Due Monday, November 9 EE571 HW#22

1.a) Go back to HW 21 problem 2 and help Josh and Tyler design a PD (ultimate lead compensator) to meet the transient specs given in problem 2 on HW21 for their A-6 intruder Yaw compensation problem.

b) Sketch the resulting compensated root locus of $G_{PD}(s)G(s)$

c) Design a PI (ultimate lag compensator) compensator for $G(s) = \frac{10}{s+4}$ such that the following specs are met:

1. t_s = 1 second
2. Damping coefficient of ζ = 0.707
3. $ess|_{step} = 0$

d) Sketch the resulting compensated root locus of $G_{PI}(s)G(s)$

e) Design a PID (ultimate lead-lag compensator) compensator for $G(s) = \frac{10}{(s+4)}$ such that the following specs are met:

1. t_s = 0.5 second
2. Damping coefficient of ζ = 0.707
3. $ess|_{ramp} = 0.1$

f) Sketch the resulting compensated root locus of $G_{PID}(s)G(s)$

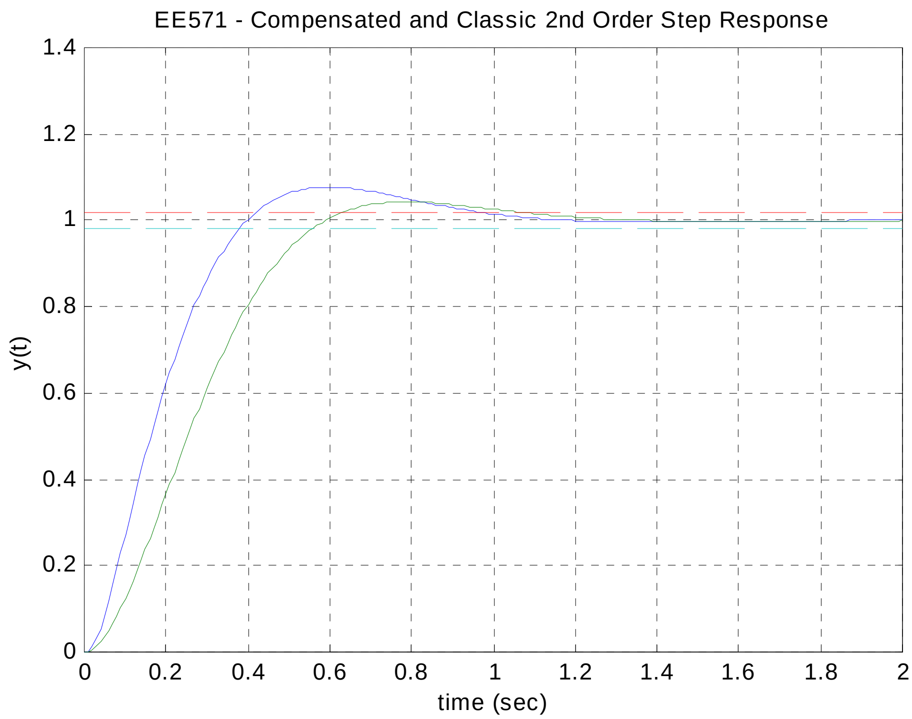
2. For my lead design in HW#21, I found the following closed-loop transfer function:

$$\frac{GcG}{1+GcG} = \frac{15872(s+6.9360)}{(s+4-j4)(s+4+j4)(s+33.59)(s+102.41)} = \frac{15872(s+6.9360)}{(s^2+8s+32)(s+33.59)(s+102.41)}$$

If my system were truly a classic second order system with dominant poles at s_1 = –4+j4 and s_2 = -4-j4, the closed-loop transfer function would be:

$$\frac{G_{classic}}{1+G_{classic}} = \frac{32}{s^2+8s+32}$$

I have gone back and plotted my lead compensated step response versus what a classic 2nd order step response would look like for the (see plot below).



a) What are the theoretical values of t_p, M_p , and t_s for dominant poles at $s_1 = -4+j4$?

b) Given your answer to part d) which plot is which (hint: the classic second-order system should have the correct overshoot and settling time)?

c) What are the values of t_p, M_p , and t_s for my lead design from the Matlab step response?

d) Note that the zero at s=-6.936 is fairly close to the dominant poles at $s_1 = -4+j4$. Given the difference between the two plots, what is your conjecture as to the effect of a zero near a pair of dominant poles?

Here's the Matlab code on how I plotted the step response. Notice, I would have used LSIM() instead of the STEP() command if my input were something other than a step! I could also obtain the same results through the attached simulink model

```
» num = 20

num =

   20

» den=poly([0,-4,-40])

den =

    1    44   160     0

» numlead=793.6*[1 6.936]

numlead =

  1.0e+003 *
```

```
    0.7936    5.5044

» denlead=[1 100]

denlead =

     1   100

» GcGnum=conv(num,numlead)

GcGnum =

  1.0e+005 *

    0.1587    1.1009

» GcGden=conv(den,denlead)

GcGden =

           1         144        4560       16000           0

» [numcloop,dencloop]=cloop(GcGnum,GcGden)

numcloop =

  1.0e+005 *

         0         0         0    0.1587    1.1009


dencloop =

  1.0e+005 *

    0.0000    0.0014    0.0456    0.3187    1.1009

» roots(dencloop)

ans =

  1.0e+002 *

  -1.0241
  -0.3359
  -0.0400 + 0.0400i
  -0.0400 - 0.0400i

» wn=sqrt(32)

wn =

    5.6569

» zeta=sqrt(2)/2
```

```
zeta =

    0.7071

» numclassic=wn^2

numclassic =

   32.0000

» denclassic=[1 2*zeta*wn wn^2]

denclassic =

    1.0000    8.0000   32.0000

» t=[0:0.01:2];
» ycloop=step(numcloop,dencloop,t);   %I would have used lsim here if my input were not step
» yclassic=step(numclassic,denclassic,t);
» plot(t,ycloop,t,yclassic,[0 2],[1.02 1.02],'--',[0 2],[.98 .98],'--')
» grid;xlabel('time (sec)');ylabel('y(t)');title('EE571 - Compensated and Classic 2nd Order Step Response')
```