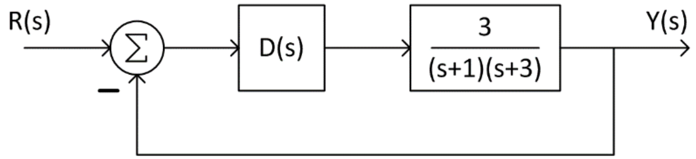
SECTION 6: ROOT-LOCUS DESIGN

ESE 430 – Feedback Control Systems

Introduction

Introduction

- Consider the following unity-feedback system

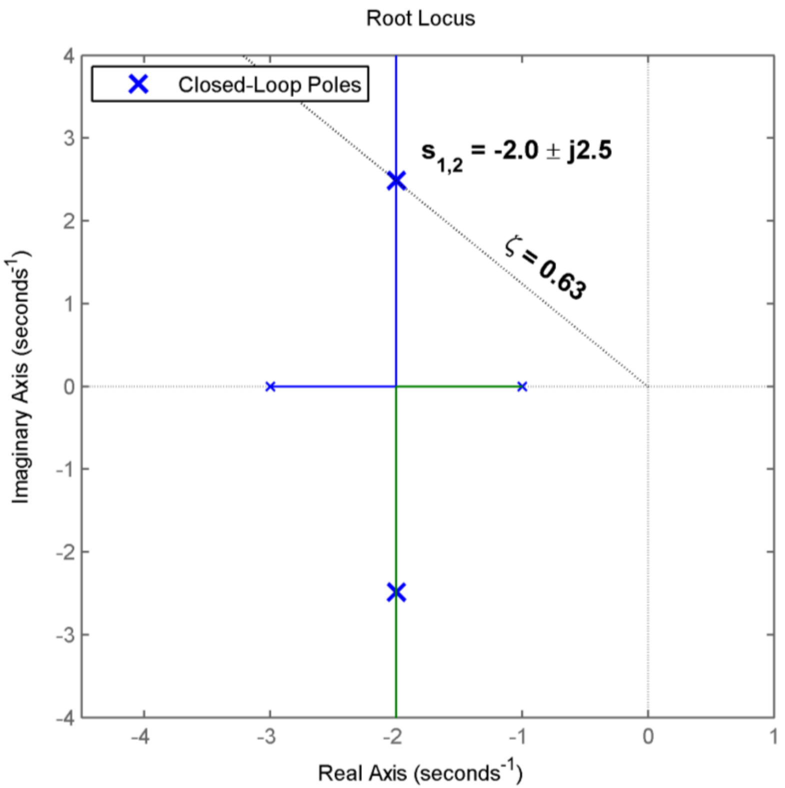


$$T(s) = \frac{3K}{s^2 + 4s + 3 + 3K}$$

- Assume $D(s) = K$
  - A proportional controller
- Design for 8% overshoot
  - Use root locus to determine K to yield required ζ

$$\zeta = -\frac{\ln(0.08)}{\sqrt{\pi^2 + \ln^2(0.08)}} = 0.63$$

- Desired poles and gain:
  - $s_{1,2} = -2 \pm j2.5$
  - $K = 2.4$



Introduction

- Overshoot is 8%, as desired, but steady-state error is large:
  - $e_{ss} = 29.4\%$
- Position constant:

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

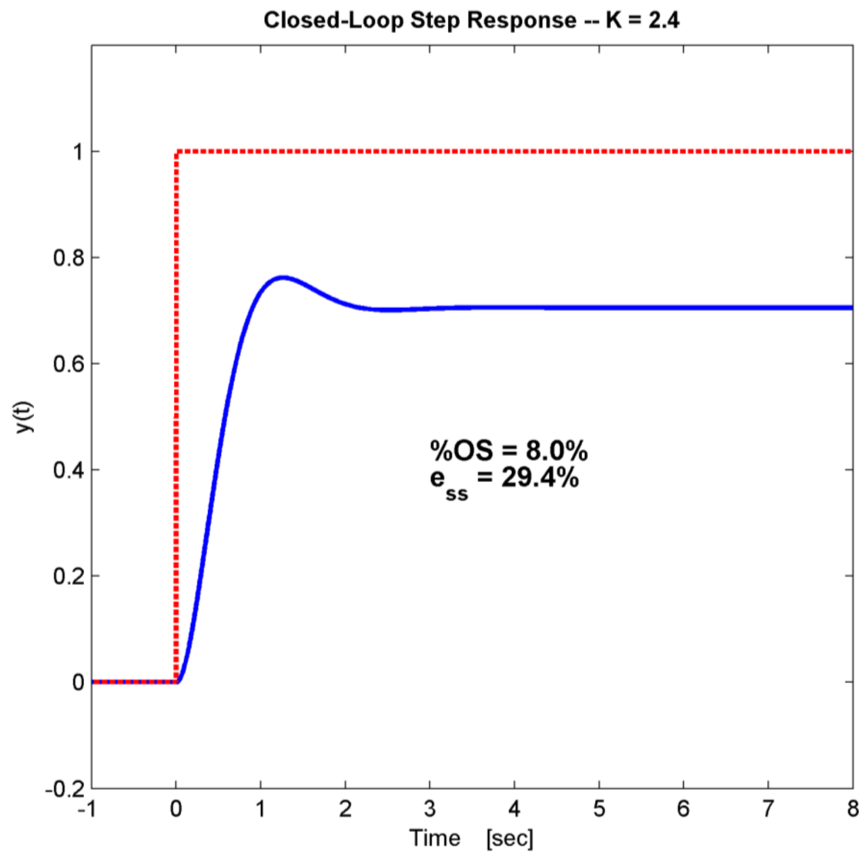
$$K_p = \lim_{s\to 0} \frac{3K}{(s+1)(s+3)} = K$$

$$K_p = 2.4$$

- Steady-state error:

$$e_{ss} = \frac{1}{1+K_p} = \frac{1}{1+K}$$

$$e_{ss} = \frac{1}{1+2.4} = 0.294$$



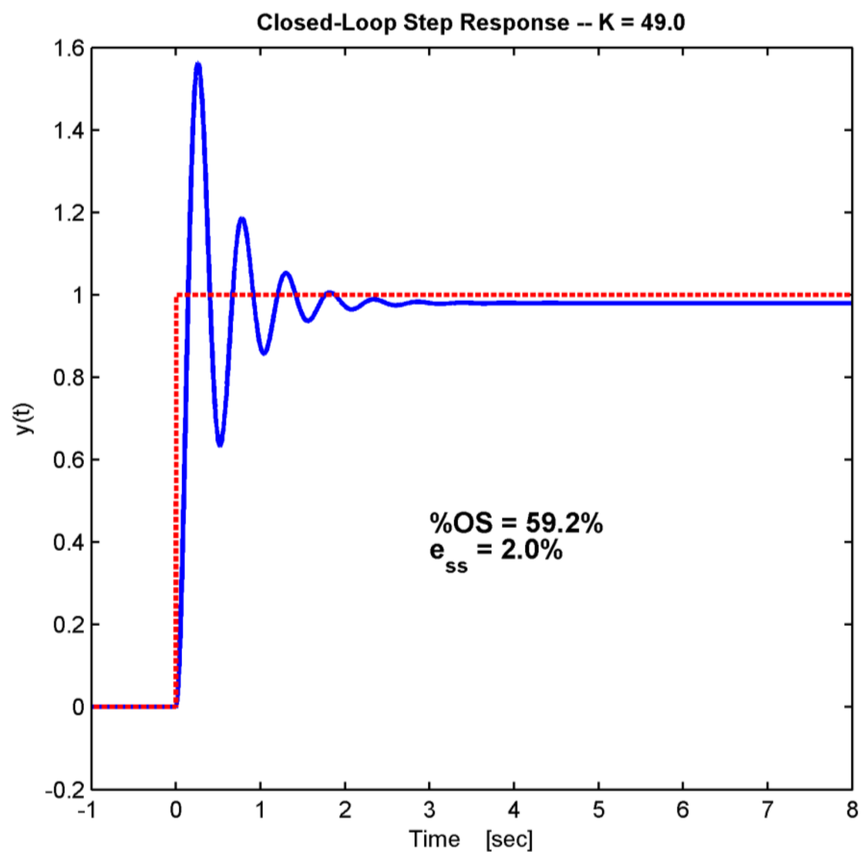
Introduction

- Let's say we want to reduce steady-state error to 2%
- Determine required gain

$$e_{ss} = \frac{1}{1+K_p} = \frac{1}{1+K} = 0.02$$

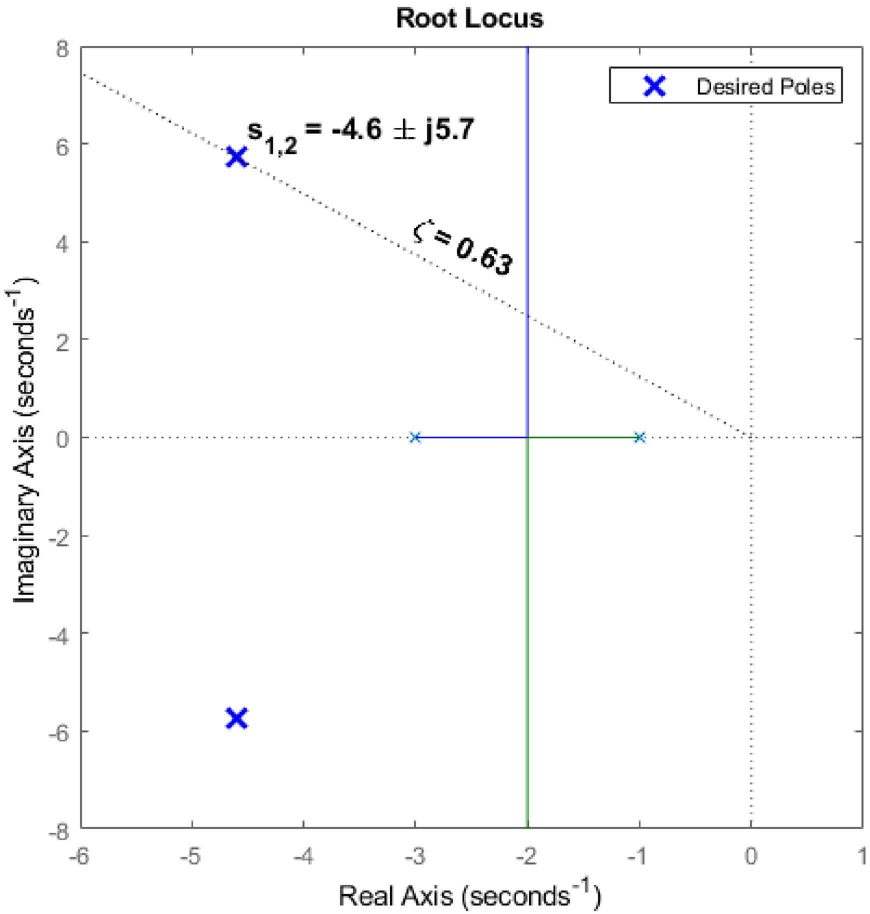
$$K = \frac{1}{0.2} - 1 = 49$$

- Transient response is degraded
  - $OS = 59.2\%$
- Can set overshoot *or* steady-state error via gain adjustment
  - Not both simultaneously



Introduction

- Now say we want $\mathrm{OS} = 8\%$ and $t_s \approx 1\ sec$, we'd need:
  $$\zeta = 0.63 \quad \text{and} \quad \sigma = 4.6$$
- Desired poles are not on the root locus
- Closed-loop poles can exist *only* on the locus
  - If we want poles elsewhere, we must *move* the locus
- Modify the locus by adding dynamics (poles and zeros) to the controller
  - A ***compensator***

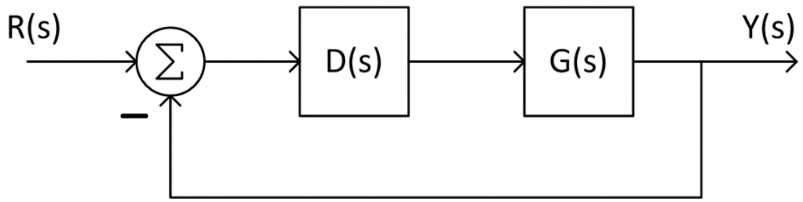


Introduction

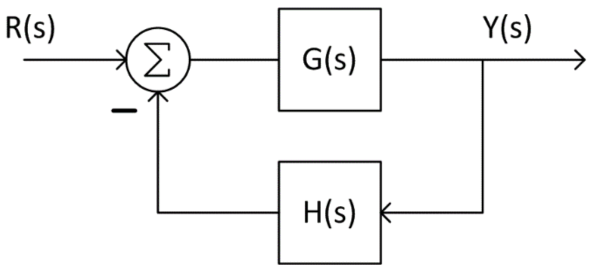
- We'll learn how to use root-locus techniques to design compensators to do the following:
  - ***Improve steady-state error***
    - Proportional-integral (PI) compensator
    - Lag compensator
  - ***Improve dynamic response***
    - Proportional-derivative (PD) compensator
    - Lead compensator
  - ***Improve dynamic response and steady-state error***
    - Proportional-integral-derivative (PID) compensator
    - Lead-lag compensator

Compensation Configurations

- Two basic compensation configurations:
  - ***Cascade compensation***



  - ***Feedback compensation***

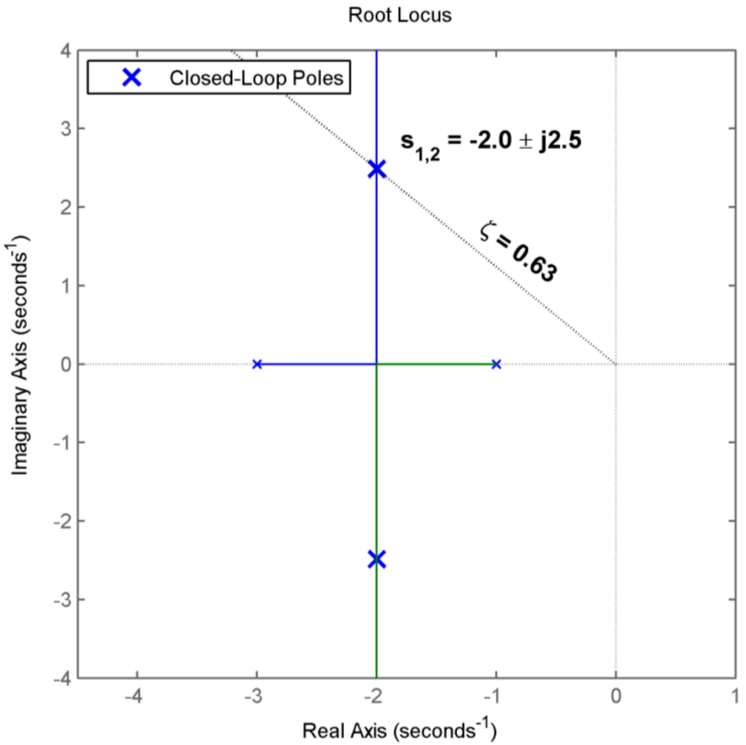


- We will focus on ***cascade*** compensation

Improving Steady-State Error

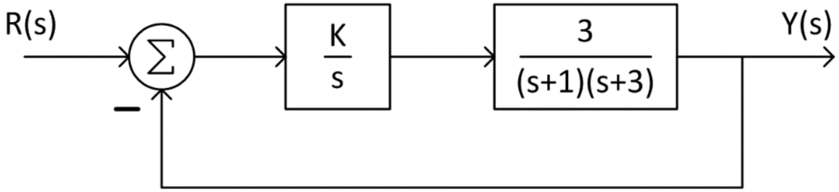
Improving Steady-State Error

- We've seen that we can improve steady-state error by adding a pole at the origin
  - An integrator
  - System type increased by one for unity-feedback
- For example, consider the previous example
  - Let's say we are happy with 8% overshoot and the corresponding pole locations
  - But, want to reduce steady-state error to 2% or less



Improving Steady-State Error

- System is type 0
  - Adding an integrator to $D(s)$ will increase it to type 1
  - Zero steady-state error for constant reference
- Let's first try a very simple approach:



- Plot the root locus for this system
  - How does the added pole at the origin affect the locus?

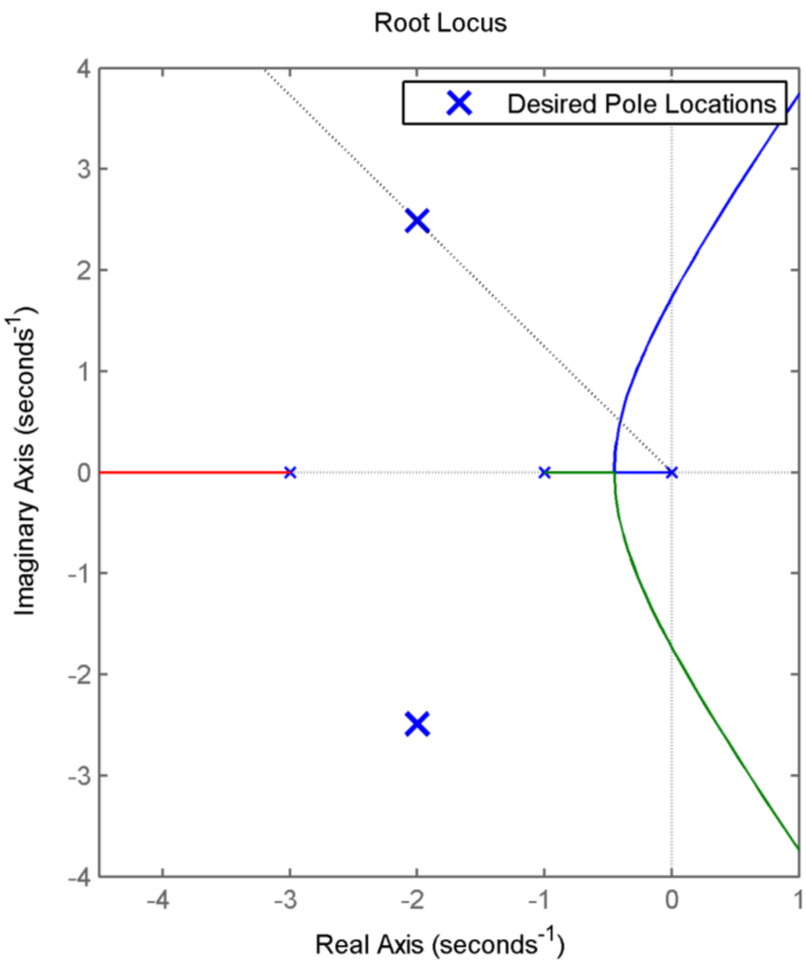
Improving Steady-State Error

- Now have $(n - m) = 3$ asymptotes to C^{∞}

  $\theta_a = 60°, 180°, 300°$

  $\sigma_a = -1.33$

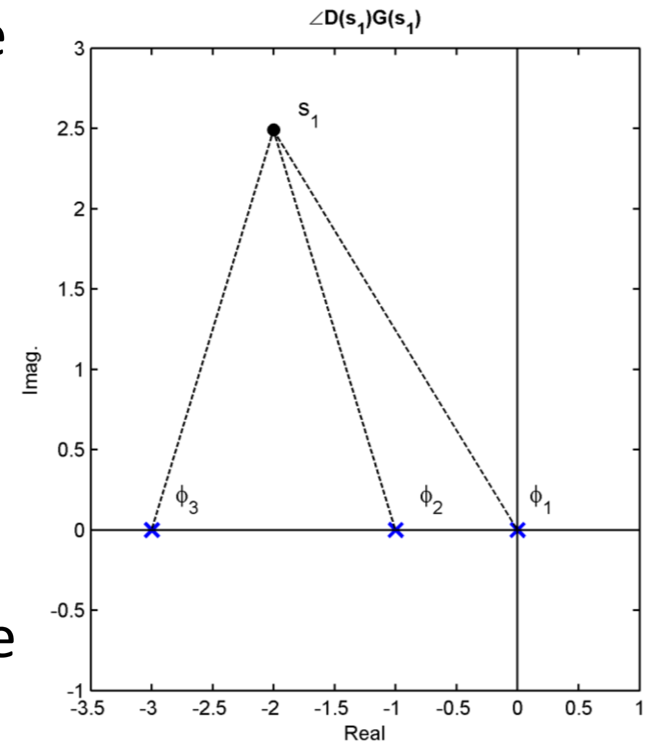
- Locus now crosses into the RHP
  - Integrator has had a *destabilizing* effect on the closed-loop system
- System is now type 1, but
- Desired poles are no longer on the root locus



Improving Steady-State Error

- Desired poles no longer satisfy the angle criterion:

$$\angle D(s_1)G(s_1) = -(\phi_1 + \phi_2 + \phi_3)$$
$$\angle D(s_1)G(s_1) = -(128.8° + 111.9° + 68.1°)$$
$$\angle D(s_1)G(s_1) = -308.8° \neq 180°$$



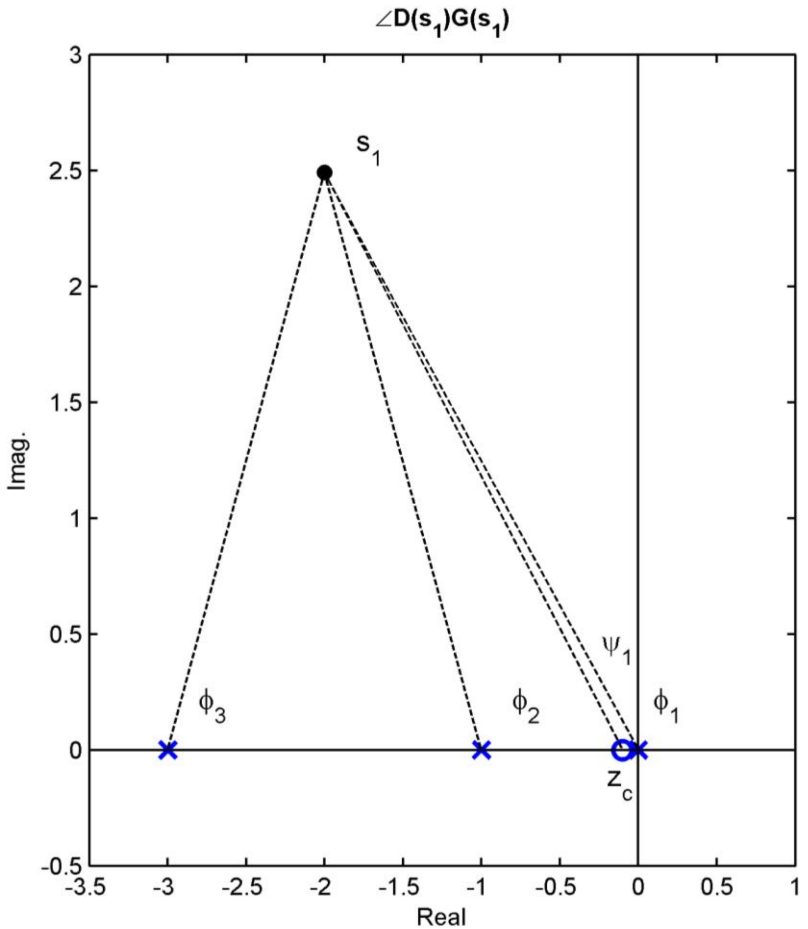
- Excess angle from the additional pole at the origin, ϕ_1
- How could we modify $D(s)$ to satisfy the angle criterion at s_1?
  - A zero at the origin would do it, of course
  - But, that would cancel the desired pole at the origin
- ***How about a zero very close to the origin?***

Improving Steady-State Error

- Now, $\psi_1 \approx \phi_1$
  - Angle contributions *nearly* cancel
  - s_1 is not on the locus, but *very close*
- The closer the zero is to the origin, the closer s_1 will be to the root locus
- Let $z_c = -0.1$
- Controller transfer function:

$$D(s) = K\frac{(s + 0.1)}{s}$$

- Plot new root locus to see how close it comes to s_1

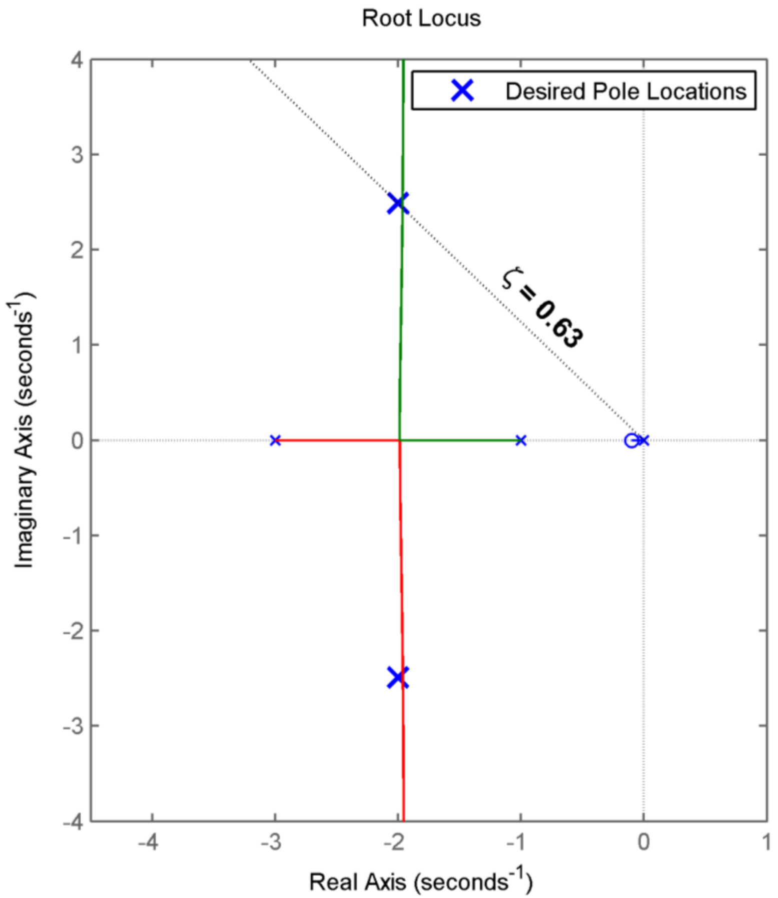


Improving Steady-State Error

- Now only two asymptotes to C^{∞}
  - $\theta_a = 90°,\ 270°$
  - $\sigma_a = -1.95$
- Real-axis breakaway point:
  - $s = -1.99$
- s_1 not on locus, but close
- Closed-loop poles with $\zeta = 0.63$:

$$s_{1,2} = -1.96 \pm j2.44$$

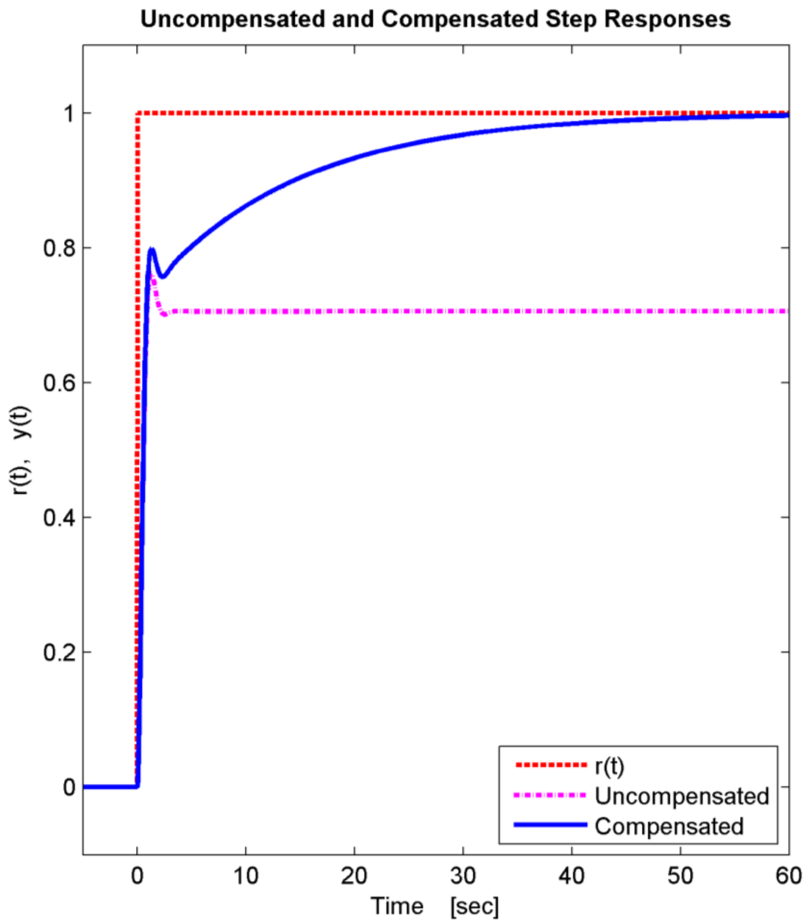
- Gain: $K = 2.37$
  - Determined from the MATLAB root locus plot



Improving Steady-State Error

- Initial transient relatively unchanged
  - Pole/zero pair near the origin nearly cancel
  - 2^{nd}-order poles close to desired location
- Zero steady-state error
  - Pole at origin increases system type to type 1
  - Slow transient as error is integrated out
- 2^{nd}-order approximation is valid
  - Poles: $s = -0.07,$

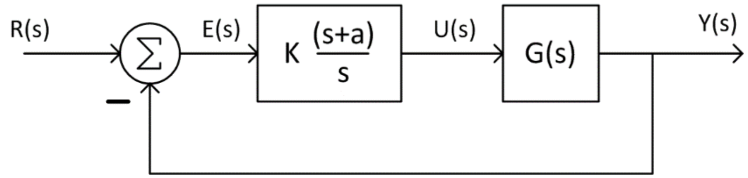
    $s = -1.96 \pm j2.44$
  - Zeros: $s = -0.1$



Ideal Integral Compensation

Proportional-Integral Compensation

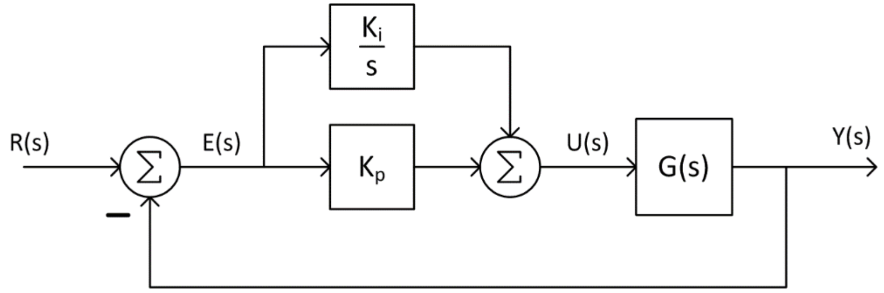
- The compensator we just designed is an ***ideal integral*** or ***proportional-integral (PI) compensator***



- Control input to plant, $U(s)$, has two components:
  - One ***proportional*** to the error, plus
  - One proportional to the ***integral*** of the error

$$U(s) = E(s)\left[K\frac{(s+a)}{s}\right] = KE(s) + \frac{Ka}{s}E(s)$$

- Equivalent to:



PI Compensation – Summary

- ***PI compensation***

$$D(s) = K\frac{(s+a)}{s} = K_p + \frac{K_i}{s}$$

  - Controller adds a ***pole at the origin*** and a ***zero nearby***

- Pole at origin (integrator) increases system type, ***improves steady-state error***

- Zero near the origin nearly cancels the added pole, leaving ***transient response nearly unchanged***

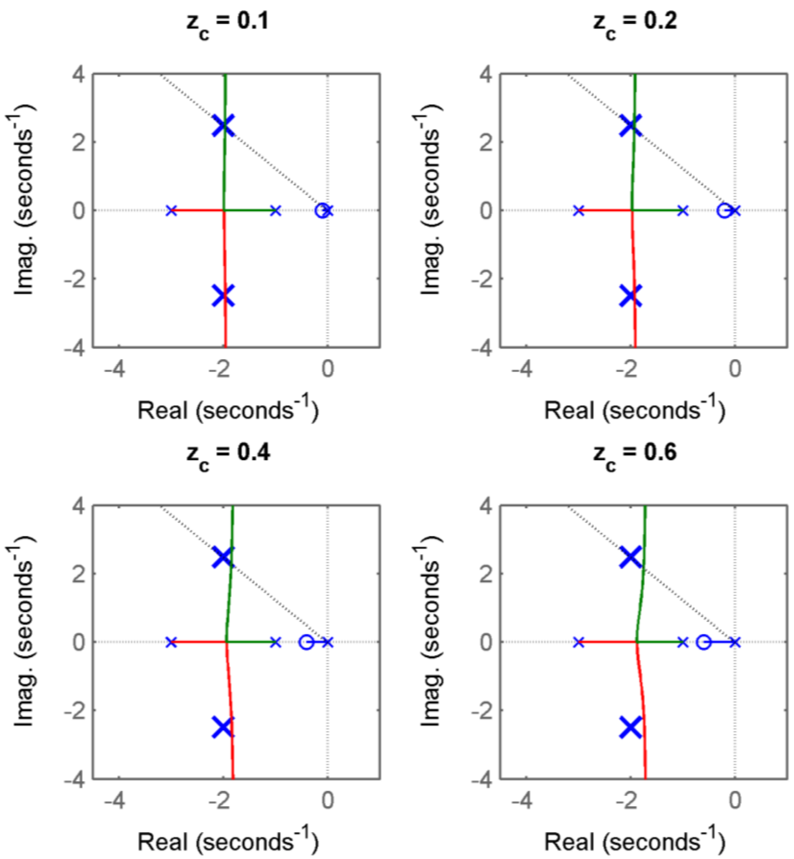
PI Compensation – Zero Location

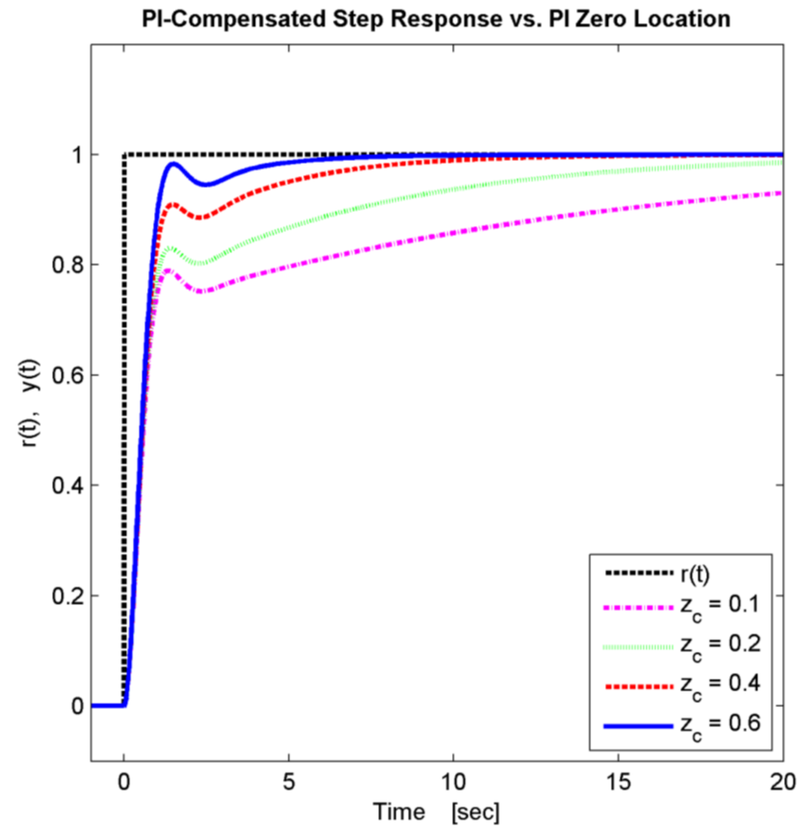
- ***Compensator zero very close to the origin:***
  - Closed-loop poles moved very little from uncompensated location
  - Relatively low integral gain, K_i
  - Closed-loop pole close to origin – slow
  - Slow transient as error is integrated out

- ***Compensator zero farther from the origin:***
  - Closed-loop poles moved farther from uncompensated location
  - Relatively higher integral gain, K_i
  - Closed-loop pole farther from the origin – faster
  - Error is integrated out more quickly

PI Compensation – Zero Location

- Root locus and step response variation with z_c:





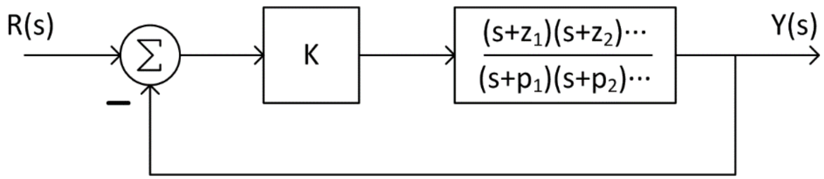
Lag Compensation

Lag Compensation

- PI compensation requires an ***ideal integrator***
  - Active circuitry (opamp) required for analog implementation
  - Susceptible to ***integrator windup***
- An alternative to PI compensation is ***lag compensation***
  - Pole placed near the origin, not at the origin
  - Analog implementation realizable with passive components (resistors and capacitors)
- Like PI compensation, lag compensation uses a closely-spaced pole/zero pair
  - Angular contributions nearly cancel
  - Transient response nearly unaffected
- System type not increased
  - Error is improved, not eliminated

Lag Compensation – Error Reduction

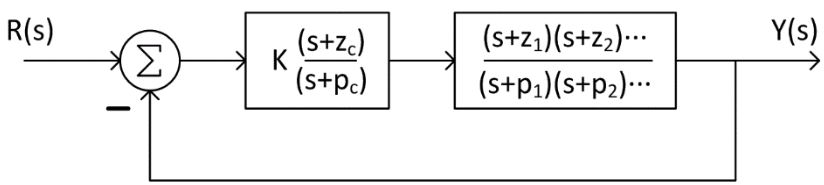
- Consider the following generic feedback system



- A type 0 system, assuming $p_i \neq 0,\ \forall i$
- Position constant:

$$K_{pu} = \lim_{s \to 0} G(s) = K \frac{z_1 z_2 \cdots}{p_1 p_2 \cdots}$$

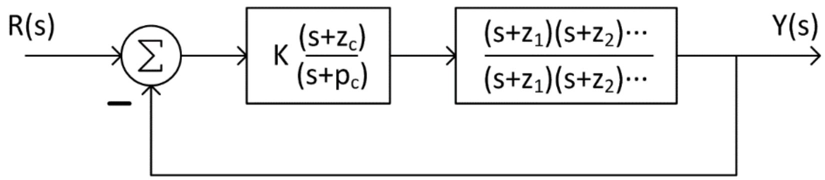
- Now, add lag compensation



- The compensated position constant :

$$K_{pc} = \left(K \frac{z_1 z_2 \cdots}{p_1 p_2 \cdots} \right) \frac{z_c}{p_c} = K_{pu} \frac{z_c}{p_c}$$

Lag Compensation – Error Reduction



$$K_{pc} = \left(K \frac{z_1 z_2 \cdots}{p_1 p_2 \cdots} \right) \frac{z_c}{p_c} = K_{pu} \frac{z_c}{p_c}$$

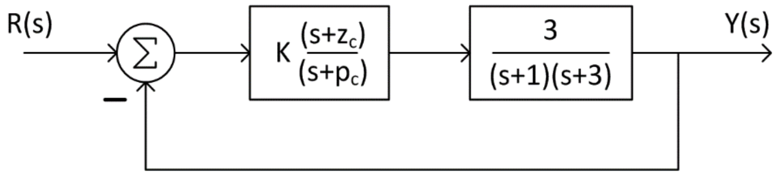
- Compensator pole is closer to the origin than the compensator zero, so

$$z_c > p_c \quad \text{and} \quad K_{pc} > K_{pu}$$

- For large improvements in e_{ss}, make $z_c \gg p_c$
  - But, to avoid affecting the transient response, we need $z_c \approx p_c$
  - As long as both z_c and p_c are very small, we can satisfy both requirements: $z_c \gg p_c$ and $z_c \approx p_c \approx 0$

Lag Compensation – Example

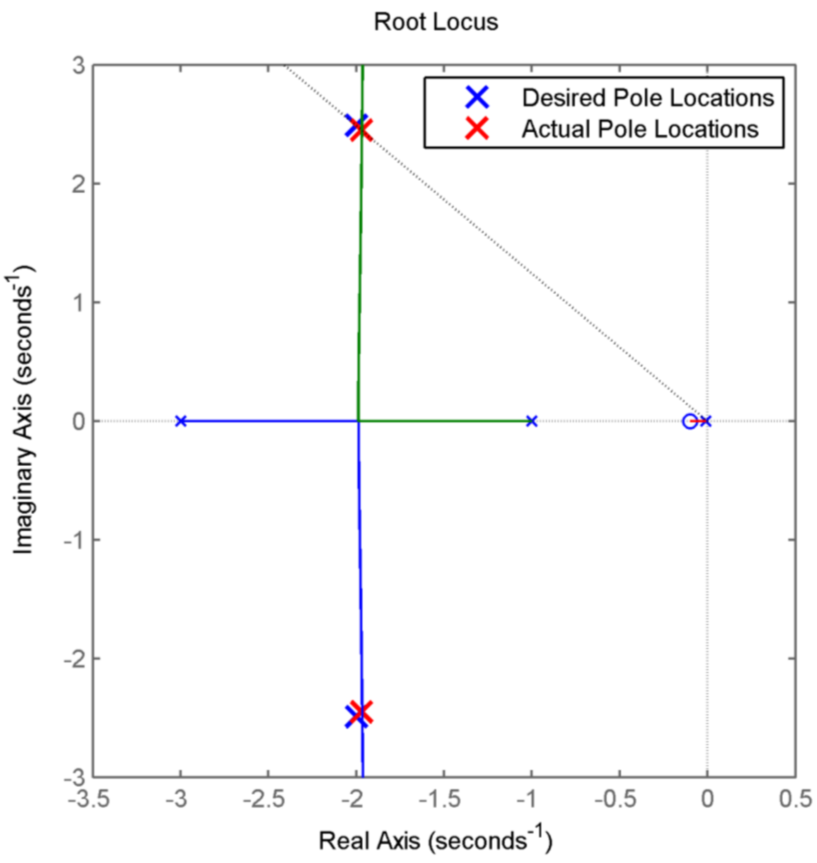
- Apply lag compensation to our previous example
  - Design for a 10x improvement of the position constant

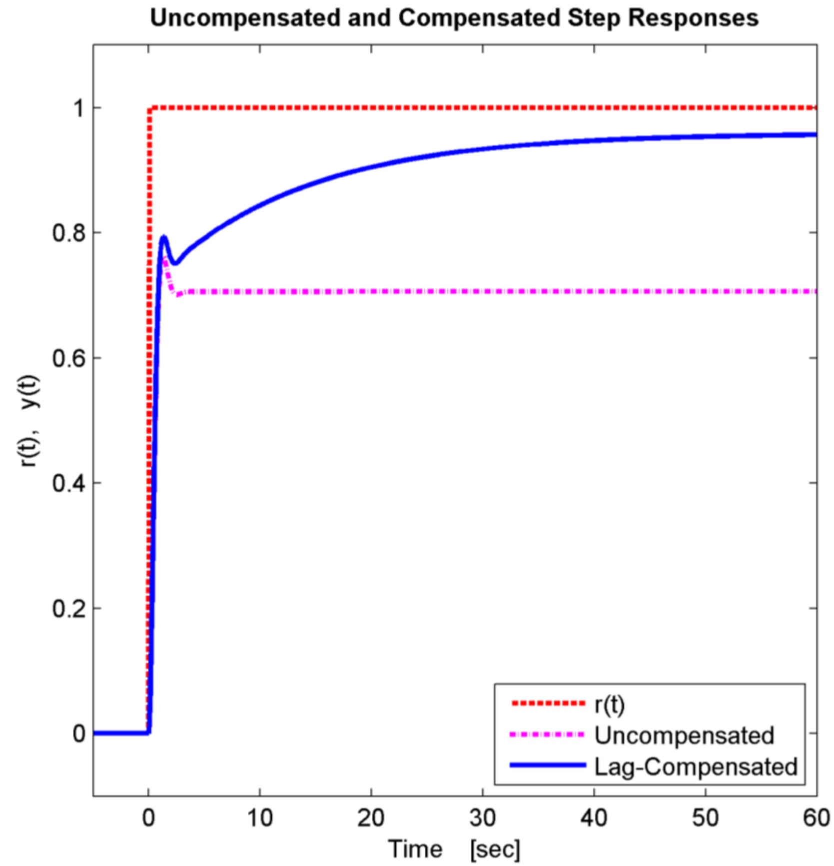


- Want $p_c \approx 0$ (relative to other poles)
  - Let $p_c = 0.01$
- Want a 10x improvement in K_p
  - $z_c = 10p_c = 0.1$
- Lag pole and zero differ by a factor of 10
  - Static error constant improved by a factor of 10
- Lag pole/zero are very close together relative to poles at $s = -1, -3$
  - Angular contributions nearly cancel
  - Transient response nearly unaffected

Lag Compensation – Example

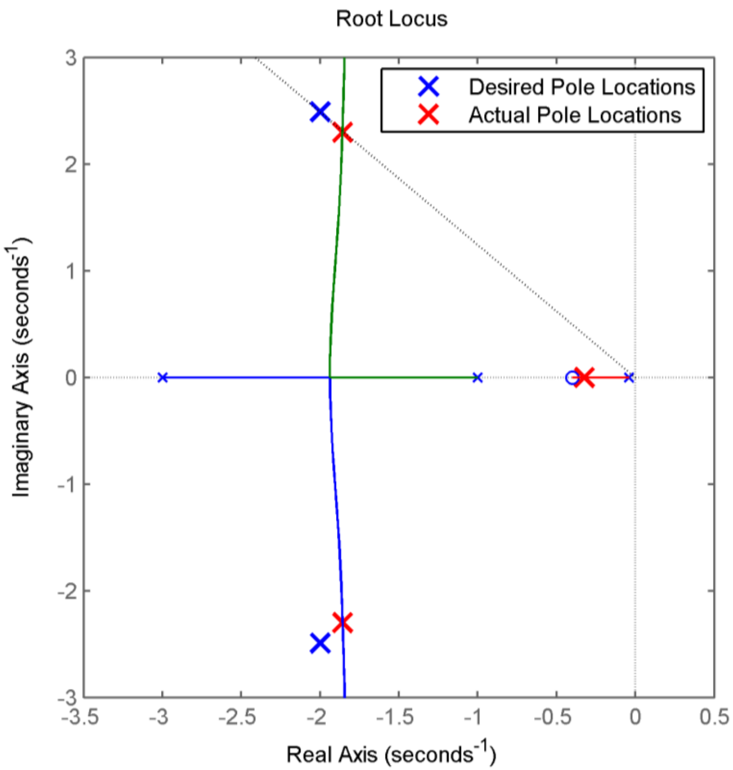
- Root locus and step response of lag-compensated system

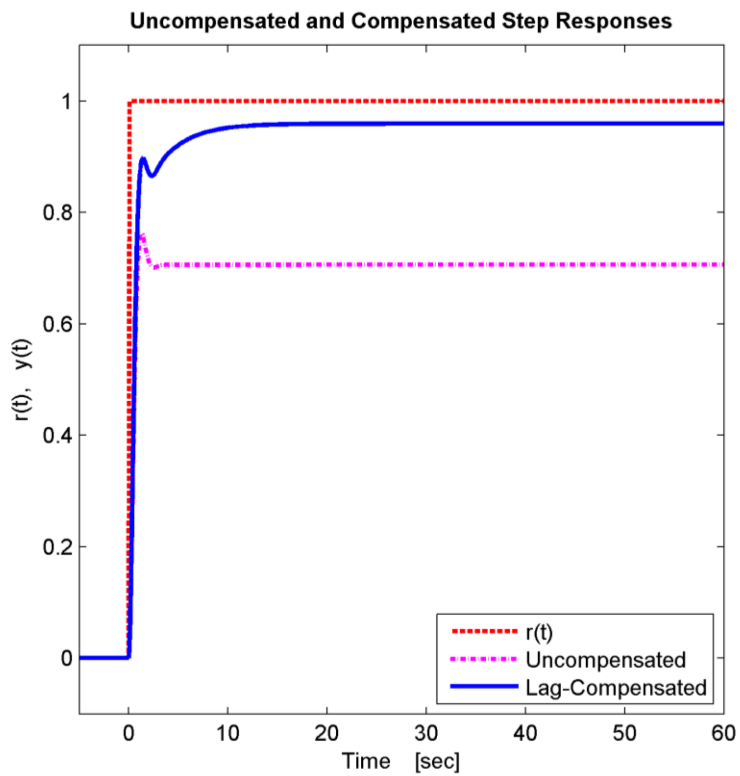




Lag Compensation – Example

- Now, let $z_c = 0.4$ and $p_c = 0.04$
  - 2nd-order poles moved more
  - Faster low-frequency closed-loop pole
  - Faster overall response





Lag Compensation – Summary

- ***Lag compensation***

$$D(s) = K\frac{(s+z_c)}{(s+p_c)}, \quad \text{where} \quad p_c < z_c$$

  - Controller adds a ***pole near the origin*** and a ***slightly-higher-frequency zero nearby***

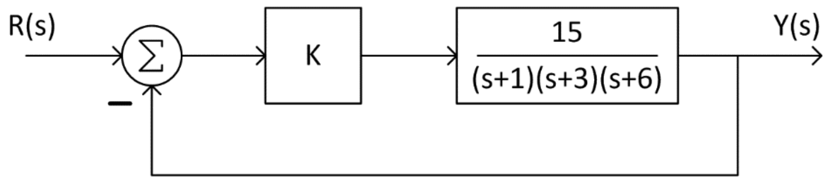
- ***Steady-state error improved*** by z_c/p_c

- Angle contributions from closely-spaced pole/zero nearly cancel
  - ***Transient response is nearly unchanged***

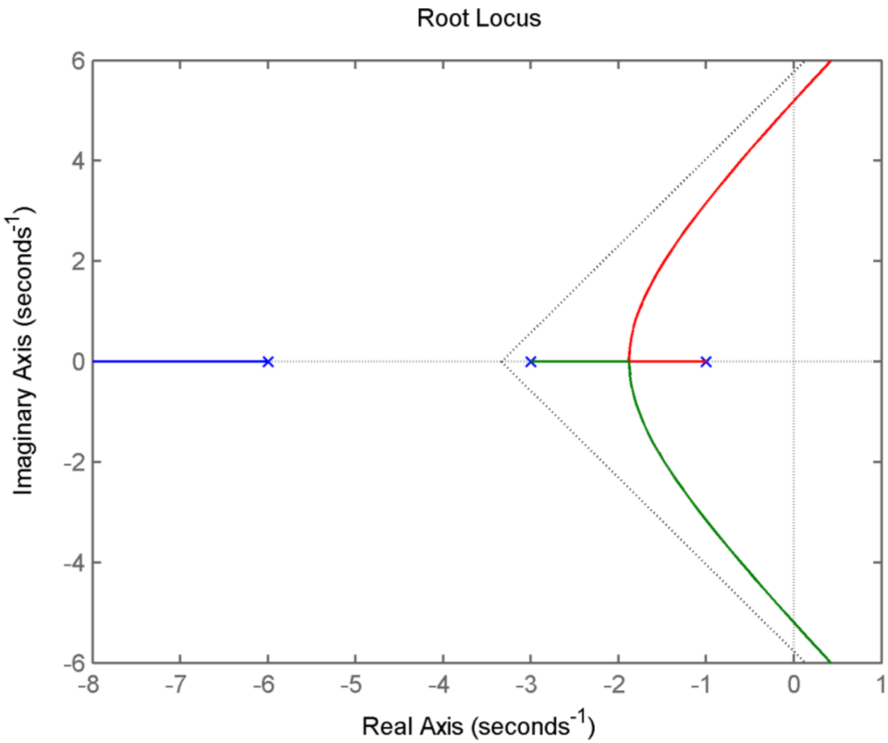
Improving Transient Response

Improving Transient Response

- Consider the following system

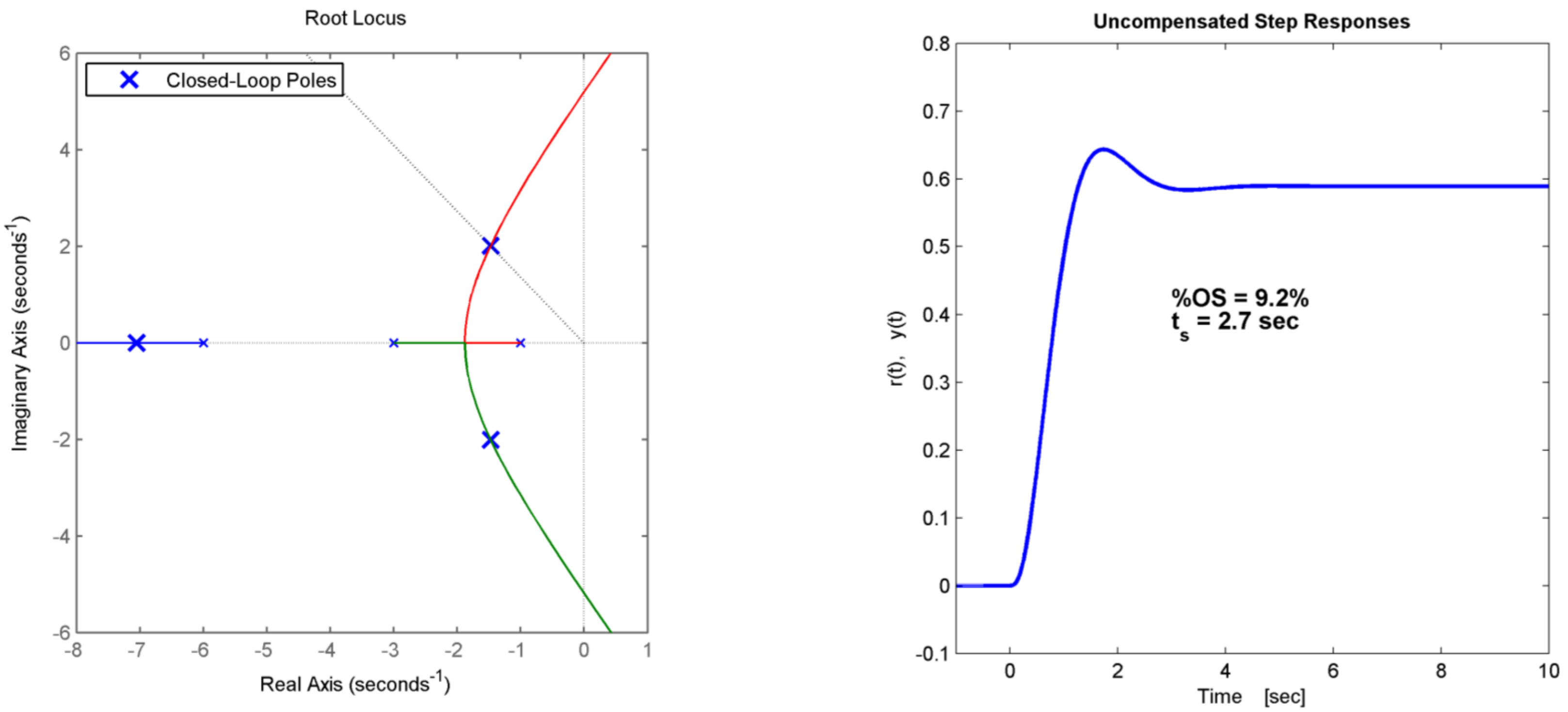


- Root locus:
  - Three asymptotes to C^{∞} at $60°$, $180°$, and $300°$
  - Real-axis breakaway point: $s = -1.88$
  - Locus crosses into RHP



Improving Transient Response

- Design proportional controller for 10% overshoot
  - $K = 1.72$



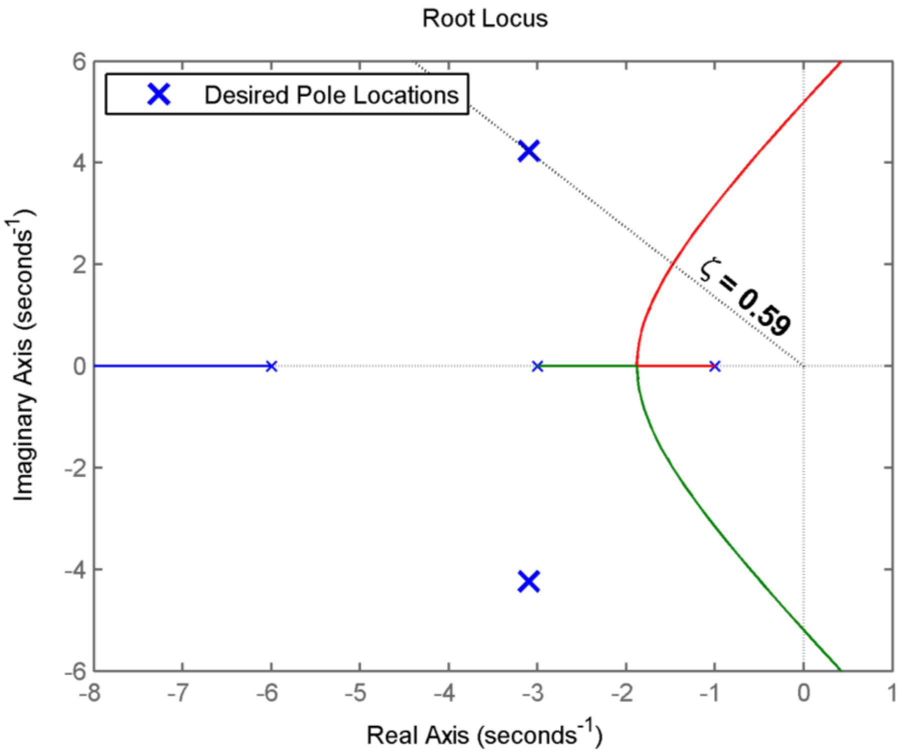
- Overshoot < 10% due to third pole

Improving Transient Response

- Now, decrease settling time to $t_s \approx 1.5\ sec$
  - Maintain same overshoot ($\zeta = 0.59$)

$$\sigma \approx \frac{4.6}{t_s} = 3.1$$

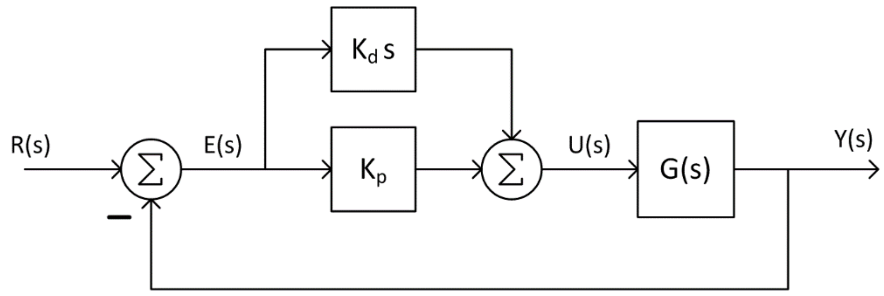
- Desired poles:
  - $s_{1,2} = -3.1 \pm j4.23$
  - Not on the locus
- Must add compensation to move the locus where we want it
  - ***Derivative compensation***



Ideal Derivative Compensation

Proportional-Derivative Compensation

- One way to improve transient response is to add the ***derivative of the error*** to the control input to the plant



- This is ***ideal derivative*** or ***proportional-derivative (PD) compensation***

$$U(s) = E(s)\left(K_p + K_d s\right) = K(s + z_c)E(s)$$

- Compensator transfer function:

$$D(s) = K(s + z_c)$$

- Compensator adds a single zero at $s = -z_c$

PD Compensation

- Compensator zero will change the root locus
  - Placement of the zero allows us to move the locus to place closed-loop poles where we want them
- One less asymptote to C^{∞}
  - $(n-m)$ decreased by one
- Asymptote origin changes

$$\sigma_a = \frac{\Sigma p_i - \Sigma z_i}{n-m}$$

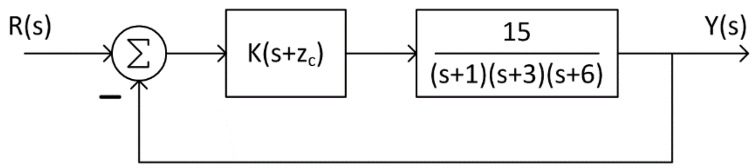
  - As z_c increases (moves left), σ_a moves right, toward the origin
  - As z_c decreases (moves right), σ_a moves further into the LHP

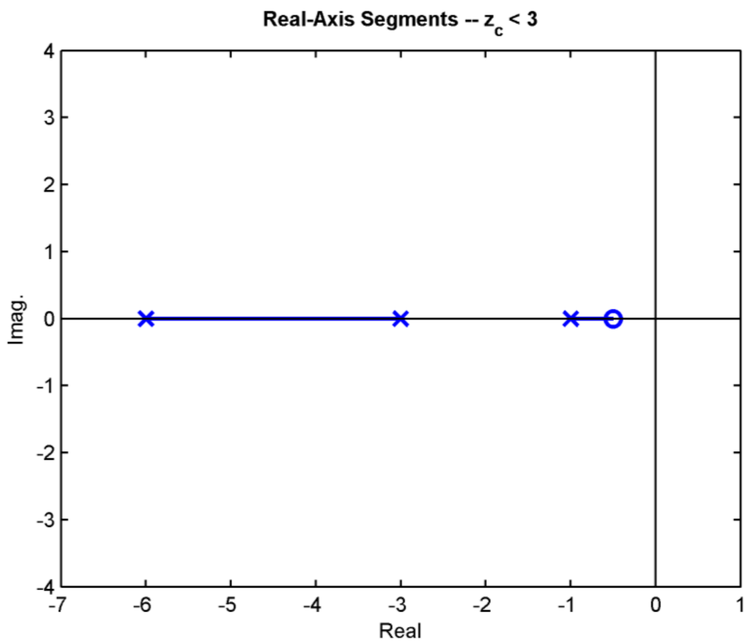
PD Compensation

- Derivative compensation allows us to speed up the closed-loop response
  - Control signal proportional to (in part) the derivative of the error
- When the reference, $r(t)$, changes quickly:
  - Error, $e(t)$, changes quickly
  - Derivative of the error, $\dot{e}(t)$, is large
  - Control input, $u(t)$, may be large
- Derivative compensation ***anticipates future error*** and compensates for it

PD Compensation – Example 1

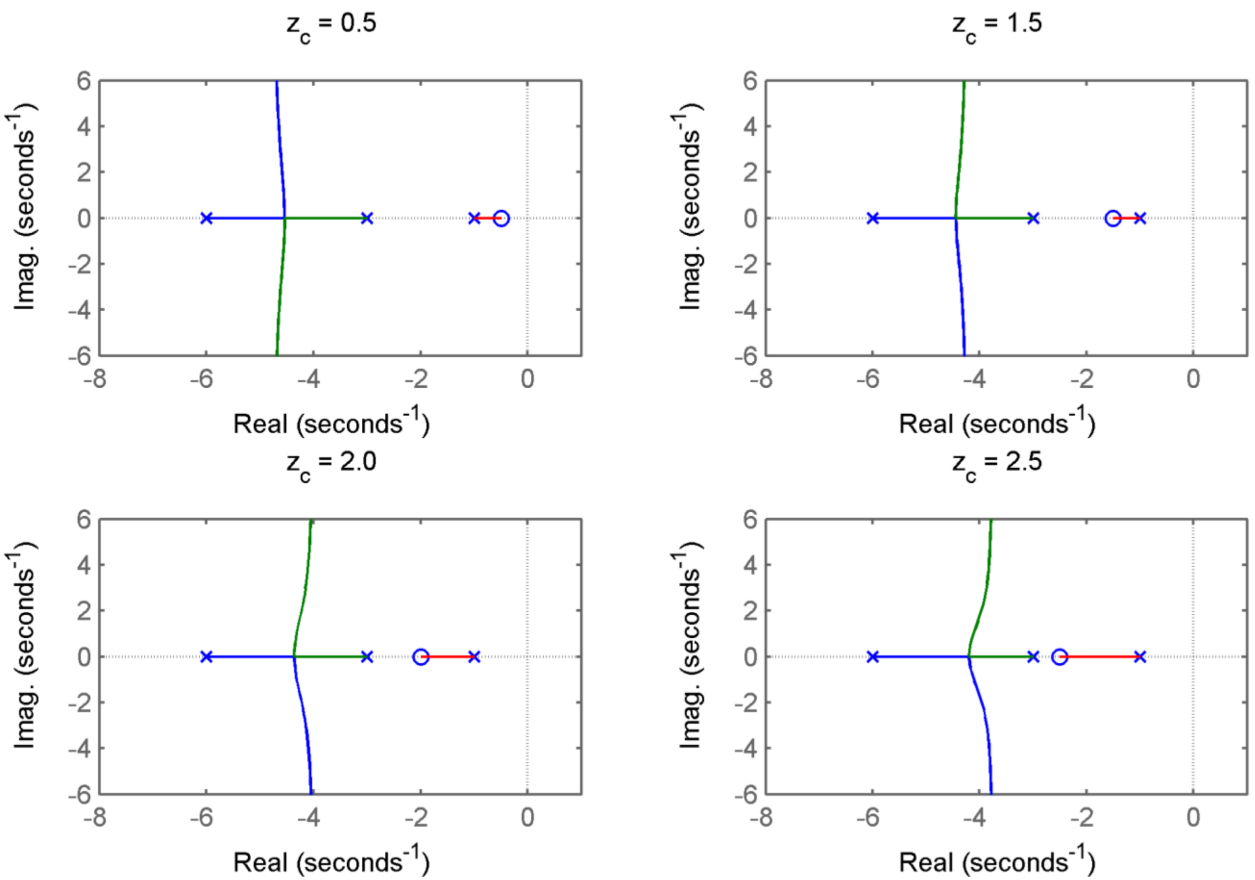
- Now add PD compensation to our example system
- Root locus depends on z_c
  - Let's first assume $z_c < 3$
- Two real-axis segments
  - $-6 \leq s \leq -3$
  - Between pole at -1 and z_c
- Two asymptotes to C^{∞}
  - $\theta_a = 90°, 270°$
  - $\sigma_a = \frac{z_c - 10}{2}$
  - As z_c varies from $0 \ldots 3$, σ_a varies from $-5 \ldots -3.5$
- Breakaway point between $-6 \ldots -3$





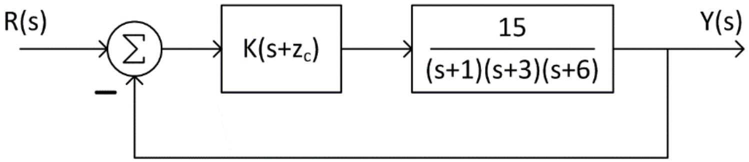
PD Compensation – Example 1

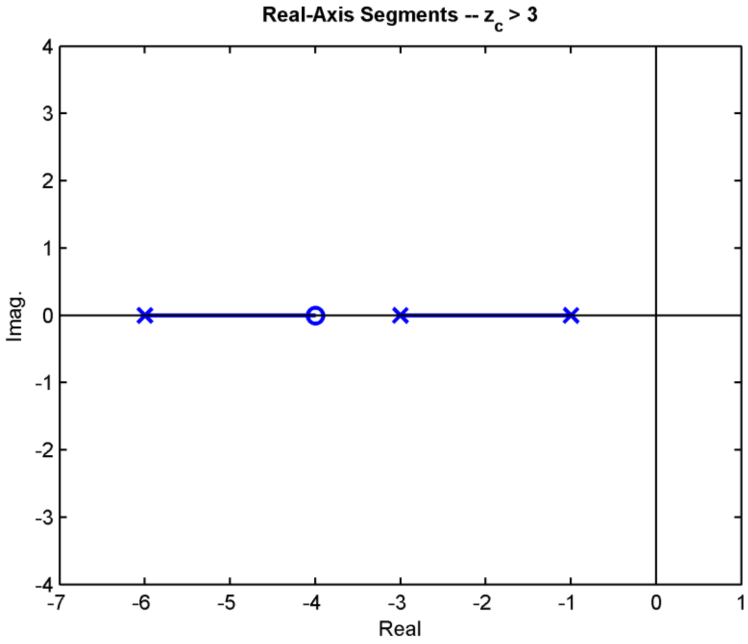
- As z_c moves to the left, σ_a moves to the right
  - Moving z_c allows us to move the locus



PD Compensation – Example 1

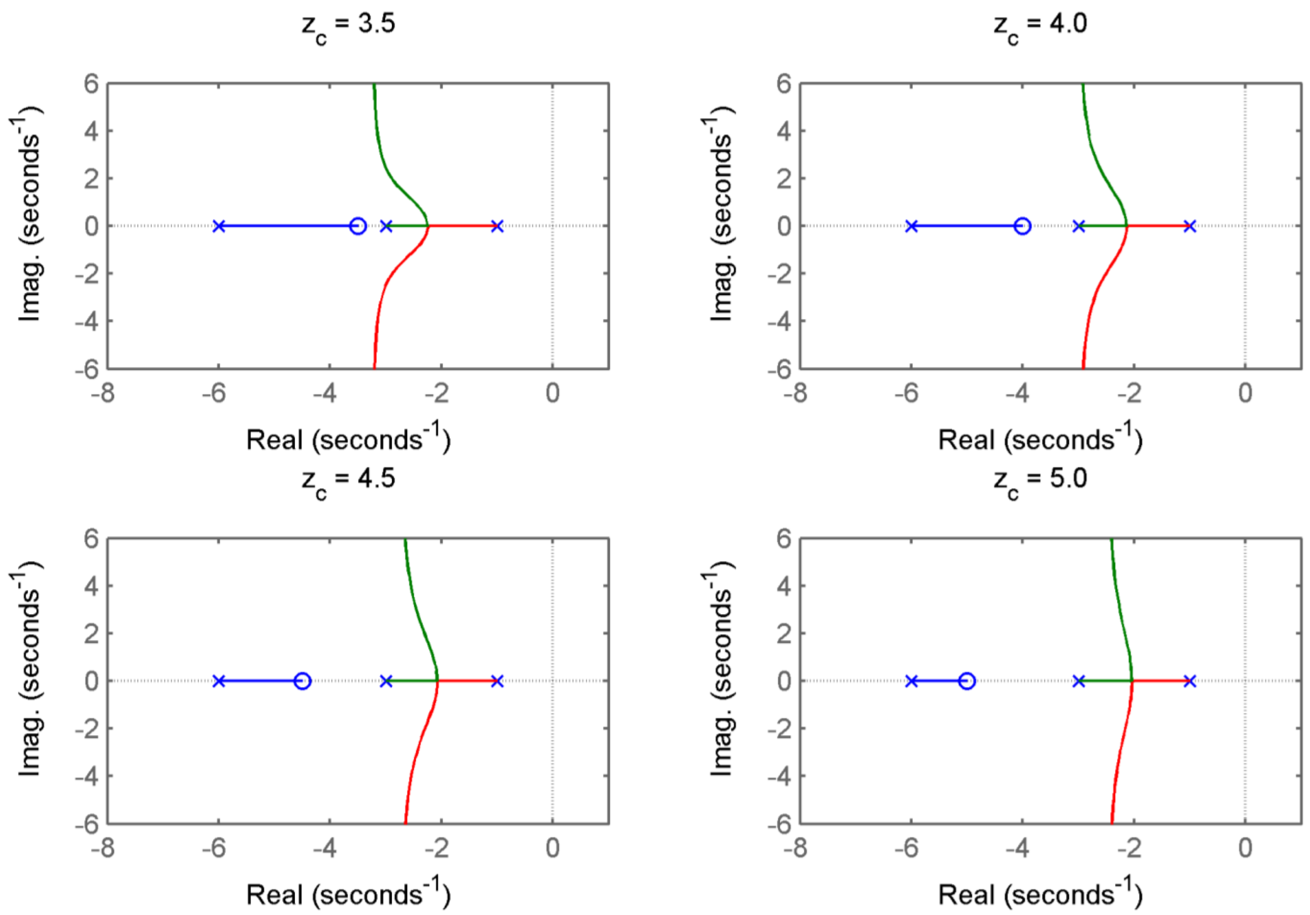
- Now move the zero further to the left: $z_c > 3$
- Still two real-axis segments
  - $-6 \leq s \leq -z_c$
  - $-3 \leq s \leq -1$
- Two asymptotes to C^{∞}
  - $\theta_a = 90°, 270°$
  - $\sigma_a = \frac{z_c - 10}{2}$
  - As z_c varies from $3 \ldots \infty$, σ_a varies from $-3.5 \ldots \infty$
- Breakaway point between $-3 \ldots -1$





PD Compensation – Example 1

- Asymptote origin continues to move to the right



PD Compensation – Calculating z_c

- For this particular system, we've seen:
  - Additional zero decreased the number of asymptotes to C^∞ by one
  - A stabilizing effect – locus does not cross into the RHP
  - Adjusting z_c allows us to move the asymptote origin left or right

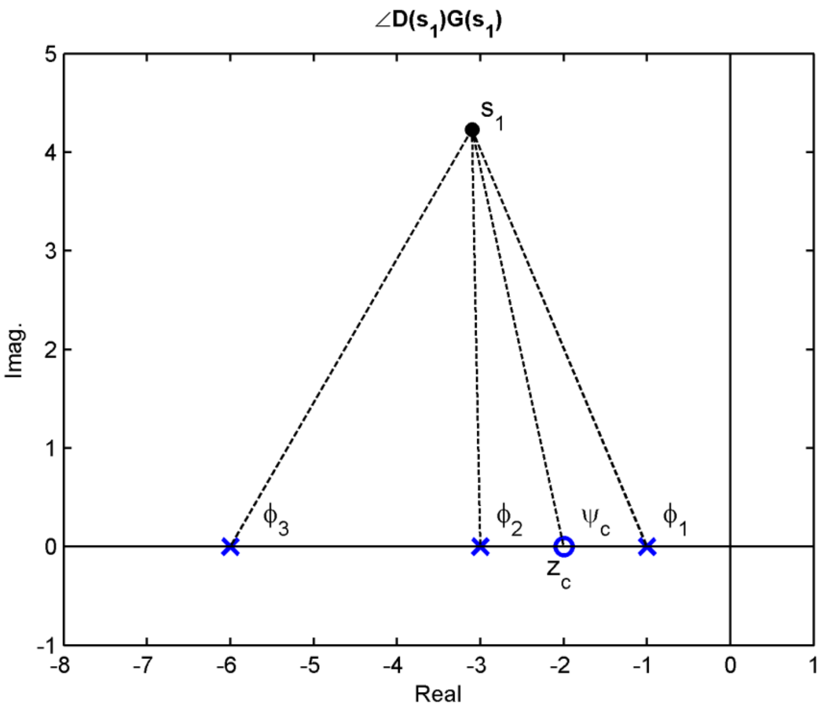
- Next, we'll determine exactly where to place z_c to place the closed-loop poles where we want them

PD Compensation – Example 2

- Desired 2nd-order poles: $s_{1,2} = -3.1 \pm j4.23$
  - Calculate required value for z_c such that these points are on the locus

- Must satisfy the angle criterion

$$\angle D(s_1)G(s_1) = 180°$$
$$\angle D(s_1)G(s_1) = \psi_c - \phi_1 - \phi_2 - \phi_3$$
$$\psi_c = 180° + \phi_1 + \phi_2 + \phi_3$$
$$\phi_1 = 116.4°$$
$$\phi_2 = 91.35°$$
$$\phi_3 = 55.57°$$



- The required angle from z_c:

$$\psi_c = 83.3°$$

- Next, determine z_c

PD Compensation – Example 2

- Compensator zero, z_c, must contribute $\psi_c = 83.3°$ at $s_{1,2} = -3.1 \pm j4.23$
- Calculate the required value of z_c

$$\psi_c = \angle(s_1 + z_c) = \angle(-3.1 + j4.23 + z_c) = 83.3°$$

$$\psi_c = \tan^{-1}\left(\frac{4.23}{z_c - 3.1}\right)$$

$$\tan(\psi_c) = \frac{4.23}{z_c - 3.1}$$

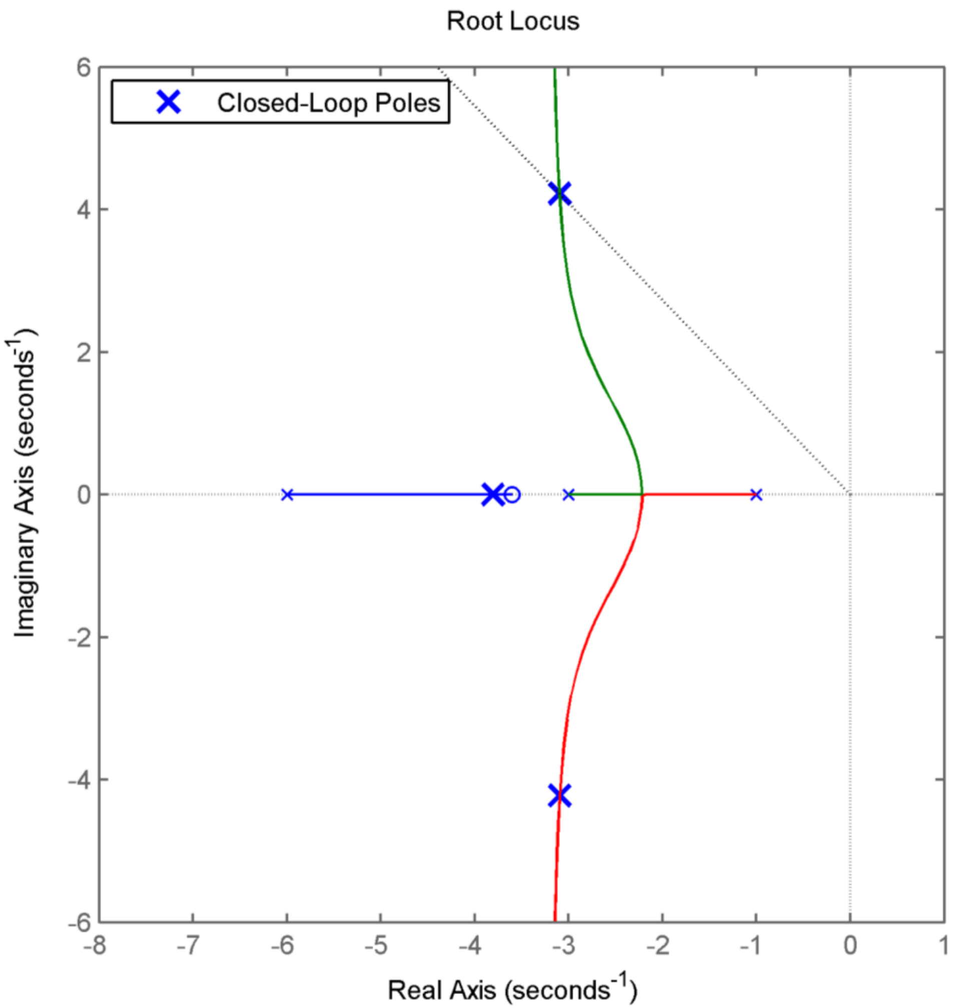
$$z_c = \frac{4.23}{\tan(\psi_c)} + 3.1 = \frac{4.23}{\tan(83.3°)} + 3.1$$

- The required compensator zero:

$$z_c = 3.6$$

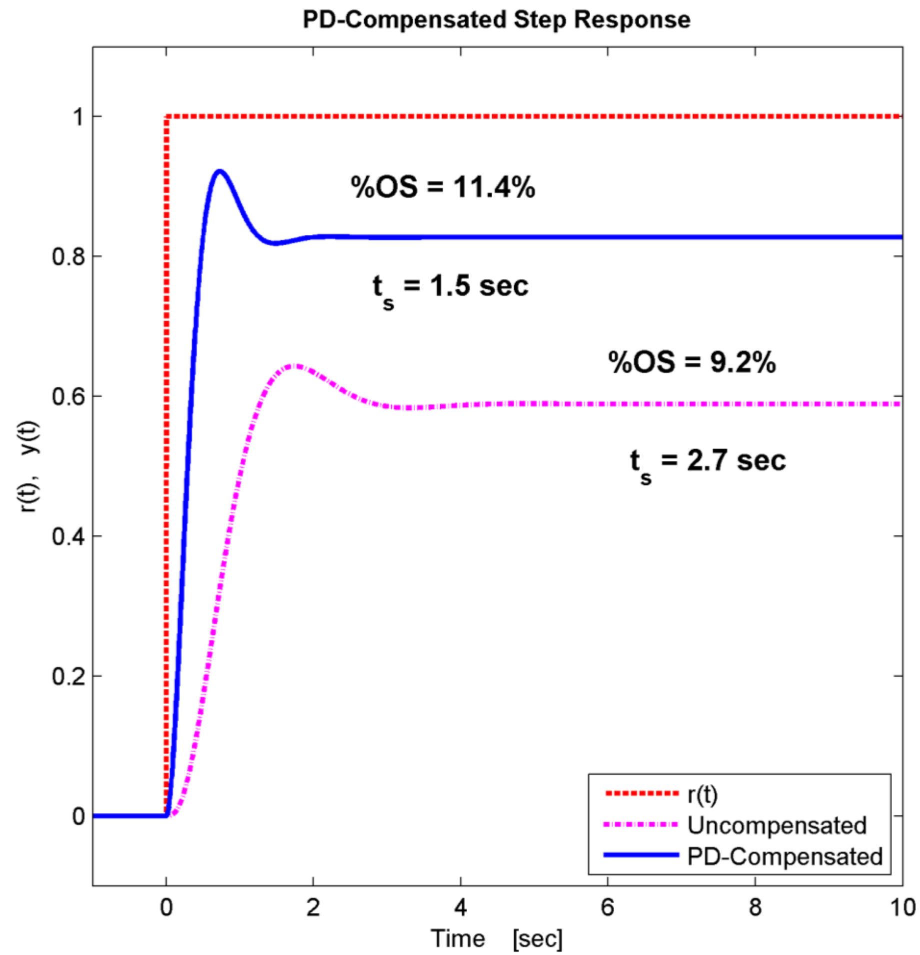
PD Compensation – Example 2

- Locus passes through desired points
- Closed-loop poles at $s = -3.1 \pm j4.23$ for $K = 1.6$
- Third closed-loop pole at $s = -3.8$
  - Close to zero at $s = -3.6$
  - 2nd-order approximation likely justified



PD Compensation – Example 2

- Settling time reduced, as desired
- Overshoot is a little higher than 10%
  - Higher order pole and zero do not entirely cancel
  - Iterate to further refine performance, if desired



PD Compensation – Summary

- ***PD compensation***

$$D(s) = K(s + z_c)$$

  - Controller adds a ***single zero***

- Angular contribution from the compensator zero allows the root locus to be modified

- Calculate z_c to satisfy the angle criterion at desired closed-loop pole locations
  - Use magnitude criterion or plot root locus to determine required gain

Lead Compensation

Sensor Noise

- Feedback control requires measurement of a system's output with some type of sensor
  - Inherently noisy
  - Measurement noise tends to be broadband in nature
    - I.e., includes energy at high frequencies
  - High-frequency signal components change rapidly
    - Large time derivatives
  - Derivative (PD) compensation amplifies measurement noise
- An alternative is ***lead compensation***
  - Amplification of sensor noise is reduced

Lead Compensation

- PD compensation utilizes an ideal differentiator
  - ***Amplifies sensor noise***
  - Active circuitry (opamp) required for analog implementation
- An alternative to PD compensation is ***lead compensation***
  - Compensator adds ***one zero*** and ***a higher-frequency pole***

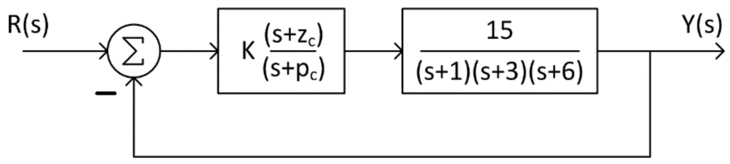
$$D(s) = K\frac{(s+z_c)}{(s+p_c)}, \quad \text{where} \quad p_c > z_c$$

  - Pole can be far enough removed to have little impact on 2nd-order dynamics
  - Additional high-frequency pole ***reduces amplification of noise***
  - Analog implementation realizable with passive components (resistors and capacitors)

Lead Compensation – Example

- Apply lead compensation to our previous example system
- Desired closed-loop poles:

$$s_{1,2} = -3.1 \pm j4.23$$



- Angle criterion must be satisfied at s_1

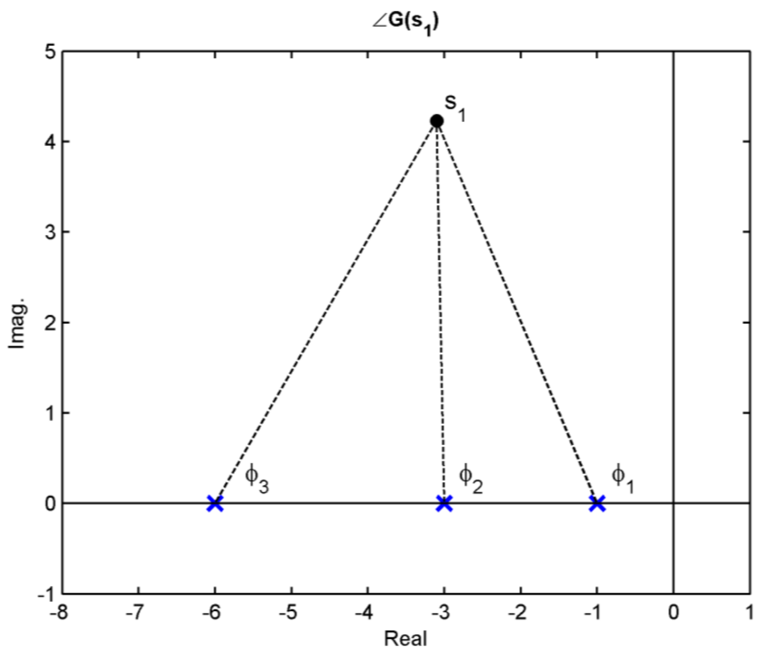
$$\angle D(s_1)G(s_1) = 180°$$

$$\angle D(s_1) + \angle G(s_1) = 180°$$

$$\angle D(s_1) = 180° - \angle G(s_1)$$

$$\angle G(s_1) = -(\phi_1 + \phi_2 + \phi_3)$$

$$\angle G(s_1) = -263.3°$$



- Required net angle contribution from the compensator:

$$\angle D(s_1) = 443.3° = 83.3°$$

Lead Compensation – Example

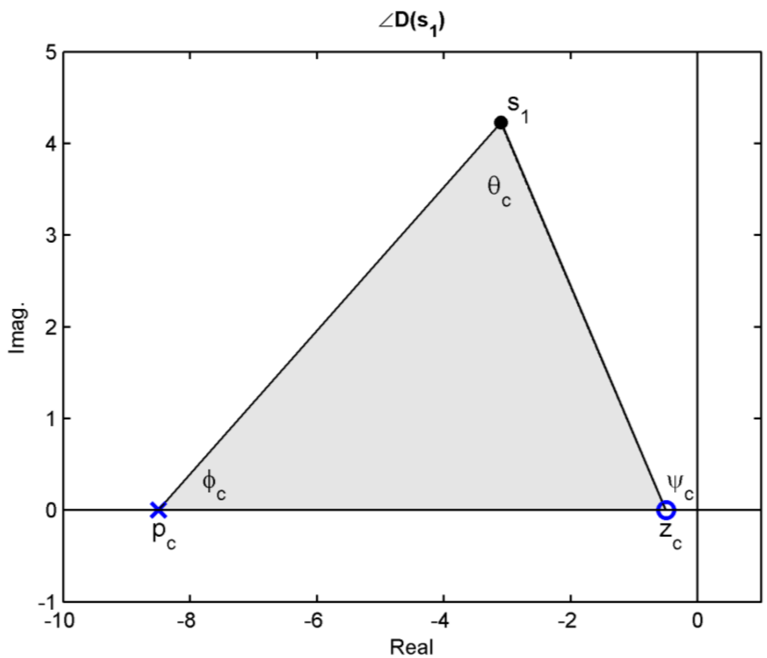
- For s_1 to be on the locus, we need $\angle D(s_1) = 83.3°$
  - Zero contributes a positive angle
  - Higher-frequency pole contributes a smaller negative angle
  - Net angular contribution will be positive, as required:

$$\angle D(s_1) = \angle(s_1 + z_c) - \angle(s_1 + p_c) = 83.3°$$

- Compensator angle is the angle of the ray from s_1 through z_c and p_c

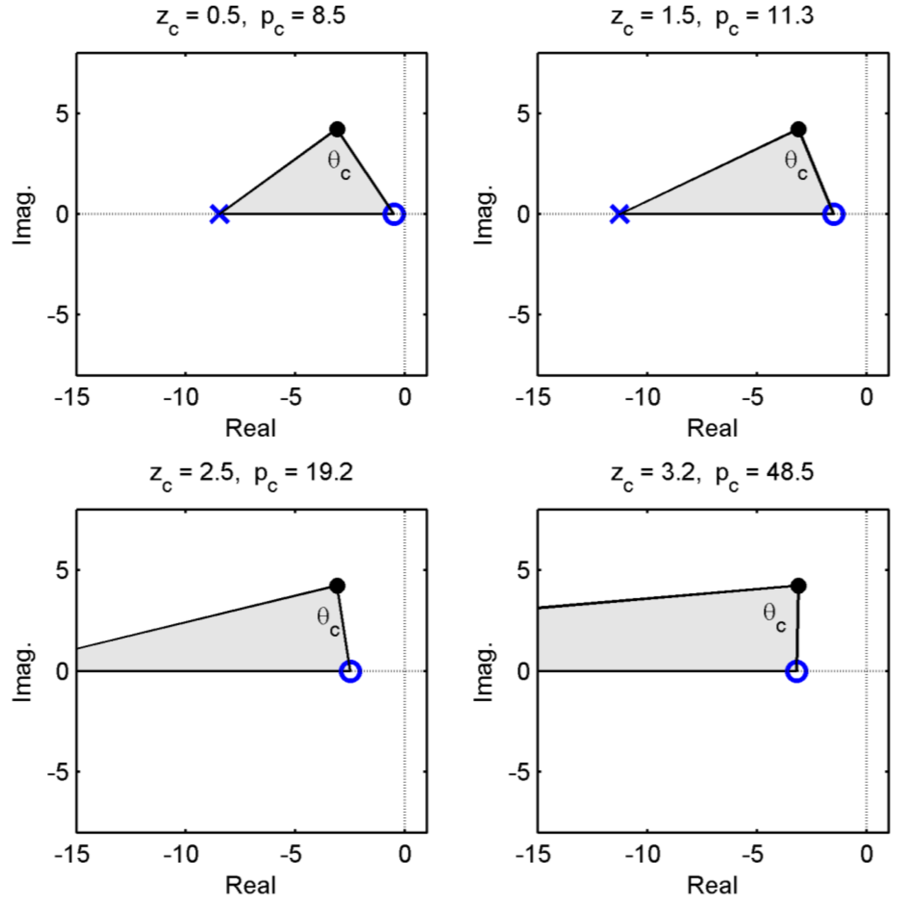
$$\angle D(s_1) = \theta_c = 88.3°$$

- Infinite combinations of z_c and p_c will provide the required θ_c



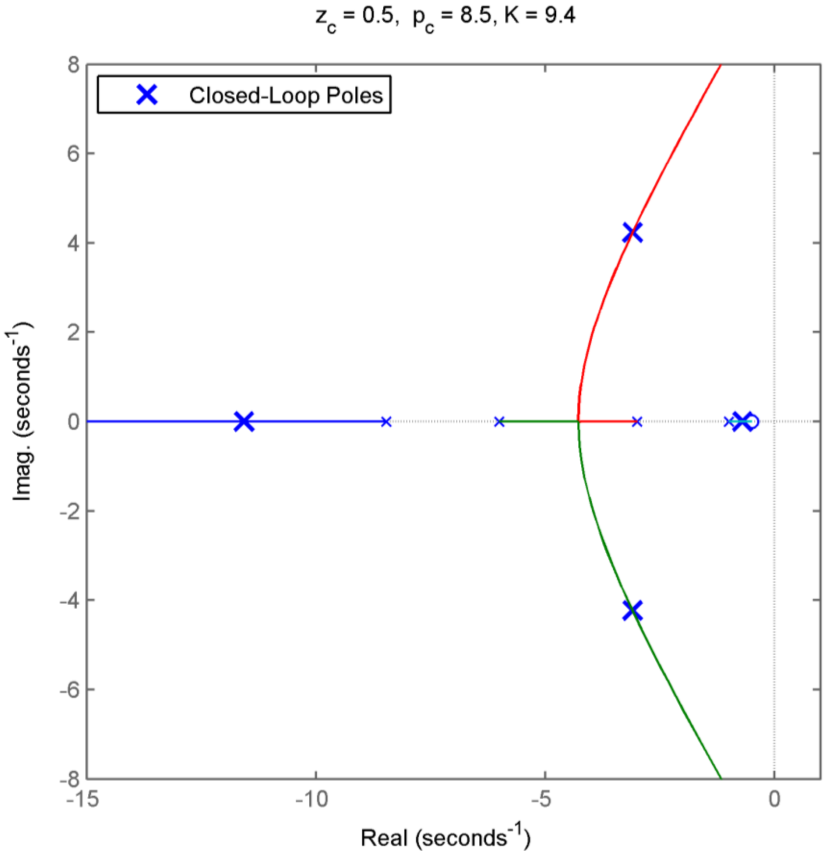
Lead Compensation – Example

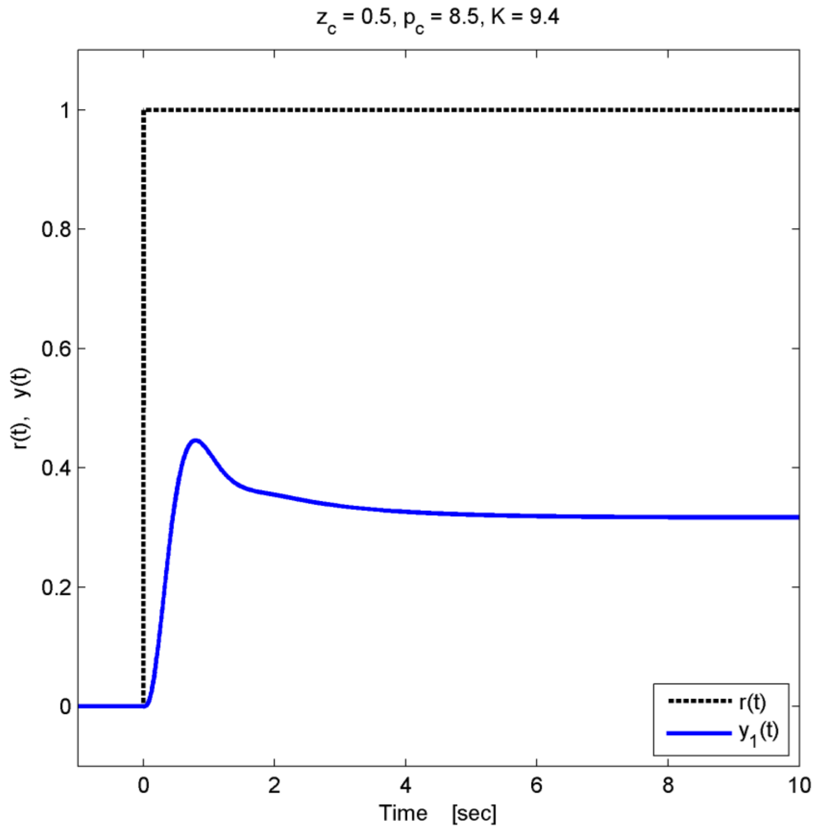
- An infinite number of possible z_c/p_c combinations
  - All provide $\theta_c = 83.3°$
  - Different static error constants
  - Different required gains
  - Different location of *other* closed-loop poles
- No real rule for how to select z_c and p_c
- Some options:
  - Set p_c as high as acceptable given noise requirements
  - Place z_c below or slightly left of the desired poles



Lead Compensation – Example

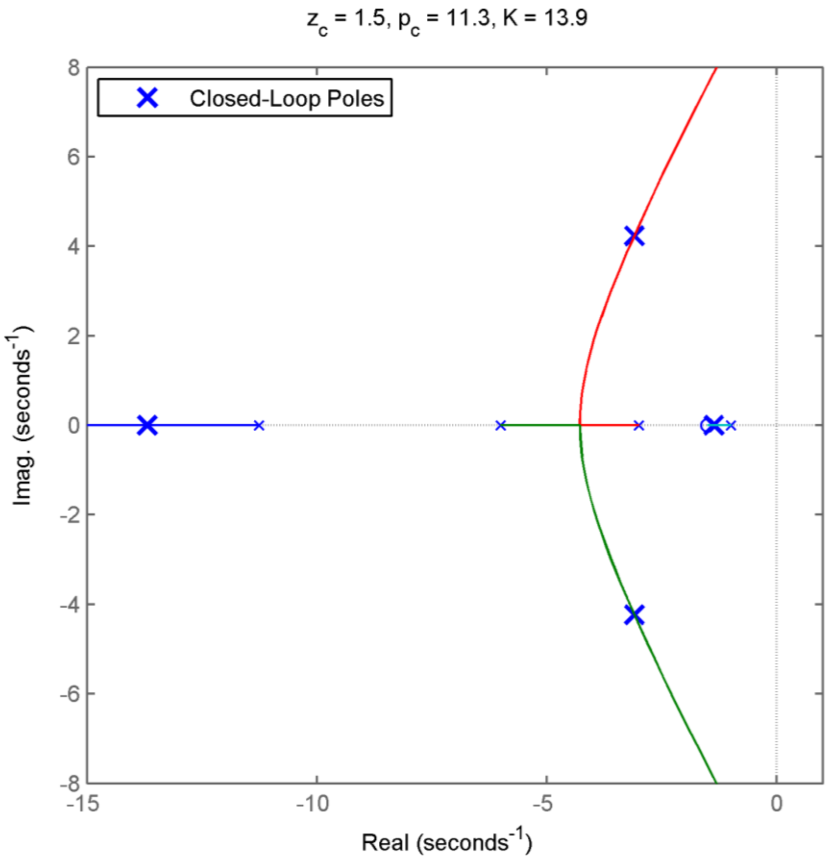
- Root locus and Step response for $z_c = 0.5$, $p_c = 8.5$
  - Lower-frequency pole/zero do not adequately cancel

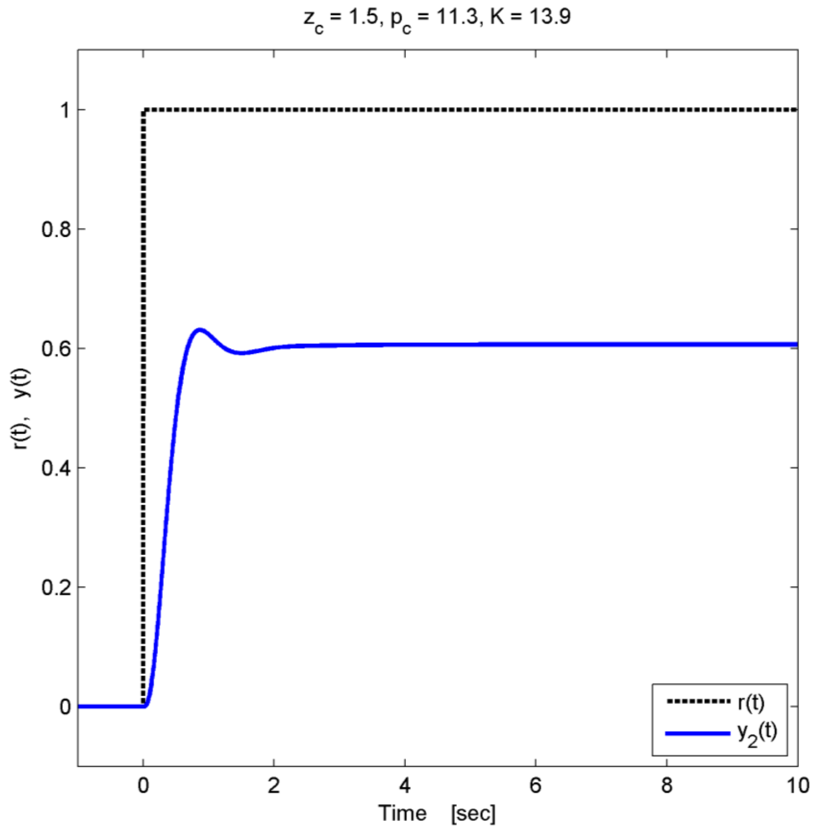




Lead Compensation – Example

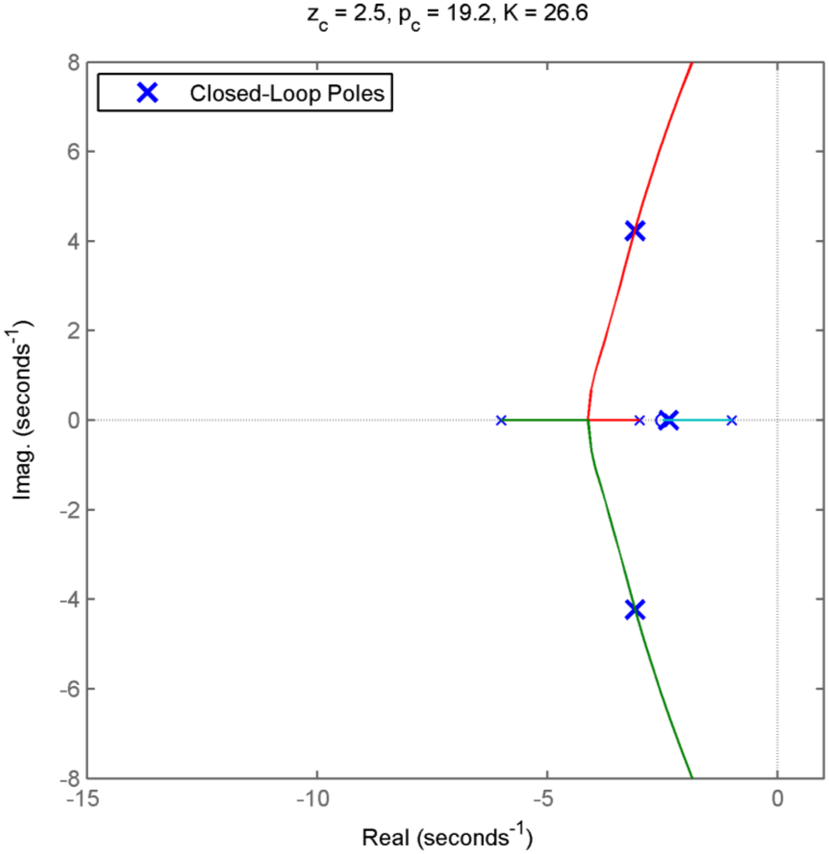
- Root locus and Step response for $z_c = 1.5, p_c = 11.3$
  - Effect of lower-frequency pole/zero reduced

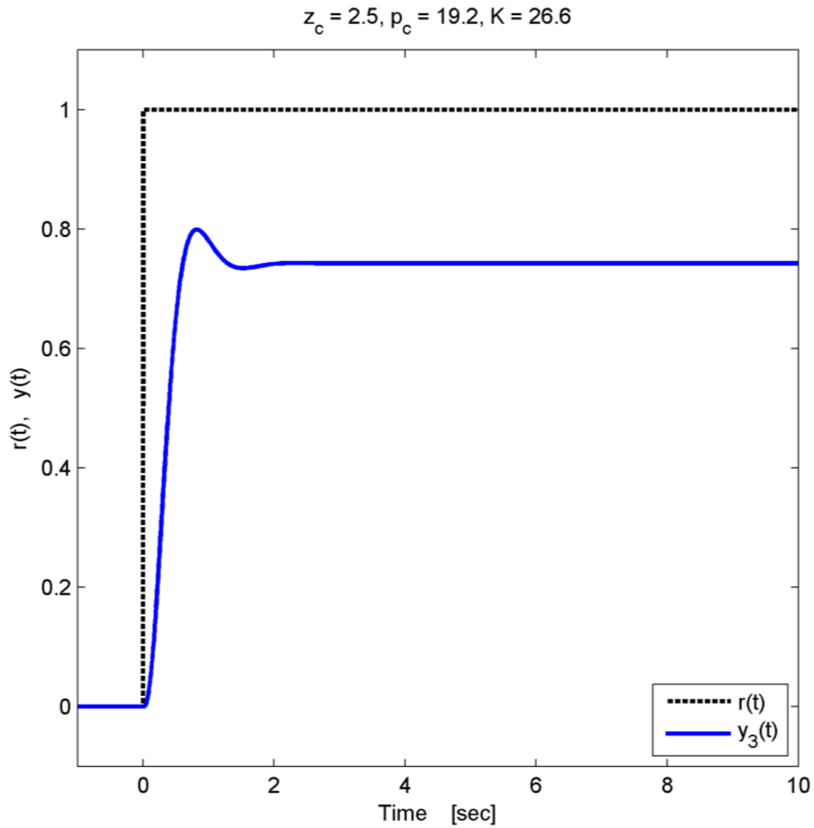




Lead Compensation – Example

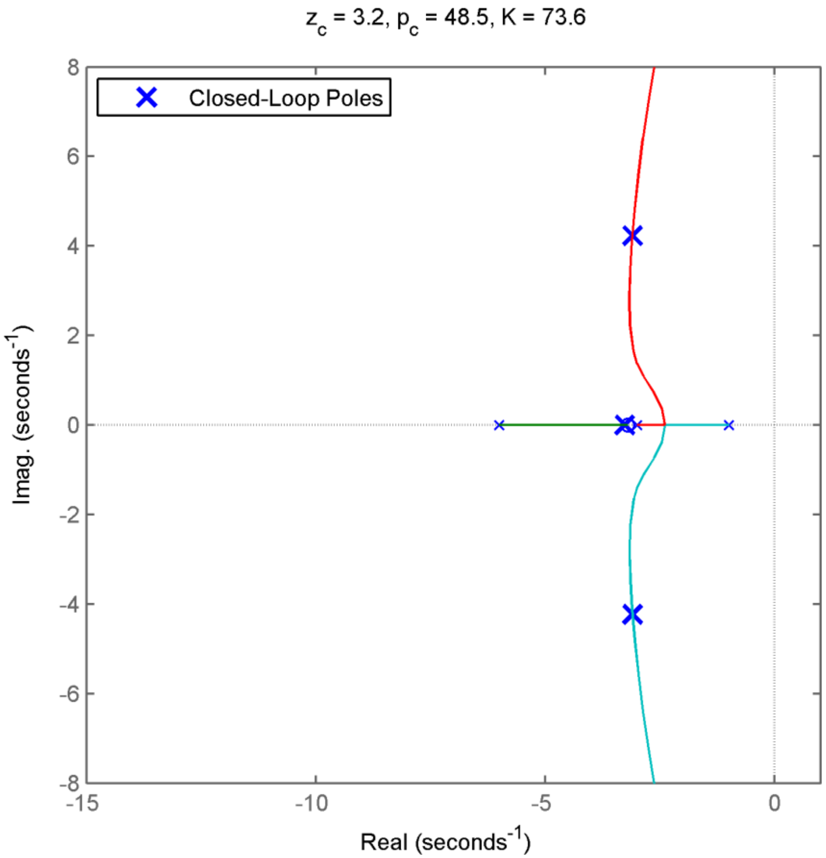
- Root locus and Step response for $z_c = 2.5$, $p_c = 19.2$
  - Lower-frequency pole/zero very nearly cancel

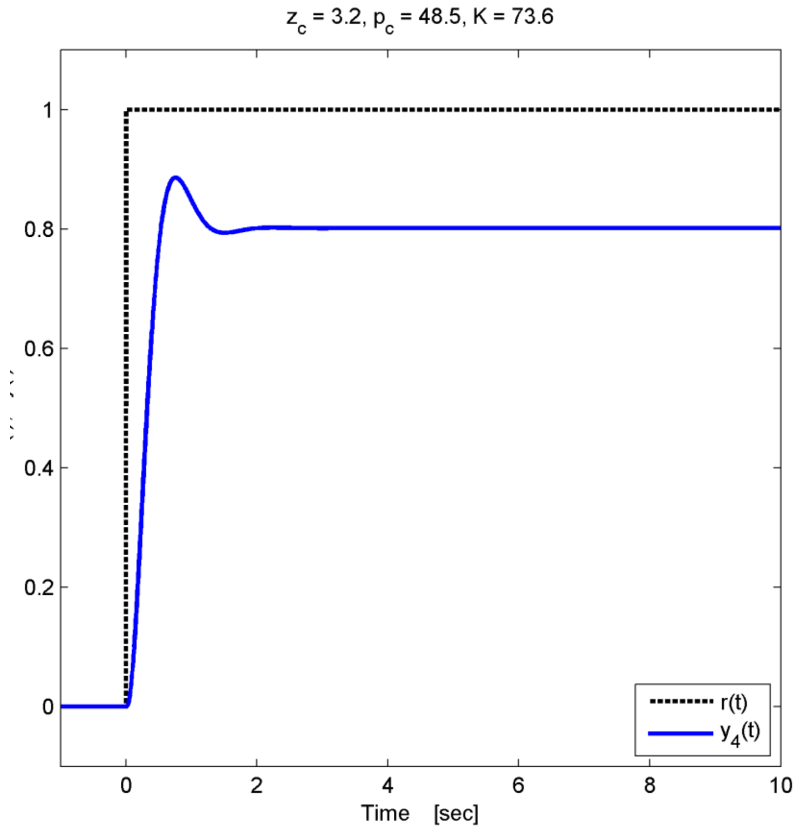




Lead Compensation – Example

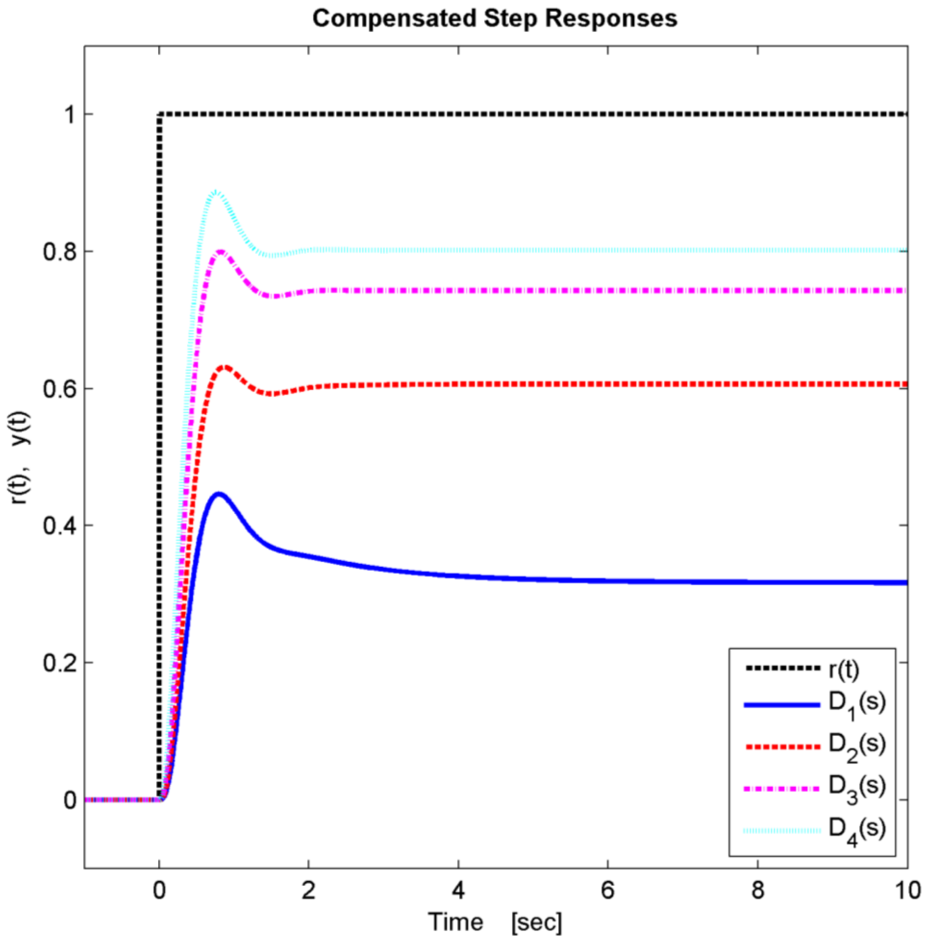
- Root locus and Step response for $z_c = 3.2, p_c = 33.1$
  - Higher-frequency pole/zero almost completely cancel





Lead Compensation – Example

- Here, $z_c = 2.5$ or $z_c = 3.2$ are good choices
- Steady-state error varies
  - Error depends on gain required for each lead implementation



Lead Compensation – Summary

- ***Lead compensation***

$$D(s) = K\frac{(s + z_c)}{(s + p_c)}, \quad \text{where} \quad p_c > z_c$$

  - Controller adds a ***lower-frequency zero*** and a ***higher-frequency pole***
- Net angular contribution from the compensator zero and pole allows the root locus to be modified
  - Allows for ***transient response improvement***
- Infinite number of possible z_c/p_c combinations to satisfy the angle criterion at the design point

Improving Error and Transient Response

Improving Error and Transient Response

- PI (or lag) control improves ***steady-state error***
- PD (or lead) control can improve ***transient response***
- Using both together can improve both error and dynamic performance
  - PD or lead compensation to achieve desired transient response
  - PI or lag compensation to achieve desired steady-state error
- Next, we'll look at two types of compensators:
  - ***Proportional-integral-derivative*** (PID) compensator
  - ***Lead-lag*** compensator

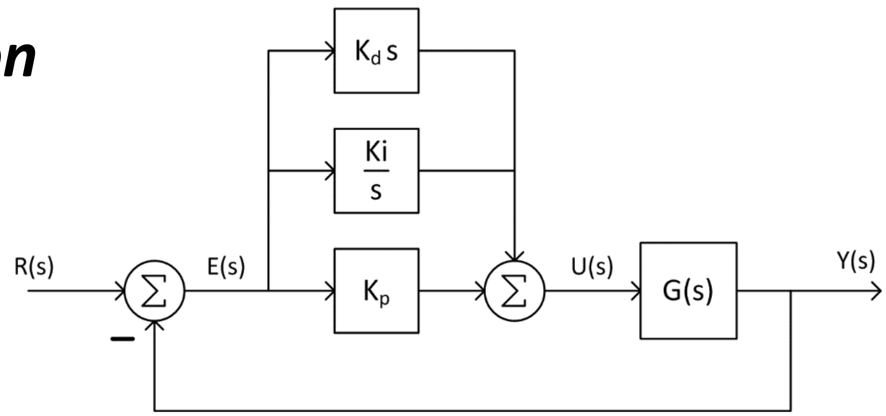
Improving Error and Transient Response

- Two possible approaches to the design procedure:
  1. First design for transient response, then design for steady-state error
     - Response may be slowed slightly in the process of improving steady-state error
  2. First design for steady-state error, then design for transient response
     - Steady-state error may be affected
- In either case, iteration is typically necessary
- We'll follow the first approach, as does the text

PID Compensation

Proportional-Integral-Derivative Compensation

- ***Proportional-integral-derivative (PID) compensation***
  - Combines PI and PD compensation
  - PD compensation adjusts transient response
  - PI compensation improves steady-state error



- Controller transfer function:

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

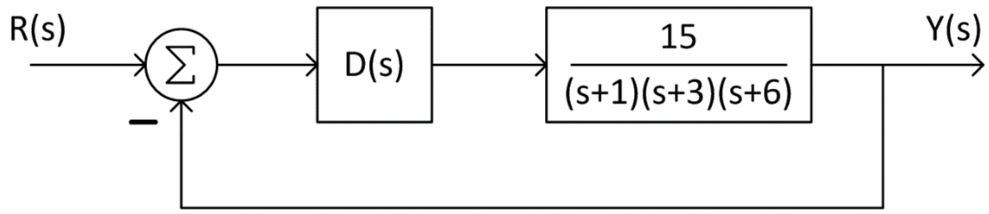
- ***Two zeros and a pole at the origin***
  - Pole/zero at/near the origin determined through PI compensator design
  - Second zero location determined through PD compensator design

PID Design Procedure

- **PID compensator design procedure:**

1. Determine closed-loop pole location to provide desired transient response
2. Design PD controller (zero location and gain) to place closed-loop poles as desired
3. Simulate the PD-compensated system, iterate if necessary
4. Design a PI controller, add to the PD-compensated system, and determine the gain required to maintain desired dominant pole locations
5. Determine PID parameters: K_p, K_i, and K_d
6. Simulate the PID-compensated system and iterate, if necessary

PID Compensation – Example



- Design PID compensation to satisfy the following specifications:
  - $t_s \approx 2\ sec$
  - $\%OS \approx 20\%$
  - Zero steady-state error to a constant reference
- First, design PD compensator to satisfy dynamic specifications

PID Compensation – Example

- Calculate desired closed-loop pole locations

$$\sigma \approx \frac{4.6}{t_s} = 2.3$$

$$\zeta = -\frac{\ln(0.2)}{\sqrt{\pi^2 + \ln^2(0.2)}} = 0.46$$

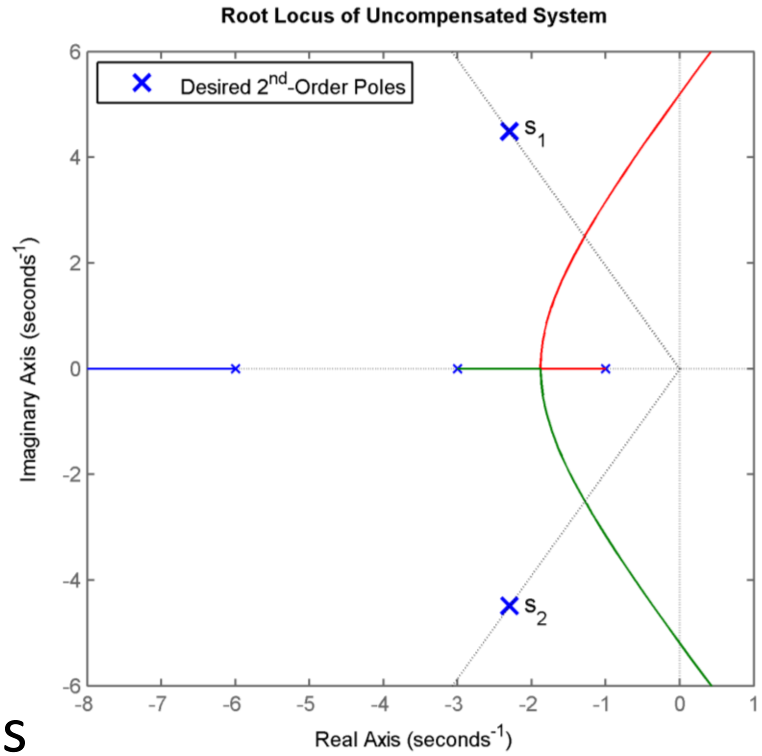
$$\omega_d = \frac{\sigma}{\zeta}\sqrt{1-\zeta^2} = \frac{2.3}{0.46}\sqrt{1-0.46^2}$$

$$\omega_d = 4.49$$

- Desired 2nd-order poles:

$$s_{1,2} = -2.3 \pm j4.49$$

- Uncompensated root locus does not pass through the desired poles
  - Gain adjustment not sufficient
  - Compensation required



PID Compensation – Example

- PD compensator design
- Determine the required angular contribution of the compensator zero to satisfy the angle criterion at s_1

$$\angle D_{pd}(s_1)G(s_1) = 180°$$

$$\angle D_{pd}(s_1) = 180° - \angle G(s_1)$$

$$\angle G(s_1) = -\phi_1 - \phi_2 - \phi_3$$

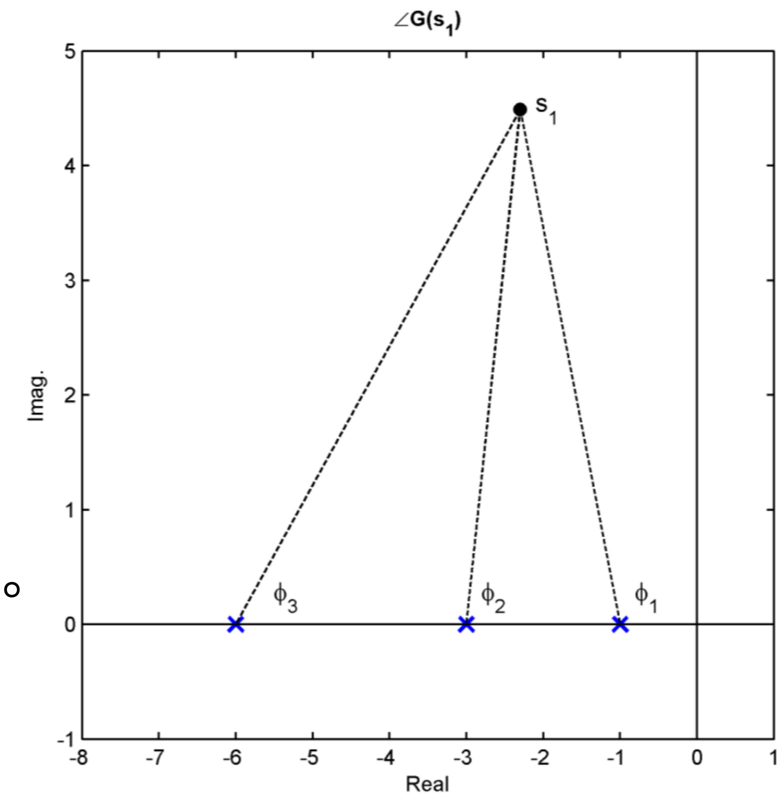
$$\phi_1 = \angle(s_1 + 1) = 106.15°$$

$$\phi_2 = \angle(s_1 + 3) = 81.14°$$

$$\phi_3 = \angle(s_1 + 6) = 50.51°$$

$$\angle G(s_1) = -237.8°$$

$$\angle D_{pd}(s_1) = 180° + 237.8° = 417.8°$$



- Required angle from PD zero

$$\psi_{pd} = 57.8°$$

PID Compensation – Example

- Use required compensator angle to place the PD zero, z_{pd}

$$\angle D_{pd}(s_1) = \angle\left(s_1 + z_{pd}\right)$$

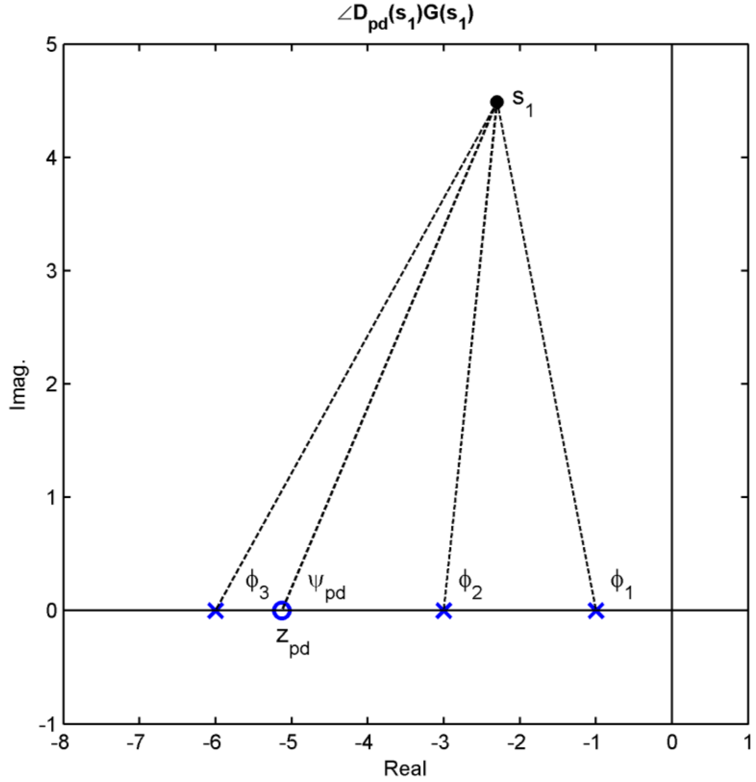
$$\angle D_{pd}(s_1) = \angle\left(-2.3 + z_{pd} + j4.49\right)$$

$$\tan\left(\psi_{pd}\right) = \frac{4.49}{z_{pd} - 2.3}$$

$$z_{pd} = \frac{4.49}{\tan(57.8°)} + 2.3 = 5.13$$

- PD compensator transfer function:

$$D_{pd}(s) = K(s + 5.13)$$



PID Compensation – Example

- PD-compensated root locus
- Determine required gain from MATLAB plot, or
- Apply the ***magnitude criterion:***

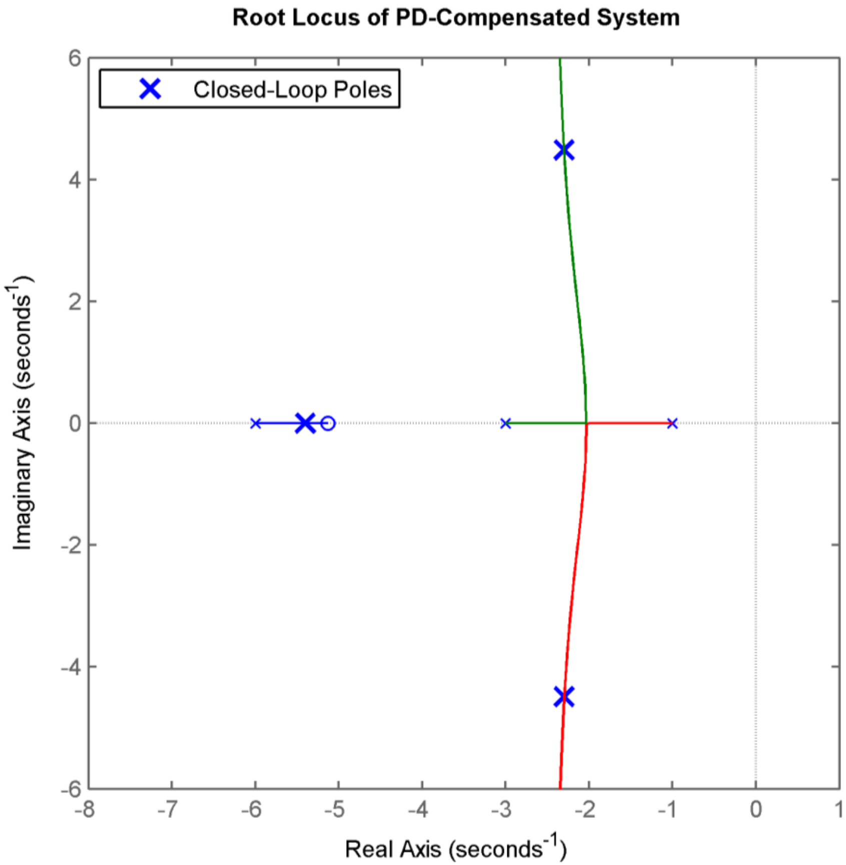
$$K = \left| \frac{1}{D_{pd}(s_1)G(s_1)} \right|$$

$$K = \left| \frac{(s_1+1)(s_1+3)(s_1+6)}{15(s_1+5.13)} \right|$$

$$K = 1.55$$

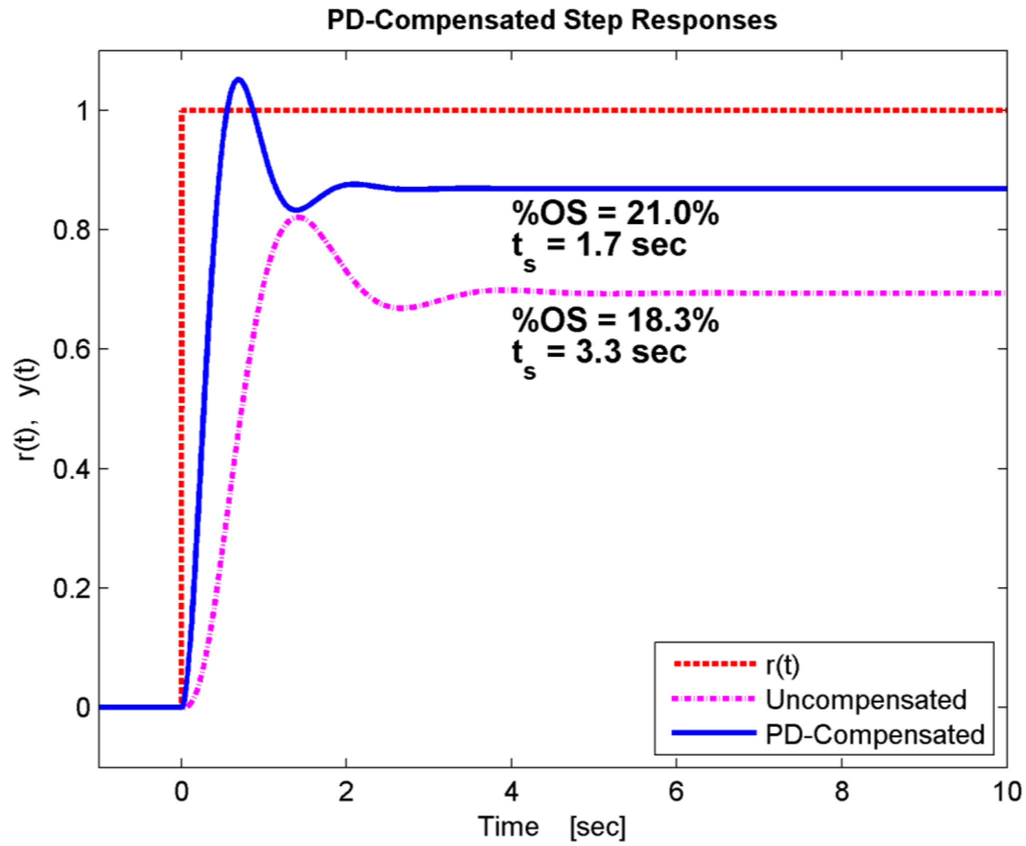
- PD compensator:

$$D_{pd}(s) = 1.55(s+5.13)$$



PID Compensation – Example

- Performance specifications not met exactly
  - Higher-frequency pole/zero do not entirely cancel
  - Close enough for now – may need to iterate when PI compensation is added



PID Compensation – Example

- Next, add PI compensation to the PD-compensated system
  - Add a pole at the origin and a zero close by

$$D_{pi}(s) = \frac{s + z_{pi}}{s}$$

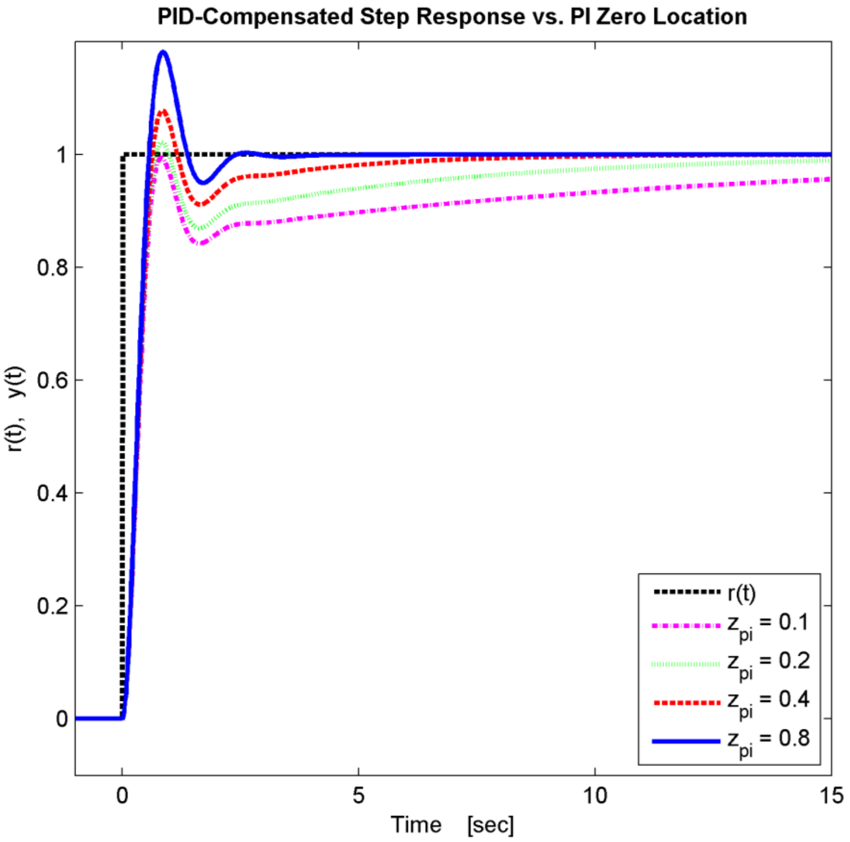
- Where should we put the zero, z_{pi}?
  - In this case, open-loop pole at the origin will become a closed-loop pole near $-z_{pi}$
  - Very small z_{pi} yields very slow closed-loop pole
    - Error integrates out very slowly
  - Small z_{pi} means PI compensator will have less effect on the PD-compensated root locus
  - Simulate and iterate

PID Compensation – Example

- Step response for various z_{pi} values:
- Here, $z_{pi} = 0.8$ works well
  - Moving z_{pi} away from the open-loop pole at the origin moves the 2nd-order poles significantly:

$$s_{1,2} = -1.86 \pm j3.63$$

  - Faster low-frequency closed-loop pole means error is integrated out more quickly



PID Compensation – Example

- The resulting PID compensator:

$$D(s) = K\frac{(s+0.8)}{s}(s+5.13)$$

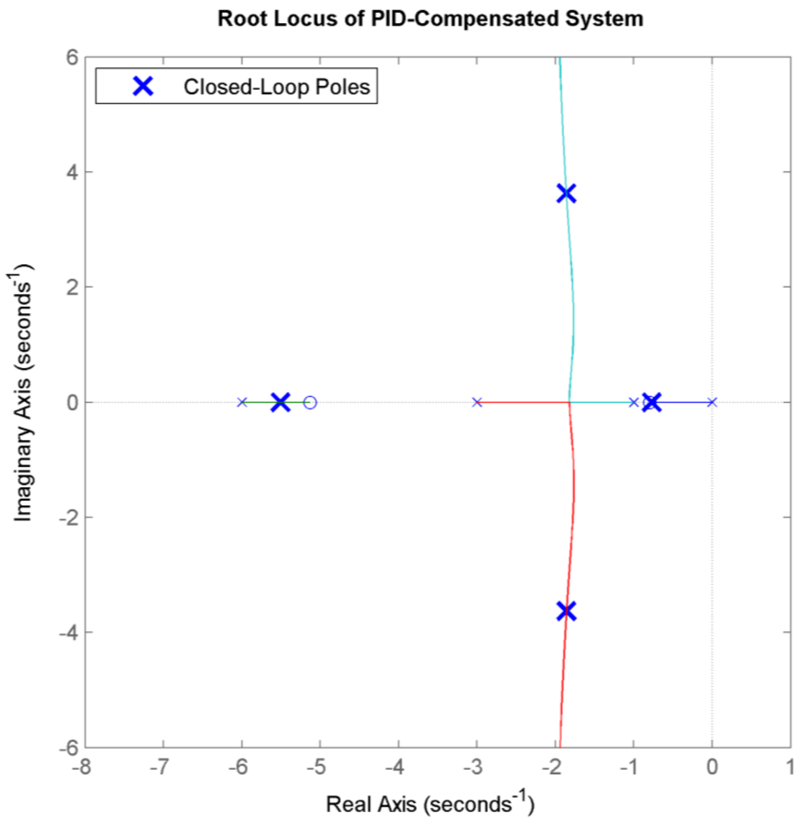
- Required gain: $K = 1.15$

$$D(s) = \frac{1.15s^2 + 6.817s + 4.718}{s}$$

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

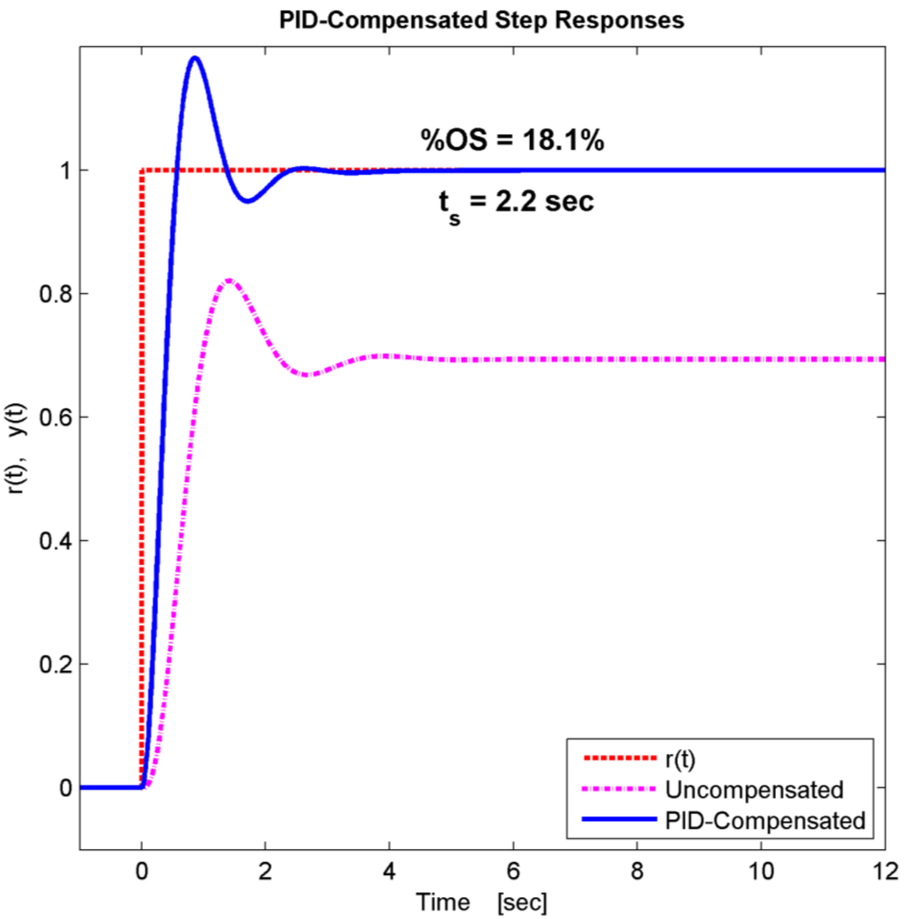
- The PID gains:

$$K_p = 6.817,\ K_i = 3.718,\ K_d = 1.15$$



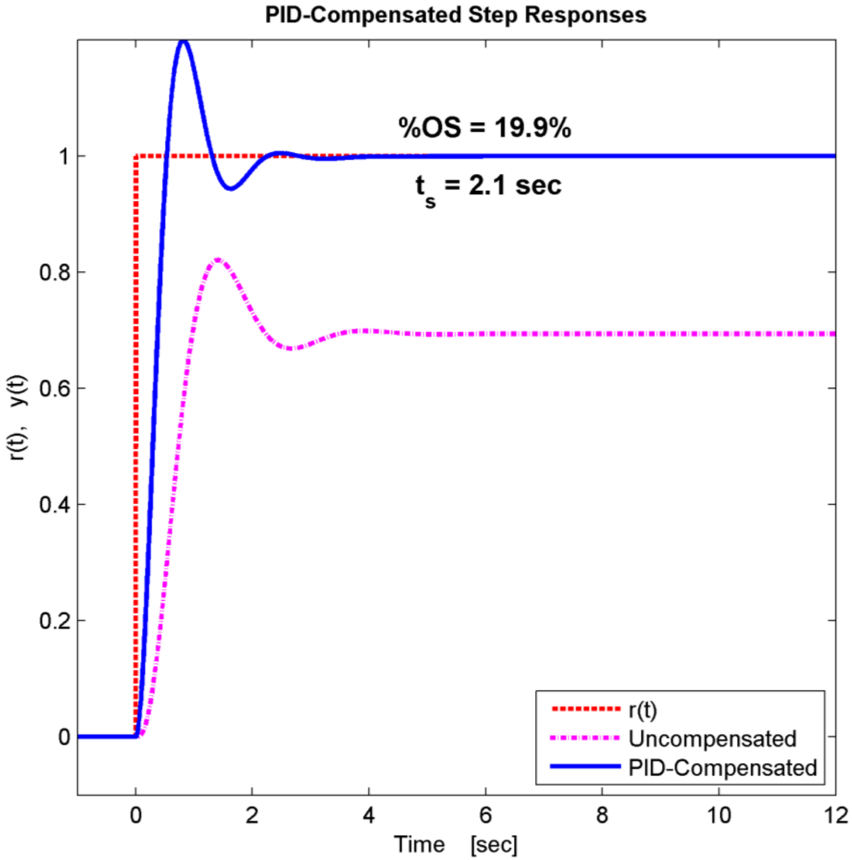
PID Compensation – Example

- Step response of the PID-compensated system:
- Settling time is a little slow
- A bit of margin on the overshoot
- Iterate
  - First, try adjusting gain alone
  - If necessary, revisit the PD compensator



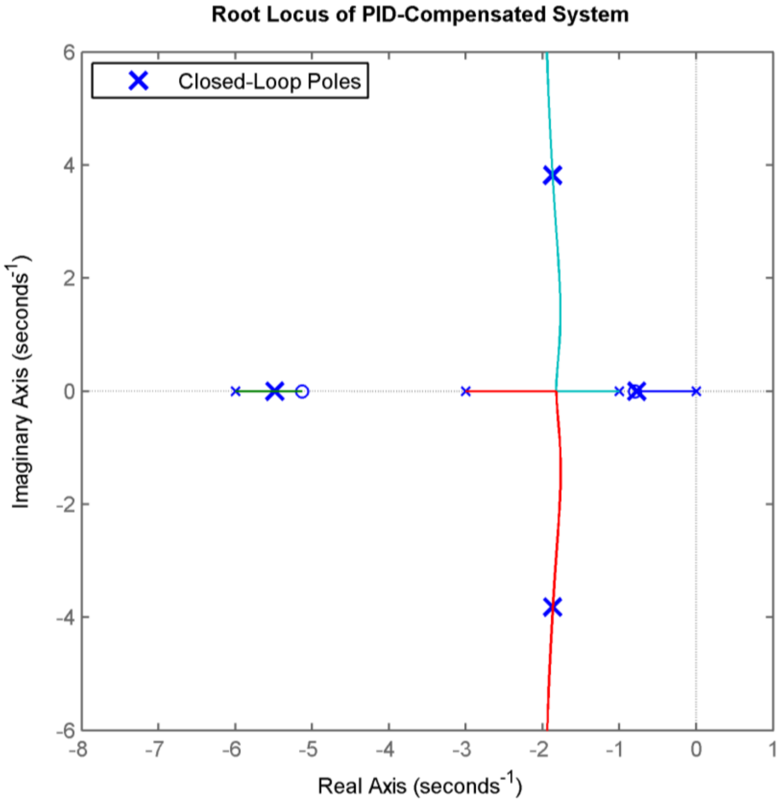
PID Compensation – Example

- Increasing gain to $K = 1.25$ speed things up a bit, while increasing overshoot
- Recall, however that root locus asymptotes are vertical
  - Increasing gain will have little effect on settling time
- If further refinement is required, must revisit the PD compensator



PID Compensation – Example

- How valid was the second-order approximation we used for design of this PID-compensated system?
- Pole at $s = -0.78$
  - Nearly canceled by the zero at $s = -0.8$
- Pole at $s = -5.5$
  - Not high enough in frequency to be negligible, but
  - Partially canceled by zero at $s = -5.13$
- But, validity of the assumption is not really important
  - Used as starting point to locate poles
  - Iteration typically required anyway



PID Compensation – Summary

- ***PID compensation***

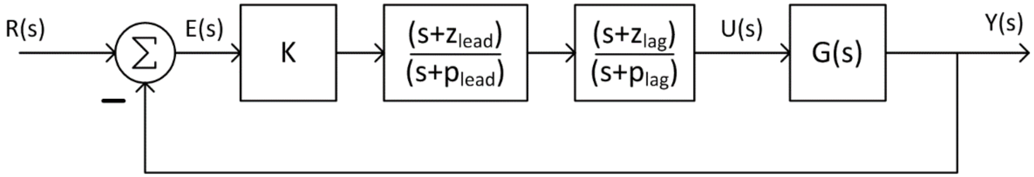
$$D(s) = K_p + \frac{K_i}{s} + K_d s = \frac{K_d s^2 + K_p s + K_i}{s}$$

  - ***Two zeros*** and a ***pole at the origin***
  - Cascade of PI and PD compensators
- PD compensator
  - Added zero allows for ***transient response improvement***
- PI compensator
  - Pole at the origin ***increases system type***
  - Nearby zero nearly cancels angular contribution of the pole, limiting its effect on the root locus

Lead-Lag Compensation

Lead-Lag Compensation

- Just as we combined derivative and integral compensation, we can combine ***lead*** and ***lag*** as well
  - ***Lead-lag compensation***
  - Lead compensator improves ***transient response***
  - Lag compensator improves ***steady-state error***



- Compensator transfer function:

$$D(s) = K\frac{(s+z_{lead})}{(s+p_{lead})}\frac{(s+z_{lag})}{(s+p_{lag})}$$

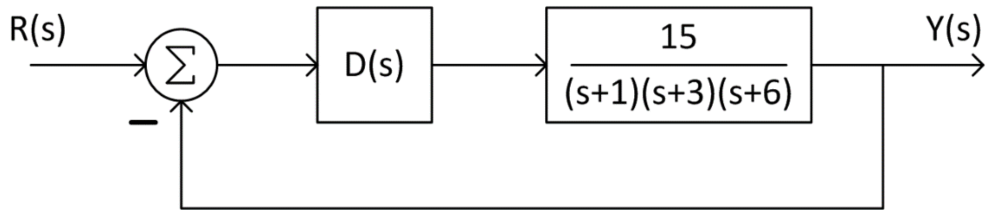
  - Lead compensator adds a pole and zero - $z_{lead} < p_{lead}$
  - Lag pole/zero close to the origin - $z_{lag} > p_{lag} \approx 0$

Lead-Lag Design Procedure

- **Lead-lag compensator design procedure:**

1. Determine closed-loop pole location to provide desired transient response
2. Design the lead compensator (zero, pole, and gain) to place closed-loop poles as desired
3. Simulate the lead-compensated system, iterate if necessary
4. Evaluate the steady-state error performance of the lead-compensated system to determine how much of an improvement is required to meet the error specification
5. Design the lag compensator to yield the required steady-state error performance
6. Simulate the lead-lag-compensated system and iterate, if necessary

Lead-Lag Compensation – Example



- Design lead-lag compensation to satisfy the following specifications:
  - $t_s \approx 2\ sec$
  - $\%OS \approx 20\%$
  - 2% steady-state error to a constant reference

- First, design the lead compensator to satisfy the dynamic specifications
- Then, design the lag compensator to meet the steady-state error requirement

Lead-Lag Compensation – Example

- Design the lead compensator to achieve the same desired dominant 2nd-order pole locations:

$$s_{1,2} = -2.3 \pm j4.49$$

  - Again, an infinite number of possibilities

- Let's assume we want to limit the lead pole to $s = -100$ due to noise considerations
  - Lower pole frequency results in amplification of less noise

$$D_{lead}(s) = K\frac{(s + z_{lead})}{(s + 100)}$$

- Apply the angle criterion to determine z_{lead}

Lead-Lag Compensation – Example

$$\angle D_{lead}(s_1)G(s_1) = 180° \quad \rightarrow \quad \angle D_{lead}(s_1) = 180° - \angle G(s_1)$$

$$\angle G(s_1) = -\phi_1 - \phi_2 - \phi_3$$

$$\phi_1 = \angle(s_1 + 1) = 106.15°$$

$$\phi_2 = \angle(s_1 + 3) = 81.14°$$

$$\phi_3 = \angle(s_1 + 6) = 50.51°$$

$$\angle G(s_1) = -237.8°$$

$$\angle D_{lead}(s_1) = 180° + 237.8°$$

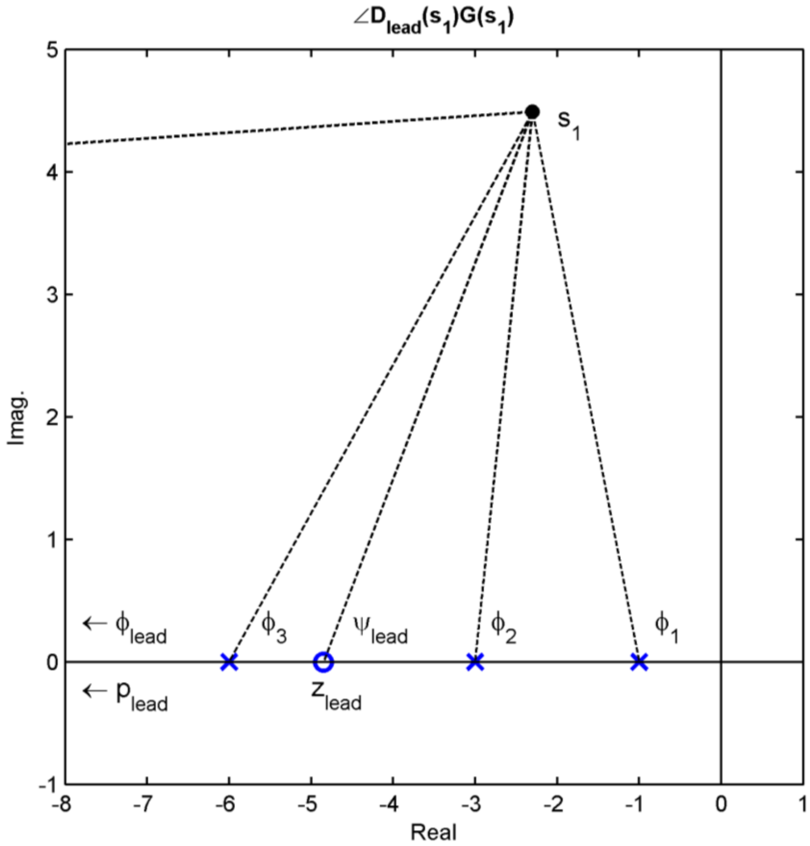
$$\angle D_{lead}(s_1) = 417.8° = 57.8°$$

$$\angle D_{lead}(s_1) = \psi_{lead} - \phi_{lead}$$

$$\phi_{lead} = \angle(s_1 + 100) = 2.63°$$

$$\psi_{lead} = \angle D_{lead}(s_1) + \phi_{lead}$$

$$\psi_{lead} = 60.43°$$



Lead-Lag Compensation – Example

- Next, calculate z_{lead} from ψ_{lead}

$$\psi_{lead} = \angle(s_1 + z_{lead})$$

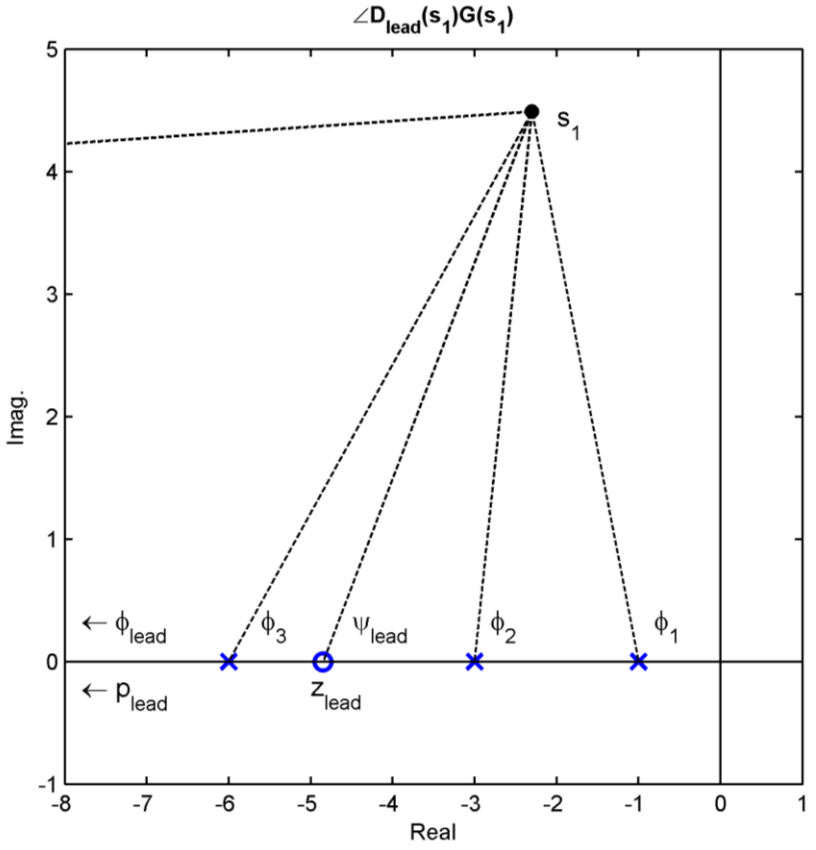
$$\psi_{lead} = \tan^{-1}\left(\frac{Im(s_1)}{Re(s_1) + z_{lead}}\right)$$

$$\tan(\psi_{lead}) = \left(\frac{Im(s_1)}{Re(s_1) + z_{lead}}\right)$$

$$z_{lead} = \frac{Im(s_1)}{\tan(\psi_{lead})} - Re(s_1)$$

$$z_{lead} = \frac{4.49}{\tan(60.43°)} + 2.3$$

$$z_{lead} = 4.85$$



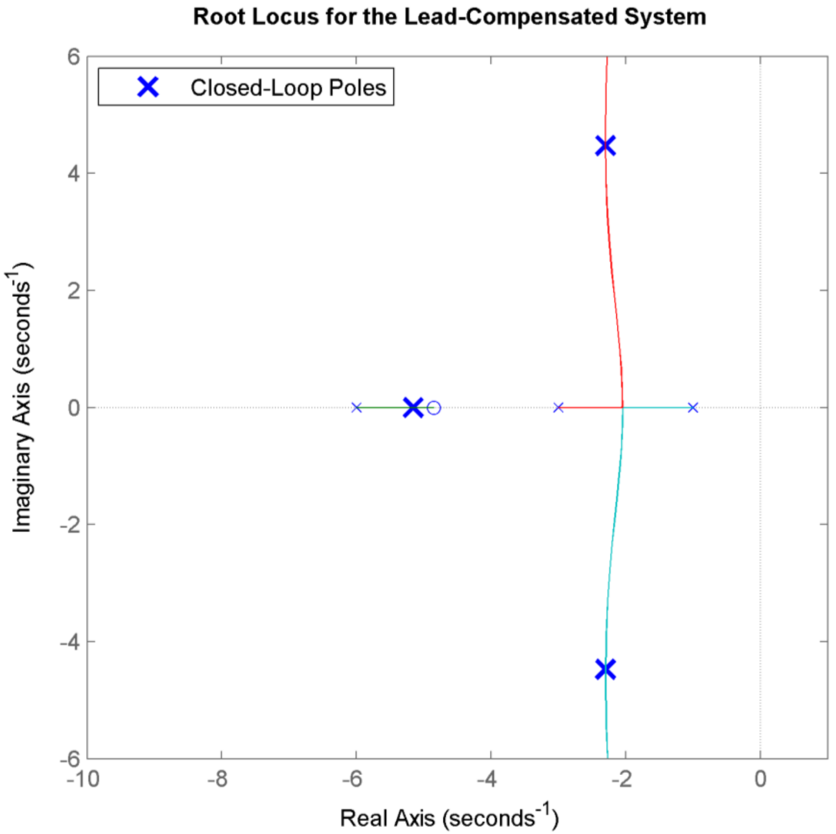
Lead-Lag Compensation – Example

- Lead compensator:

$$D_{lead}(s) = K\frac{(s+4.85)}{(s+100)}$$

- From magnitude criterion or MATLAB plot, $\mathrm{K} = 156$
- Lead compensator transfer function:

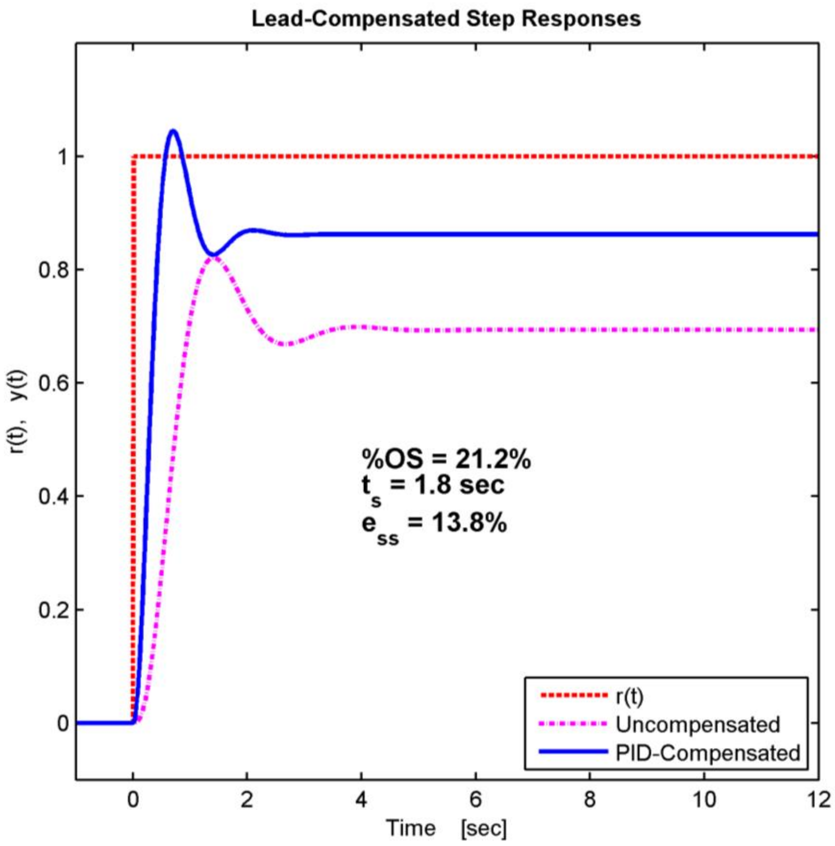
$$D_{lead}(s) = 156\frac{(s+4.85)}{(s+100)}$$



- Next, simulate the lead-compensated system to verify dynamic performance and to evaluate steady-state error

Lead-Lag Compensation – Example

- Performance specifications not met exactly
  - Higher-frequency pole/zero do not entirely cancel
  - Close enough for now – may need to iterate when lag compensation is added, anyway
- Steady-state error is 13.8%



Lead-Lag Compensation – Example

- Desired position constant:

$$e_{ss} = \frac{1}{1 + K_p} = 0.02$$

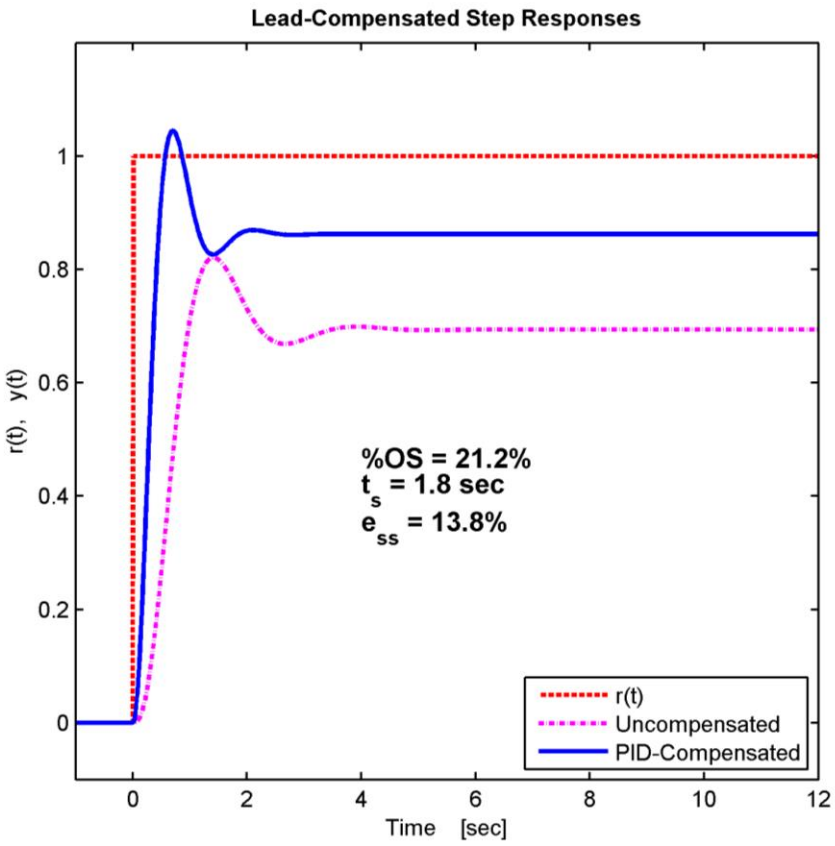
$$K_p = \frac{1}{e_{ss}} - 1 = 49$$

- K_p for lead-compensated system

$$K_{p,lead} = \frac{1}{0.138} - 1 = 6.26$$

- Required error constant improvement

$$\frac{z_{lag}}{p_{lag}} = \frac{K_p}{K_{p,lead}} = 7.83$$



Lead-Lag Compensation – Example

- Arbitrarily set $p_{lag} = 0.01$
- To achieve desired error, we need

$$z_{lag} = 8 \cdot p_{lag} = 0.08$$

- The lag compensator transfer function:

$$D_{lag}(s) = \frac{(s + 0.08)}{(s + 0.01)}$$

- Magnitudes from lag pole/zero effectively cancel, so required gain is unchanged:

$$K = 156$$

- Lead-lag compensator transfer function:

$$D(s) = 156 \frac{(s + 4.85)}{(s + 100)} \frac{(s + 0.08)}{(s + 0.01)}$$

Lead-Lag Compensation – Example

- Root locus and closed-loop poles/zeros for the lead-lag-compensated system:
- Second-order poles:

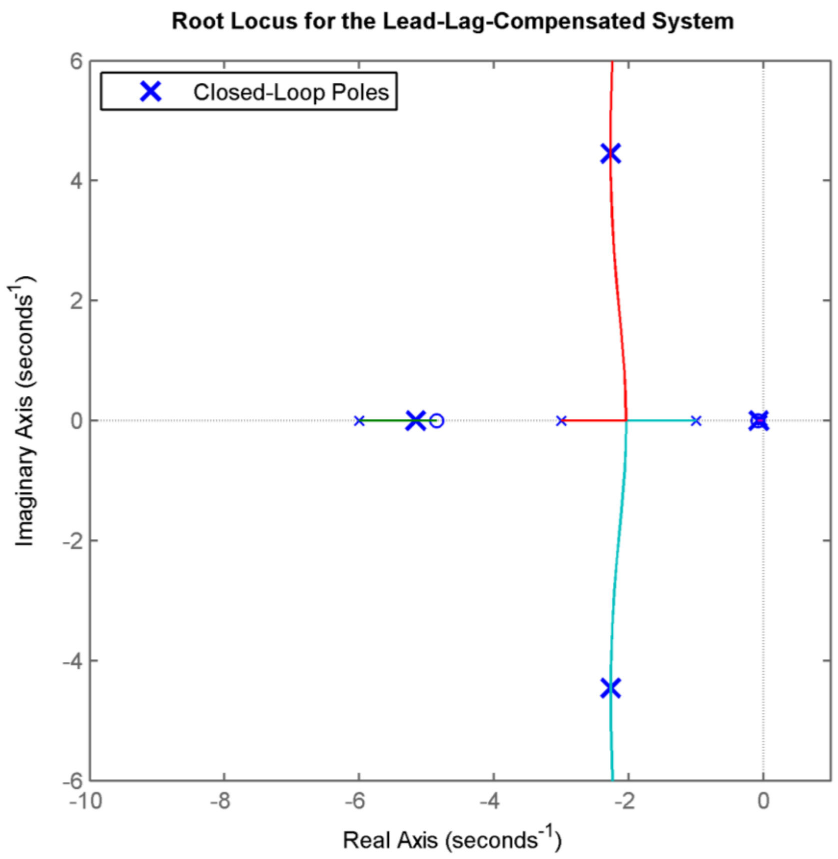
$$s_{1,2} = -2.27 \pm j4.46$$

- Other closed-loop poles:

$$s = -100.2$$
$$s = -5.2$$
$$s = -0.07$$

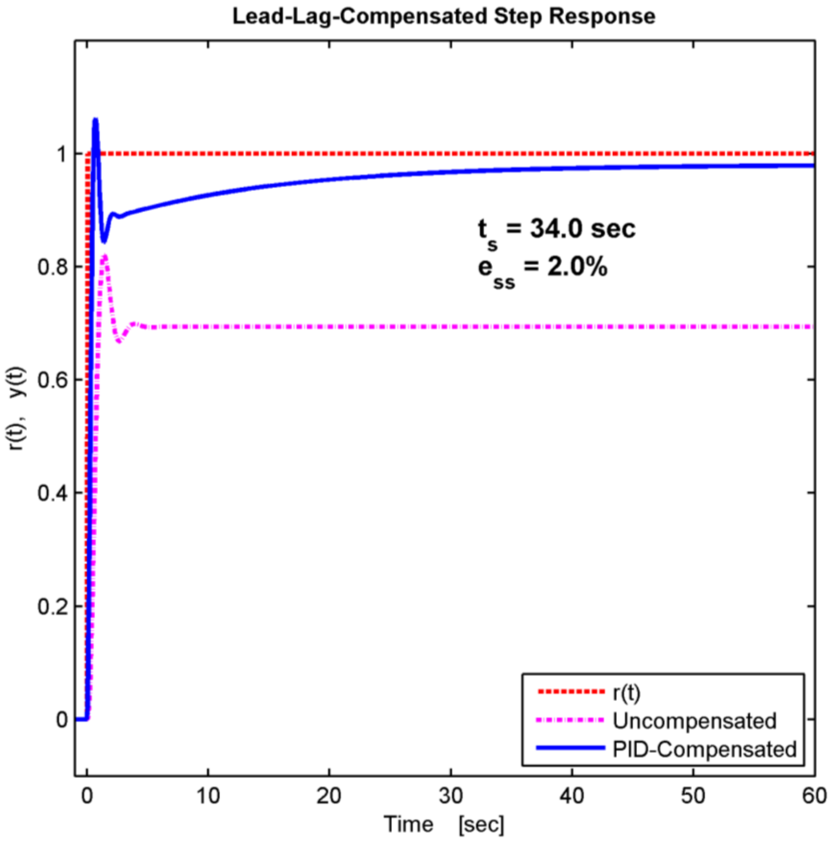
- Closed-loop zeros:

$$s = -0.08$$
$$s = -4.85$$



Lead-Lag Compensation – Example

- Step response for the lead-lag-compensated system:
- Steady-state error requirement is satisfied
- Slow closed-loop pole at $s = -0.07$ results in very slow tail as error is eliminated
- Can speed this up by moving the lag pole/zero away from the origin
  - Dominant poles will move



Lead-Lag Compensation – Example

- Increase the lag pole/zero frequency by 8x

$$p_{lag} = 0.08 \text{ and } z_{lag} = 0.64$$

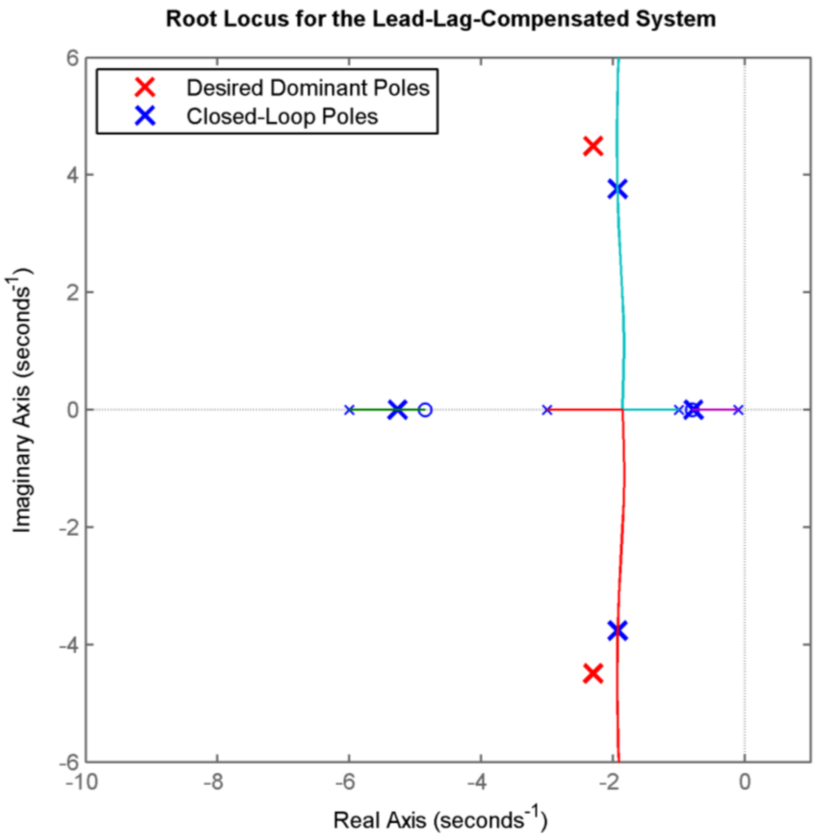
- Lag pole/zero now affect the root locus significantly
  - Dominant poles move:

$$s_{1,2} = -1.9 \pm j3.8$$

  - Required gain for $\zeta = 0.46$ changes:

$$K = 123$$

- Reduced gain will reduce K_p
  - z_{lag}/p_{lag} ratio must increase



Lead-Lag Compensation – Example

- Some iteration shows reasonable transient and error performance for:

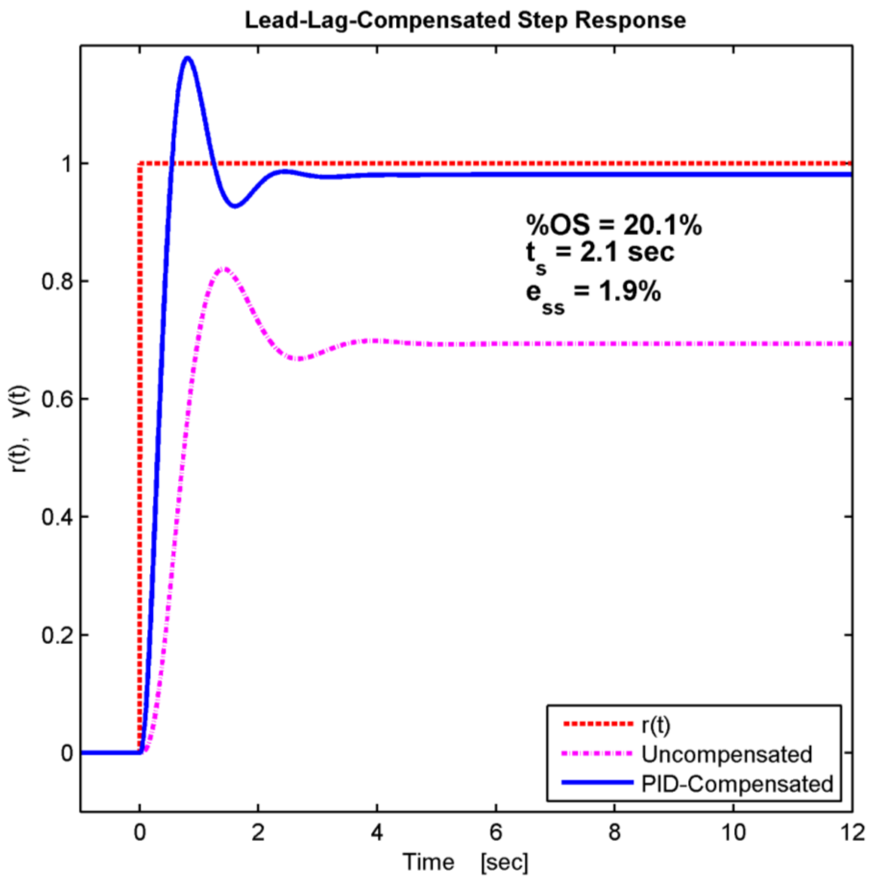
$$p_{lag} = 0.08$$

$$z_{lag} = 0.8$$

$$K = 130$$

- Lead-lag compensator:

$$D(s) = 130\frac{(s + 4.85)}{(s + 100)}\frac{(s + 0.8)}{(s + 0.08)}$$



Lead-Lag Compensation – Summary

- ***Lead-lag compensation***

$$D(s) = K\frac{(s+z_{lead})}{(s+p_{lead})}\frac{(s+z_{lag})}{(s+p_{lag})}, \quad p_{lead} > z_{lead} \text{ and } p_{lag} < z_{lag}$$

  - ***Two zeros*** and ***two poles***
  - Cascade of lead and lag compensators
- Lead compensator
  - Added pole/zero improves ***transient response***
- Lag compensator
  - ***Steady-state error improved*** by z_{lag}/p_{lag}
  - Nearby zero partially cancels angular contribution of the pole, limiting its effect on the root locus
  - May introduce a slow transient

Summary

Compensator Summary

Type	Transfer function	Improves	Comments
PI	$K\frac{(s+z_c)}{s}$	Error	• Pole at origin • Zero near origin • Increases system type • May introduce a slow transient • Active circuitry required • Susceptible to integrator windup
Lag	$K\frac{(s+z_c)}{(s+p_c)}$	Error	• Pole *near* the origin • Small negative zero • $z_c > p_c$ • Error constant improved by z_c/p_c • May introduce a slow transient • Passive circuitry implementation possible

Compensator Summary

Type	Transfer function	Improves	Comments
PD	$K(s+z_c)$	Transient response	• Zero at $-z_c$ contributes angle to satisfy angle criterion at desired closed-loop pole location • Active circuitry required • Amplifies sensor noise
Lead	$K\frac{(s+z_c)}{(s+p_c)}$	Transient response	• Lower-frequency zero • Higher-frequency pole • Net angle contribution satisfies angle criterion at design point • Added pole helps reduce amplification of higher-frequency sensor noise • Passive circuitry implementation possible

Compensator Summary

Type	Transfer function	Improves	Comments
PID	$K_p + \frac{K_i}{s} + K_d s$	Error & transient response	• PD compensation improves transient response • PI compensation improves steady-state error • Active circuitry • Amplifies noise
Lead-lag	$K\frac{(s+z_{lead})}{(s+p_{lead})}\frac{(s+z_{lag})}{(s+p_{lag})}$	Error & transient response	• Lead compensation improves transient response • Lag compensation improves steady-state error • Passive circuitry implementation possible • Amplification of high-frequency noise reduced