06.7 rldesign.PLead Proportional-lead design

Similar to how proportional-lag controllers can be considered passively realizable PI controllers, proportional-lead controllers can be considered passively realizable PD controllers. The idea is to choose a design point ψ through which we construct the root locus to pass. As with PD control, this point is chosen to meet primarily transient response characteristics, and the controller contributes the proper phase such that the root locus passes through the point; however, we have both a pole and a zero to set in the compensator:

$$C(s) = K_2 \frac{s - z_c}{s - p_c}. \tag{1}$$

We will "arbitrarily" choose either p_c or z_c and the phase criterion for our design point ψ will set the other. However, the "arbitrary" selection of p_c or z_c in fact affects both the transient response and the steady-state error (if it is finite).

Let's work out the details. A way to approach designing a controller for a plant G with lead compensator C is to consider the compensator effects on the phase criterion, which must always be satisfied at points on the root locus:

$$\angle(G(s)C(s)) = \pi. \tag{2}$$

In order for a desired point $s = \psi$ to be on the root locus, then,[7]

$$\begin{aligned} \angle(G(\psi)C(\psi)) &= \pi \\ \angle G(\psi) + \angle C(\psi) &= \pi \Rightarrow \\ \angle C(\psi) &= \pi - \angle G(\psi) \Rightarrow \\ \angle(\psi - z_c) - \angle(\psi - p_c) &= \pi - \angle G(\psi). \end{aligned}$$

Let this angle $\angle(\psi - z_c) - \angle(\psi - p_c)$, called the **compensator angle**, be given the symbol

$$\theta_c \equiv \angle(\psi - z_c) - \angle(\psi - p_c). \tag{3}$$

[7] The 2π modulo in these expressions is suppressed for clarity.

So we can choose to arbitrarily set the location of either z_c or p_c and the other will be set by the phase criterion. Therefore we have either

$$\angle(\psi - p_c) = \underbrace{\angle(\psi - z_c)}_{\text{arbitrary}} - \theta_c \quad \text{or} \tag{4a}$$

$$\angle(\psi - z_c) = \theta_c - \underbrace{\angle(\psi - p_c)}_{\text{arbitrary}}. \tag{4b}$$

And, from trigonometry,

$$p_c = \mathrm{Re}(\psi) - |\mathrm{Im}(\psi)| / \tan(\theta_c - \angle(\psi - z_c)) \quad \text{or} \tag{5a}$$

$$z_c = \mathrm{Re}(\psi) - |\mathrm{Im}(\psi)| / \tan(\theta_c + \angle(\psi - p_c)). \tag{5b}$$

This result is to be used in the design procedure that follows.

Design procedure

The following procedure provides a starting-point for proportional-lead controller design. Let's assume the transient response requirement is such that, according to the second-order approximation, we desire a closed-loop pole to be located at $s = \psi$.

1. Design a proportional controller to meet transient response requirements by choosing the gain K_1 for the dominant closed-loop poles to be as close as possible to ψ.
2. Include a cascade lead compensator of the form

$$K_2 \frac{s - z_c}{s - p_c}, \tag{6}$$

where we arbitrarily set either z_c or p_c; initially, $K_2 = 1$. The other parameter must be chosen to satisfy Eq. 5a or Eq. 5b. For convenience, we repeat the key formulas:

$$\theta_c = \pi - \angle G(\psi) \quad \text{and, after setting arbitrarily } z_c \text{ or } p_c,$$

$$p_c = \mathrm{Re}(\psi) - |\mathrm{Im}(\psi)| / \tan(\angle(\psi - z_c) - \theta_c) \quad \text{or}$$

$$z_c = \mathrm{Re}(\psi) - |\mathrm{Im}(\psi)| / \tan(\theta_c - \angle(\psi - p_c)).$$

3. By construction ψ is on the root locus, so the gain can be computed directly from Eq. 2:

$$K_2 = \frac{1}{|K_1 C(\psi) G(\psi)|}. \tag{7}$$

4. Construct the closed-loop transfer function with the controller

$$K_1 K_2 \frac{s - z_c}{s - p_c}. \tag{8}$$

5. Simulate the time response to see if it meets specifications. Tune.

A design example

Let a system have plant transfer function

$$\frac{37500}{s^4 + 70s^3 + 1625s^2 + 14000s + 37500}. \tag{9}$$

Design a P-lead controller such that the closed-loop settling time is about 0.4 seconds and the overshoot is about 10%.

Determining ψ

We use Matlab for the design.[8] First, we must determine what the specified transient response criteria imply for the locations of our closed-loop poles. Let one of these desired pole locations be called ψ. The transient response performance criteria are as follows.

```
Ts = .4; % sec ... spec settling time
OS = 10; % percent ... spec overshoot
```

The second-order approximation from Chapter 03 trans tells us that the settling time specification implies a specific $\mathrm{Re}(\psi)$ and the overshoot a specific angle $\angle\psi$. From previous results, the desired pole location ψ (assuming the second-order approximation is valid) is given by the expression

$$\psi = -\frac{4}{T_s}\left(1 - j\frac{\pi}{\ln(100/\%OS)}\right). \tag{10}$$

[8] See ricopic.one/control/source/plead_controller_design_example.m for the source.

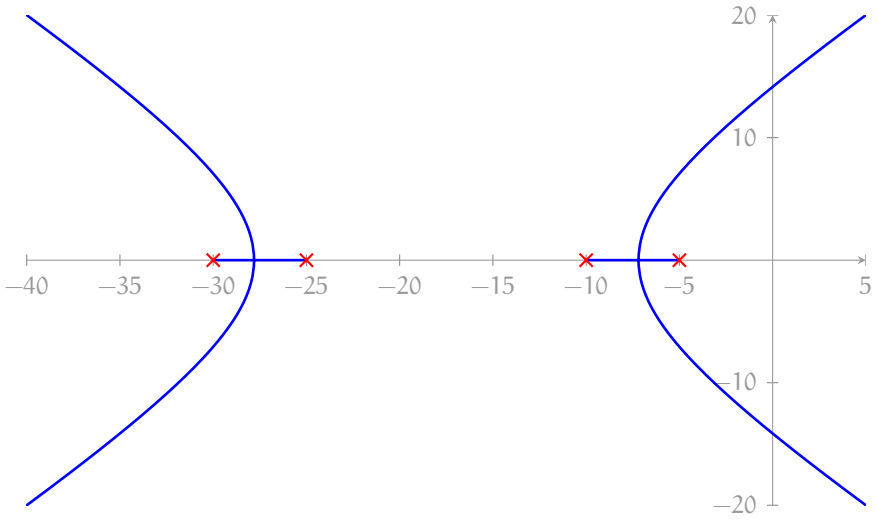

Figure PLead.1: root locus without compensation.

This formula holds beyond the scope of this problem. We define it as an anonymous function.

```
psi_fun = @(Ts,pOS) -4/Ts*(1-1j*pi/log(100/pOS));
psi = psi_fun(Ts,OS);
disp(sprintf('psi = %0.3g + j %0.3g',real(psi),imag(psi)))
```

```
psi = -10 + j 13.6
```

P control

We design a proportional controller that gets us as close as possible to ψ. The root locus is shown in Figure PLead.1.

```
G = tf([37500],[1,70,1625,14000,37500]);
figure
rlocus(G)
```

Although we cannot get close to ψ on the root locus, we can at least meet our %OS specification by choosing a gain of about

$$K_1 = 1.1. \tag{11}$$

Let's construct the compensator and corresponding closed-loop transfer function G_P for gain control.

```
K_1 = 1.1;
G_P = feedback(K_1*G,1);
```

Lead compensation

Now, we use cascade derivative compensation with compensator

$$K_2 \frac{s - z_c}{s - p_c}. \tag{12}$$

For now, we set $K_2 = 1$. Let's also set $p_c = -40, -100$, and -400 to see how we fair with different "arbitrary" choices. From Eq. 5b, we compute the compensator zero

$$\theta_c = \pi - \angle G(\psi) \qquad \text{and} \qquad z_c = \mathrm{Re}(\psi) - |\mathrm{Im}(\psi)| / \tan(\theta_c + \angle(\psi - p_c)).$$

```
p_c = [-40,-100,-400];
theta_c = pi - angle(evalfr(G,psi));
theta_p_c = angle(psi*ones(size(p_c))-p_c);
z_c = real(psi) - abs(imag(psi))./tan(theta_c + theta_p_c);
disp(sprintf('theta_c = %0.3g deg',rad2deg(theta_c)))
for i = 1:length(p_c)
  disp(sprintf(...
    'pole phase contribution = %0.3g deg',...
    rad2deg(theta_p_c(i))...
  ))
  disp(sprintf('z_c = %0.3g',z_c(i)))
end
```

```
theta_c = 96.7 deg
pole phase contribution = 24.5 deg
z_c = -1.75
```

```
pole phase contribution = 8.62 deg
z_c = -6.26
pole phase contribution = 2 deg
z_c = -7.91
```

By construction, ψ is on the root locus, so we can find K_2 directly from Eq. 2.

```
C_sans = stack(1,tf(1,1)); % initialize model array
C = stack(1,tf(1,1)); % initialize model array
for i = 1:length(p_c)
  C_sans(i) = zpk(z_c(i),p_c(i),1); % without gain
  K_2(i) = 1/abs(evalfr(K_1*C_sans(i)*G,psi));
  C(i) = K_1*K_2(i)*C_sans(i);
  disp(sprintf('K_2 = %0.3g',K_2(i)))
end
```

```
K_2 = 4.88
K_2 = 15.2
K_2 = 66.7
```

Let's compute the closed-loop controller C_{lead}, and the closed-loop transfer function G_{lead}.

```
G_Plead = stack(1,tf(1,1));
for i = 1:length(p_c)
  G_Plead(i) = feedback(C(i)*G,1);
end
```

Simulate

Our placement of the ψ depended on the second-order approximation's accuracy, which in this case is questionable. In any case, we simulate the step response to test the efficacy of the P-lead controller design and to compare it with the P controller.

```
t_a = linspace(0,2.5,200); % s ... sim time
y_P = step(G_P,t_a); % P controlled step response
for i = 1:length(p_c)
```

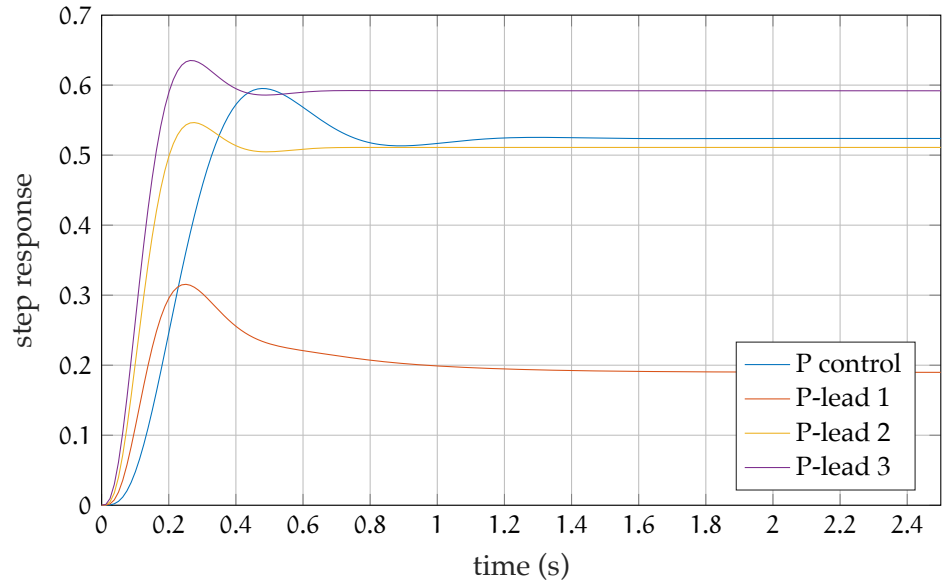

Figure PLead.2: step responses for proportional and proportional-lead controllers.

```
  y_Plead(:,i) = step(G_Plead(i),t_a); % P-lead step resp.
end
```

```
figure
plot(t_a,y_P);
hold on;
for i = 1:length(p_c)
  plot(t_a,y_Plead(:,i));
end
xlabel('time (s)');
ylabel('step response');
grid on
legend(...
  'P control','P-lead 1','P-lead 2','P-lead 3',...
  'location','southeast'...
);
```

The responses, shown in Figure PLead.2, suggest the lead-compensated controllers are at least close to meeting the transient specifications. The

steady-state error is worse for compensator locations that are less-negative and better for those that are more-negative. For this reason, we remember that our "arbitrary" choice of one of our compensator parameters still affects the steady-state (and sometimes transient) response. Let's use `stepinfo` to compute more accurate transient response characteristics for the different controllers.

```
disp('P control')
si_P = stepinfo(y_P,t_a);
disp(sprintf('settling time: %0.3g',si_P.SettlingTime))
disp(sprintf('percent overshoot: %0.3g\n',si_P.Overshoot))
for i = 1:length(p_c)
  si_Plead = stepinfo(y_Plead(:,i),t_a);
  disp(sprintf('p_c: %0.3g',p_c(i)))
  disp(sprintf(...
    'settling time: %0.3g',si_Plead.SettlingTime ...
  ))
  disp(sprintf(...
    'percent overshoot: %0.3g\n',si_Plead.Overshoot...
  ))
end
```

```
P control
settling time: 0.906
percent overshoot: 13.6

p_c: -40
settling time: 1.28
percent overshoot: 66.2

p_c: -100
settling time: 0.371
percent overshoot: 6.95

p_c: -400
settling time: 0.37
percent overshoot: 7.31
```

We see that most of the P-lead controllers meet the settling time and percent overshoot requirements. However, the first one is problematic. This is

mostly due to the second-order approximation being significantly violated in this case. We see from the time response that the initial overshoot happens quickly, but the return to steady-state is slow. If desired, the gain K_2 and compensator pole and zero locations could be tuned, iteratively.