Assignment 4

March 26, 2024

In this assignment we will design a controller for an airplane elevator. The elevator is used to control the pitch of the aircraft which in turn determines the altitude of the aircraft. We wish to design a controller for the elevator such that the aircraft is able to:

- maintain its elevation even in the presence of disturbances such as wind and sudden wind gusts,
- increase altitude of 1 m in 10s (the settling time to a unit step of the closed loop system should be < 10 s),

You must ensure that your design is as robust as possible. Specifically, it must have a stability margin ≤ 1.5.

For this assignment you can use any controller or approach that you have learned about in this course. The transfer function of the airplane altitude vs. elevator angle is given by:

$$P(s) = \frac{0.4}{s^3 + 0.1s^2 + 20s + 0.5}.$$

This model is linearized about a specific velocity and an elevator angle of 0 degrees. This means that the model is only valid for elevator angles between -15 and 15 degrees. When you are validating your design, check the magnitude of the input. If it is much larger than ± 15 degrees for the flight path and the disturbances that are provided in the code below, then you may want to revisit your design.

Note that for a unit step, the input might be outside of the 15 degree bounds. This is OK. The important part is to be (largely) within the bounds for the given reference flight path and the disturbance signal.

Depending on your design, you may want to implement PD controllers. The Python package does not let you implement non-proper transfer functions. This means that you cannot directly implement a PD controller. However, recall the modification to the PD controller that we discussed in class. Thus, when implementing a PD controller $K(s) = K_d s + K_p$, use the modified version $K(s) = \frac{K_d s}{0.01s+1} + K_p$

You must submit a report that shows:

1. the steps you took to design your controller,
2. an analysis of your design which includes:

- Gang of 4 plots,
- step response of closed loop system,
- example of following a specified flight pattern (see code below for specific flight pattern),
- example of response to a gust of wind (see code below).

With your submission, include a text file with your controller in the following format:

```
numerator = []
denominator = []
```

I will write a simulator to make a short video of all the controllers that are submitted. Submissions will be anonymized in the video.

Code Snippets

For plotting of the Gang of Four use the following code snippet:

```
import enel441_utilities as eu
fig, ax = eu.plot_gang_of_four(P,[K]) #P is the plant, K is your controller
```

Code snippet to plot step response of closed loop system:

```
L = P*K
T = L/(1+L)

t,y = ct.step_response(T)
fig,ax = plt.subplots(1)
ax.plot(t,y)
ax.set_title('Step Response of Closed-Loop System')
ax.set_xlabel('Time (s)')
ax.set_ylabel('Elevation increase (m)' )
```

Code snippet to plot input and output of plant in response to a disturbance:

```
N = 1000
t = np.linspace(0,80,N)

S = 1/(1+P*K)

gust = np.zeros(N)
gust[200:210] = 1*np.ones(10)

t = np.linspace(0,80,N)
```

```
t,y = ct.forced_response(S*P,T=t,U=gust)
t,u = ct.forced_response(S,T=t,U=gust)
fig, ax = plt.subplots(2,1)
fig.tight_layout(pad=5.0)
ax[0].plot(t,gust, label='Gust')
ax[0].plot(t,y, label='Elevation of Aircraft')
ax[0].set_title('Response of aircraft to disturbances')
ax[0].set_xlabel('Time (s)')
ax[0].set_ylabel('Change in Elevation (m)')
ax[0].legend()

ax[1].plot(t,u, label='Elevator Angle')
ax[1].set_title('Input to Plant')
ax[1].set_xlabel('Time (s)')
ax[1].set_ylabel('Angle (degrees)')
```

Code snippet to plot tracking of a specified flight pattern:

```
import scipy as sp
N = 10000
b1, a1 = sp.signal.butter(5, 0.001, 'low')
r = 100*sp.signal.lfilter(b1, a1, np.random.randn(N))


t = np.linspace(0,600,N)

t,y = ct.forced_response(T,T=t,U=r)
t,u = ct.forced_response(K*S,T=t,U=r)
fig, ax = plt.subplots(2,1, figsize=(6,6))
fig.tight_layout(pad=5.0)
ax[0].plot(t,r, label='Desired Elevation')
ax[0].plot(t,y, label='Elevation of aircraft')
ax[0].set_title('Response to Reference')
ax[0].set_xlabel('Time (s)')
ax[0].set_ylabel('Deviation in Elevation (m)')
ax[0].legend()

ax[1].plot(t,u, label='Elevator Angle')
ax[1].set_title('Input to Plant')
ax[1].set_xlabel('Time (s)')
ax[1].set_ylabel('Elevator Angle (degrees)')
```