots + E_n/R_n)$  $E_o = -(A + B + 10C + 10D)$ <p>NO PATCHING TO SWITCH CONTROL "SW"</p>	
<p>SUMMER (amplifiers 5 & 6)</p>	 $E_o = -(.1A + .1B + .1C)$	
<p>INVERTER (amplifiers 7 & 8)</p>	$E_o = -A$	

FUNCTION	OPERATION	PATCHING
<p>INTEGRATOR (amplifiers 1 - 4)</p>	<p>Fundamental Integrator Operation</p>  $E_o = -R_f \int (E_1/R_1 + E_2/R_2 + \dots + E_n/R_n) dt - E_{ic}$  $E_o = -\int (A + C + D + 10E + 10F) dt - A$	
<p>ATTENUATOR (pots 1-6)</p>	 $E_o = K (A)$	
<p>ATTENUATOR (pots 7 & 8)</p>	 $E_o = K (A)$	
<p>VOLTAGE DIVIDER (pots 7 & 8)</p>	 $E_o = K (A - B) + B$	

Feedback Control of Dynamic Systems

Fourth Edition

Gene F. Franklin
Stanford University

J. David Powell
Stanford University

Abbas Emami-Naeini
SC Solutions, Inc.



Prentice Hall
Upper Saddle River, New Jersey 07458

Library of Congress Cataloging-in-Publication Data

Franklin, Gene F.

Feedback control of dynamic systems / Gene F. Franklin,
J. David Powell, Abbas Emami-Naeini.—4th ed.

p. cm.

Includes index.

ISBN 0-13-032393-4

1. Feedback control systems. I. Powell, J. David. II. Emami-Naeini, Abbas. III. Title.
CIP Data available.

Vice President and Editorial Director, ECS: *Marcia J. Horton*

Acquisitions Editor: *Eric Frank*

Editorial Assistant: *Jessica Romeo*

Vice President and Director of Production and Manufacturing, ESM: *David W. Riccardi*

Executive Managing Editor: *Vince O'Brien*

Managing Editor: *David A. George*

Production Editor: *Irwin Zucker*

Composition: *PreTeX, Inc.*

Director of Creative Services: *Paul Belfanti*

Creative Director: *Carole Anson*

Art Director: *Jonathan Boylan*

Assistant to the Art Director: *John Christiana*

Art Editor: *Xiaohong Zhu*

Interior Designer: *Maria Guglielmo*

Cover Designer: *Stacey Abraham*

Manufacturing Manager: *Trudy Piscioti*

Manufacturing Buyer: *Lisa McDowell*

Marketing Manager: *Holly Stark*

About the Cover: *Photograph of computer hard drive courtesy of Corbis Images. Photograph of satellite orbiting the earth courtesy of NASA. Photograph of Boeing 747 jet aircraft courtesy of Tony Stone Images.*



© 2002, 1994, 1991, 1986 by Prentice Hall

Prentice-Hall, Inc.

Upper Saddle River, New Jersey 07458

All rights reserved. No part of this book may be reproduced, in any format or by any means, without permission in writing from the publisher.

The author and publisher of this book have used their best efforts in preparing this book. These efforts include the development, research, and testing of the theories and programs to determine their effectiveness. The author and publisher make no warranty of any kind, expressed or implied, with regard to these programs or the documentation contained in this book. The author and publisher shall not be liable in any event for incidental or consequential damages in connection with, or arising out of, the furnishing, performance, or use of these programs.

MATLAB and Simulink are registered trademarks of The MathWorks, Inc., 3 Apple Hill Drive, Natick, MA, 01760-2098.

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

ISBN 0-13-032393-4

Pearson Education Ltd., *London*

Pearson Education Australia Pty. Limited, *Sydney*

Pearson Education Singapore, Pte. Ltd.

Pearson Education North Asia Ltd., *Hong Kong*

Pearson Education Canada Inc., *Toronto*

Pearson Educación de México, S.A. de C.V.

Pearson Education—Japan, *Tokyo*

Pearson Education Malaysia, Pte. Ltd.

Pearson Education, *Upper Saddle River, New Jersey*

▲ 3.8 Obtaining Models from Experimental Data

There are several reasons for using experimental data to obtain a model of the dynamic system to be controlled. In the first place, the best theoretical model built from equations of motion is still only an approximation of reality. Sometimes, as in the case of a very rigid spacecraft, the theoretical model is extremely good. Other times, as with many chemical processes such as paper-making or metalworking, the theoretical model is very approximate. In every case, before the final control design is done, it is important and prudent to verify the theoretical model with experimental data. Secondly, in situations where the theoretical model is especially complicated or the physics of the process is poorly understood, the only reliable information on which to base the control design is the experimental data. Finally, the system is sometimes subject to on-line changes, which occur when the environment of the system changes. Examples include when an aircraft changes altitude or speed, a paper machine is given a different composition of fiber, or a nonlinear system moves to a new operating point. On these occasions we need to “retune” the controller by changing the control parameters. This requires a model for the new conditions, and experimental data are often the most effective, if not the only, information available for the new model.

There are four kinds of experimental data for generating a model.

1. **Transient response**, such as comes from an impulse or a step.
2. **Frequency response data**, which result from exciting the system with sinusoidal inputs at many frequencies.
3. **Stochastic steady-state information**, as might come from flying an aircraft through turbulent weather or from some other natural source of randomness.
4. **Pseudorandom-noise data**, as may be generated in a digital computer.

Each class of experimental data has its properties, advantages, and disadvantages.

Transient response data are quick and relatively easy to obtain. They are also often representative of the natural signals to which the system is subjected. Thus a model derived from such data can be reliable for designing the control system. On the other hand, in order for the signal-to-noise ratio to be sufficiently high, the transient response must be highly noticeable. Thus the method is rarely suitable for normal operations, so the data must be collected as part of special tests. A second disadvantage is that the data do not come in a form suitable for standard control systems designs, and some parts of the model, such as poles and zeros, must be computed from the data.¹⁶ This computation can be simple in special cases or complex in the general case.

¹⁶ Ziegler and Nichols (1943), building on the earlier work of Callender et al. (1936), use the step response directly in designing the controls for certain classes of processes. See Chapter 4 for details.

Four sources of experimental data

Transient response

Frequency response

Frequency-response data (see Chapter 6) is simple to obtain but substantially more time-consuming than transient-response information. This is especially so if the time constants of the process are large, as often occurs in chemical processing industries. As with the transient-response data, it is important to have a good signal-to-noise ratio, so obtaining frequency-response data can be very expensive. On the other hand, as we will see in Chapter 6, frequency-response data are exactly in the right form for frequency-response design methods, so once the data have been obtained, the control design can proceed immediately.

Stochastic steady-state

Normal operating records from a natural stochastic environment at first appear to be an attractive basis for modeling systems since such records are by definition nondisruptive and inexpensive to obtain. Unfortunately, the quality of such data is inconsistent, tending to be worst just when the control is best, because then the upsets are minimal and the signals are smooth. At such times some or even most of the system dynamics are hardly excited. Because they contribute little to the system output, they will not be found in the model constructed to explain the signals. The result is a model that represents only part of the system and is sometimes unsuitable for control. In some instances, as occurs when trying to model the dynamics of the electroencephalogram (brain waves) of a sleeping or anesthetized person to locate the frequency and intensity of alpha waves, normal records are the only possibility. Usually they are the last choice for control purposes.

Pseudorandom noise (PRBS)

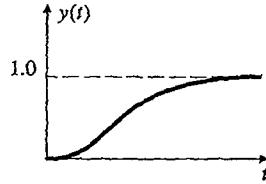
Finally, the pseudorandom signals that can be constructed using digital logic have much appeal. Especially interesting for model making is the pseudorandom binary signal (PRBS). The PRBS takes on the value $+A$ or $-A$ according to the output (1 or 0) of a feedback shift register. The feedback to the register is a binary sum of various states of the register that have been selected to make the output period (which must repeat itself in finite time) as long as possible. For example, with a register of 20 bits, $2^{20} - 1$ (over a million) steps are produced before the pattern repeats. Analysis beyond the scope of this text has revealed that the resulting signal is almost like a broad-band random signal. Yet this signal is entirely under the control of the engineer who can set the level (A) and the length (bits in the register) of the signal. The data obtained from tests with a PRBS must be analyzed by computer, and both special-purpose hardware and programs for general-purpose computers have been developed to perform this analysis.

3.8.1 Models from Transient-Response Data

To obtain a model from transient data we assume that a step response is available. If the transient is a simple combination of elementary transients, then a reasonable low-order model can be estimated using hand calculations. For example, consider the step response shown in Fig. 3.45. The response is monotonic and smooth. If we assume that it is given by a sum of exponentials, we can write

$$y(t) = y(\infty) + Ae^{-\alpha t} + Be^{-\beta t} + Ce^{-\gamma t} + \dots \quad (3.83)$$

Figure 3.45
A step response
characteristic of many
chemical processes



Subtracting off the final value and assuming that $-\alpha$ is the slowest pole, we write

$$\begin{aligned} y - y(\infty) &\cong Ae^{-\alpha t} \\ \log_{10}[y - y(\infty)] &\cong \log_{10} A - \alpha t \log_{10} e \\ &\cong \log_{10} A - 0.4343\alpha t. \end{aligned} \quad (3.84)$$

This is the equation of a line whose slope determines α and intercept determines A . If we fit a line to the plot of $\log_{10}[y - y(\infty)]$ (or $\log_{10}[y(\infty) - y]$ if A is negative), then we can estimate A and α . Once these are estimated, we plot $y - [y(\infty) + Ae^{-\alpha t}]$, which as a curve approximates $Be^{-\beta t}$ and on the log plot is equivalent to $\log_{10} B - 0.4345\beta t$. We repeat the process, each time removing the slowest remaining term, until the data stop being accurate. Then we plot the final model step response and compare it with data so we can assess the quality of the computed model. It is possible to get a good fit to the step response and yet be far off from the true time constants (poles) of the system. However, the method gives a good approximation for control of processes whose step responses look like Fig. 3.45.

EXAMPLE 3.35

Determining the Model from Time-Response Data

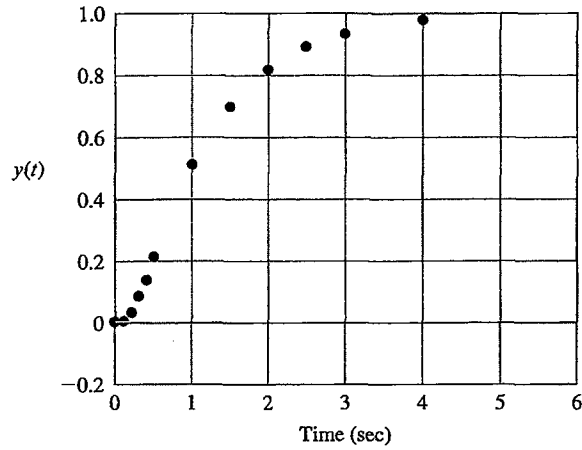
Find the transfer function that generates the data given in Table 3.1 and which are plotted in Fig. 3.46.

TABLE 3.1

Step-Response Data			
t	$y(t)$	t	$y(t)$
0.1	0.000	1.0	0.510
0.1	0.005	1.5	0.700
0.2	0.034	2.0	0.817
0.3	0.085	2.5	0.890
0.4	0.140	3.0	0.932
0.5	0.215	4.0	0.975
		∞	1.000

Source: Sinha and Kuszta (1983).

Figure 3.46
Step response data in
Table 3.1



Solution. Table 3.1 shows and Fig. 3.46 implies that the final value of the data is $y(\infty) = 1$. We know that A is negative because $y(\infty)$ is greater than $y(t)$. Therefore, the first step in the process is to plot $\log_{10}[y(\infty) - y]$, which is shown in Fig. 3.47. From the line (fitted by eye) the values are

$$\log_{10} |A| = 0.125,$$

$$0.4343\alpha = \frac{1.602 - 1.167}{\Delta t} = \frac{0.435}{1} \Rightarrow \alpha \cong 1.$$

Figure 3.47
 $\log_{10}[y(\infty) - y]$ versus t

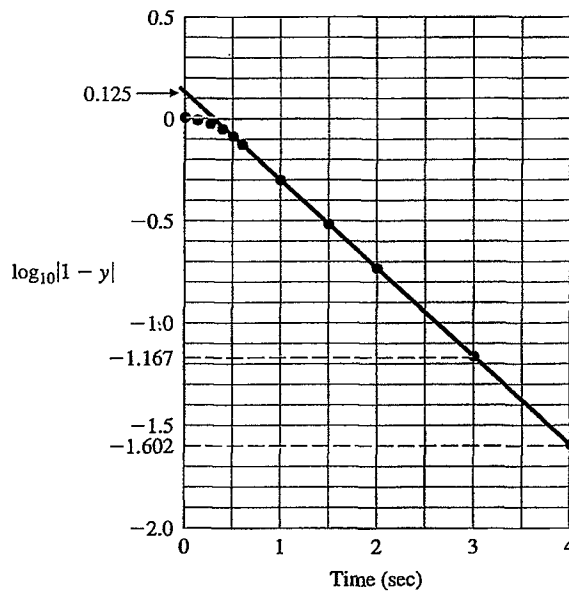
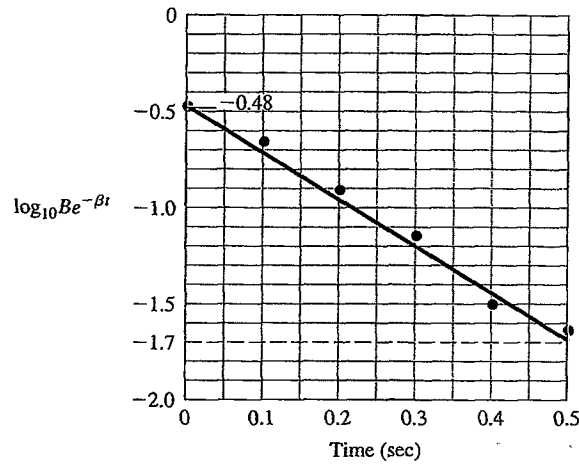


Figure 3.48
 $\log_{10}[y - (1 + Ae^{-\alpha t})]$
 versus t



Thus

$$A = -1.33,$$

$$\alpha = 1.0.$$

If we now subtract $1 + Ae^{\alpha t}$ from the data and plot the log of the result, we find the plot of Fig. 3.48. Here we estimate

$$\log_{10} B = -0.48,$$

$$0.4343\beta = \frac{-0.48 - (-1.7)}{0.5} = 2.5,$$

$$\beta \cong 5.8,$$

$$B = 0.33.$$

Combining these results, we arrive at the y estimate

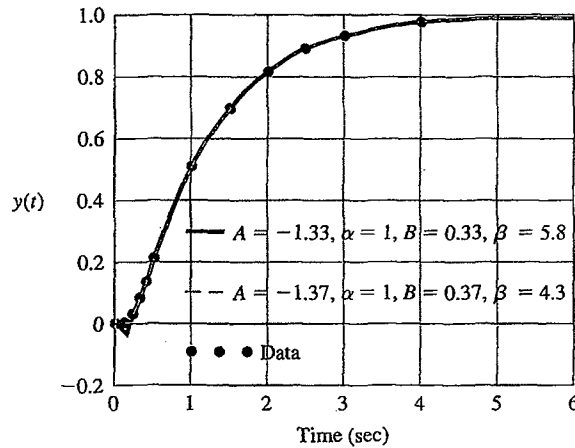
$$\hat{y}(t) \cong 1 - 1.33e^{-t} + 0.33e^{-5.8t}. \quad (3.85)$$

Equation (3.85) is plotted as the colored line in Fig. 3.49 and shows a reasonable fit to the data, although some error is noticeable near $t = 0$.

From $\hat{y}(t)$ we compute

$$\begin{aligned} \hat{Y}(s) &= \frac{1}{s} - \frac{1.33}{s+1} + \frac{0.33}{s+5.8} \\ &= \frac{(s+1)(s+5.8) - 1.33s(s+5.8) + 0.33s(s+1)}{s(s+1)(s+5.8)} \\ &= \frac{-0.58s + 5.8}{s(s+1)(s+5.8)} \end{aligned}$$

Figure 3.49
Model fits to the
experimental data



The resulting transfer function is

$$G(s) = \frac{-0.58(s - 10)}{(s + 1)(s + 5.8)}$$

Notice that this method has given us a system with a zero in the RHP, even though the data showed no values of y that were negative. Very small differences in the estimated value for A , all of which approximately fit the data, can cause values of β to range from 4 to 6. This illustrates the sensitivity of pole locations to the quality of the data and emphasizes the need for a good signal-to-noise ratio.

By using a computer to perform the plotting, we are better able to iterate the four parameters to achieve the best overall fit. The data presentation in Figs. 3.47 and 3.48 can be obtained directly by using a semilog plot. This eliminates having to calculate \log_{10} and the exponential expression to find the values of the parameters. The equations of the lines to be fit to the data are $y(t) = Ae^{\alpha t}$ and $y(t) = Be^{\beta t}$, which are straight lines on a semilog plot. The parameters A and α , or B and β , are iteratively selected so that the straight line comes as close as possible to passing through the data. This process produces the improved fit shown by the dashed black line in Fig. 3.49. The revised parameters, $A = -1.37$, $B = 0.37$, and $\beta = 4.3$ result in the transfer function

$$G(s) = \frac{-0.22s + 4.3}{(s + 1)(s + 4.3)}$$

The RHP zero is still present, but it is now located at $s \cong +20$ and has no noticeable effect on the time response.

This set of data was fitted quite well by a second-order model. In many cases a higher-order model is required to explain the data and the modes may not be as well-separated.

If the transient response has oscillatory modes, then these can sometimes be estimated by comparing them with the standard plots of Fig. 3.23. The period will give the frequency ω_d , and the decay from one period to the next will afford

an estimate of the damping ratio. If the response has a mixture of modes not well separated in frequency, then more sophisticated methods need to be used. One such is **least-squares identification**, in which a numerical optimization routine selects the best combination of parameters so as to minimize the fit error. The fit error is defined to be a scalar **cost function**

$$J = \sum_i (y_{\text{data}} - y_{\text{model}})^2, \quad i = 1, 2, 3, \dots \text{ for each data point,}$$

so that fit errors at all data points are taken into account in determining the best value for the parameters.

3.8.2 Models from Other Data

As mentioned early in Section 3.8, we can also generate a model using frequency-response data, which are obtained by exciting the system with a set of sinusoids and plotting $H(j\omega)$. In Chapter 6 we will show how such plots can be used directly for design. Alternatively, we can use the frequency response to estimate the poles and zeros of a transfer function using straight-line asymptotes on a logarithmic plot.

The construction of dynamic models from normal stochastic operating records or from the response to a PRBS can be based either on the concept of cross-correlation or on the least-squares fit of a discrete equivalent model, both topics in the field of **system identification**. They require substantial presentation and background that are beyond the scope of this text. An introduction to system identification can be found in Chapter 8 of Franklin et al. (1998), and a comprehensive treatment is given in Ljüing (1999). Based largely on the work of Professor Ljüing, the MATLAB Toolbox on Identification provides substantial software to perform system identification and to verify the quality of the proposed models.

SUMMARY

- The Laplace transform is the primary tool used to determine the behavior of linear systems. The Laplace transform of a time function $f(t)$ is given by

$$\mathcal{L}[f(t)] = F(s) = \int_{0^-}^{\infty} f(t)e^{-st} dt. \quad (3.86)$$

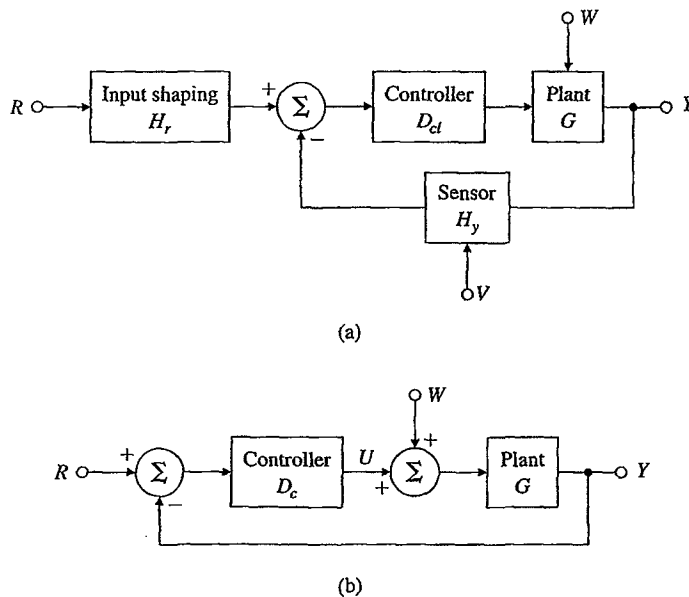
- This relationship leads to the key property of Laplace transforms, namely,

$$\mathcal{L}[\dot{f}(t)] = sF(s) - f(0^-) \quad (3.87)$$

- This property allows us to find the transfer function of a linear ODE. Given the transfer function $G(s)$ of a system and the input $u(t)$, with transform $U(s)$, the system output transform is

$$Y(s) = G(s)U(s).$$

Figure 4.2
Feedback control system



4.1 A Case Study of Speed Control

It is very important to maintain constant speed of the spindle in high-performance hard-disk drives used for computer data storage and, in fact, speed control with an electric motor is a generic problem calling for control. As derived in Chapter 2, Eqs. (4.1) and (4.2) below describe the dynamics of a DC motor with negligible armature inductance and including a term, $T_\ell(t)$, for load-torque disturbance.

$$J_m \ddot{\theta}_m + b \dot{\theta}_m = K_t i_a + \tau_\ell, \tag{4.1}$$

$$K_e \dot{\theta}_m + R_a i_a = v_a, \tag{4.2}$$

If we take the Laplace transform of Eqs. (4.1) and (4.2) and let the velocity $\dot{\theta}_m = \omega_m$ with transform $\Omega_m(s) \triangleq s\Theta(s)$ and the transform of the load torque be $T_\ell(s)$, we obtain the transformed equations of speed control as

$$s J_m \Omega_m(s) + b \Omega_m(s) = K_t I_a(s) + \tau_\ell(s), \tag{4.3}$$

$$K_e \Omega_m(s) + R_a I_a(s) = V_a(s). \tag{4.4}$$

After solving for I_a from Eq. (4.4) and substituting it into Eq. (4.3), the equation for motor speed becomes

$$(J_m R_a s + b R_a + K_t K_e) \Omega_m(s) = K_t V_a(s) + R_a T_\ell(s), \tag{4.5}$$

or

$$\left(\frac{J_m R_a}{b R_a + K_t K_e} s + 1 \right) \Omega_m(s) = \frac{K_t}{b R_a + K_t K_e} V_a(s) + \frac{R_a}{(b R_a + K_t K_e)} T_\ell(s). \quad (4.6)$$

We can simplify the coefficients in this equation by defining new parameters as

$$(\tau s + 1) \Omega_m(s) = A V_a(s) + B T_\ell(s), \quad (4.7)$$

where the time constant and gains are given by

$$\begin{aligned} \tau &= \frac{J_m R_a}{b R_a + K_t K_e} && \text{(time constant, sec),} \\ A &= \frac{K_t}{b R_a + K_t K_e} && \text{(voltage gain, rad/volt-sec),} \\ B &= \frac{R_a}{b R_a + K_t K_e} && \text{(torque gain, rad/N·m-sec).} \end{aligned} \quad (4.8)$$

We may also rewrite Eq. (4.7) in transfer-function form:

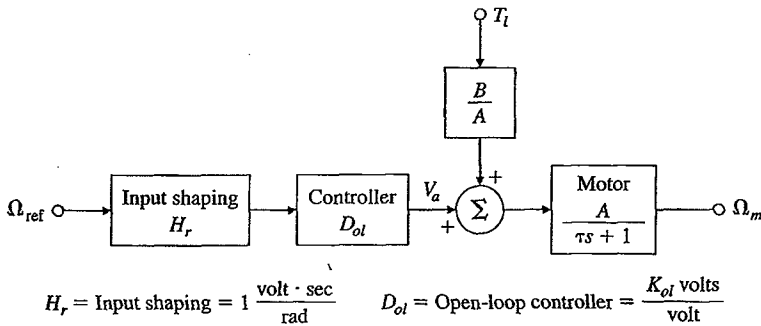
$$\Omega_m(s) = \frac{A}{\tau s + 1} V_a(s) + \frac{B}{\tau s + 1} T_\ell(s). \quad (4.9)$$

If the inputs are constants, say $v_a = a$ and $\tau_\ell = b$, then the steady-state solution of these equations is¹

$$\omega_{ss} = Aa + Bb. \quad (4.10)$$

The transfer functions from Eq. (4.9) are shown in Fig. 4.3 for open-loop control with a controller having transfer function $D_{ol}(s)$. For this configuration we

Figure 4.3
Open-loop speed-control system



¹ From the final value theorem presented in Section 3.1.6, if $Y(s) = G(s)U(s)$ and $U(s) = 1/s$, the final value of $y(t)$ is $y_{ss} = G(0)$. Thus we just have to set $s = 0$ in a *stable* transfer function to find the final or steady-state value to a unit step input.

will refer to the transfer function from Ω_{ref} to Ω_m as the **open-loop transfer function**,

$$T_{ol}(s) = H_r D_{ol} \frac{A}{\tau s + 1}.$$

Modeling speed control by a block diagram

In contrast, the block diagram in Fig. 4.4 shows a closed-loop system which requires an output sensor. In this case the sensor is a tachometer, which is usually a small permanent-magnet DC machine which produces a voltage proportional to the shaft speed ($= \omega_m$). In this case, the **closed-loop transfer function** is

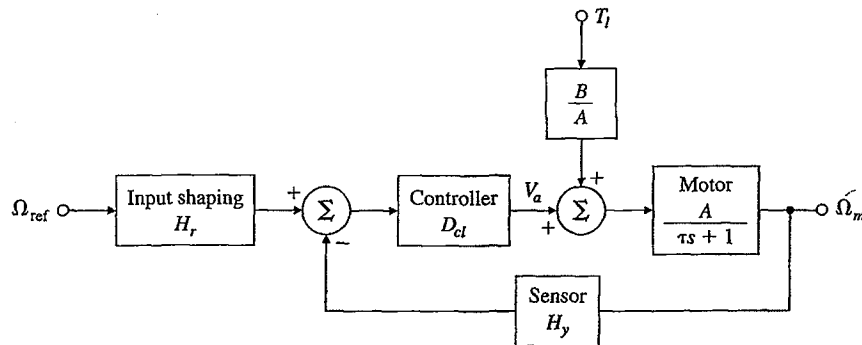
$$T_{cl}(s) = \frac{\Omega_m}{\Omega_{ref}} = H_r \frac{D_{cl} \frac{A}{\tau s + 1}}{1 + D_{cl} H_y \frac{A}{\tau s + 1}}.$$

We will often refer to the transfer function of the elements around a closed loop as the **loop gain**, which in this case is

$$D_{cl} H_y \frac{A}{\tau s + 1}.$$

In these figures and in the analysis below, the gain or scale factor of the sensor is taken to be $H_y = 1$ volt·sec/rad. Techniques for treating more practical values will be described shortly. The input-shaping filter, H_r , is a units conversion device also taken here to have gain 1 volt·sec/rad. The controllers in the figures are taken to be electronic circuits with transfer functions having dimensions *volts/volt*.

Figure 4.4
Feedback speed control system



$$H_r = \text{Input shaping} = 1 \frac{\text{volt} \cdot \text{sec}}{\text{rad}}$$

$$H_y = \text{Sensor} = 1 \frac{\text{volt} \cdot \text{sec}}{\text{rad}}$$

$$D_{cl} = \text{Closed-loop controller} = \frac{K_{cl} \text{ volts}}{\text{volt}}$$

4.1.1 Disturbance Rejection

Now let us compare open-loop control with feedback control with respect to how well each system maintains a constant steady-state reference speed in the face of load or disturbance torques. For the open-loop controller (Fig. 4.3), the amplifier voltage is taken to be

$$v_a = D_{ol}\omega_{ref} = K_{ol}\omega_{ref}. \quad (4.11)$$

Open-loop speed control

The constant gain $D_{ol}(s) = K_{ol}$ is determined so that in steady-state the output speed, $\omega_m = \omega_{ss}$ equals the constant reference speed ω_{ref} when the load torque τ_ℓ is zero. From Eq. (4.10) this value is found to be

$$K_{ol} = \frac{1}{A}.$$

In the steady state ($s = 0$) and with no load torque ($\tau_\ell = 0$), Eq. (4.10) gives the output speed as

$$\omega_m = \omega_{ss} = Av_a = A\frac{1}{A}\omega_{ref} = \omega_{ref}.$$

With the feedback controller (Fig. 4.4) the controller transfer function is taken to be $D_{cl} = K_{cl}$ so that

$$v_a = K_{cl}(\omega_{ref} - \omega_m). \quad (4.12)$$

Closed-loop speed control

Combining the motor model given by Eq. (4.7) with the transform of the feedback controller equation described by Eq. (4.12), the closed-loop system equations are

$$[(\tau s + 1) + AK_{cl}]\Omega_m(s) = AK_{cl}\Omega_{ref}(s) + BT_\ell(s). \quad (4.13)$$

For no load torque ($\tau_\ell = 0$) and in the steady state ($s = 0$) the speed is given by

$$\omega_{ss} = \frac{AK_{cl}}{1 + AK_{cl}}\omega_{ref}. \quad (4.14)$$

If the gain K_{cl} is selected large so that $AK_{cl} \gg 1$, then, although some error remains, $\omega_{ss} \cong \omega_{ref}$.

With these controls, both approaches provide the desired result of good tracking, $\omega_{ss} \cong \omega_{ref}$, although on the face of it the open-loop seems to be more accurate. However, the gains have been set with zero load torque; now let us compute what effect a nonzero torque will have on the steady-state speed in the two cases. Equation (4.10) shows that in the open-loop system the steady-state speed is

$$\omega_{ss} = AK_{ol}\omega_{ref} + B\tau_\ell. \quad (4.15)$$

Here, with $K_{ol} = 1/A$, the speed with load torque for the open-loop case is

$$\omega_{ss} = \omega_{ref} + B\tau_\ell,$$

and the variation in speed caused by the load torque for the open-loop case is $\omega_{ss} - \omega_{ref} = \delta\omega$, where

$$\delta\omega = B\tau_\ell.$$

So we see that the speed error is proportional to the disturbing load torque, and the control designer has no influence over the parameter B which determines the size of the error.

For the feedback case [Eq. (4.13)], the steady-state speed for constant values of ω_m and τ_ℓ becomes

$$\omega_{ss} = \frac{AK_{cl}}{1 + AK_{cl}}\omega_{ref} + \frac{B}{1 + AK_{cl}}\tau_\ell. \quad (4.16)$$

If it is possible for the designer to pick values of K_{cl} so that $AK_{cl} \gg 1$ and $AK_{cl} \gg B$, a very small error will result with or without a disturbing load torque. In fact, as long as $AK_{cl} > 0$, comparison of Eqs. (4.15) and (4.16) reveals that the feedback-controller speed errors due to a load torque will be less than the open-loop errors by exactly $1 + AK_{ol}$. This is our first result on the advantages of feedback:

Advantage of feedback

System errors to constant disturbances can be made smaller with feedback than they are in open-loop systems by a factor of $1 + AK_{cl}$ where AK_{cl} is the loop gain at $s = 0$.

EXAMPLE 4.1

Steady-State Error to Disturbance, Open-Loop Case

For a certain small servomotor and load, it is determined that $\tau = \frac{1}{60}$ sec, $A = 10$ rad/V-sec, and $B = 50$ rad/N·m-sec. The reference speed is $\omega_{ref} = 100$ rad/sec. Find the steady-state open-loop armature voltage needed to get this speed with zero load torque. What is the steady-state speed error with this load voltage if the load torque is $\tau_\ell = -0.1$ N·m?

Solution. For the open-loop system with an armature voltage input v_a and a load torque τ_ℓ , the steady-state output is given by Eq. (4.10) with $K_{ol} = \frac{1}{10}$ and thus $v_a = \omega_{ref}/10$.

$$\omega_{ss} = 10v_a + 50\tau_\ell.$$

Assuming $\tau_\ell = 0$, the output is

$$\omega_{ss} = 100 \text{ rad/sec} = \omega_{ref}.$$

Now suppose a constant load torque of $\tau_\ell = -0.1$ N·m is applied. The steady-state value of the output with the disturbance is

$$\omega_{ss} = 100 + 50(-0.1) = 95 \text{ rad/sec}.$$

Thus the error is 5 rad/sec or 5%.

Now consider the same example using proportional feedback of the output.

EXAMPLE 4.2

Steady-State Error to Disturbance, Closed-Loop Case

Consider the feedback control structure shown in Fig. 4.4, where tachometer feedback and proportional control are employed. You are to improve the ability of the system to reject steady-state disturbances by a factor of at least 100 compared with the open-loop system. The parameter values are the same as those given in Example 4.1.

Solution. From Eq. (4.16) we see that to get an improvement factor of 100 we require $1 + AK_{cl} = 100$, from which $K_{cl} = 9.9$. With this value of K_{cl} the steady-state value of the tracking output assuming that the load torque $T_\ell = 0$, is²

$$\begin{aligned}\omega_{ss} &= \frac{AK_{cl}}{1 + AK_{cl}}\omega_{ref} = \frac{99}{100}100 \\ &= 99 \text{ rad/sec.}\end{aligned}$$

The steady-state output due to the input of a constant disturbance torque is [Eq. 4.16)]

$$\omega_{ss} = \frac{B \tau_\ell}{1 + K_{cl}A} = \frac{50 \tau_\ell}{1 + 10K_{cl}}$$

The output with both reference and disturbance is, $\omega_{ss} = 99 - 0.05 = 98.95$ rad/sec which is a change of only 0.051% from the speed without disturbance. Thus the feedback has improved the disturbance rejection by a factor of about 100 over the open-loop system.

4.1.2 Sensitivity of System Gain to Parameter Changes

As another comparison of open- and closed-loop control, we consider the sensitivity of the steady-state gain or transfer function at $s = 0$ to parameter changes. The change might come about because of external effects such as temperature changes or might simply be due to an error in the value of the parameter from the start. Suppose that the motor gain in operation differs from its original design value of A to be $A + \delta A$. In the open-loop case the nominal gain is³ $T_{ol} = K_{ol}A$, and the new overall system gain would be

$$T_{ol} + \delta T_{ol} = K_{ol}(A + \delta A) = K_{ol}A + K_{ol}\delta A = T_{ol} + K_{ol}\delta A,$$

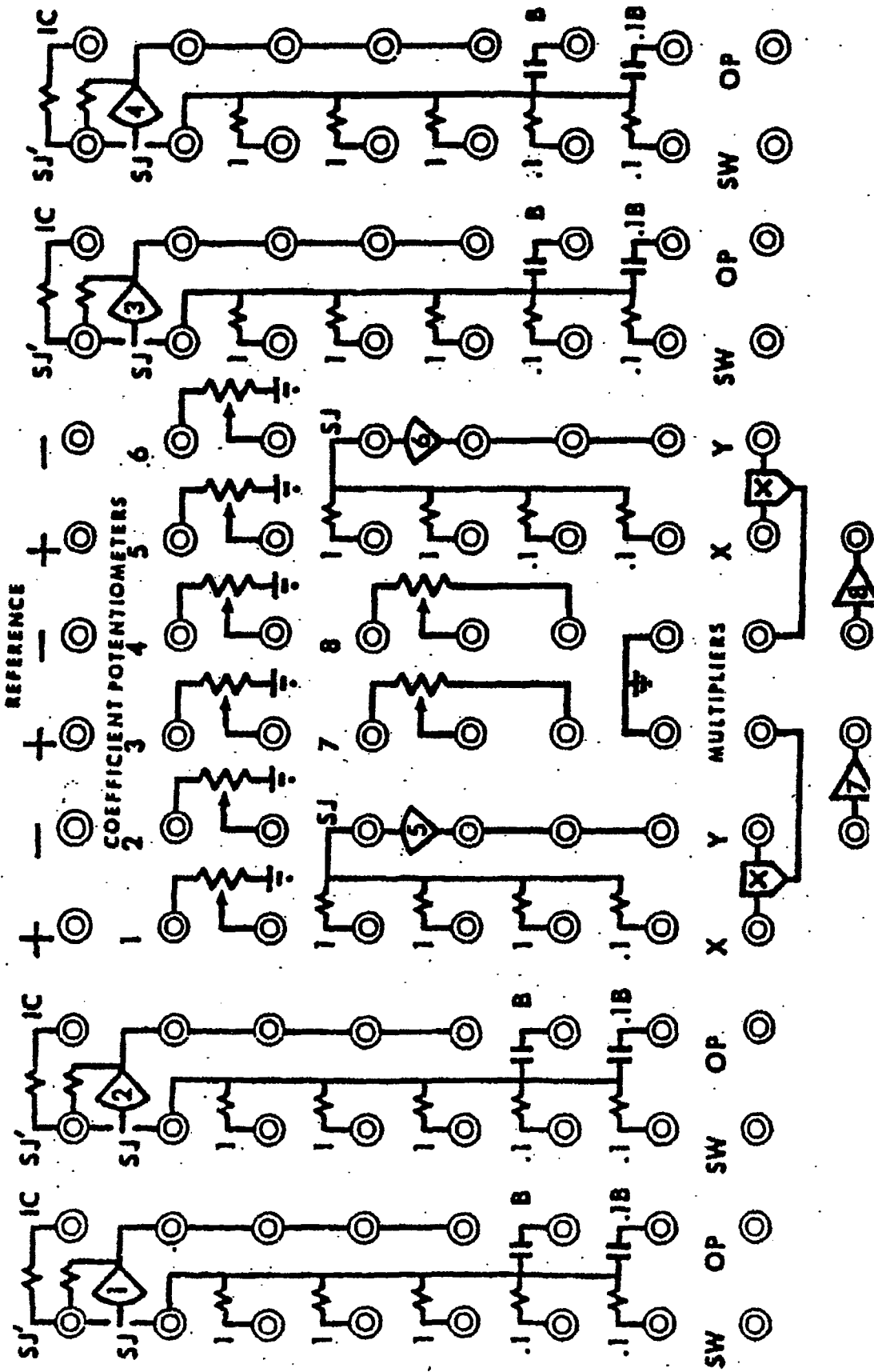
where T_{ol} is the open-loop transfer function at $s = 0$. Using the fact that $K_{ol} = 1/A$, the normalized error in the gain is

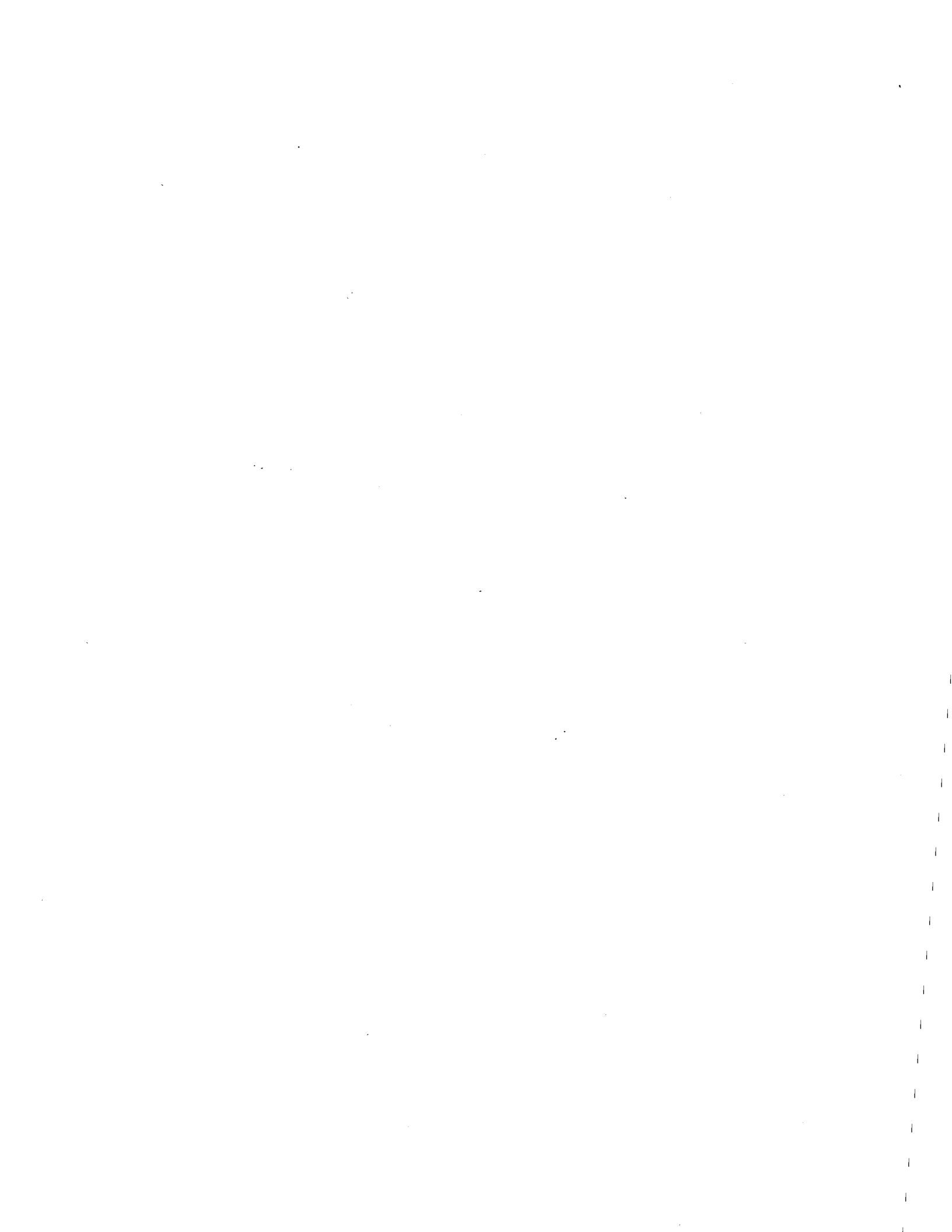
$$\frac{\delta A}{A}$$

² In process control it is common practice to note that if the set point were *reset* to $\omega'_{ref} = 100/99\omega_{ref}$, then the output will be equal to the reference value.

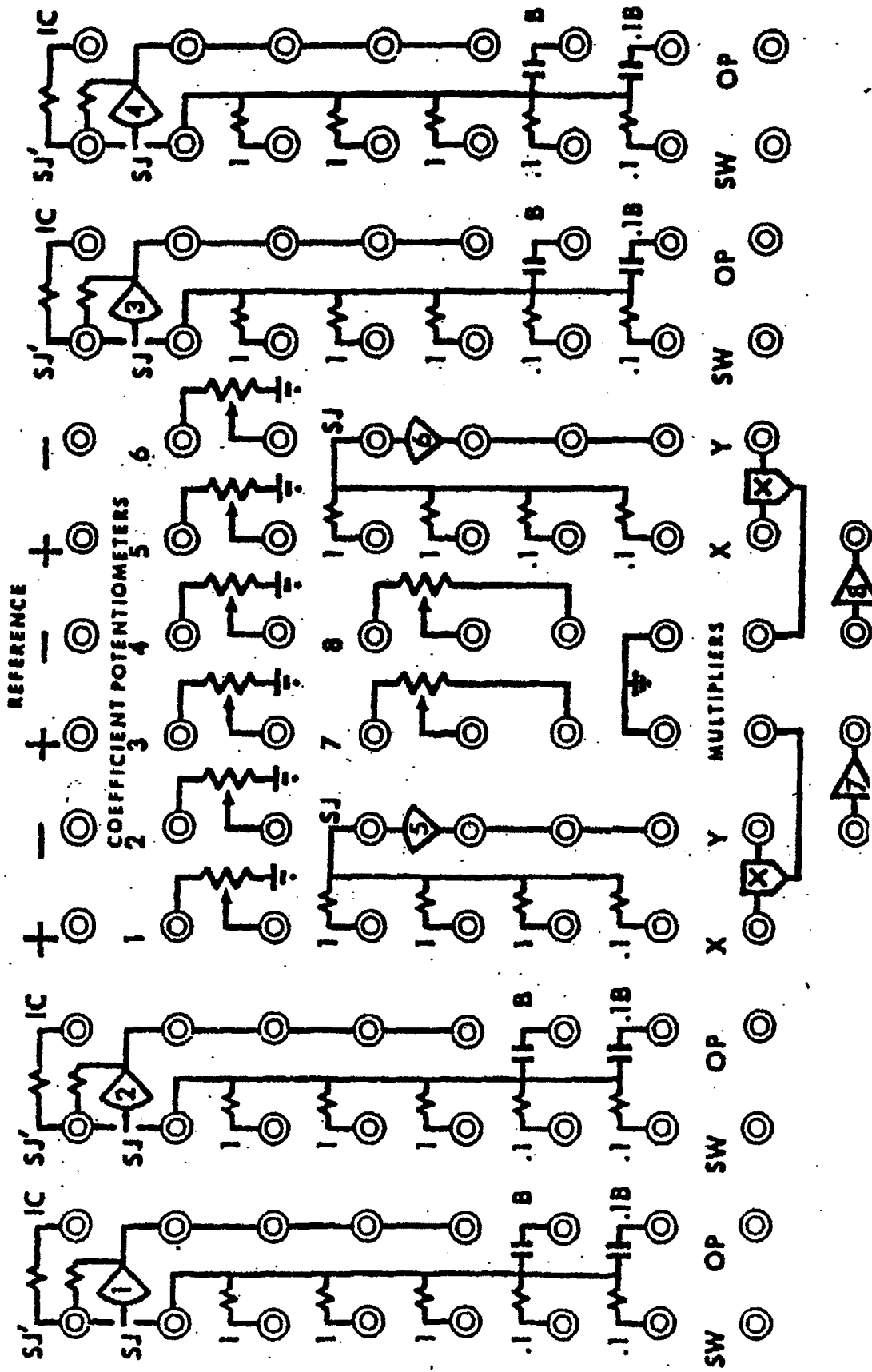
³ We use T_{ol} and T_{cl} for the open-loop and closed-loop transfer functions, respectively. These are not to be confused with the transform of the disturbance torque, T_ℓ used earlier.

3. GP-6 PATCH PANEL





3. GP-6 PATCH PANEL



3. GP-6 PATCH PANEL

