MCE 441/541

HW#4

Part 1

In the Simulink file provided: First set the integral and the derivative parameters in the PID controller to zero, which turns the PID controller into P controller. Insert the numerator and denominator values, which are [1 1] and [1 10 24 23 14] respectively in the transfer function block. Then change the proportional parameter in the PID controller and plot the result, \( y \), against \( t \) in Matlab. Using trial-and-error to obtain the best plot with constant amplitude. (See Figure 1) Doing so, obtains the best proportional parameter value equal to 123. This value is \( K_u \).

Find \( T_u \) by using command “ginput(n)” set \( n=2 \) for input of two points. It is easier if the graph is zoomed in to the two peaks that you want to click before typing in the command. (See Figure 2) Upon clicking the two points, the coordinates of these points are obtained. Subtract the x-coordinate to get the period \( T_u \), which is 14.073 - 12.4288 = 1.6442.
With the values of $K_u = 123$ and $T_u = 1.6442$, complete the Ziegler-Nichols PID Tuning table.

<table>
<thead>
<tr>
<th></th>
<th>P</th>
<th>PI</th>
<th>PID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kp</td>
<td>61.5</td>
<td>55.35</td>
<td>73.8</td>
</tr>
<tr>
<td>Ki</td>
<td>0</td>
<td>40.4</td>
<td>89.77</td>
</tr>
<tr>
<td>Kd</td>
<td>0</td>
<td>0</td>
<td>15.17</td>
</tr>
</tbody>
</table>

Set the tuned gains in the PID block and simulate. Plot unit step response for each tuned gains. Find settling time and percent overshoot for each plot. (See Figure 3-8)

```matlab
plot(t,y)
xlabel('Time')
ylabel('Amplitude')
title('P')
hold on

% Find percent overshoot: $((peak\ value - final\ value)/final\ value)*100$
((1.403-.8146)/.8146)*100

% Find settling time
plot([0 50],[.8146*.98 .8146*.98 ],'r-')
plot([0 50],[.8146*1.02 .8146*1.02],'r-')
% Then use zoom in and data cursor to find settling time
```
Percent overshoot is 72.2318%

Settling time is 11.09 sec.
plot(t,y)
xlabel('Time')
ylabel('Amplitude')
title('PI')
hold on

%Find percent overshoot: ((peak value - final value)/final value)*100
((1.757-1)/1)*100

%Find settling time
plot([0 50],[.98 .98 ],'r-')
plot([0 50],[1.02 1.02],'r-')

%Use zoom in and data cursor to find settling time

Figure 5:
Percent overshoot is 75.7%

Settling time is 46.07 sec.

```matlab
plot(t,y)
xlabel('Time')
ylabel('Amplitude')
title('PID')
hold on

%Find percent overshoot: ((peak value - final value)/final value)*100
((1.611-1)/1)*100
%Find settling time
plot([0 50], [.98 .98], 'r-')
plot([0 50], [1.02 1.02], 'r-')
%Use zoom in and data cursor to find settling time
```
Percent overshoot is 61.1%
Settling time is 5.69 sec.

PID controller would be best for this system since it produces the least percent overshoot and settling time.

Use “sisotool” to fine tuning the system by import the system into the SISO design tool. First set up the system in the transfer function format.

```matlab
num=[1 1];
den=[1 10 24 23 14];
System=tf(num,den)
sisotool
```

Then import the system into SISO by click File>Import. Then choose “System” in the SISO Models block, click the arrow to import it into the plant. Click OK. The system will be plotted as Root Locus and Open-Loop Bode. (See Figure 9).

![Figure 9: System Root Locus and Open-Loop Bode Plot](image)

To show a plot of step response click Analysis->Response to Step Command. Right click on the plot choose Systems and uncheck Closed-Loop: r to u. Right click again, choose Characteristics and check Settling Time and Peak Response. Click on the dots in the graph to see the system data. (See figure 10)
The object is to keep the controller transfer function as simple as possible, while achieving less settling time and percent overshoot. First, an extra integrator is needed to achieve the steady-state error requirement. Right click on Root Locus plot choose Add Pole/Zero>Integrator. Then, adding a pair of complex zeros near the complex poles to improve the settling time. Grab the red dot with the mouse to change the gain value to obtain the best settling time and percent overshoot by looking at the step response plot. (See Figure 11-12). Obtaining settling time of 1.56 seconds and percent overshoot of 1.22% after fine tuning the system. By matching

\[ G_c(s) = \frac{K_d s^2 + K_p s + K_i}{s} \]

with C(s) in Figure 11, obtains \( K_d = 49.04 \), \( K_p = 40.257 \), \( K_i = 56.7 \).
Figure 11: Root Locus and Open-Loop Bode plot of the system after fine tuning

Figure 12: Step response of the system after fine tuning
Part 2

From the best tuning in part 1,

\[ \frac{K_p}{K_d} = \frac{40.257}{49.04} = 0.821 \]
\[ \frac{K_i}{K_d} = \frac{56.7}{49.04} = 1.156 \]

Obtaining the characteristic equation:

\[ 1 + GK = 0 \]

Where \( G = \frac{s+1}{(s^2+s+1)(s+1)(s+7)} \) and \( K = \frac{K_d(s^2+\left(\frac{K_p}{K_d}\right)s+\left(\frac{K_i}{K_d}\right))}{s} = \frac{K_d(s^2+0.821s+1.156)}{s} \)

Poles = 0, -2, -7, -0.5±0.866i

Zeros = -1, -0.4105±0.9937

Finding center of asymptotes from

\[ \sigma_A = \frac{\sum\text{Poles} - \sum\text{Zeros}}{n-m} = \frac{-0.0895}{5-3} = -0.44 \]

Finding angle of the asymptotes from

\[ \phi_A = \frac{180(2K+1)}{n-m} \text{ where } K=0 \text{ and } 1. \]

\[ \phi_A = 90^\circ \text{ and } 270^\circ \]

Determine the breakaway points. From the characteristic equation \( 1+GK=0 \) solve for \( K_d \), which is

\[ K_d = \frac{-s(s^2+s+1)(s+2)(s+7)}{(s+1)(s^2+0.821s+1.156)} \]

Differentiate \( K_d \) with respect to \( s \) by using Matlab and set it equal to zero. Then find the roots of this equation. The real value of the root is the break-away point, since it is the only \( s \) value that lies on the real axis. (See Matlab commands and results below).

\[ \text{>> syms } s \]
\[ \text{>> Kd}=(-s*(s^2+s+1)*(s+2)*(s+7))/((s+1)*(s^2+0.821*s+1.156)); \]
\[ \text{>> DKd}=\text{diff}(\text{Kd},s); \]
\[ \text{>> simplify}(\text{DKd}) \]

\[ \text{ans} = \]
\[ -1000*(16184+53176*s+103209*s^2+113136*s^3+44328*s^4+2000*s^5+15463*s^6+85794*s^7)/(s+1)^2/(1000*s^2+821*s+1156)^2 \]
The breakaway point is at -4.3389. Figure 11 verifies this result. See Figure 13 for a detailed hand-sketch of the root locus.

There is no positive value of Kd that will destabilize the closed loop since all values of Kd are in the left hand plane.

Part 3
From $K = \frac{K_d(s^2 + \frac{K_p}{K_d}s + \frac{K_i}{K_d})}{s} = K_d s + K_p + K_i/s$, where $K_d = 49.04$.

Obtain the characteristic equation $1 + GK = 0$ with $G = \frac{s+1}{(s^2+s+1)(s+2)(s+7)}$.

Simplify the equation to find the coefficient of each $s$ term.

```matlab
syms Kp Ki s
K = (49.04*s + Kp + Ki/s);
G = (s+1)/((s^2+s+1)*(s+2)*(s+7));
ChaEq = 1 + (K*G);
simplify(ChaEq)
```

```
ans =
1/25*(25*s^5+250*s^4+1826*s^3+1801*s^2+350*s+25*Kp*s^2+25*Kp*s+25*Ki*s+25*Ki)/s/(s^2+s+1)/(s+2)/(s+7)
```

Rewrite the equation:

$$(25s^5) + (250s^4) + (1826s^3) + (1801 + 25Kp*s^2) + (350 + 25Kp + 25Ki*s^1) + (25Ki) = 0$$

Divided through by 25:

$s^5 + 10s^4 + 73.04s^3 + (72.04 + Kp*s^2) + (14 + Kp + Ki*s^1) + Ki = 0$

Routh-Hurwitz Table

<table>
<thead>
<tr>
<th>$s^5$</th>
<th>$s^4$</th>
<th>$s^3$</th>
<th>$s^2$</th>
<th>$s^1$</th>
<th>$s^0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10</td>
<td>[73.04-(72.04+Kp)]/10 = B1</td>
<td>[10*(14+Kp+Ki)-Ki]/10 = B2</td>
<td>Ki</td>
<td>Ki</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[B1*(72.04+Kp) - 10*B2]/B1 = C1</td>
<td>Ki</td>
<td>Ki</td>
<td>Ki</td>
</tr>
<tr>
<td></td>
<td></td>
<td>[((C1<em>B2)-Ki</em>B1)]/C1 = D1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ki</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

B1, C1, D1, and Ki values need to be more than zero.
Create Matlab commands that will plot the allowable values of Kp and Ki. Figure 14 shows the plot of these commands.

```matlab
for Kp=-1000:10:1000
    for Ki=-1000:10:1000
        C1=(((730.4-(72.04+Kp))/10)*(72.04+Kp)-(10*(14+Kp+Ki)-Ki))/((730.4-(72.04+Kp))/10);
        if ( (Kp<658.36) && (((730.4-(72.04+Kp))/10)*(72.04+Kp)-(10*(14+Kp+Ki)-Ki))/((730.4-(72.04+Kp))/10) > 0) && (((10*(14+Kp+Ki)-Ki)/(730.4-(72.04+Kp))/10)*((730.4-(72.04+Kp))/10)*(72.04+Kp)-(10*(14+Kp+Ki)-Ki))/((730.4-(72.04+Kp))/10) > 0) && (Ki > 0)
            plot(Ki,Kp,'x')
        end
    end
end

title('Allowable values of Kp and Ki')
xlabel('Ki')
ylabel('Kp')
```

Figure 14: 2D shaded region of allowable values Kp and Ki for stability