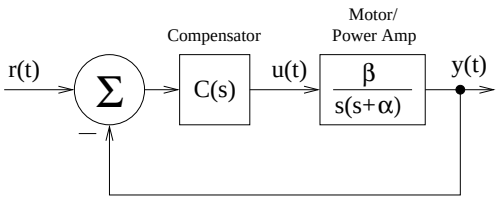
ELE458/459 **Homework 1** **Spring 2026**

1 Introduction

In this course you will design and simulate control systems using Matlab and Simulink. Simulink will also be used to control hardware systems in the laboratory. This handout provides a brief introduction to the way we will use Matlab and Simulink this semester. Suppose we want to simulate the response of the following system to a step input of height H.



Some important variables from the above figure are defined as follows:

$$r(t) = \text{step signal of height } H$$

$$u(t) = \text{plant input signal}$$

$$y(t) = \text{plant output signal}$$

This homework starts with the design of two compensators, $C_1(s)$ and $C_2(s)$. The first compensator is just a gain, $C_1(s) = K$. That can be simulated as the compensator shown in the figure above by setting $c_1 = c_2 = 1$. The second compensator, $C_2(s)$ is a phase-lead compensator $C_2(s) = K(s + \alpha)/(s + \alpha')$. Finally, a third compensator consisting of a phase-lead compensator and a prefilter will be designed.

2 Design Methodology

There are two types of information that are used in the formulation of a design problem. On one hand, there are performance specifications which state in certain measurable terms what we want the hardware to do. On the other hand, there are certain constraints that limit the performance of the system. The design problem is to create a system that will meet the specifications without violating any of the constraints.

Specifications for a position control problem include percent overshoot of the step response, settling time, and steady-state error. These three numbers provide a quantitative way of assessing the performance of the system. The main constraint in this project is that the **input to the plant should be between -5 and +5 volts.** The motor power amplifier has a 5 volt saturation.

Once the performance specifications and constraints for a particular design problem are understood, the procedure used to find an acceptable system is the following *iterative design cycle*:

1. Design a control system.

2. Perform a simulation study and check if the constraints are satisfied and if performance specifications are achieved. Repeat step 1 if necessary.

3. Implement the control system in the lab and check its performance. Repeat steps 1 and 2 if necessary.

3 Procedure for $C_1(s)$ and $C_2(s)$

Create a directory (folder) for this assignment. Start Matlab and then change to the desired directory in Matlab. The next step is to write a Matlab program (m-file) to define the variables and design the control system.

The requirements for the first compensator are as follows: $C_1(s) = K$ where the gain K is chosen to give a critically damped closed-loop system with the smallest settling time such that the plant input does not exceed a magnitude of 5 volts.

The requirements of the second compensator are as follows: $C_2(s) = K(s+\alpha)/(s+\alpha')$ where α comes from the plant transfer function and K and $alpha'$ are chosen to give a critically damped closed-loop system with the smallest settling time such that the plant input does not exceed a magnitude of 5 volts.

Suppose you wanted this file to be called `hw1_design.m`. At the Matlab prompt type `edit hw1_design`. Matlab will add the `.m` extension. In this case the m-file should contain the following lines: (start with the values for $C_1(s)$).

```
H=60;
beta=2400;
alpha=22;
K= ...    [use the gain value for C1(s) or C2(s)]
c1= ...  [use 1 for C1(s) or alpha for C2(s)]
c2= ...  [use 1 for C1(s) or alpha_prime for C2(s)]
```

Click on the green *Run* arrow to execute this Matlab program. You will then see these variables in the Matlab workspace.

The second step in this process is to create a Simulink model of the control system. The following steps show you how to do that.

1. Type *simulink* at the Matlab prompt. On the next screen that pops up, click on the "Blank Model" tab (and wait a while while Simulink gets started!).

2. In the window that pops up (untitled) click on the *Modeling* tab, then click *Model Settings*, then click *Data Import/Export* and uncheck the *Single Simulation Output* box. Then click *OK*. **The following is needed only for this Homework and for Lab 1.** Click on *Solver*. In the *Solver* section change the type from Variable-step to Fixed-step. In the Fixed-step box put this number: 1e-4.

   Click on the *Simulation* tab, then click on the *Library Browser* icon in the upper left (red, white, and blue icon). If the arrow pointing to *Simulink* is pointing down you will see a submenu of items. It the arrow is pointing to the word *Simulink*, click on it to show the submenu.

   In the submenu, double-click on *Continuous* and then click, drag, and release a *Transfer Fcn* block to the Simulink workspace (the window labeled *Untitled*). Then click and drag a second transfer function to the workspace. Click on the black arrow next to *Continuous* to close this block menu. We will finish the Simulink model in step 5 and following.

3. When you give the untitled model a name and save it, it will be saved as *name.slx*. The next time you start Matlab you can double click on *name.slx*, which will start Simulink and open this model. Go ahead and save this model as `hw1_sim`. **Note that the Simulink model and the m-file may not have the identical name.**

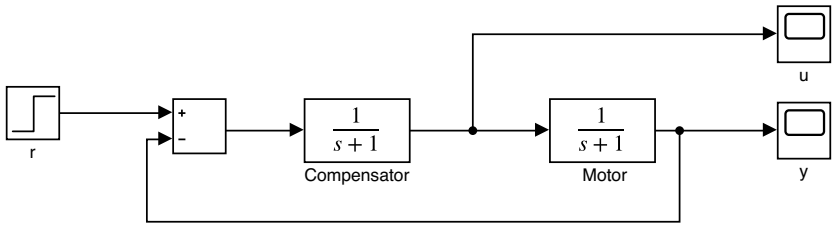
4. The white box at the top of the Simulink model labeled *Stop Time* has the number 10 in it. This default value runs the simulation for 10 seconds. For this handout we will change this number to 1 (the simulation will run for one second).

5. In the Simulink Library Browser, in the left-hand column under Simulink click on *Math Operations*. Click and drag an *Add* block to the workspace. Double click on this block and change the second sign to be negative. Close this submenu by clicking on the black arrow next to *Math Operations*.

6. In the Simulink Library Browser click on *Sources*. Click and drag the *Step* block to the workspace; Click on the word *Sinks* and drag two *Scope* blocks to the Simulink workspace.

7. Connect the Simulink blocks to create the block diagram shown on the first page of this handout. (See also the sample Simulink model under item 8 below.) Connect one scope to the output of the second transfer function block, and the other scope to the input of this transfer function.

8. Click on the transfer function blocks and then click on the blue names. Change them to *Compensator* and *Motor*, respectively. Click on the names under the scopes and change them to `u` and `y`. At this point the model should look as follows:



9. Double click on the *Scope* blocks and click on the *Configuration Properties* icon (this is the gear-shaped icon located at the far left). Click on the *Logging* tab. Check the box that says *Log data to workspace*. Type in a variable name (I suggest you use u for the plant input and y for the plant output). Change the format to *Array*. This will save the variable as an array (matrix) with two columns. **The first column contains time values and the second column contains the corresponding variable values at those times.**

10. In your Simulink workspace, double click on the step block; change the *Step Time* (starting time) to be zero and the *Final Value* to be H.

    Double click on the Compensator block; change the numerator to be `K*[1 c1]`, which denotes the polynomial $K(s+c_1)$, and the denominator to be `[1 c2]`, which denotes $(s+c_2)$. The variables `K`, `c1`, and `c2` must exist in the Matlab workspace (which they do if you ran the m-file). Their numerical values will be used in this Simulink simulation.

    Double click on the Plant block; change the numerator to be `beta` and the denominator to be `[1 alpha 0]`, which denotes the polynomial $s^2 + \alpha s + 0 = s(s + \alpha)$.

11. In the Simulink window, click on the green **Run** arrow to run the simulation. You will see the results on the scopes. (Double click a scope block to make the scope output visible.) **Note that you may also run (or re-run) a simulation by clicking on the green arrow at the top of any scope block.**

    To see the results in Matlab, you may use the following Matlab script (type "edit graphs" to create this m-file). You can copy and paste the commands given below into Matlab, but **you will have to correct the opening single quotes in the Matlab file**. Change the input and output units to their actual units.

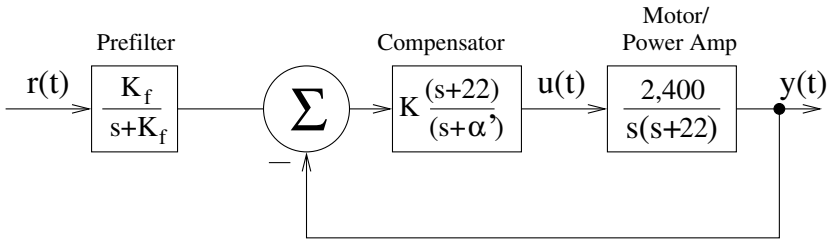
```
subplot(211)
plot(y(:,1),y(:,2));grid
title('Plant Output')
xlabel('Time (sec.)')
ylabel('Output Units')
subplot(212)
plot(u(:,1),u(:,2));grid
title('Plant Input')
xlabel('Time (sec.)')
ylabel('Input Units')
```

12. It is easy to determine the exact settling time from the Matlab plot of the plant output. The final value of the plant output is H. The one-percent settling time is the time at which the output gets and stays within one percent of this value. Because there is no overshoot, the settling time for this problem is the time at which the output crosses 0.99*H. There are three ways to find the settling time (I recommend the third way) (1) you may use the data cursor on the Matlab figure; (2) you can type the variable name for the plant input at the Matlab prompt, which will result in two columns of numbers, time and voltage. Find the time at which the output crosses 0.99*H; (3) finally, you could use the following Matlab commands. **These commands can be added to the `graphs.m` program, so that every time new data are plotted the corresponding settling time and max plant and input values are printed in the Matlab window.** These command should **not** be added to the "design" m-file from page 1.

```
ind=max(find(y(:,2)<=0.99*H));
t1=y(ind,1);
y1=y(ind,2);
t2=y(ind+1,1);
y2=y(ind+1,2);
if abs(y1-0.99*H)<H/1e6
    Ts=t1; %if y1 is close enough to 0.99*H
else
    m=(y2-y1)/(t2-t1); % use linear interpolation
    Ts=t2-(y2-0.99*H)/m
end
max_output=max(y(:,2))
max_input=max(u(:,2))
```

4 Procedure for Third Control System: Phase-Lead Compensator with Prefilter

The third control system uses a prefilter filter $P(s) = K_f/(s + K_f)$ in addition to the phase-lead compensator, as shown in the following diagram:



There are three different parameters to vary in this design: K, α', and K_f. **HINTS:** Due to the cancellation between the compensator zero and the plant pole at -22, the overall system in the figure above is third order. Two of the poles of the closed-loop system are the poles of the feedback loop, and the third pole is the prefilter pole at $-K_f$. This control system is designed by choosing values for K, α', and K_f. We know that choosing $K = (\alpha'/2)^2/2,400$ results in the feedback loop having a double pole at $-\alpha'/2$. In previous designs, a double pole was required to get a critically damped response without overshoot. For the present design, however, there is a third, real pole. It turns out that it is possible for a 3rd-order system to have complex-conjugate poles and have a step response without overshoot. Such a system can be faster than a system with all real poles.

We know that if the gain K is increased beyond that used to get a double pole, the resulting poles will be complex. Thus, we choose

$$K = \frac{f \cdot (\alpha'/2)^2}{2,400} \tag{1}$$

where f is a "factor" giving a 25-50% increase in the value of K (i.e. $1.25 \leq f \leq 1.5$). For any such choice of f, the real parts of the complex poles will be $-\alpha'/2$. It makes sense to choose the prefilter pole so that it is neither faster nor slower than these complex poles. Thus, K_f should be chosen to be $K_f = \frac{\alpha'}{2}$. Pick a fixed value of f and conduct a search by varying the value of α' and using (1) and (2) to calculate K and K_f. You can start with the value of α' that was used for the second control system. Run the simulation to see the settling time and max plant input for this design. You should see that the max plant input is well below 5 volts. Increase the value of α' until max plant input is just under 5, while checking that the maximum value of the plant output is less than or equal to 60 (no overshoot).