Lecturer

Set up MATLAB

In [1]:

```
cd matlab
pwd
clear all
format compact
```

```
ans =

    '/Users/eechris/dev/eglm03-textbook/content/02/matlab'
```

Steady-state and Transient Response

This chapter is concerned with the analysis of steady-state and transient response performance of control systems.

The second-order system response and its relationship to the closed-loop poles and zeros is revised. The effect of an additional zero or an additional pole on the 2^{nd} order response is examined and pole-zero cancellation is discussed.

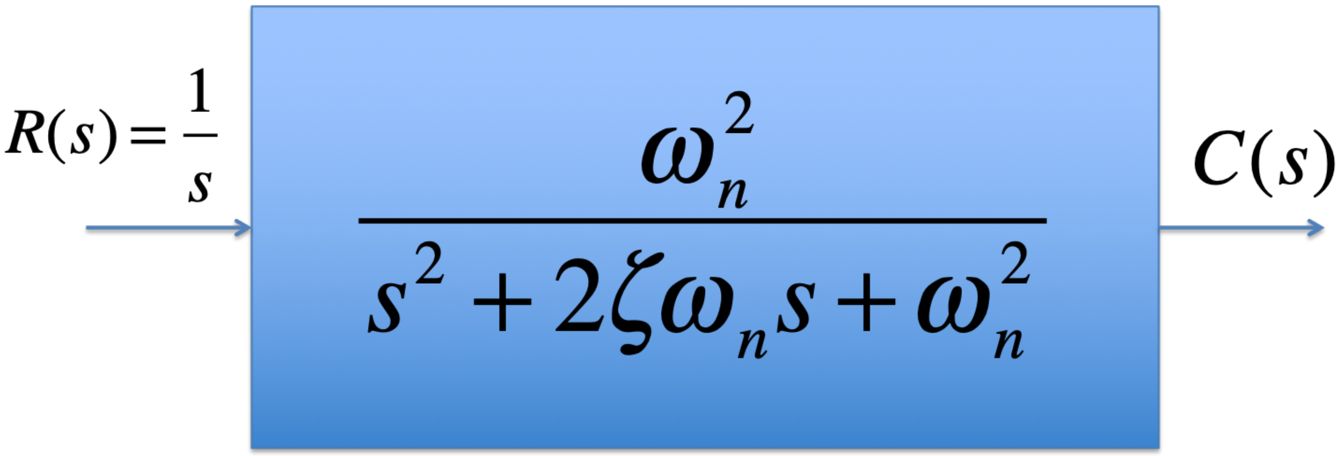
System type-number and its relationship to steady-state error response is revised.

Reading

You should read sections 4.2 **Time Domain Criteria** and 4.1 **Steady-State Criteria** of the Handout (/eglm03-textbook/handouts/csd) **Control System Design Methods, Compensation Strategies and Design Criteria.**

Transient Performance

A Second-Order System



Where are the system poles and what does the model 2nd Order response look like for each of these cases?

ω_n	ζ
3	3
3	1
3	0.8
3	0.5
3	0

Effect of Damping on 2nd Order Response

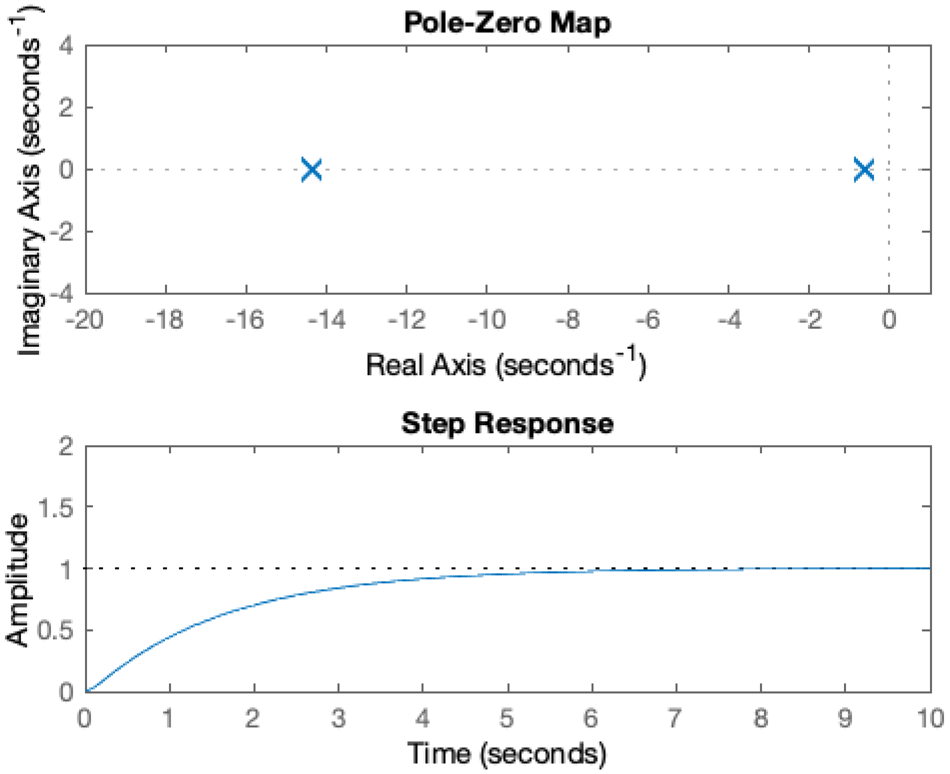
In [2]:

```
wn = 3;
z = [3, 2.5, 2, 1.5, 1, 0.9, 0.8, 1/sqrt(2), 0.5, 0.4, 0.3, 0.2, 0.1, 0];
```

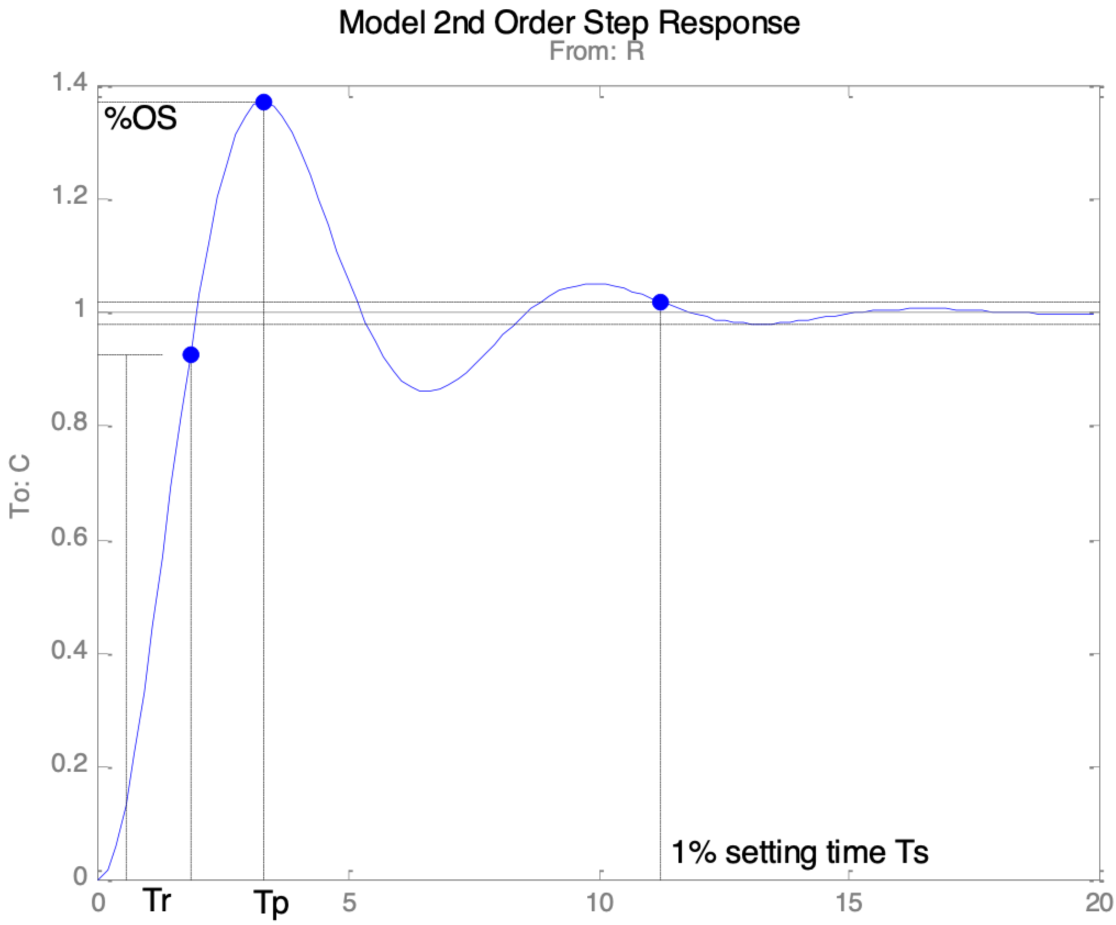
In [3]:

```
zeta = z(2);
G = tf(wn^2, [1, 2*zeta*wn, wn^2])
subplot(211),pzmap( G),axis([-20, 1, -4, 4])
subplot(212),step( G),axis([0,10,0,2])
```

```
G =
 
         9
  --------------
  s^2 + 15 s + 9
 
Continuous-time transfer function.
```



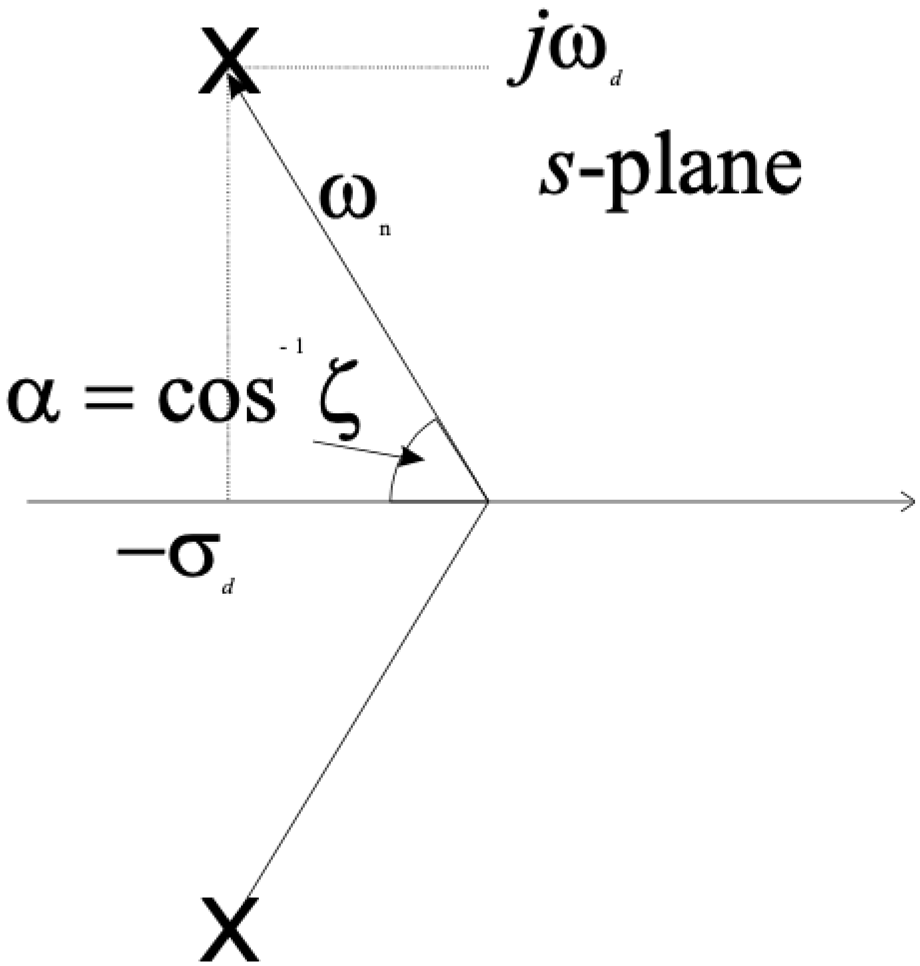
Or download and run this script second_resp.m (second_resp.m) in MATLAB.



How do the natural frequency and damping ratio relate to pole locations?

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

$$\begin{aligned} P_{1,2} &= -\zeta\omega_n \pm j\omega_n\sqrt{1-\zeta^2} \\ &= -\sigma_d \pm j\omega_d \end{aligned}$$



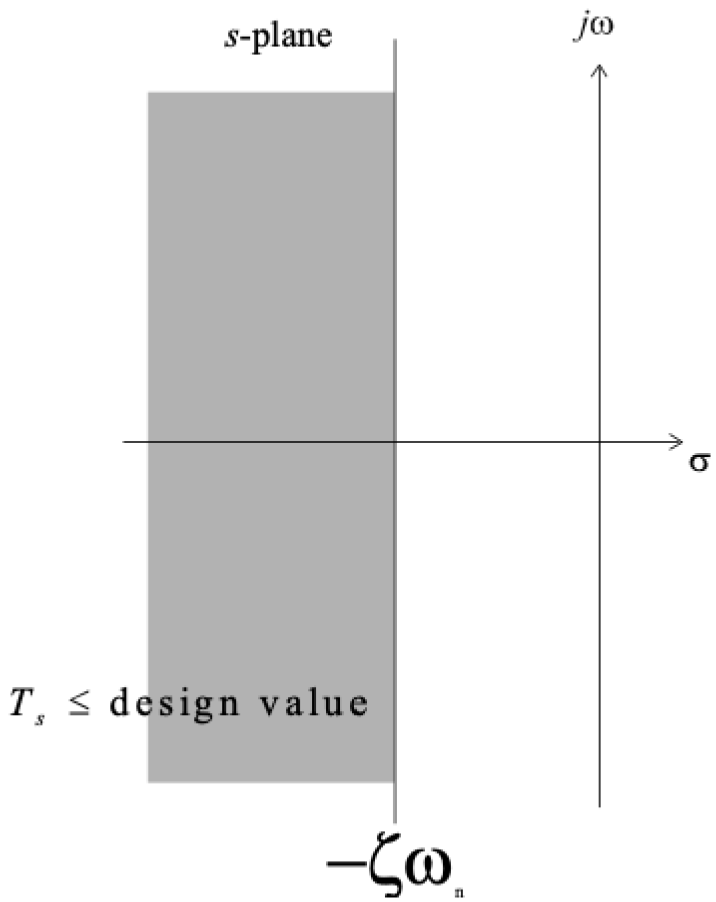
How do the transient performance criteria map to the closed loop poles?

Settling time T_s

Settling time is related to relative stability and speed of response.

1% settling time:

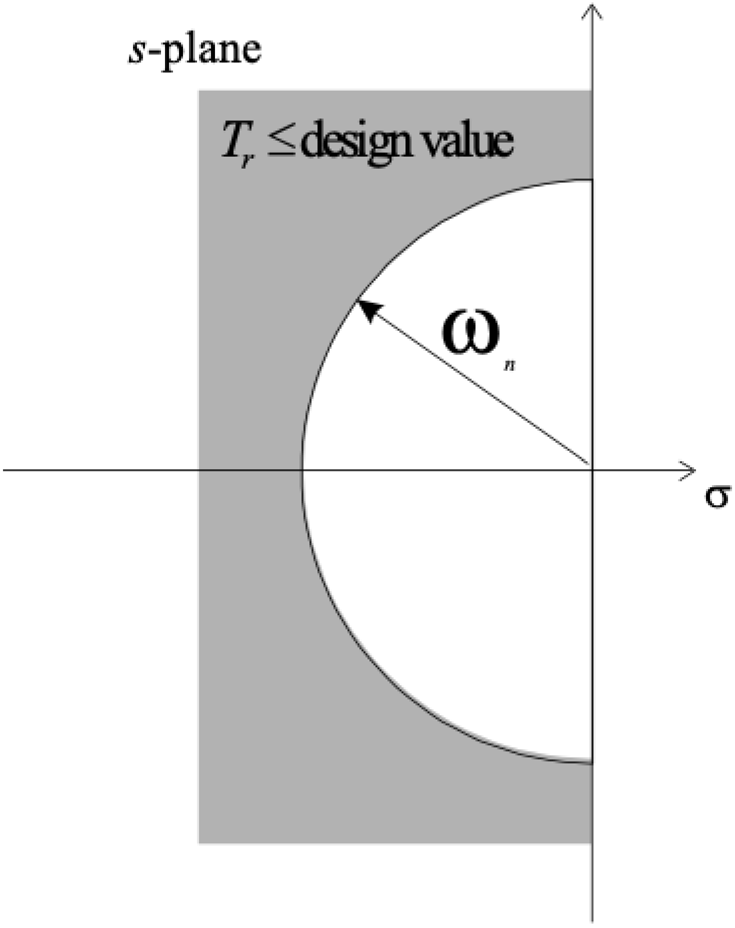
$$T_s \approx \frac{4.6}{\sigma_d}$$



Rise Time T_r

Rise time is related to speed of response

$$T_r \approx \frac{1.8}{\omega_n}$$

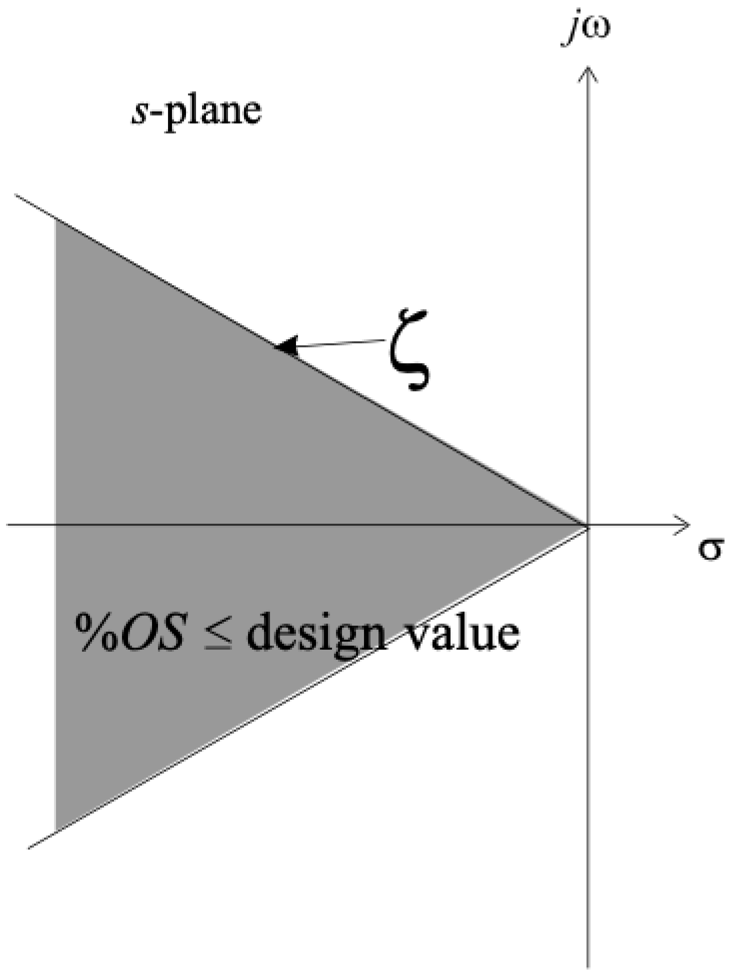


Percentage overshoot (%OS or M_p)

Percentage overshoot is related to damping

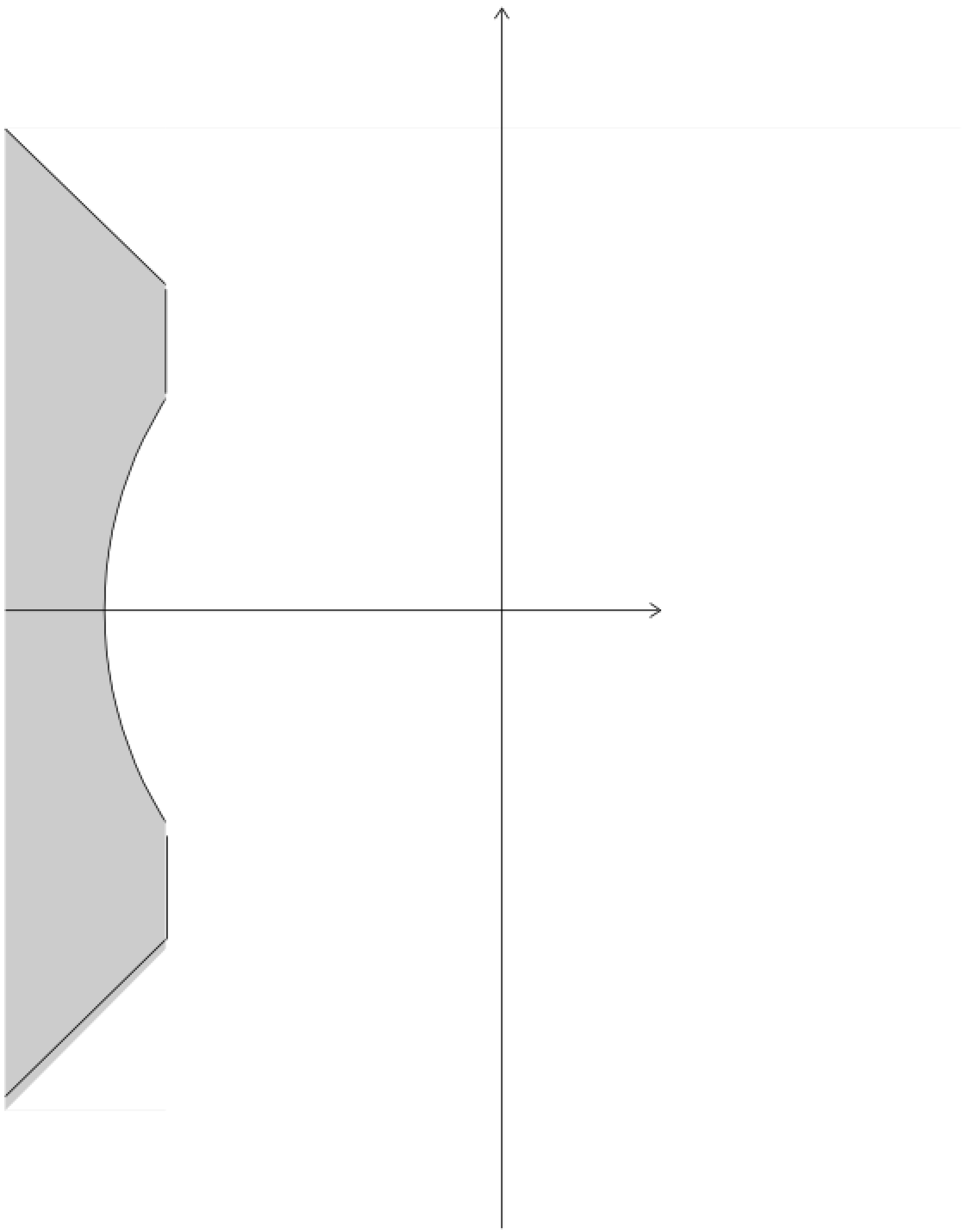
$$M_p = \exp\left(\frac{-\pi\zeta}{\sqrt{1-\zeta^2}}\right) \times 100$$

$$M_p \approx \left(1 - \frac{\zeta}{0.6}\right) \times 100 \;\; 0 \leqslant \zeta \leqslant 0.6$$



Combined constraints

- If system has inadequate rise time (too slow) we must raise the natural frequency
- If system has too much overshoot we need to increase damping
- If transient persists too long, move the poles further to the left in the s-plane



What if the system is not second order?

- What is the effect of an extra zero?

- What is the effect of an extra pole?
- What if there are many poles and zeros?

Effect of an Extra Zero

First normalize transfer function:

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

Then add a zero

$$G(s) = \frac{C(s)}{R(s)} = \frac{\left(\frac{s}{\alpha\zeta\omega_n}\right) + 1}{\left(\frac{2}{\omega_n}\right)^2 + 2\zeta\left(\frac{s}{\omega_n}\right) + 1}$$

Note that α is a multiplier of the real part of the complex poles $\zeta\omega_n$.

2nd order system with extra zero

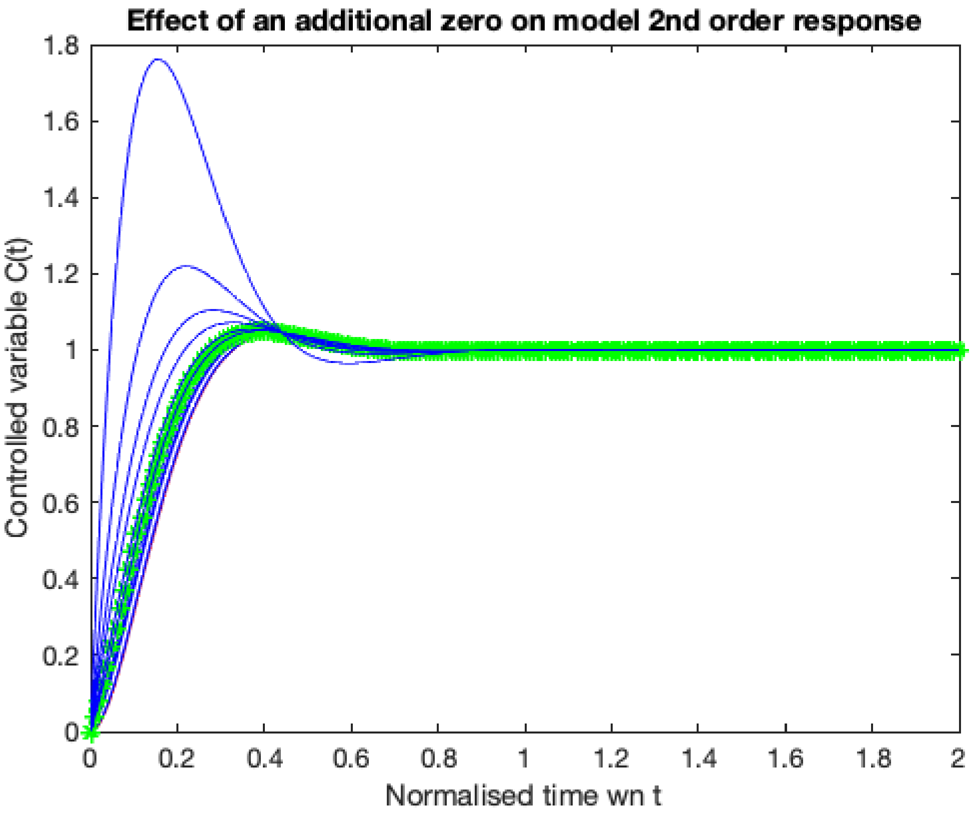
Matlab demo (run zero2nd.m (matlab/zero2nd.m)):

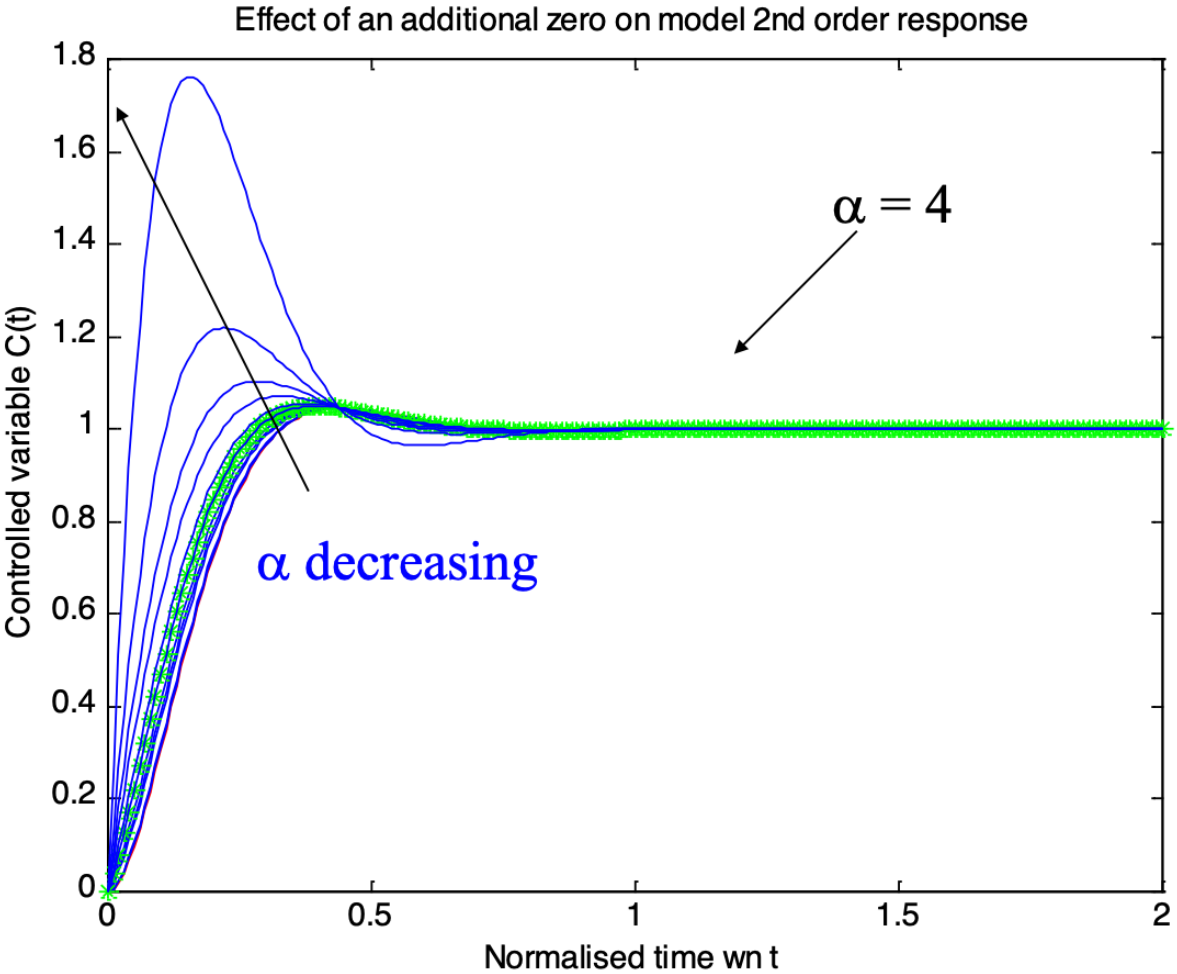
In [4]:

```
zero2nd
```

```
clf
wn = 10;
zeta = 0.7;
t = 0:0.01:2;
s = tf('s');
Tc = tf(1/((s/wn)^2 + 2*zeta*(s/wn) + 1))

Tc =

          1000
  ---------------------
  10 s^2 + 140 s + 1000

Continuous-time transfer function.

[c]=step(Tc,t);
plot(t,c,'r-')
title('Effect of an additional zero on model 2nd order response')
ylabel('Controlled variable C(t)')
xlabel('Normalised time wn t')
hold on
for alpha = [100,50,10,8,6,4,3,2,1.5,1,0.5]

   T2 = tf((s/(alpha*zeta*wn)+1)/((s/wn)^2 + 2*zeta*(s/wn) + 1));
   [c,t]=step(T2,t);
   if (alpha == 4)
   plot(t,c,'b-')
end

   T2 = tf((s/(alpha*zeta*wn)+1)/((s/wn)^2 + 2*zeta*(s/wn) + 1));
   [c,t]=step(T2,t);
   if (alpha == 4)
   plot(t,c,'b-')
end

   T2 = tf((s/(alpha*zeta*wn)+1)/((s/wn)^2 + 2*zeta*(s/wn) + 1));
   [c,t]=step(T2,t);
   if (alpha == 4)
   plot(t,c,'b-')
end

   T2 = tf((s/(alpha*zeta*wn)+1)/((s/wn)^2 + 2*zeta*(s/wn) + 1));
   [c,t]=step(T2,t);
   if (alpha == 4)
   plot(t,c,'b-')
end

   T2 = tf((s/(alpha*zeta*wn)+1)/((s/wn)^2 + 2*zeta*(s/wn) + 1));
   [c,t]=step(T2,t);
   if (alpha == 4)
   plot(t,c,'b-')
end

   T2 = tf((s/(alpha*zeta*wn)+1)/((s/wn)^2 + 2*zeta*(s/wn) + 1));
   [c,t]=step(T2,t);
   if (alpha == 4)
      plot(t,c,'g*')
   end
   plot(t,c,'b-')
end
```

```
    T2 = tf((s/(alpha*zeta*wn)+1)/((s/wn)^2 + 2*zeta*(s/wn) + 1));
    [c,t]=step(T2,t);
    if (alpha == 4)
    plot(t,c,'b-')
end

    T2 = tf((s/(alpha*zeta*wn)+1)/((s/wn)^2 + 2*zeta*(s/wn) + 1));
    [c,t]=step(T2,t);
    if (alpha == 4)
    plot(t,c,'b-')
end

    T2 = tf((s/(alpha*zeta*wn)+1)/((s/wn)^2 + 2*zeta*(s/wn) + 1));
    [c,t]=step(T2,t);
    if (alpha == 4)
    plot(t,c,'b-')
end

    T2 = tf((s/(alpha*zeta*wn)+1)/((s/wn)^2 + 2*zeta*(s/wn) + 1));
    [c,t]=step(T2,t);
    if (alpha == 4)
    plot(t,c,'b-')
end

    T2 = tf((s/(alpha*zeta*wn)+1)/((s/wn)^2 + 2*zeta*(s/wn) + 1));
    [c,t]=step(T2,t);
    if (alpha == 4)
    plot(t,c,'b-')
end
```




Effect of an additional zero on model 2nd order response
Controlled variable C(t)
Normalised time wn t
α = 4
α decreasing
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
0
0.5
1
1.5
2

Design curves (see handout):

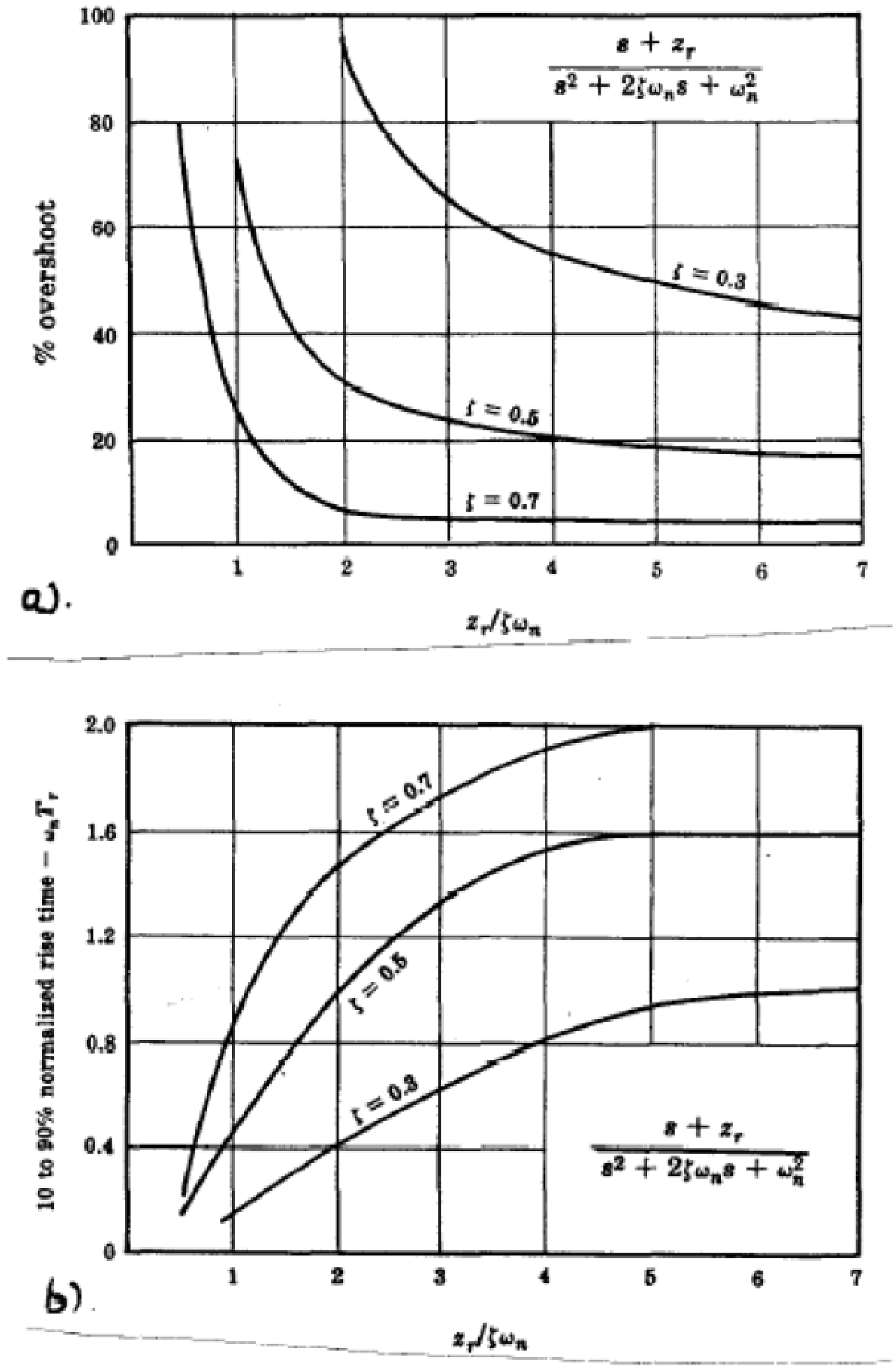


Figure 20 Effect of an extra zero at $s = -z_r$ on a second order system
a). % overshoot M_p vs $z_r/\zeta\omega_n$
b). normalized rise time $\omega_n t_r$ vs $z_r/\zeta\omega_n$

.. how about adding an extra pole?

$$G(s) = \frac{C(s)}{R(s)} = \frac{1}{\left(\left(\frac{s}{\alpha\zeta\omega_n}\right) + 1\right)\left(\left(\frac{s}{\omega_n}\right)^2 + 2\zeta\left(\frac{s}{\omega_n}\right) + 1\right)}$$

Note that α is a multiplier of the real part of the complex poles.

2nd order system with extra pole

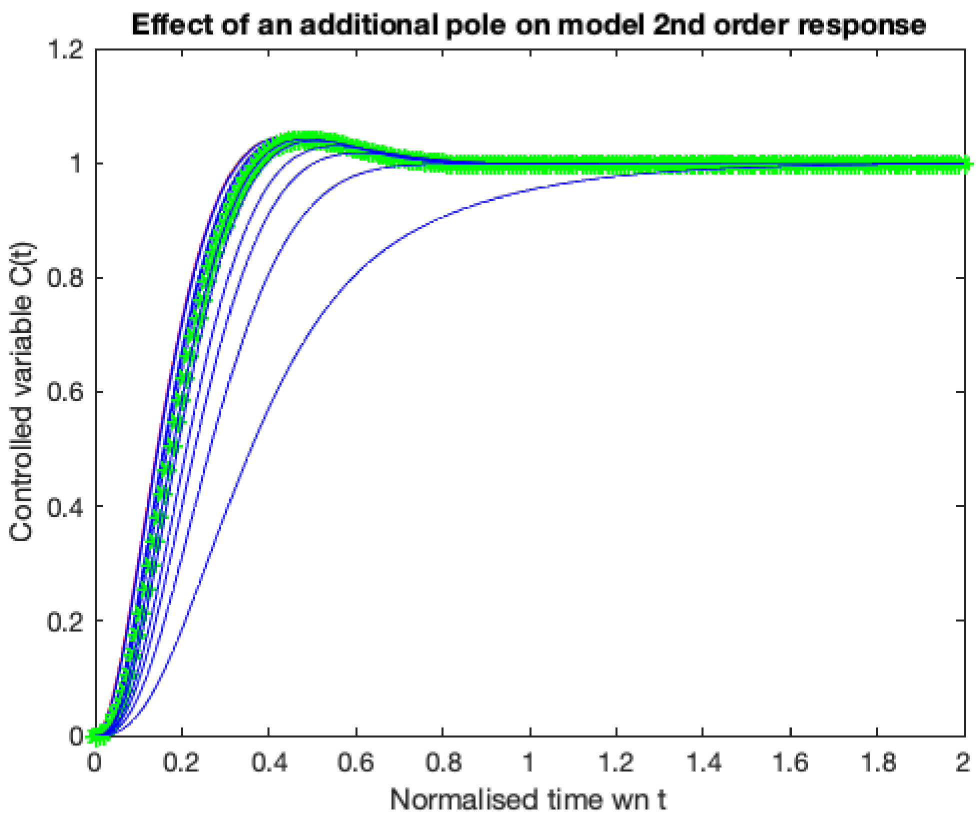
Matlab demo (run pole2nd.m (matlab/pole2nd.m)):

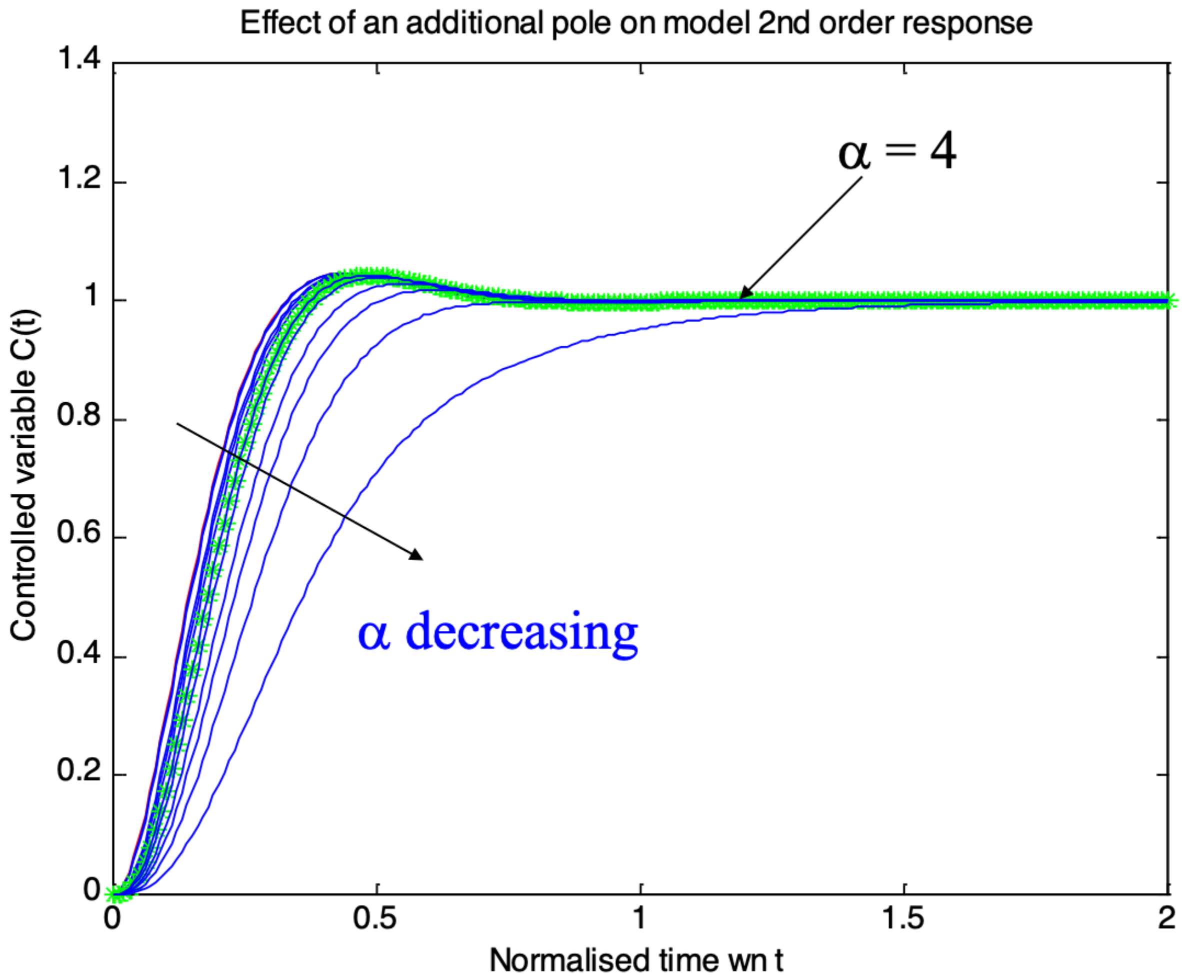
In [5]:

```
pole2nd
```

```
clf
wn = 10;
zeta = 0.7;
t = 0:0.01:2;
s = tf('s');
Tc = tf(1/((s/wn)^2 + 2*zeta*(s/wn) + 1))

Tc =

          1000
  ---------------------
  10 s^2 + 140 s + 1000

Continuous-time transfer function.

[c]=step(Tc,t);
plot(t,c,'r-')
title('Effect of an additional pole on model 2nd order response')
ylabel('Controlled variable C(t)')
xlabel('Normalised time wn t')
hold on

for alpha = [100,50,10,8,6,4,3,2,1.5,1,0.5]
   T2 = tf(1/((s/(alpha*zeta*wn)+1)*((s/wn)^2 + 2*zeta*(s/wn) +
1)));
   [c,t]=step(T2,t);
   if (alpha == 4)
   plot(t,c,'b-')
end
   T2 = tf(1/((s/(alpha*zeta*wn)+1)*((s/wn)^2 + 2*zeta*(s/wn) +
1)));
   [c,t]=step(T2,t);
   if (alpha == 4)
   plot(t,c,'b-')
end
   T2 = tf(1/((s/(alpha*zeta*wn)+1)*((s/wn)^2 + 2*zeta*(s/wn) +
1)));
   [c,t]=step(T2,t);
   if (alpha == 4)
   plot(t,c,'b-')
end
   T2 = tf(1/((s/(alpha*zeta*wn)+1)*((s/wn)^2 + 2*zeta*(s/wn) +
1)));
   [c,t]=step(T2,t);
   if (alpha == 4)
   plot(t,c,'b-')
end
   T2 = tf(1/((s/(alpha*zeta*wn)+1)*((s/wn)^2 + 2*zeta*(s/wn) +
1)));
   [c,t]=step(T2,t);
   if (alpha == 4)
   plot(t,c,'b-')
end
   T2 = tf(1/((s/(alpha*zeta*wn)+1)*((s/wn)^2 + 2*zeta*(s/wn) +
1)));
   [c,t]=step(T2,t);
   if (alpha == 4)
      plot(t,c,'g*')
   end
   plot(t,c,'b-')
end
```

```
    T2 = tf(1/((s/(alpha*zeta*wn)+1)*((s/wn)^2 + 2*zeta*(s/wn) +
1)));
    [c,t]=step(T2,t);
    if (alpha == 4)
    plot(t,c,'b-')
end
    T2 = tf(1/((s/(alpha*zeta*wn)+1)*((s/wn)^2 + 2*zeta*(s/wn) +
1)));
    [c,t]=step(T2,t);
    if (alpha == 4)
    plot(t,c,'b-')
end
    T2 = tf(1/((s/(alpha*zeta*wn)+1)*((s/wn)^2 + 2*zeta*(s/wn) +
1)));
    [c,t]=step(T2,t);
    if (alpha == 4)
    plot(t,c,'b-')
end
    T2 = tf(1/((s/(alpha*zeta*wn)+1)*((s/wn)^2 + 2*zeta*(s/wn) +
1)));
    [c,t]=step(T2,t);
    if (alpha == 4)
    plot(t,c,'b-')
end
    T2 = tf(1/((s/(alpha*zeta*wn)+1)*((s/wn)^2 + 2*zeta*(s/wn) +
1)));
    [c,t]=step(T2,t);
    if (alpha == 4)
    plot(t,c,'b-')
end
```




Effect of an additional pole on model 2nd order response
$\alpha = 4$
α decreasing
Controlled variable C(t)
Normalised time wn t
1.4
1.2
1
0.8
0.6
0.4
0.2
0
0.5
1.5
2

Design curves (see handout):

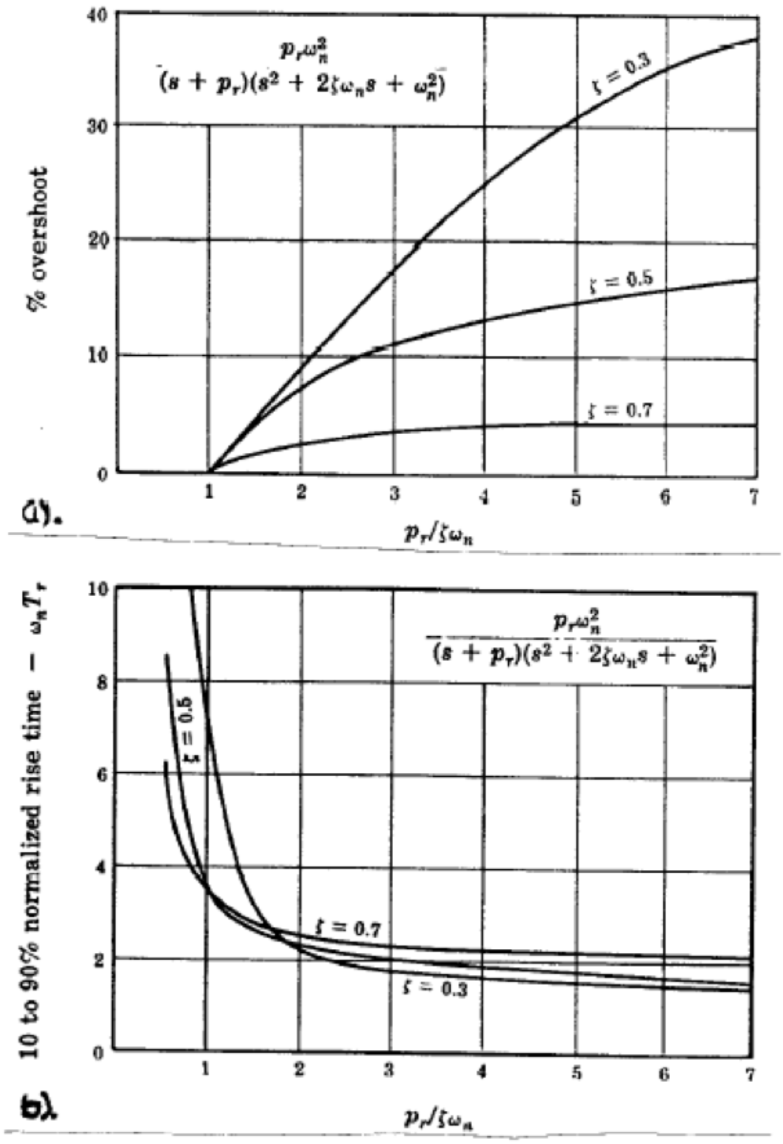


Figure 19 Effect of an extra pole at $s = -p_r$ on a second order system
a). % overshoot M_p vs $p_r/\zeta\omega_n$
b). normalized rise time $\omega_n t_r$ vs $p_r/\zeta\omega_n$

Dominant poles and order reduction

Because the time response of many real systems will be dominated by two or three low frequency poles, a complex high order system can often be simplified by ignoring the effects of high-frequency poles and zeros or a pole that is effectively cancelled by a zero. This MATLAB script file demonstrates this.

Matlab demo (Run reduction.m (matlab/reduction.m))

In this example we ignore any poles or zeros that are located 4 or more times the real part of the dominant poles $s = -1 \pm j$ or poles that a cancelled by a closed-loop zero and see that the seventh order system is effectively only a third-order system.

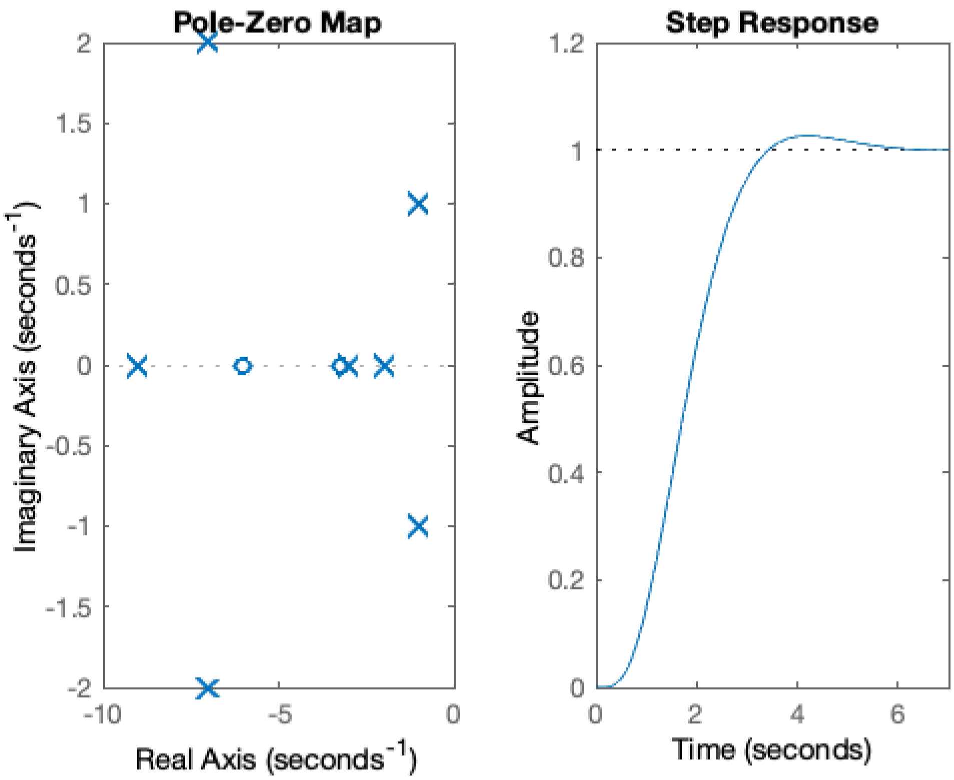
In [6]:

```
sigma = 1;
wd = 1;
disp('Full order system')
zeros = [-6*sigma; -3.2*sigma]
poles = [-9*sigma
   -7*sigma+j*2*wd
   -7*sigma-j*2*wd
   -3*sigma
   -2*sigma
   -sigma+j*wd
   -sigma-j*wd]
g = zpk(zeros,poles,prod(abs(poles))/prod(abs(zeros)));
```

```
Full order system
zeros =
   -6.0000
   -3.2000
poles =
  -9.0000 + 0.0000i
  -7.0000 + 2.0000i
  -7.0000 - 2.0000i
  -3.0000 + 0.0000i
  -2.0000 + 0.0000i
  -1.0000 + 1.0000i
  -1.0000 - 1.0000i
```

In [7]:

```
subplot(121)
pzmap(poles,zeros)
subplot(122)
step(g)
```



Now remove redundant terms

Step 1: remove high frequency pole at $-9 * \sigma$

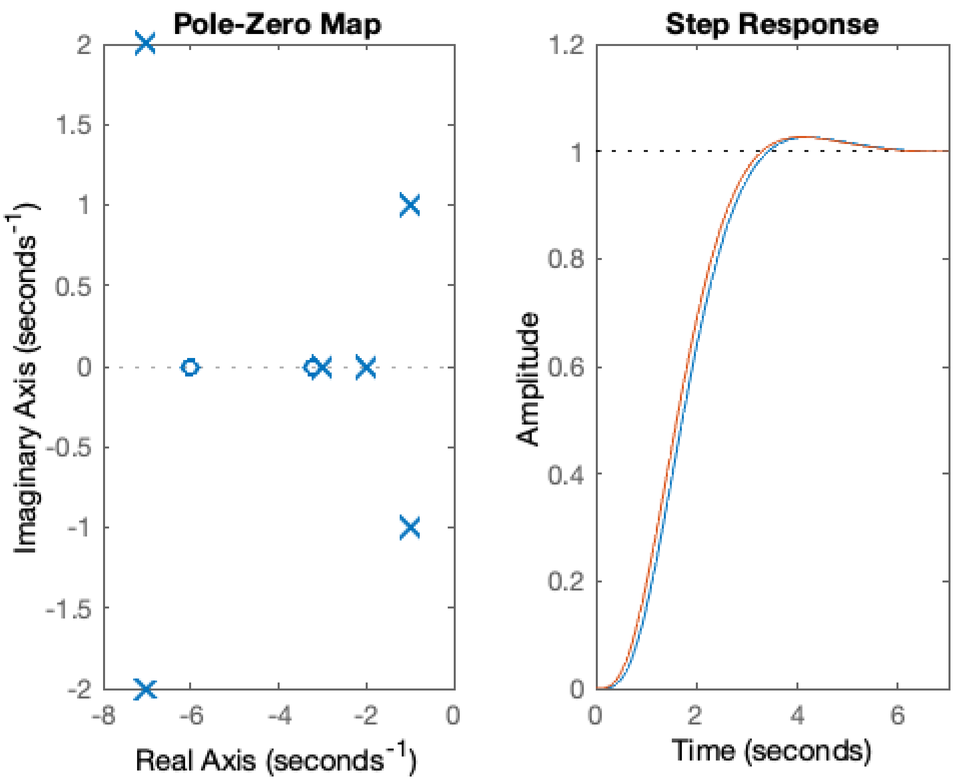
In [8]:

```
z1 = zeros
p1 = poles(2:7)
g1 = zpk(z1,p1,prod(abs(p1))/prod(abs(z1)));
```

```
z1 =
   -6.0000
   -3.2000
p1 =
  -7.0000 + 2.0000i
  -7.0000 - 2.0000i
  -3.0000 + 0.0000i
  -2.0000 + 0.0000i
  -1.0000 + 1.0000i
  -1.0000 - 1.0000i
```

In [9]:

```
subplot(121)
pzmap(p1,z1)
subplot(122)
step(g,g1)
```



Step 2: remove complex hf pole pair

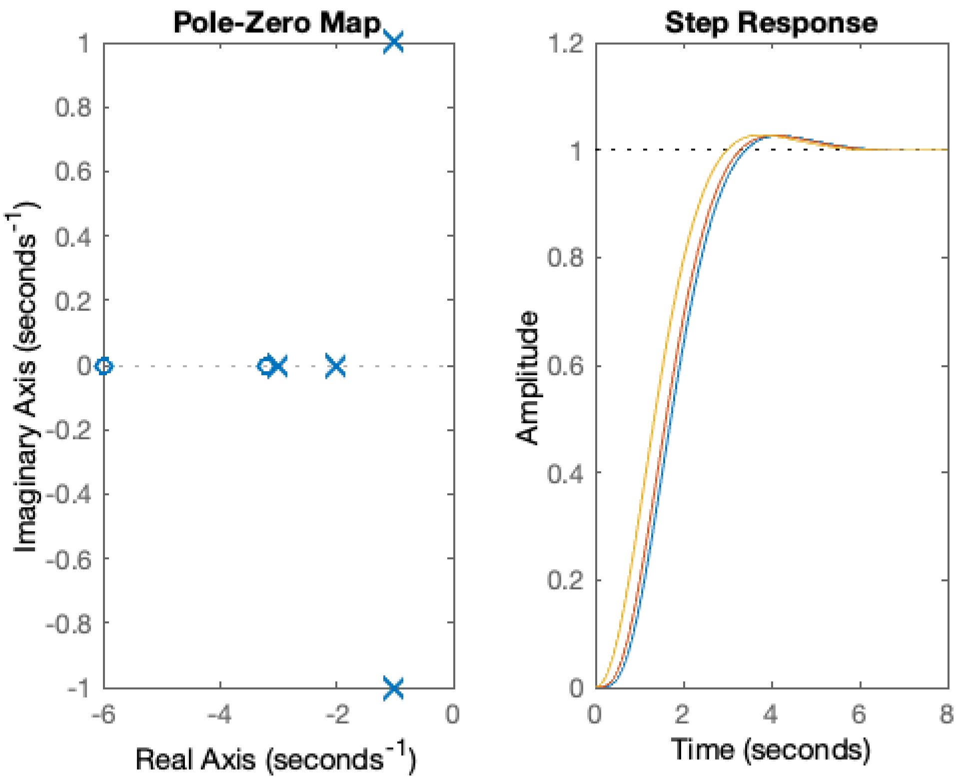
In [10]:

```
z2 = z1
p2 = p1(3:6)
g2 = zpk(z2,p2,prod(abs(p2))/prod(abs(z2)));
```

```
z2 =
   -6.0000
   -3.2000
p2 =
  -3.0000 + 0.0000i
  -2.0000 + 0.0000i
  -1.0000 + 1.0000i
  -1.0000 - 1.0000i
```

In [11]:

```
subplot(121)
pzmap(p2,z2)
subplot(122)
step(g,g1,g2)
```



Step 3: remove hf zero

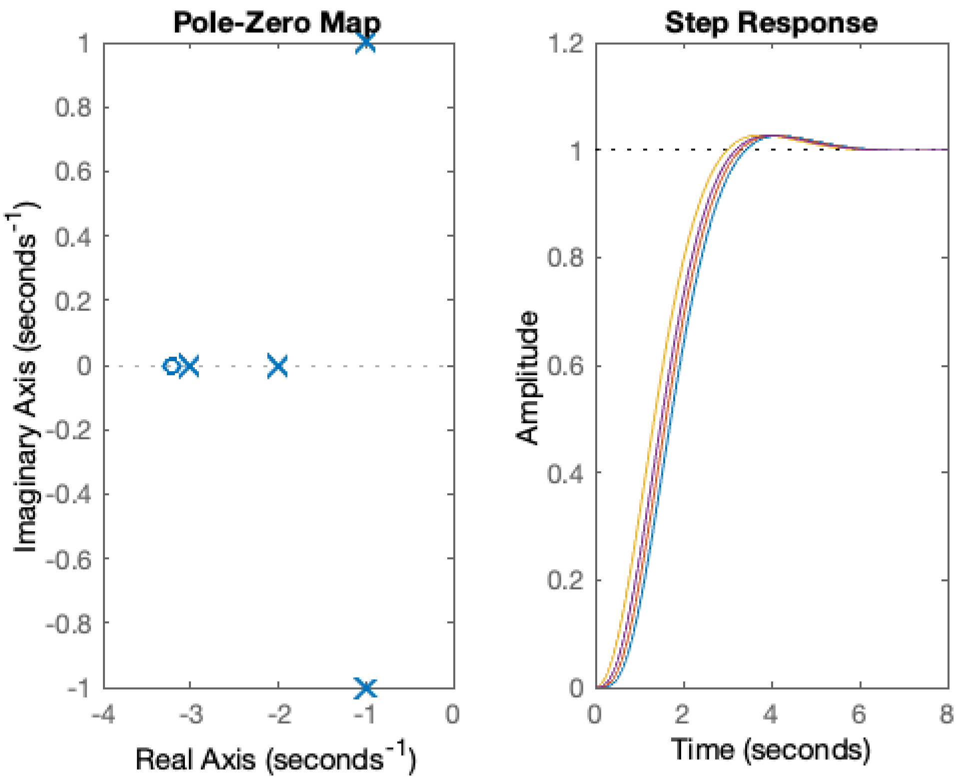
In [12]:

```
z3= z2(2)
p3 = p2
g3 = zpk(z3,p3,prod(abs(p3))/prod(abs(z3)));
```

```
z3 =
   -3.2000
p3 =
  -3.0000 + 0.0000i
  -2.0000 + 0.0000i
  -1.0000 + 1.0000i
  -1.0000 - 1.0000i
```

In [13]:

```
subplot(121)
pzmap(p3,z3)
subplot(122)
step(g,g1,g2,g3)
```



Step 4: remove pole-zero cancellation terms

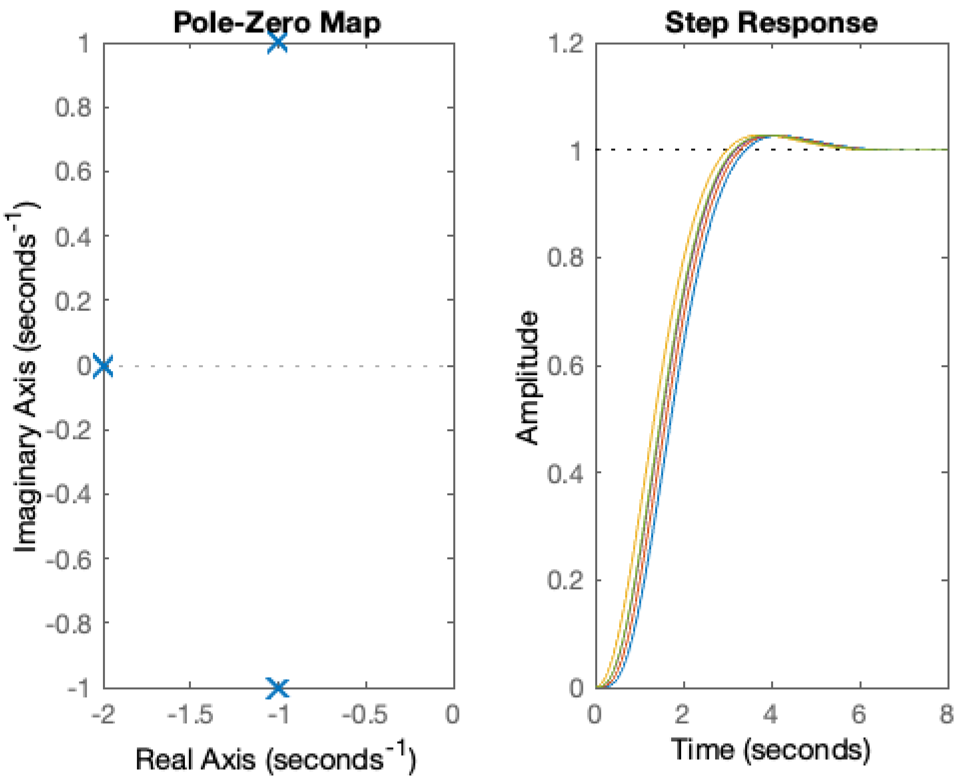
In [14]:

```
z4= []
p4 = p3(2:4)
g4 = zpk(z4,p4,prod(abs(p4))/prod(abs(z4)));
```

```
z4 =
     []
p4 =
  -2.0000 + 0.0000i
  -1.0000 + 1.0000i
  -1.0000 - 1.0000i
```

In [15]:

```
subplot(121)
pzmap(p4,z4)
subplot(122)
step(g,g1,g2,g3,g4)
```



Step 5: remove last non-dominant pole')

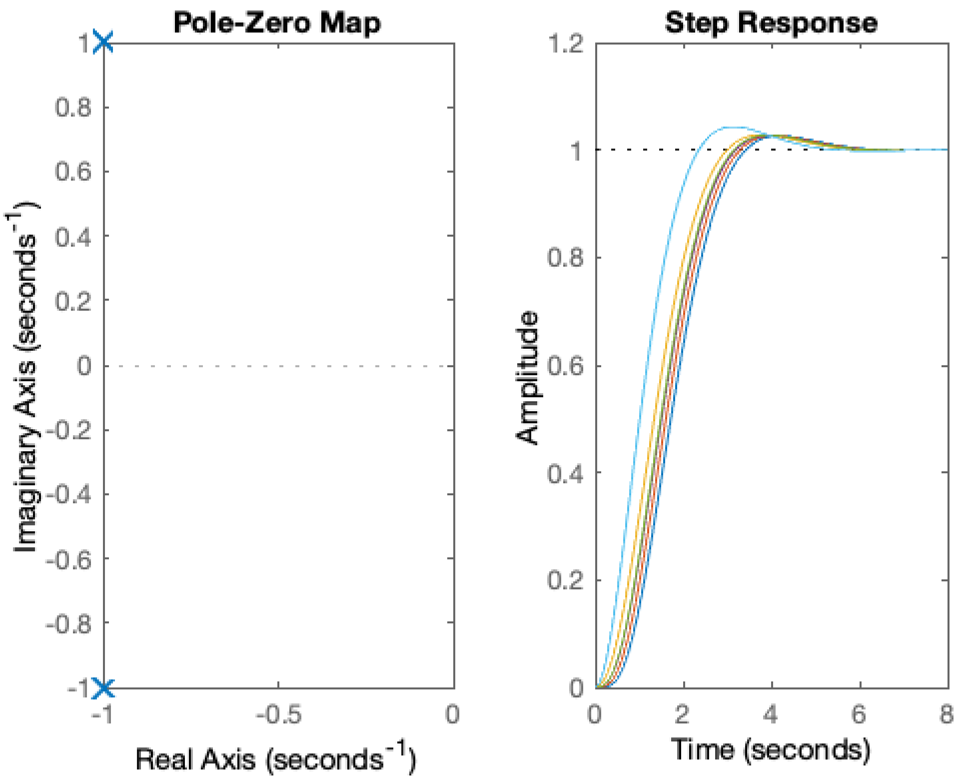
In [16]:

```
z5 = z4
p5 = p4(2:3)
g5 = zpk(z5,p5,prod(abs(p5))/prod(abs(z5)));
```

```
z5 =
     []
p5 =
  -1.0000 + 1.0000i
  -1.0000 - 1.0000i
```

In [17]:

```
subplot(121)
pzmap(p5,z5)
subplot(122)
step(g,g1,g2,g3,g4,g5)
```



Original system

In [18]:

```
g
```

```
g =
 
                298.12 (s+6) (s+3.2)
  -------------------------------------------------
  (s+9) (s+3) (s+2) (s^2 + 2s + 2) (s^2 + 14s + 53)
 
Continuous-time zero/pole/gain model.
```

Reduced order system

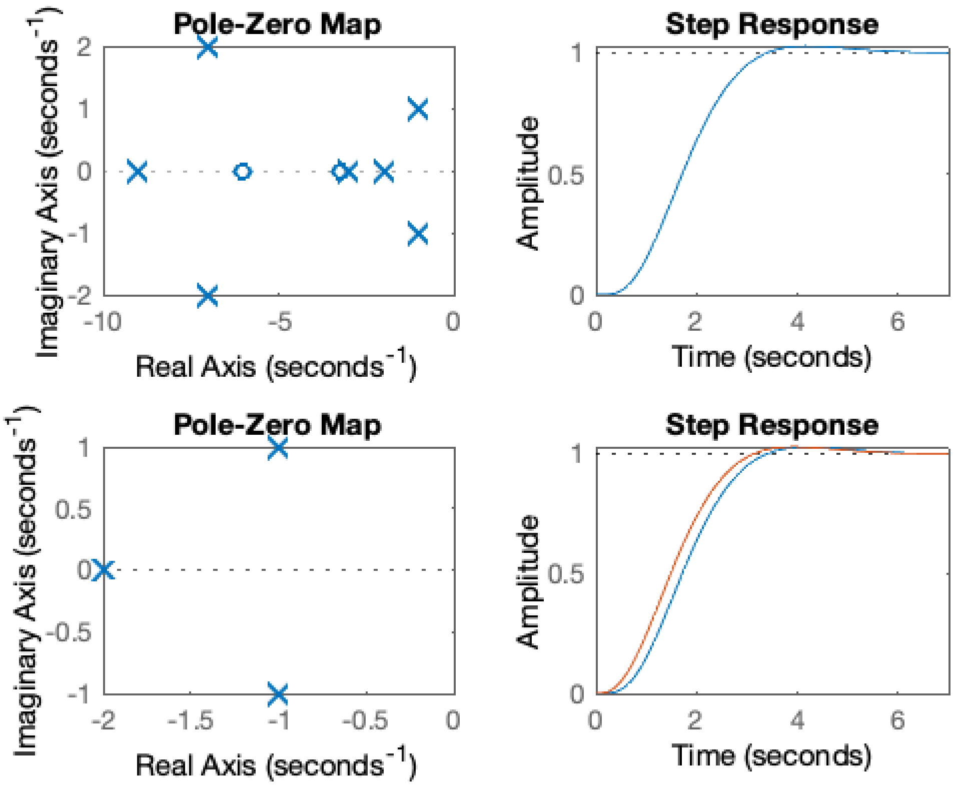
In [19]:

```
g4
```

```
g4 =

           4
  --------------------
  (s+2) (s^2 + 2s + 2)

Continuous-time zero/pole/gain model.
```

In [20]:

```
subplot(221)
pzmap(poles,zeros)
subplot(222)
step(g)
subplot(223)
pzmap(p4,z4)
subplot(224)
step(g,g4)
```



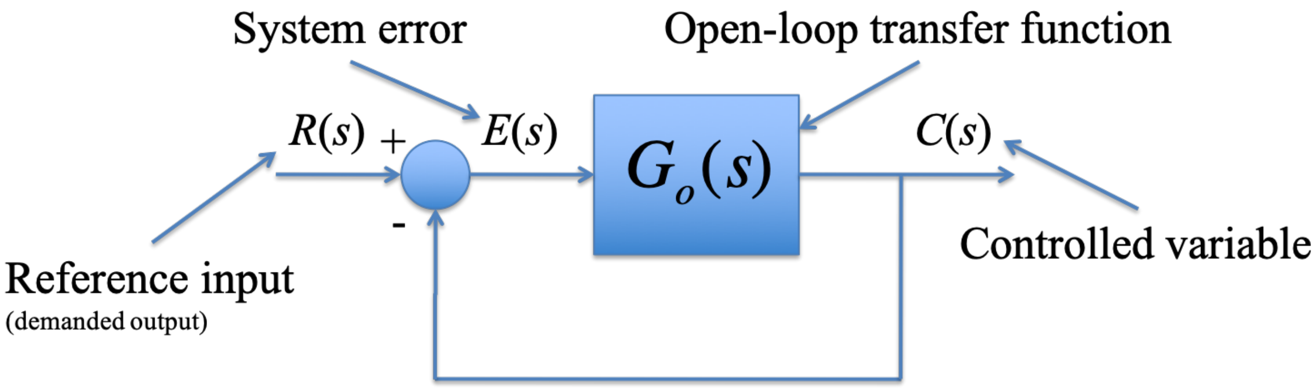
What are the steady-state performance criteria?

Steady-state response

- Canonical system
- Disturbance rejection
- System type for non-unity gain feedback

Canonical System

(unity-gain feedback)



$$E(s) = R(s) - C(s)$$

$$E(s) = \frac{G_o(s)}{1 + G_o(s)} R(s)$$

Steady-state Performance

For a unity-gain negative feedback system with open-loop transfer function Go(s) the steady-state error (SSE) response of the closed-loop system is related to system type number according to the table shown below.

		System Type Number		
		Type 0	**Type 1**	**Type 2**
Type of input	**SSE**	**Step**	**Velocity**	**Acceleration**
Step	$\frac{1}{1+K_p}$	$\frac{1}{1+K_p}$	∞	∞
Ramp	$\frac{1}{K_v}$	0	$\frac{1}{K_v}$	∞
Parabola	$\frac{1}{K_a}$	0	0	$\frac{1}{K_a}$

Position error constant for step input: $R(s) = 1/s$:

$$K_p = \lim_{s \to \infty} G_o(s)$$

Velocity error constant for ramp input: $R(s) = 1/s^2$:

$$K_v = \lim_{s \to \infty} sG_o(s)$$

Acceleration error constant for parabolic input: $R(s) = 1/s^3$:

$$K_a = \lim_{s\to\infty} s^2 G_o(s)$$

Special Cases

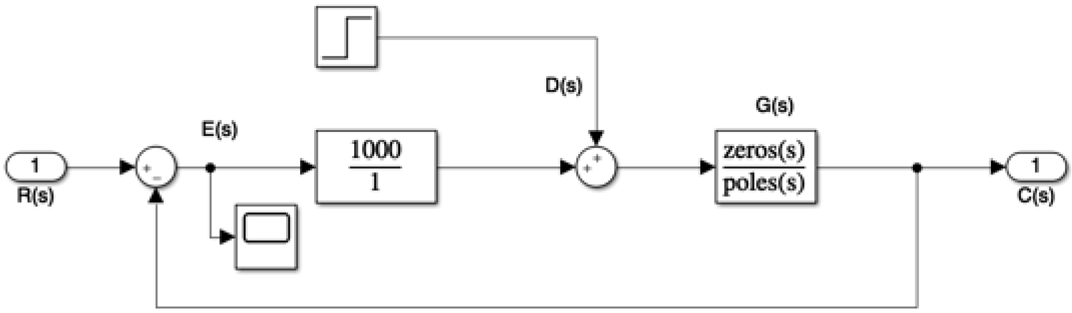
For these models calculate the error response ($E(s) = G_o(s)N_d(s)$ for the "disturbance rejection" case and $E(s) = R(s) - C(s)$ for the "non-unity-gain-feedback") case and use the final value theorem to calculate the steady state step error.

Compare your result with the result of the simulation.

You should note that in both cases the plant transfer function has type number 1. Do the rules of system type number as you understand them carry over to these special cases?

Disturbance rejection? (Compliance)

Assuming that the system is originally at steady-state ($E(s) = R(s) - C(s) = 0$) what is the steady-state error to a step change in the disturbance in $n_d(t)$? ($N_d(s) = 1/s$)

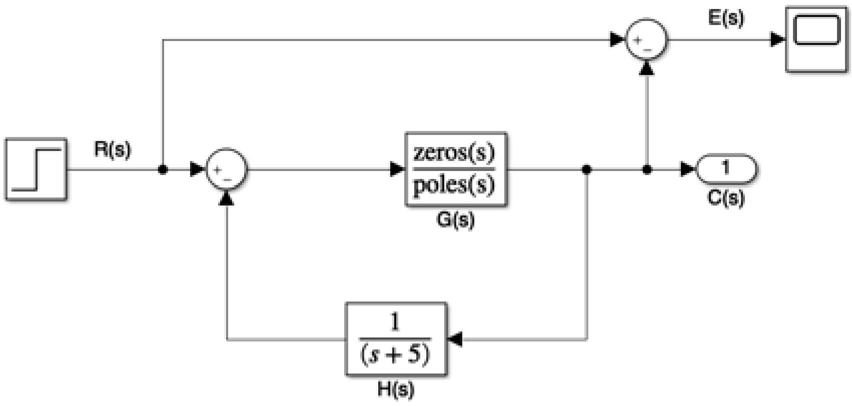


[Model file disturbance_rejection.mdl (matlab/disturbance_rejection.mdl)]

In [21]:

```
disturbance_rejection
```

Non-unity gain feedback



[Model file non_unity_gain_feedback.mdl (matlab/non_unity_gain_feedback.mdl)]

In [22]:

```
non_unity_gain_feedback
```

Further Reading

The System Metrics (https://en.wikibooks.org/wiki/Control_Systems/System_Metrics) section of the Control Systems Wikibook (https://en.wikibooks.org/wiki/Control_Systems) amplifies some of the topics covered in this chapter.

The topics covered in this chapter are also amplified in

- Nise. Chapter 4: Time Response.
- Dorf and Bishop. Chapter 5: The Performance of Feedback Control Systems.