XMUT315 Control Systems Engineering

Note 9b: Applications of Controllers-Compensators

Topic

- Applications of controllers or compensators.
- Examples of applications of Proportional, Derivative, and Integral controllers and their combinations.
- Examples of applications of Lag, Lead, and Lag-lead compensators.
- Practical circuit implementations of controllers or compensators.
- Tuning in of the controllers.

1. Application of Controllers

We will consider the following unity-feedback system. The output of the controller (u), which is equal to the control input to the plant, is calculated in the time domain from the feedback error (e) as follows:

$$u(t) = c(t)e(t)$$

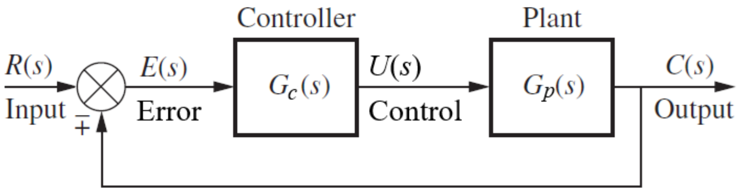


Figure 1: Controller and plant in feedback control system

First, let us look at how the controller works in a closed-loop system using the block diagram shown above.

The variable (e) represents the tracking error, the difference between the desired output (r) and the actual output (c).

This error signal (e) is fed to the controller, and the controller computes this error signal with the parameter(s) of the controller with respect to time.

Depending on the type of controller, these parameters could be K_p for proportional controller, K_i/s for integral controller, $K_d s$ for derivative controller or any of their combinations such as $K_p + K_i/s$ for PI controller, $K_p + K_d s$ for PD controller and $K_p + K_i/s + K_d s$ for PID controller.

The control signal (u) to the plant is equal to the error (e) times the magnitude of the parameters of the controller (G_c). This control signal (u) is fed to the plant (G_p) and the new output (c) is obtained. The new output (c) is then fed back and compared to the reference (r) to find the new error signal (e). The controller takes this new error signal and computes an update of the control input. This process continues while the controller is in effect.

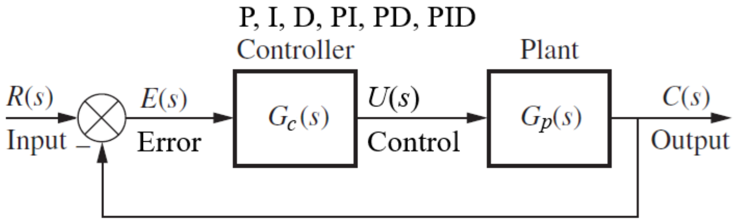


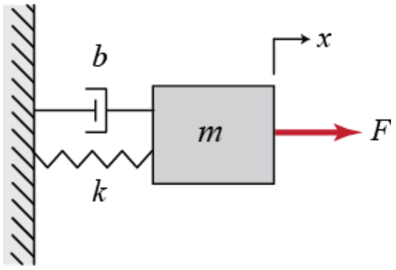
Figure 2: Control system with example controllers

The goal of the examples of controller applications e.g. Examples for Tutorial 1-5 in this section is to show how each of the terms of the controller: K_p, K_i, and K_d, contributes to obtaining the common goals of fast rise time, minimal overshoot, and zero steady-state error.

Example for Tutorial 1 – Analyse System

Suppose we have a simple mass-spring-damper system as shown in the figure below. The governing equation of this system is:

$$m\left(\frac{d^2x}{dx^2}\right) + b\left(\frac{dx}{dt}\right) + kx = F$$



a. Derive the transfer function of the system. [2 marks]

b. If m = 1 kg, b = 10 N s/m, k = 20 N/m, and F = 1 N, determine the transfer function of the system. [2 marks]

c. Simulate the step response of the open-loop system in MATLAB. [4 marks]

d. Describe the result of the simulation in part (c) in terms of DC gain and steady-state error, rise time and settling time. What are the characteristics of the controller needed to fix the problems? [4 marks]

Answer

a. Taking the Laplace transform of the transfer function equation of the system, we get:

$$ms^2X(s) + bsX(s) + kX(s) = F(s)$$

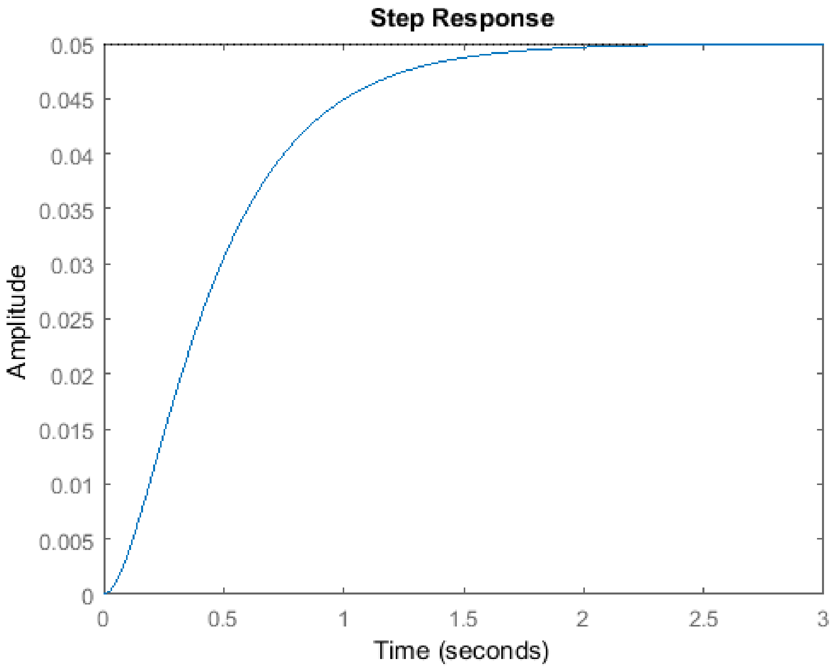
The transfer function equation between the input force and the output displacement then becomes:

$$\frac{X(s)}{F(s)} = \frac{1}{ms^2 + bs + k}$$

b. Let: m = 1 kg, b = 10 N s/m, k = 20 N/m, and F = 1 N. Substituting these values into the transfer function obtained in part (a):

$$\frac{X(s)}{F(s)} = \frac{1}{s^2 + 10s + 20}$$

c. Let's first view the open-loop step response. The following figure shows the step response of the open-loop system.



d. The DC gain of the plant transfer function is 1/20 = 0.05, so 0.05 is the final value of the output to a unit step input. This corresponds to a steady-state error of 0.95, which is quite large. Furthermore, the rise time is about one second, and the settling time is about 1.5 seconds.

   Thus, we need to design a controller that will reduce the rise time, reduce the settling time, and eliminate the steady-state error.

1.1. Proportional Controllers

For a proportional controller, the control signal (u) to the plant is equal to the proportional gain (K_p) times the magnitude of the error. The output of a proportional controller, which is equal to the control input to the plant, is calculated in the time domain from the feedback error as follows:

$$u(t) = K_p e(t)$$

Thus, the transfer function of a proportional controller is found by taking the Laplace transform of system equation:

$$G_c(s) = K_p$$

Where: K_p = proportional gain.

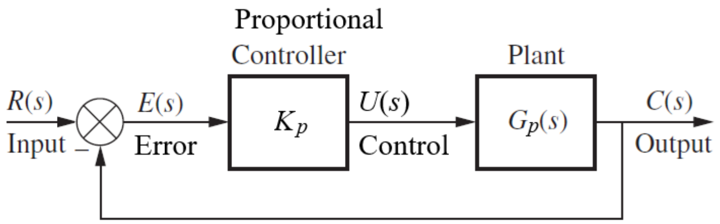


Figure 3: A unity feedback control system with a proportional controller

Increasing the proportional gain (K_p) has the effect of proportionally increasing the control signal for the same level of error. The fact that the controller will "push" harder for a given level of error tends to cause the closed-loop system to react more quickly, but also to overshoot more. Another effect of increasing K_p is that it tends to reduce, but not eliminate, the steady-state error.

When we have the proportional controller, we see that the proportional controller (K_p) reduces the rise time, increases the overshoot, and reduces the steady-state error.

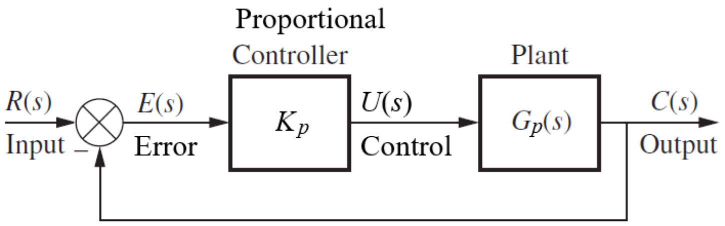
Tutorial for Example 2 – Adding P Controller

For the given simple mass-spring-damper system in Tutorial for Example 1, add a proportional controller in series with the system.

a. Derive the transfer function equation of the system. [4 marks]

b. Using the trial-and-error methods in MATLAB, determine the appropriate value of the parameter of the controller. Then, simulate the transient response of the system. [6 marks]

c. Describe the response of the system in terms of rise time, overshoot, settling time and steady-state error. [2 marks]



Answer

a. The transfer function equation of a proportional controller is:

$$G_c(s) = K_p$$

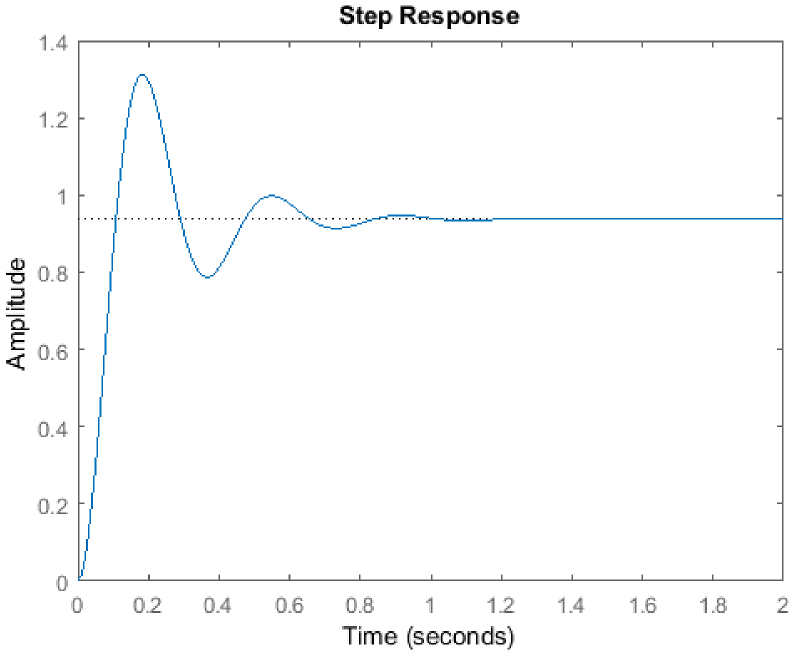
The closed-loop transfer function of our unity-feedback system with a proportional controller is as follow, where $C(s)$ is our output and our reference $R(s)$ is the input:

$$T(s) = \frac{C(s)}{R(s)} = \frac{G_c G_p}{1 + G_c G_p}$$

$$= \frac{K_p\left(\frac{1}{s^2 + 10s + 20}\right)}{1 + K_p\left(\frac{1}{s^2 + 10s + 20}\right)}$$

$$= \frac{K_p}{s^2 + 10s + \left(20 + K_p\right)}$$

b. Let the proportional gain (K_p) equal 300. The following figure shows the step response of the example system with proportional controller.



c. The above plot shows that the proportional controller reduces both the rise time and the steady-state error, increases the overshoot, and decreases the settling time by a small amount.

1.2. Derivative Controllers

For a derivative controller, the control signal (u) to the plant is equal to the derivative gain (K_d) times the derivative of the error. The output of a derivative controller, which is equal to the control input to the plant, is calculated in the time domain from the feedback error as follows:

$$u(t) = K_d \frac{de(t)}{dt}$$

Thus, the transfer function of a derivative controller is found by taking the Laplace transform of system equation:

$$G_c(s) = K_d s$$

Where: K_d = derivative gain.

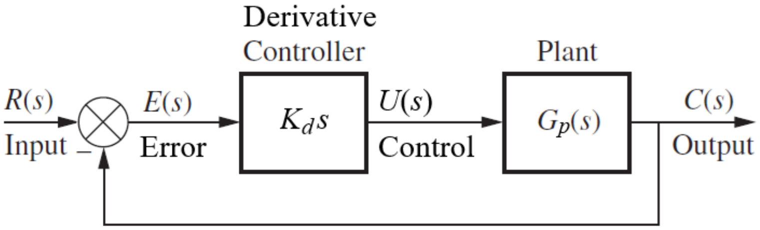


Figure 4: A unity feedback control system with a derivative controller

The addition of a derivative term to the controller (K_d) adds the ability of the controller to "anticipate" error.

With simple proportional control, if K_p is fixed, the only way that the control will increase is if the error increases. With derivative control, the control signal can become large if the error begins sloping upward, even while the magnitude of the error is still relatively small. This anticipation tends to add damping to the system, thereby decreasing overshoot. The addition of a derivative term, however, does not affect the steady-state error.

Now, let's take a look at the characteristics of the PD control. We see that the addition of derivative control (K_d) tends to reduce both the overshoot and the settling time.

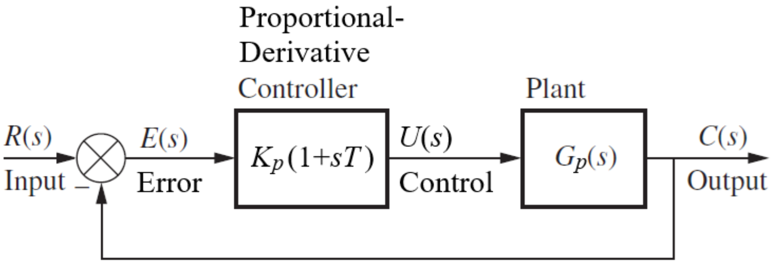


Figure 5: A unity feedback control system with a proportional-derivative controller

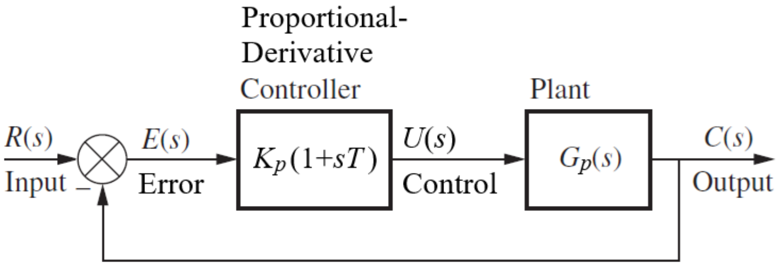
Tutorial for Example 3 – Adding PD Controller

For the given simple mass-spring-damper system in Tutorial for Example 1, add a proportional-derivative controller in series with the system.

a. Derive the transfer function equation of the system. [4 marks]

b. Using the trial-and-error methods in MATLAB, determine the appropriate values of the parameters of the controller. Then, simulate the transient response of the system. [6 marks]

c. Describe the response of the system in terms of rise time, overshoot, settling time and steady-state error. [2 marks]



Answer

a. The transfer function of the PD controller is (note: $K_d = K_p T_d$):

$$G_c(s) = K_d s + K_p$$

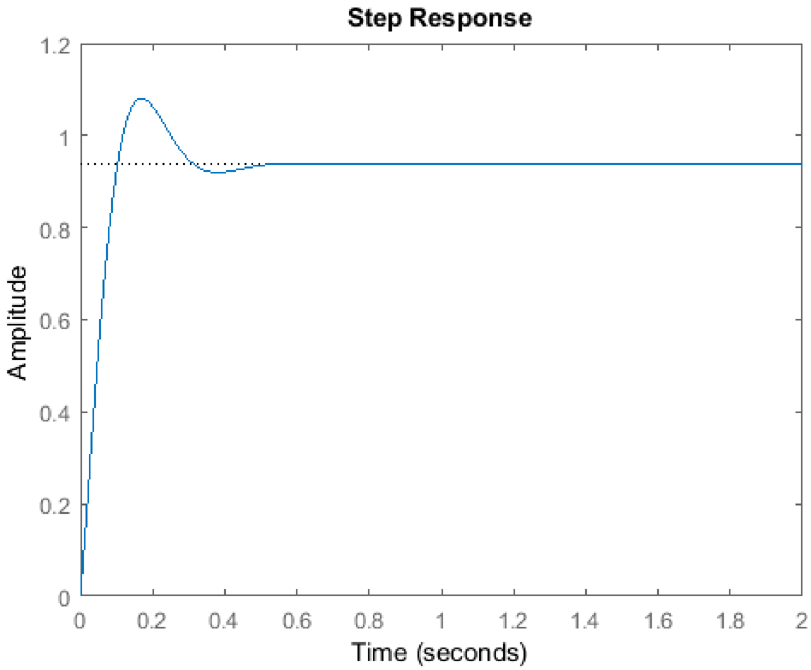
The closed-loop transfer function of the given system with a PD controller is:

$$T(s) = \frac{C(s)}{R(s)} = \frac{G_c(s)G_p(s)}{1 + G_c(s)G_p(s)}$$

$$= \frac{(K_d s + K_p)\left(\frac{1}{s^2 + 10s + 20}\right)}{1 + (K_d s + K_p)\left(\frac{1}{s^2 + 10s + 20}\right)}$$

$$= \frac{K_d s + K_p}{s^2 + (10 + K_d)s + (20 + K_p)}$$

b. Let K_p equal 300 as before and let K_d equal 10. The following figure shows the step response of the example system with PD controller.



c. This plot shows that the addition of the derivative term reduces both the overshoot and the settling time, but it has a negligible effect on the rise time and the steady-state error.

1.3. Integral Controllers

For an integral controller, the control signal (u) to the plant is equal to the integral gain (K_i) times the integral of the error. The output of an integral controller, which is equal to the control input to the plant, is calculated in the time domain from the feedback error as follows:

$$u(t) = K_i \int e(t)dt$$

Thus, the transfer function of an integral controller is found by taking the Laplace transform of system equation:

$$G_c(s) = \frac{K_i}{s}$$

Where: K_i = integral gain.

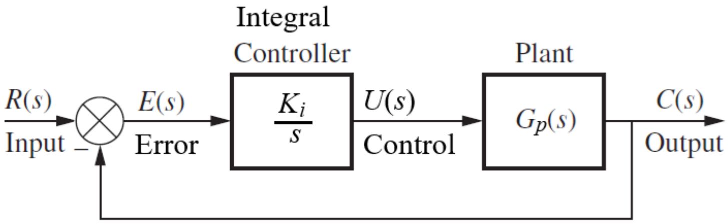


Figure 6: A unity feedback control system with an integral controller

The addition of an integral term to the controller (K_i) tends to help reduce steady-state error. If there is a persistent, steady error, the integrator builds and builds, thereby increasing the control signal and driving the error down.

A drawback of the integral term, however, is that it can make the system more sluggish (and oscillatory) since when the error signal changes sign, it may take a while for the integrator to "unwind."

Let's investigate PI control. We see that the addition of proportional-integral control (K_i) tends to decrease the rise time, increase both the overshoot and the settling time, and reduces the steady-state error.

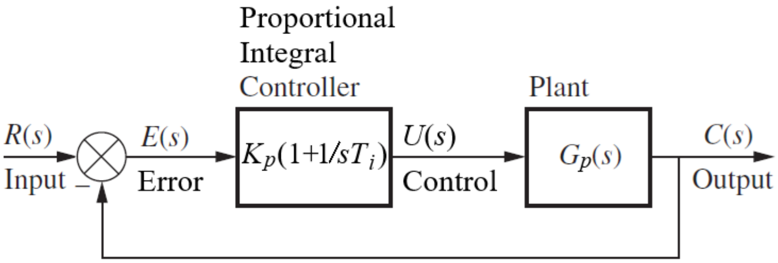


Figure 7: A unity feedback control system with a proportional-integral controller

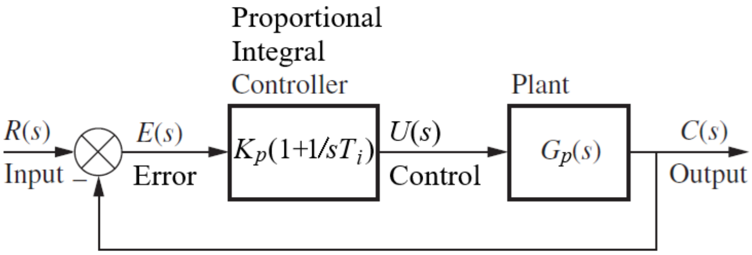
Tutorial for Example 4 – Adding PI Controller

For the given simple mass-spring-damper system in Tutorial for Example 1, add a proportional-integral controller in series with the system.

a. Derive the transfer function equation of the system. [4 marks]

b. Using the trial-and-error methods in MATLAB, determine the appropriate values of the parameters of the controller. Then, simulate the transient response of the system. [6 marks]

c. Describe the response of the system in terms of rise time, overshoot, settling time and steady-state error. [2 marks]



Answer

a. The transfer function of the PI controller is (note: $K_i = K_p/T_i$):

$$G_c(s) = K_i/s + K_p$$

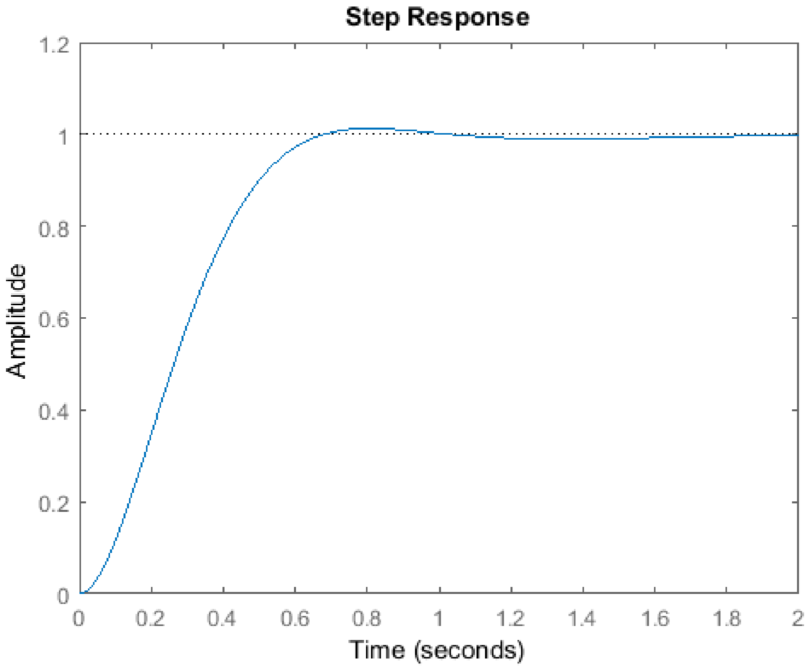
For the given system, the closed-loop transfer function with a PI controller is:

$$T(s) = \frac{C(s)}{R(s)} = \frac{G_c(s)G_p(s)}{1 + G_c(s)G_p(s)}$$

$$= \frac{\left(K_i/s + K_p\right)\left(\frac{1}{s^2 + 10s + 20}\right)}{1 + \left(K_i/s + K_p\right)\left(\frac{1}{s^2 + 10s + 20}\right)}$$

$$= \frac{K_p s + K_i}{s^3 + 10s^2 + \left(20 + K_p\right)s + K_i}$$

b. Let's reduce K_p to 30 and let K_i equal 70. The following figure shows the step response of the example system with PI controller.



c. Compared with the response of the system with PD controller, the response of the system is less oscillatory than before. Notice that the steady-state error is eliminated from the response.

But, on the other hand, the settling time of the system with PI controller is longer than the system with PD controller.

1.4. PID Controllers

For a PID controller, the control signal (u) to the plant is equal to the proportional gain (K_p) times the magnitude of the error plus the integral gain (K_i) times the integral of the error plus the derivative gain (K_d) times the derivative of the error.

The output of a PID controller, which is equal to the control input to the plant, is calculated in the time domain from the feedback error as follows:

$$u(t) = K_p e(t) + K_i \int e(t)dt + K_d \frac{de(t)}{dt}$$

The transfer function of a PID controller is found by taking the Laplace transform of system equation:

$$K_p + \frac{K_i}{s} + K_d s = \frac{K_d s^2 + K_p s + K_i}{s}$$

Where: K_p = proportional gain, K_i = integral gain, and K_d = derivative gain.

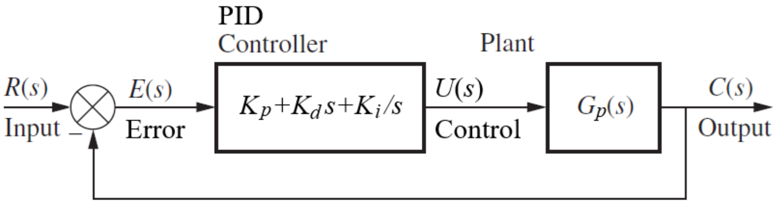


Figure 8: A unity feedback control system with proportional-integral-and-derivative controller

The PID controller tends to combine the characteristics of PI and PD controller. So, it is capable for improving both the transient response and steady-state characteristics of the system.

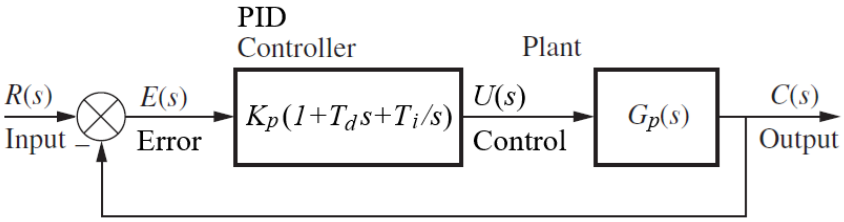
Tutorial for Example 5 – Adding PID Controller

For the given simple mass-spring-damper system in Tutorial for Example 1, add a proportional-integral-derivative controller in series with the system.

a. Derive the transfer function equation of the system. [4 marks]

b. Using the trial-and-error methods in MATLAB, determine the appropriate values of the parameters of the controller. Then, simulate the transient response of the system. [6 marks]

c. Describe the response of the system in terms of rise time, overshoot, settling time and steady-state error. [2 marks]



Answer

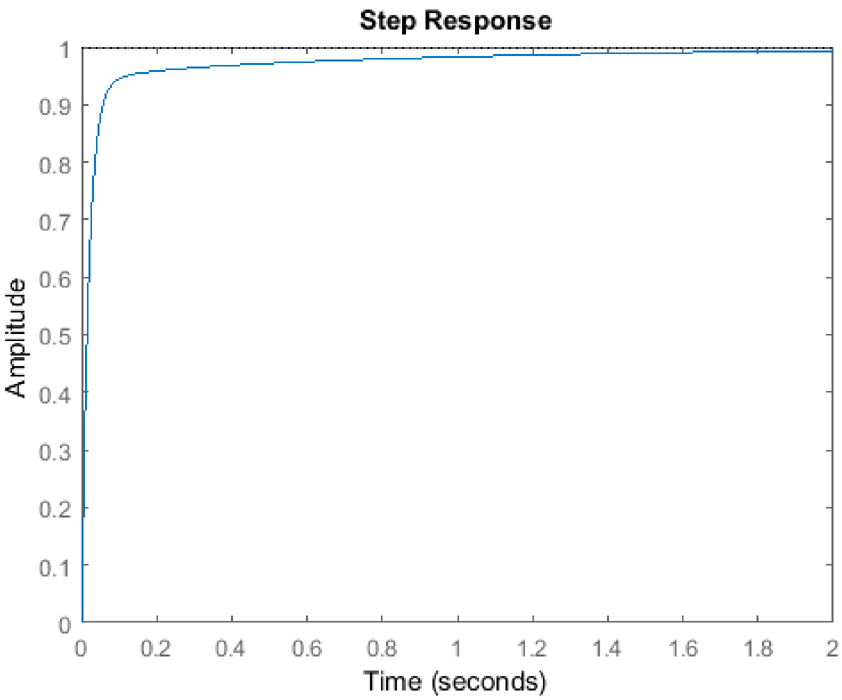
a. Now, let us examine PID control. The transfer function of the PID controller is (note $K_d = K_p T_d$ and $K_i = K_p/T_i$):

$$G_c(s) = K_d s + K_i/s + K_p$$

The closed-loop transfer function of the given system with a PID controller is:

$$\begin{aligned} T(s) = \frac{C(s)}{R(s)} &= \frac{G_c(s)G_p(s)}{1 + G_c(s)G_p(s)} \\ &= \frac{\left(K_d s + K_i/s + K_p\right)\left(\frac{1}{s^2 + 10s + 20}\right)}{1 + \left(K_d s + K_i/s + K_p\right)\left(\frac{1}{s^2 + 10s + 20}\right)} \\ &= \frac{K_d s^2 + K_p s + K_i}{s^3 + (10 + K_d)s^2 + \left(20 + K_p\right)s + K_i} \end{aligned}$$

b. After several iterations of tuning, the gains K_p = 350, K_i = 300, and K_d = 50 provided the desired response. The following figure shows the step response of the example system with PID controller.



c. Now, we have designed a closed-loop system with no overshoot, fast rise time, and no steady-state error.

We have reduced the proportional gain (K_p) because the integral controller also reduces the rise-time and increases the overshoot as the proportional controller does (double effect). The response of the system shows that the integral controller eliminated the steady-state error in this case.

1.5. Summary of Applications of Controllers

The following tables list the summary of controller's applications.

Controller Name	**Transfer Function Equation**	**Characteristics**
P	K_p	Reduces the rise time, increases the overshoot, and reduces the steady-state error.
I	$\frac{K_i}{s}$	Reduces steady-state error.
D	$K_d s$	Increases the transient response responsiveness and characteristics.
PI	$K_p + K_i/s$	Decrease the rise time, increase both the overshoot and the settling time, and reduces the steady-state error.
PD	$K_p + K_d s$	Reduce both the overshoot and the settling time.

PID	$\frac{K_d s^2 + K_p s + K_i}{s}$	Improve both the transient response and steady-state characteristics.

Table 1: Summary of the details of controllers

The general effects of each controller parameter : K_p, K_i, and K_d on a closed-loop system are summarized in the table below. Note, these guidelines are held in many cases, but not all. If you truly want to know the effect of tuning the individual gains, you will have to do more analysis or will have to perform testing on the actual system.

Controller	Rise Time	Overshoot	Settling Time	Steady-State Error
K_p	Decrease	Increase	Small Change	Decrease
K_i	Decrease	Increase	Increase	Decrease
K_d	Small Change	Decrease	Decrease	No Change

Table 2: General effects of controller parameters on a closed-loop system

2. Applications of Compensator

Controllers and compensators are slightly different in terms of the of their characteristics and purpose, their practical implementations, and their designs. We will be looking at some control systems with example compensators i.e. Lead, Lag, and Lead-Lag compensators.

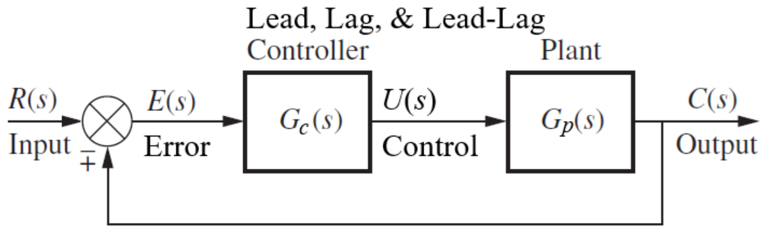


Figure 9: Control system with example compensators

2.1. Phase-lag Compensator

Lag compensator is commonly employed in the control systems to improve steady-state conditions and transient response of the systems.

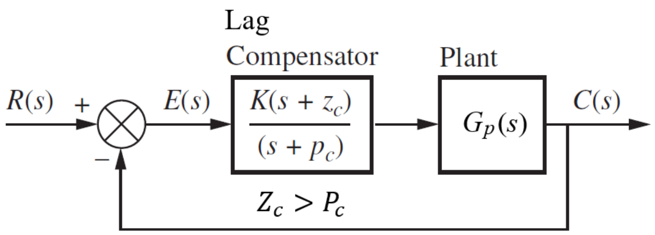


Figure 10: Block diagram of a Lag compensator with the plant

Lag compensators reduce steady-state error, so sometimes we want smaller steady-state error rather than shorter rise and settling time as in a Lead compensator.

The integrator in PI controller can cause some practical problems, e.g., “integrator windup” due to actuator saturation. PI controller is often approximated by “lag control.”

$$G_c(s) = \frac{(s - z_0)}{(s - p_0)} \quad \text{with} \quad |p_0| < |z_0|$$

That is, the pole is closer to the origin than the zero. Because $|z_0| < |p_0|$, the phase " added to the open-loop transfer function is negative. . . “phase lag”. Pole often placed very close to zero. e.g., $p_0 \approx 0.01$. Zero is placed near pole. e.g., $z_0 \approx 0.1$.

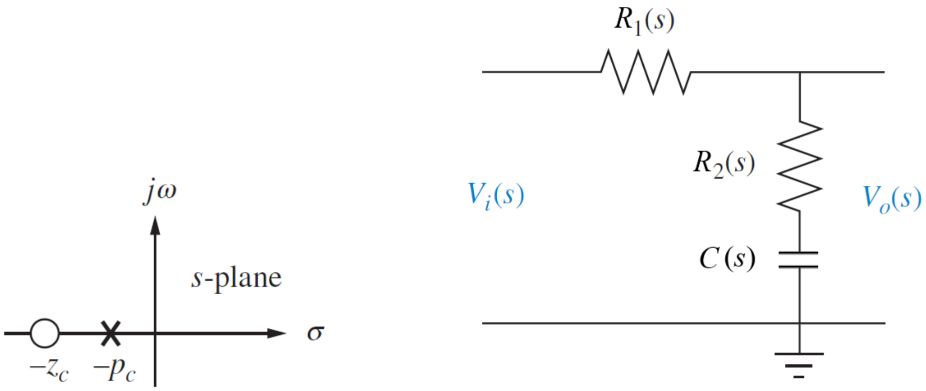


Figure 11: poles and zeros of Lag compensator in the s-plane and its circuit implementation

We want $|G_c(s)| \approx 1$ for all s to preserve transient response (and hence, have nearly the same root locus as for a proportional controller). The idea is to improve steady-state error but to modify the transient response as little as possible. That is, using proportional control, we have pole locations we like already, but poor steady-state error. So, we add a Lag compensator to minimally disturb the existing good pole locations but improve steady-state error.

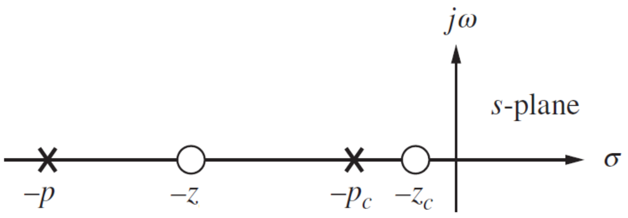


Figure 12: Poles and zeros of the system in the s-plane

Good steady-state error without overflow problems. Very similar to proportional control. The uncompensated system had a loop gain of:

$$K(\text{before}) = \lim_{s\to 0} G(s)$$

The Lag-compensated system has loop gain:

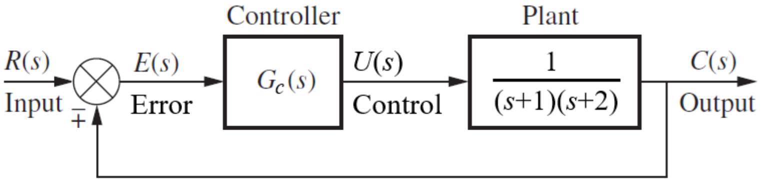
$$K(\text{after}) = \lim_{s\to 0} G_c(s)G(s) = \left(\frac{Z_0}{P_0}\right)\lim_{s\to 0} G(s)$$

Since $|z_0| > |p_0|$, there is an improvement in the position/velocity/acceleration error constant of the system, and a reduction in steady-state error. Transient response is mostly unchanged, but slightly slower settling due to small-magnitude slow “tail” caused by lag compensator.

Example for Tutorial 6 – Adding Lag Compensator

When $G_C(s) = 1$, the control system given below suffers from issues in both steady-state and transient response conditions.

- Steady-state: non-zero steady-state error.
- Transient response: sluggish system that takes time to settle down.



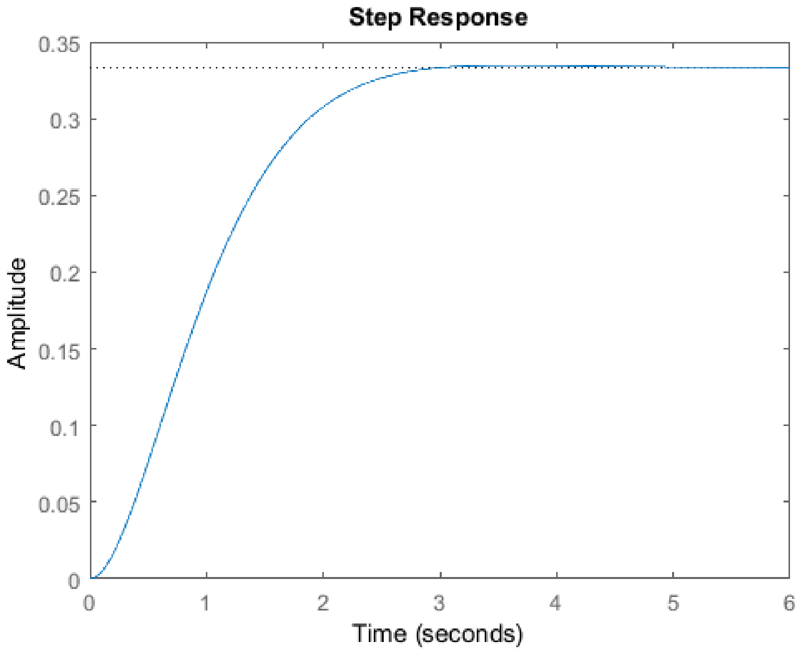
a. Simulate the uncompensated system in MATLAB. Comment on the result of the simulation. [6 marks]

b. Using the trial-and-error methods in MATLAB, design a Lead compensator that will be able to fix the problem observed in part (a). [4 marks]

c. Simulate in MATLAB and compare the uncompensated and compensated systems. Observe whether the compensator has achieved its purpose. [6 marks]

Answer

a. The simulation of the uncompensated system in MATLAB is shown in the figure below.

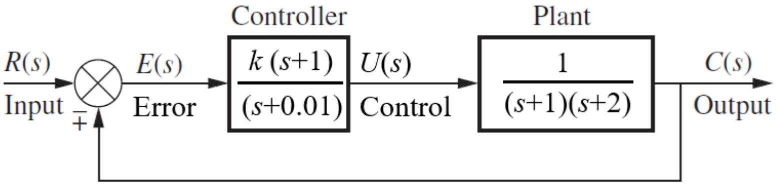


Looking into the step response of the given system, there are issues as highlighted before e.g. non-zero steady-state error and slow (i.e. sluggish) response of the system.

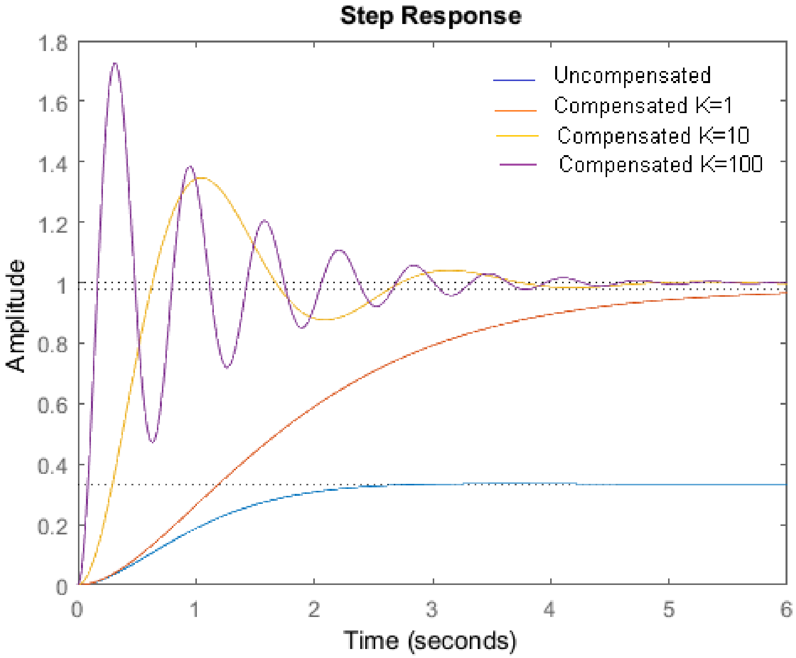
b. By trial and error in MATLAB, the gain and the pole and zero of the Lead compensators are determined:

$$G_c(s) = \frac{K(s+1)}{s+0.01}$$

The following figure shows the given system with Lead compensator.



c. The following figure shows the simulation of the system with various gains in MATLAB (with K = 1, 10, and 100).



Looking into the step response of the compensated system with the Lag compensator, the plot shows smaller steady-state error than uncompensated system. The plots shown below are with K = 1 (orange line), K = 10 (yellow line), and K = 100 (purple line). Notice the growing oscillation as you increase the system gain (K), but settling time increases for all cases.

2.2. Phase-lead Compensator

Lead compensator is typically used in the control systems to improve the transient response and stability of the systems.

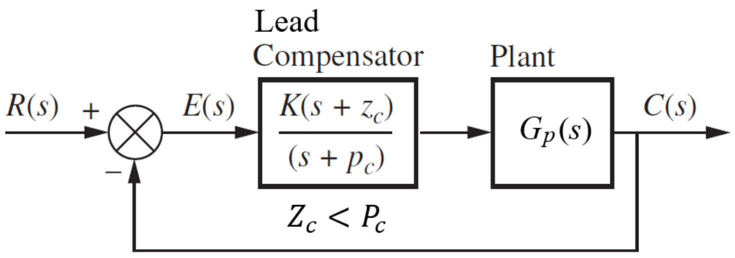


Figure 13: Block diagram of a Lead compensator with the plant

The Lead compensators improve transient response and stability, but they do not typically reduce steady-state error. Derivative magnifies noise. Instead of D-control or PD-control use “lead control.”

$$G_c(s) = \frac{(s - z_0)}{(s - p_0)} \quad \text{with} \quad |z_0| < |p_0|$$

That is, the zero is closer to the origin than the pole.

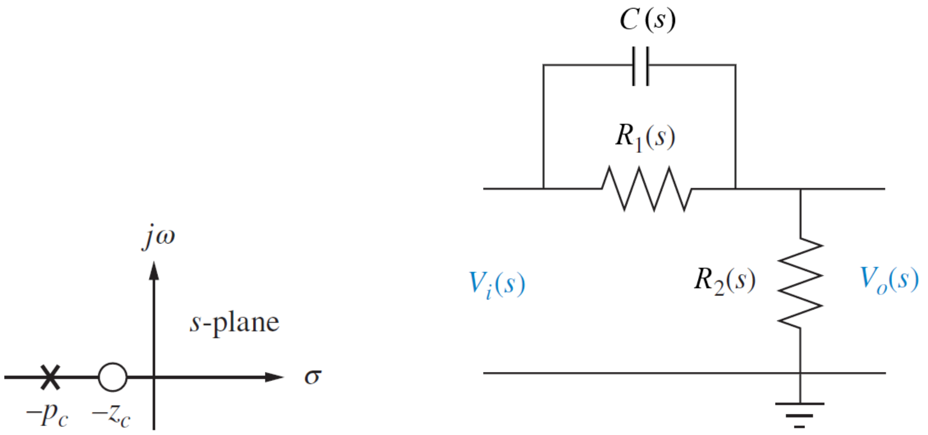


Figure 14: poles and zeros of Lead compensator in the s-plane and its circuit implementation

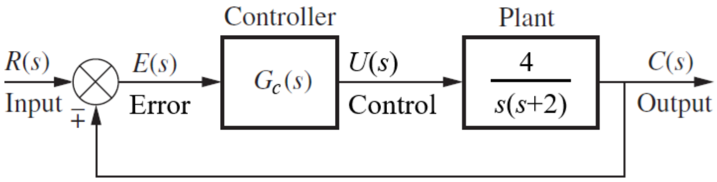
Lead compensator has the same form as Lag compensator, but with different intent:

- Lag compensator does not change locus much since $P_0 \approx Z_0 \approx 0$. Instead, the Lag compensator improves steady-state error.
- Lead compensator changes locus. Pole and zero locations chosen so that locus will pass through some desired point $s = s_1$.

Example for Tutorial 7 – Adding Lead Compensator

When $G_C(s) = 1$, the control system given in the figure below suffers from issues in the transient response conditions:

- Rise time: take some time for the system to rise up.
- Settling time: sluggish system that takes time to settle down.



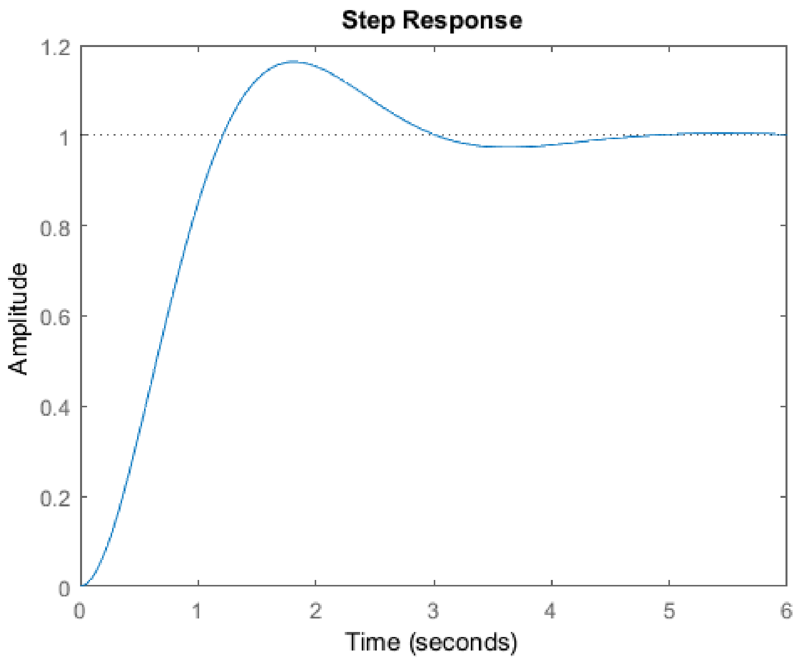
a. Simulate the uncompensated system in MATLAB. Comment on the result of the simulation. [6 marks]

b. Using the trial-and-error methods in MATLAB, design a Lead compensator that will be able to fix the problem observed in part (a). [4 marks]

c. Simulate in MATLAB and compare the uncompensated and compensated systems. Observe whether the compensator has achieved its purpose. [6 marks]

Answer

a. The following diagram shows the simulation of the uncompensated system in MATLAB.

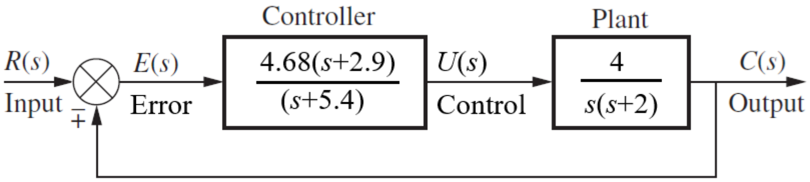


Looking into the step-response of the given system, there are issues as highlighted before e.g. slow (sluggish) response of the system e.g. long rise time and settling time.

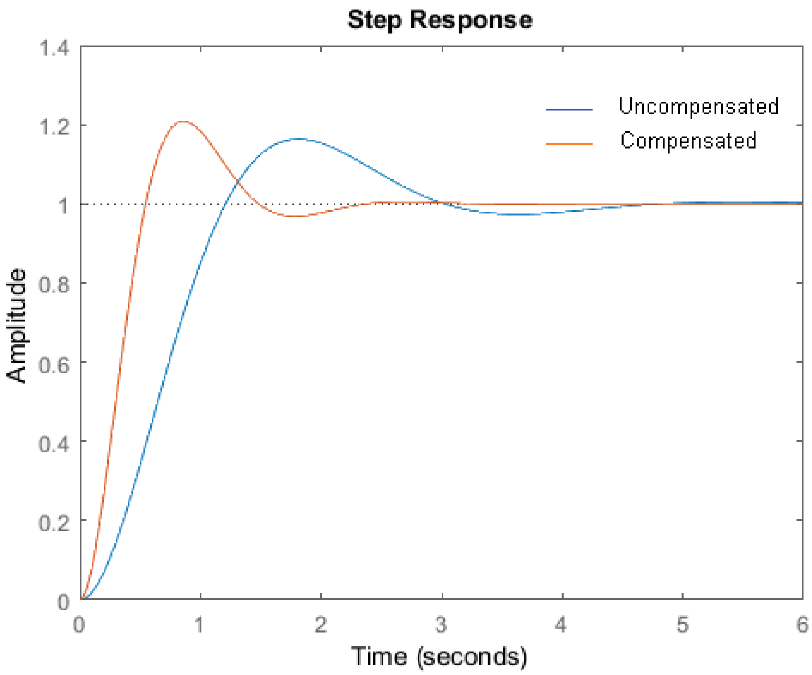
b. By trial and error, the gain and the pole and zero of the Lead compensators are determined.

$$G_c(s) = \frac{4.68(s+2.9)}{s+5.4}$$

The following figure shows the compensated system with the Lead compensator.



c. The following figure shows the simulation of uncompensated and compensated system in MATLAB.



From the step response plot of the uncompensated and compensated systems with lead compensator as shown in the figure below, the compensated system (red line) reaches steady state faster (i.e. shorter rise and settling times) than uncompensated system (blue line). Although, on the other hand, it has a higher percentage overshoot, M_p.

2.3. Lead-Lag Compensator

For Lead-Lag compensator, it combines Lead compensator and Lag compensator. Lead-lag compensator provides the benefits of both Lead and Lag compensators e.g. improve performance in terms of steady state and transient responses.

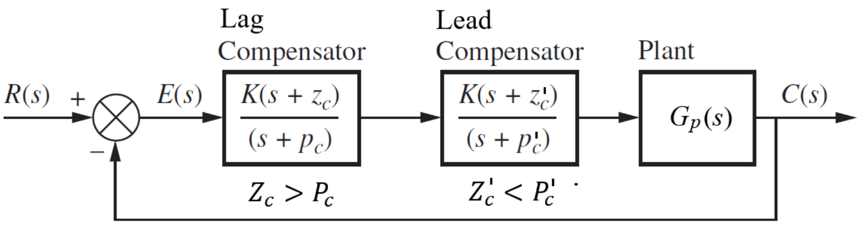


Figure 15: Block diagram of a lead-lag compensator with the plant

The transfer function of the Lead-Lag compensator is as given below.

$$\frac{(s - z_{lag})}{(s - p_{lag})} \frac{(s - z_{lead})}{(s - p_{lead})}$$

with

$$|p_{lag}| < |z_{lag}| \text{ and } |z_{lead}| < |p_{lead}|$$

The Lead-Lag compensator improves both steady-state error and transient response performance. Design of the Lead-Lag compensator requires careful design of its individual parts e.g. Lag compensator and Lead compensator. Trial and error are typically employed to get the best set up for the Lead-Lag compensator.

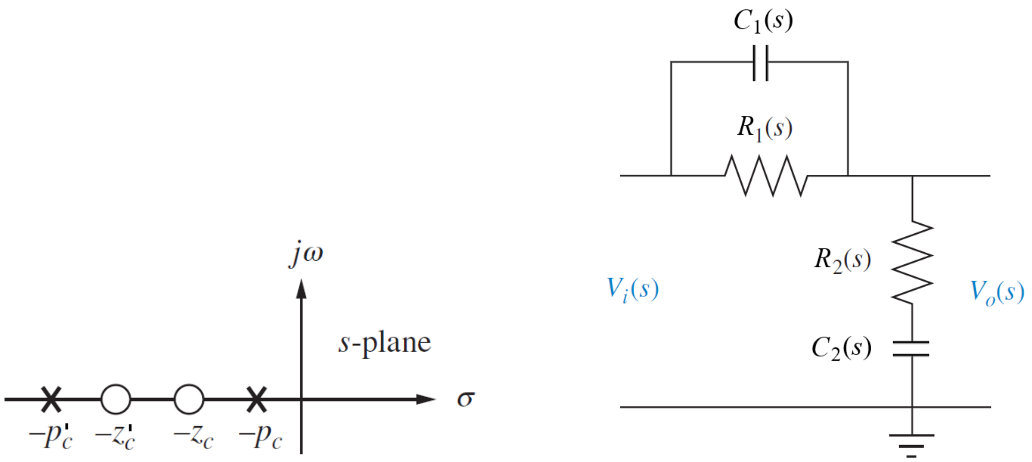


Figure 16: Poles and zeros of lead-lag compensator in the s-plane and its circuit implementation

If we must satisfy both the transient and steady-state specifications:

1. Design a Lead compensator to meet transient specifications first.
2. Include Lead compensator with plant after its design is final.
3. Design a Lag compensator (where “plant” = actual plant and Lead compensator combined) to meet steady-state specification.

When a lead-lag compensator is added into the system, the following figure shows step response and ramp response of the uncompensated, Lead compensated, and Lead-Lag compensated systems.

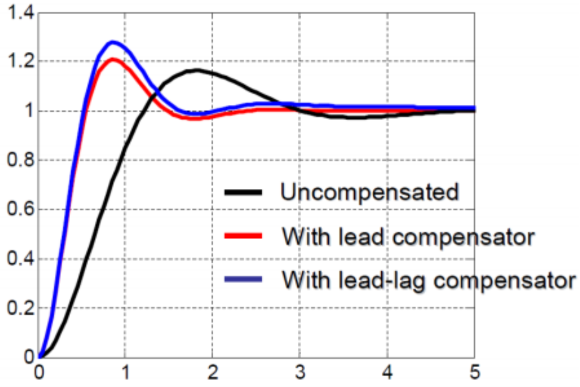


Figure 17: Step responses of uncompensated and compensated systems with Lead and Lead-Lag compensators

As shown in the figures, the Lead-Lag compensator provides the benefits of both the Lead and Lag compensators e.g. improve performance of the system in terms of steady-state and transient response.

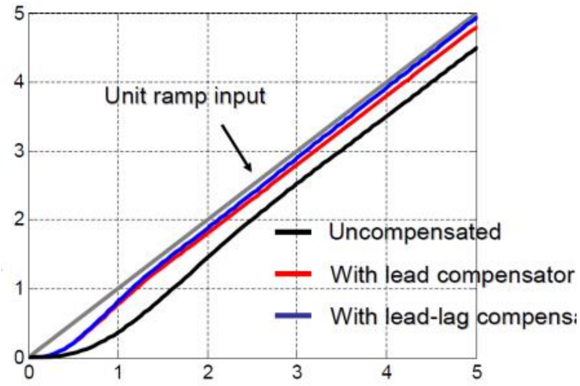


Figure 18: Ramp responses of uncompensated and compensated systems with Lead and Lead-Lag compensators

2.4. Summary of Applications of Compensators

The following table outlines a brief summary of applications of the compensators used in control systems.

Name of Compensator	**Transfer Function Equation**	**Characteristics**
Lag	$\frac{(s-z_0)}{(s-p_0)}$ with $\lvert p_0\rvert < \lvert z_0\rvert$	It improves steady-state error.
Lead	$\frac{(s-z_0)}{(s-p_0)}$ with $\lvert z_0\rvert < \lvert p_0\rvert$	It improves transient response performance.

Lead-lag	$\frac{(s-z_{lag})}{(s-p_{lag})}\frac{(s-z_{lead})}{(s-p_{lead})}$ with $\lvert p_{lag}\rvert < \lvert z_{lag}\rvert$ and $\lvert z_{lead}\rvert < \lvert p_{lead}\rvert$	It improves both steady-state error and transient response performance.

Table 3: Summary of details of compensators

3. Practical Implementations

The practical implementation of controllers or compensators in the course are implemented as op amp-based amplifier circuits. For to the practical implementations of controllers P, I, D, and any of their combinations, these are typically performed using the op-amp based amplifier circuit. But, the practical implementations of Lead, Lag, and Lead-lag compensators could be realised with the passive devices such as R, L, and C.

A true P, I, and D controller needs arbitrary gain for its proportional (P) part, true integration for its integration (I) part with infinite gain at low frequency, and true differentiation for its derivative (D) part with infinite gain at high frequency. The op amp-based circuits could realise them, but circuits with passive components can't do this well.

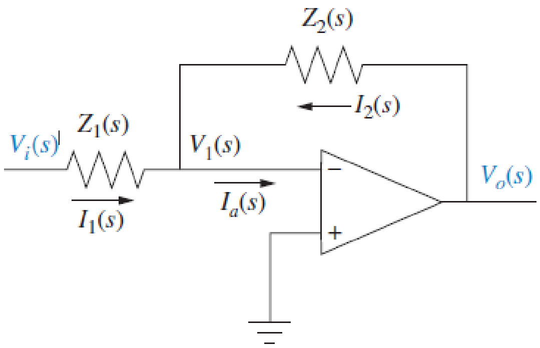


Figure 19: Impedances in the inverting operational amplifier circuit

For the practical circuit implementation for the controllers and compensators, we derived as the transfer function equation of an inverting operational amplifier whose configuration is shown above.

Implementing KCL on the inverting pin of the op amp, the currents in the given node are:

$$I_1(s) + I_2(s) = I_a(s)$$

With $I_a = I_- = I_+ = 0$, applying Ohm's law, the equation above becomes:

$$\frac{V_1(s) - V_i(s)}{Z_1(s)} + \frac{V_o(s) - V_1(s)}{Z_2(s)} = 0$$

As the non-inverting pin is grounded, $V_1(s) = V_- = V_+ = 0$ and rearranging the equation above, thus:

$$\frac{V_o(s)}{V_i(s)} = \frac{Z_2(s)}{Z_1(s)}$$

3.1. Practical Active Circuits for Controllers

By judicious choice of $Z_1(s)$ and $Z_2(s)$, this circuit shown in the figure below is used as a building block to implement compensators and controllers, such as PID controllers and lag-lead compensators using operational amplifiers.

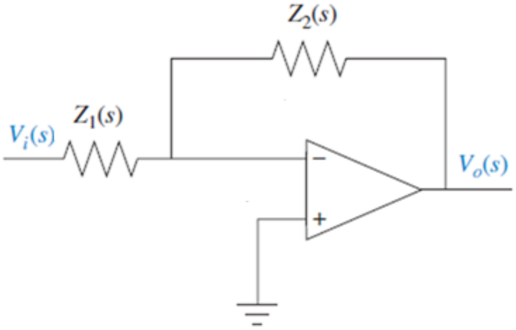


Figure 20: Practical active circuits for implementing controllers or compensators

The following table outlines the practical active circuit for realising controllers.

Controller	$\boldsymbol{Z_1(s)}$	$\boldsymbol{Z_2(s)}$	$\boldsymbol{G_c(s) = \frac{Z_1(s)}{Z_2(s)}}$
Proportional (gain)	R_1	R_2	$-\frac{R_1}{R_2}$
Integral	R	C	$-\frac{\left(\frac{1}{RC}\right)}{s}$
Derivative	C	R	$-RCs$
Proportional-Integral (PI)	R_1	R_2 and C (in series)	$-\left(\frac{R_1}{R_2}\right)\left(\frac{s+\frac{1}{R_2C}}{s}\right)$
Proportional-Derivative (PD)	C and R_1(in parallel)	R_2	$-R_2C\left(s+\frac{1}{R_1C}\right)$
Proportional-Integral-Derivative (PID)	C_1 and R_1(in parallel)	R_2 and C_2 (in series)	$-\left(\frac{R_2}{R_1}+\frac{C_1}{C_2}+R_2C_1s+\frac{\frac{1}{R_1C_2}}{s}\right)$

Table 4: Summary of practical implementations of active circuit controllers

3.2. Practical Circuits for Compensators

The following table shows the practical active circuits for realising the compensator.

Compensator	$\boldsymbol{Z_1(s)}$	$\boldsymbol{Z_2(s)}$	$\boldsymbol{G_c(s) = \frac{Z_1(s)}{Z_2(s)}}$
Lag	C_1 and R_1 (in parallel)	C_2 and R_2 (in parallel)	$-\left(\frac{C_1}{C_2}\right)\left(\frac{s+\frac{1}{R_1C_1}}{s+\frac{1}{R_2C_2}}\right)$ where $R_2C_2 > R_1C_1$
Lead	C_1 and R_1 (in parallel)	C_2 and R_2 (in parallel)	$-\left(\frac{C_1}{C_2}\right)\left(\frac{s+\frac{1}{R_1C_1}}{s+\frac{1}{R_2C_2}}\right)$ where $R_2C_2 < R_1C_1$
Lead-Lag	Cascading lag compensator with lead compensator (as shown below).		

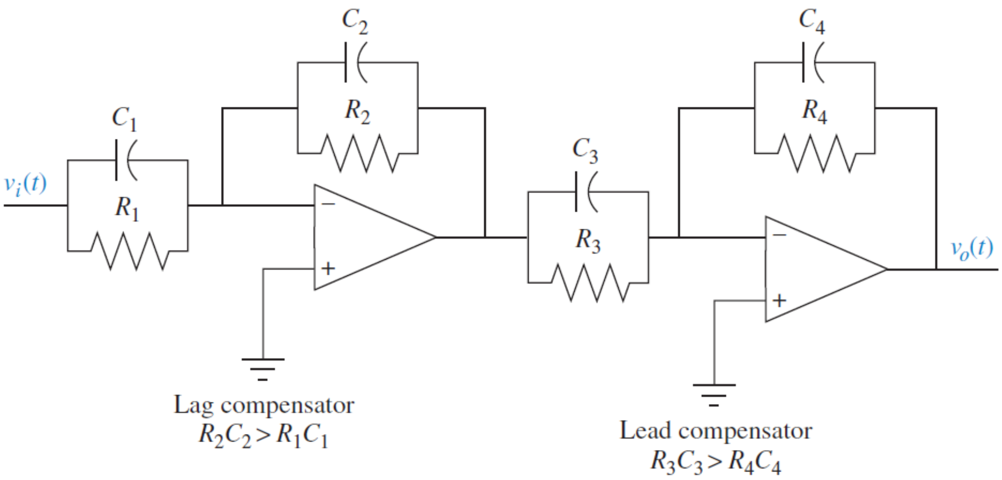
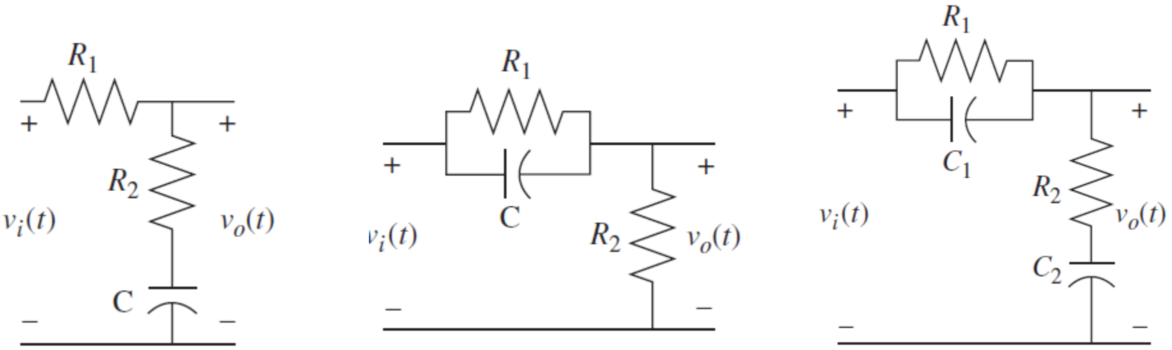


Table 5: Summary of practical implementations of active circuit compensators

The following table shows the practical passive circuits for realising the compensator.

Compensator	$\boldsymbol{V_i(s)}$	$\boldsymbol{V_o(s)}$	$\boldsymbol{G_c(s) = \frac{V_o(s)}{V_i(s)}}$

Lag	R_1, R_2, and C (all in series)	C and R_2 (both in series)	$\left(\frac{R_1}{R_1+R_2}\right)\left(\frac{s+\frac{1}{R_2C}}{s+\frac{1}{(R_1+R_2)C}}\right)$
Lead	C and R_1 (both in parallel) and in series with R_2	R_2	$\frac{s+\frac{1}{R_1C}}{s+\frac{1}{R_1C}+\frac{1}{R_2C}}$
Lead-Lag	C_1 and R_1(both in parallel) and in series with C_2 and R_2 (both in series)	C_1 and R_2 (both in series)	$\frac{\left(s+\frac{1}{R_1C_1}\right)\left(s+\frac{1}{R_2C_2}\right)}{s+\left(\frac{1}{R_1C_1}+\frac{1}{R_2C_2}+\frac{1}{R_2C_1}\right)s+\frac{1}{R_1R_2C_1C_2}}$



Lag compensator

Lead compensator

Lag-lead compensator

Table 6: Summary of practical implementations of passive circuit compensators

Example for Tutorial 8 - PID Controller Circuit

Design a practical implementation of the PID controller using op-amp based circuit when the transfer function of the PID controller is: [10 marks]

$$G_c(s) = \frac{(s+55.92)(s+0.5)}{s}$$

Answer

The transfer function equation of the controller can be put in the form:

$$G_c(s) = s + 56.42 + \frac{27.96}{s}$$

Equating the transfer function of the controller with the transfer function of PID controller from the table.

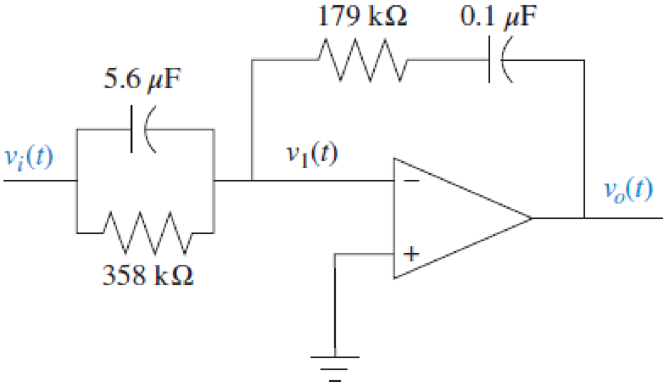
$$s + 56.42 + \frac{27.96}{s} = \left(\frac{R_2}{R_1} + \frac{C_1}{C_2}\right) + R_2C_1s + \frac{\frac{1}{R_1C_2}}{s}$$

Comparing the transfer function of the PID controller in the table with the controller, we obtain the following three relationships:

$$\frac{R_2}{R_1} + \frac{C_1}{C_2} = 56.42 \qquad R_2C_1 = 1 \qquad 1/R_2C_1 = 27.96$$

Since there are four unknowns and three equations, we arbitrarily select a practical value for one of the elements. Selecting C_2 = 0.1 µF, the remaining values are found to be R_1 = 357.65 kΩ, R_2 = 178,891 kΩ, and C_1 = 5.59 µF.

The complete op-amp based circuit for the PI controller is shown in the figure below, where the circuit element values have been rounded off.



4. Configuring the Controller

In majority of the cases, compensator is commonly designed for tackling a particular issue or problem in control system with its specific set up or arrangement.

On the other hand, controller is typically intended and designed to be able to be adjusted or tuned-in to manage the operation of the system.

There are many approaches to configure controllers. But these are typically classified into e.g. ad-hoc (on the spot), experimentation, or prescriptive formulas.

4.1. Methods for Configuring/Tuning-In Controller

There are various ways to configure and to tune in the controller in control system.

Tuning Method	**Advantages**	**Disadvantages**

Tuning	No math required.	Requires experience.
Software tools	Consistent tuning, can employ computer-automated control system design techniques, may include devices analysis, allows simulation before implementation, and can support non-steady-state (NSS) tuning.	Some cost or training involved.
Ziegler–Nichols	Proven method.	Process upset, some trial-and-error, very aggressive tuning.
Tyreus-Luyben	Proven method.	Process upset, some trial-and-error, very aggressive tuning.
Cohen–Coon	Good process models.	Some math required and only good for first-order processes.
Åström-Hägglund	Can be used for auto tuning; amplitude is minimum, so this method has lowest process upset.	The process itself is inherently oscillatory.

Table 7: Common methods for configuring the controllers

4.2. General Tips for Designing a PID Controller

When you are designing a PID controller for a given system, follow the steps shown below to obtain the desired response.

1. Obtain an open-loop response and determine what needs to be improved.
2. Add proportional control to improve the rise-time.
3. Add a derivative control to reduce the overshoot.
4. Add integral control to reduce the steady-state error.
5. Adjust each of the gains K_p, K_i, and K_d until you obtain a desired overall response. You can always refer to the table shown to find out which controller controls which characteristics.

Lastly, please keep in mind that you do not need to implement all three controllers (proportional, derivative, and integral) into a single system, if not necessary.

For example, if a PI controller meets the given requirements (like the above example), then you do not need to implement a derivative controller on the system. Keep the controller as simple as possible.