Quintin Nelson

Kade Carlson

AERSP 304

4/25/22

# Project 3 Report

Participation:

Quintin Nelson: 50%

Kade Carlson: 50%

We pledge that we have neither given nor received assistance on this project.

Signed: Quintin Nelson

Signed: Kade Carlson

**Problem Formulation:**

The objective of this project was to design a pitch motion controller for an aircraft. To design the controller, a transfer function of the system was obtained. The equations of motion for the aircraft were then rewritten in state-space form. MATLAB commands ss and ss2tf were then used to find the open-loop transfer function. This transfer function was then used to find the system's poles, zeros, and output response for a step input. The closed-loop transfer function was then found, and the step response was graphed. The step response was additionally found explicitly by finding a function y(t), which made a graph that was then used for comparison. The MATLAB function sisotool was then used to view response over time and the root locus of the system. A lead compensator was then added, and several K gain values were plotted to determine if the compensator could be used to satisfy certain requirements.

**Solution Methodology:**

The following equation of motion of an aircraft were considered:

$$\dot{\alpha} = -0.313\alpha + 56.7q + 0.232\delta_e$$
$$\dot{q} = -0.0139\alpha - 0.426q + 0.0203\delta_e$$
$$\dot{\theta} = 56.7q$$

Where α is the angle of the attack, q is the pitch rate and θ is the pitch angle. δe represents the elevator input.

A state vector $\bar{x} = \{\alpha, q, \theta\}^T$ was then defined, as shown in Figure 1. This was then put into state-space form, where A, B, C, and D are defined.

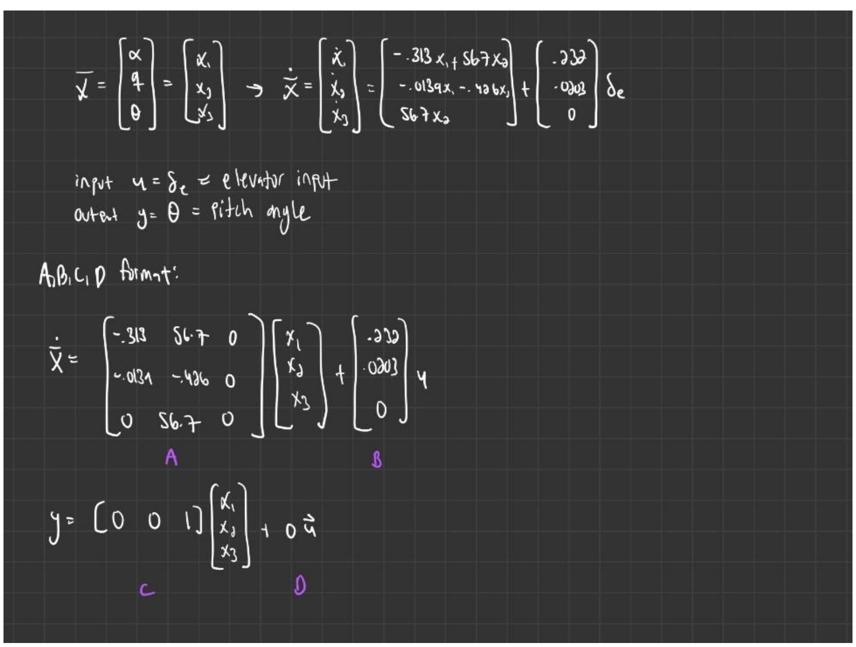


Figure 1. State-space system

The A, B, C, and D matrices can then be used to create a transfer function, which was done in MATLAB.

The four matrices were defined and the ss command was used to create a state space system. The ss2tf and the tf functions was then used to convert the state space into a transfer function. This was an open-loop transfer function

$$G(s) = \frac{\Theta(s)}{\Delta e(s)} = \frac{Y(s)}{U(s)}$$

where Y is the output and U is the input. The equation for G can be seen in Figure 2.

The tftzp command was then used to compute the zeros, poles, and associated K value from the open-loop transfer function. The output response from a .2 radian step input was then graphed using the step command.

Next, assuming unity feedback where K = 1, the closed-loop transfer function was then found using the feedback command. The diagram of the system is shown in Figure 2.

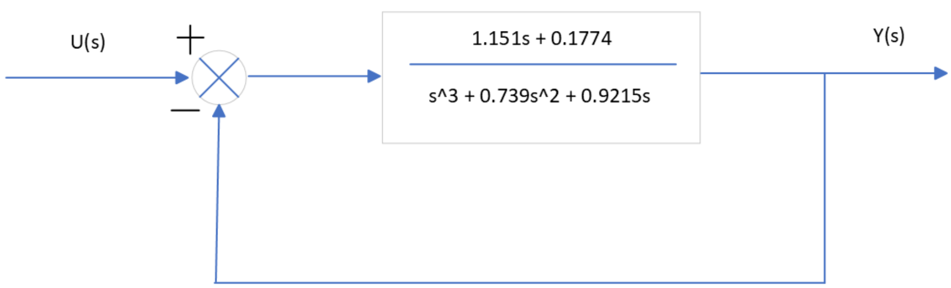


Figure 2. Transfer function diagram

The closed-loop transfer function then resulted in the following equation:

```
          1.151 s + 0.1774
-----------------------------------
s^3 + 0.739 s^2 + 2.072 s + 0.1774
```

The .2 radian step input was then graphed again for the closed loop transfer function.

The closed-loop system could also be solved for and graphed explicitly by solving an equation for y in terms of t. To do this, the closed-loop equation was rewritten as

$$G\_cl(s) = \frac{Y(s)}{U(s)}$$

$$Y(s) = G\_cl(s)U(s)$$

where

$$U(s) = \frac{.2}{s}$$

The residue command was then used to break up the Y(s) equation using partial fraction decomposition. The inverse laplace transform of the resulting equation was then taken, resulting in the following equation:

$$e^{-.3255t}\left[2\cdot\left(-.08\sin(1.3816t) - .2802\cos(1.3816t)\right)\right] - .4395e^{-.0881t} + 1$$

This was then graphed and compared to the MATLAB-generated graph.

Finally, the sisotool function in MATLAB was used to design the controller. The response graphs and the root locus was focused on to discuss changes to the system.

The following design requirements were considered for this simulation:

Table 1. Design Requirements

| Requirement | Value |
|---|---|
| Settling Time | Less than 10 seconds |
| % Overshoot | Less than 10% |
| Natural frequency | At least 0.9 rad/s |

These were placed onto the root locus to determine if the system could meet the requirements.

A lead compensator was then added to the system, adding a pole at -3 and a zero at -0.9, using the form

$$C(s) = K\frac{s+z}{s+p}$$

where K was then varied to be 2, 50, and 200.

**Results and Discussion:**

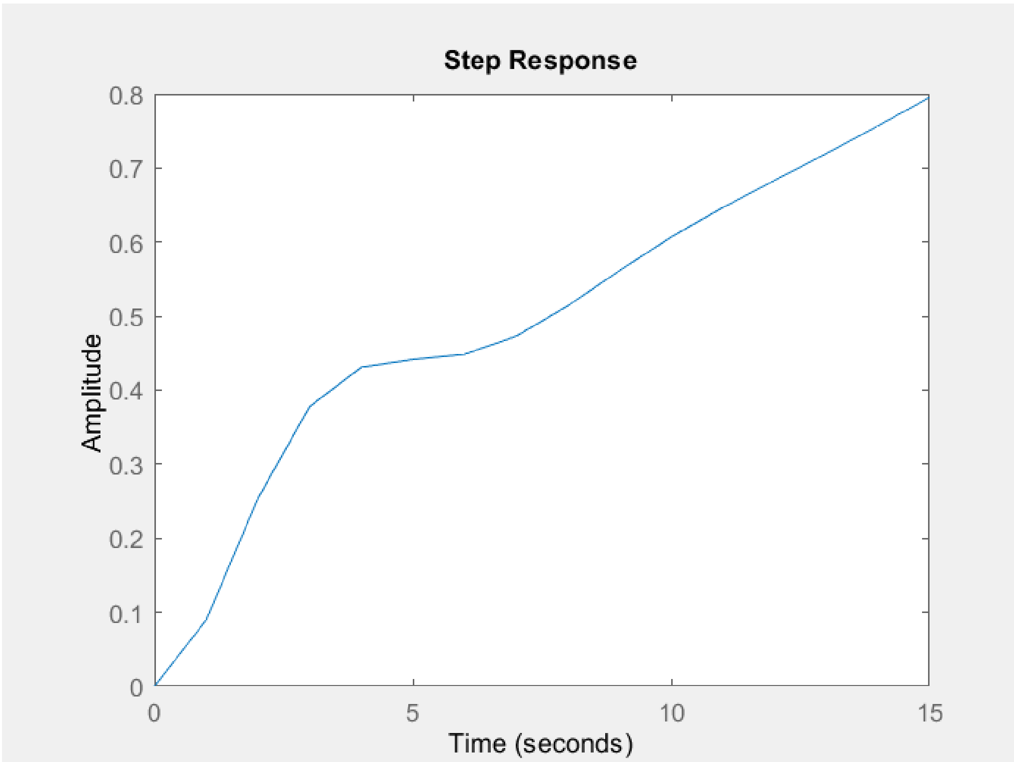


Figure 3: Amplitude vs. Time (s)

The open loop transfer function has no feedback so it will continue forever as shown in the figure. The amplitude will go on forever with no error correction. Therefore, the open loop transfer function was put into a closed-loop feedback system to achieve the desired amplitude.

Table 2: Zeros and Poles for open-loop transfer function

| Zeros | Poles |
|---|---|
| -0.1541 | 0 |
| - | -0.3695 + 0.8860i |
| - | -0.3695 - 0.8660i |

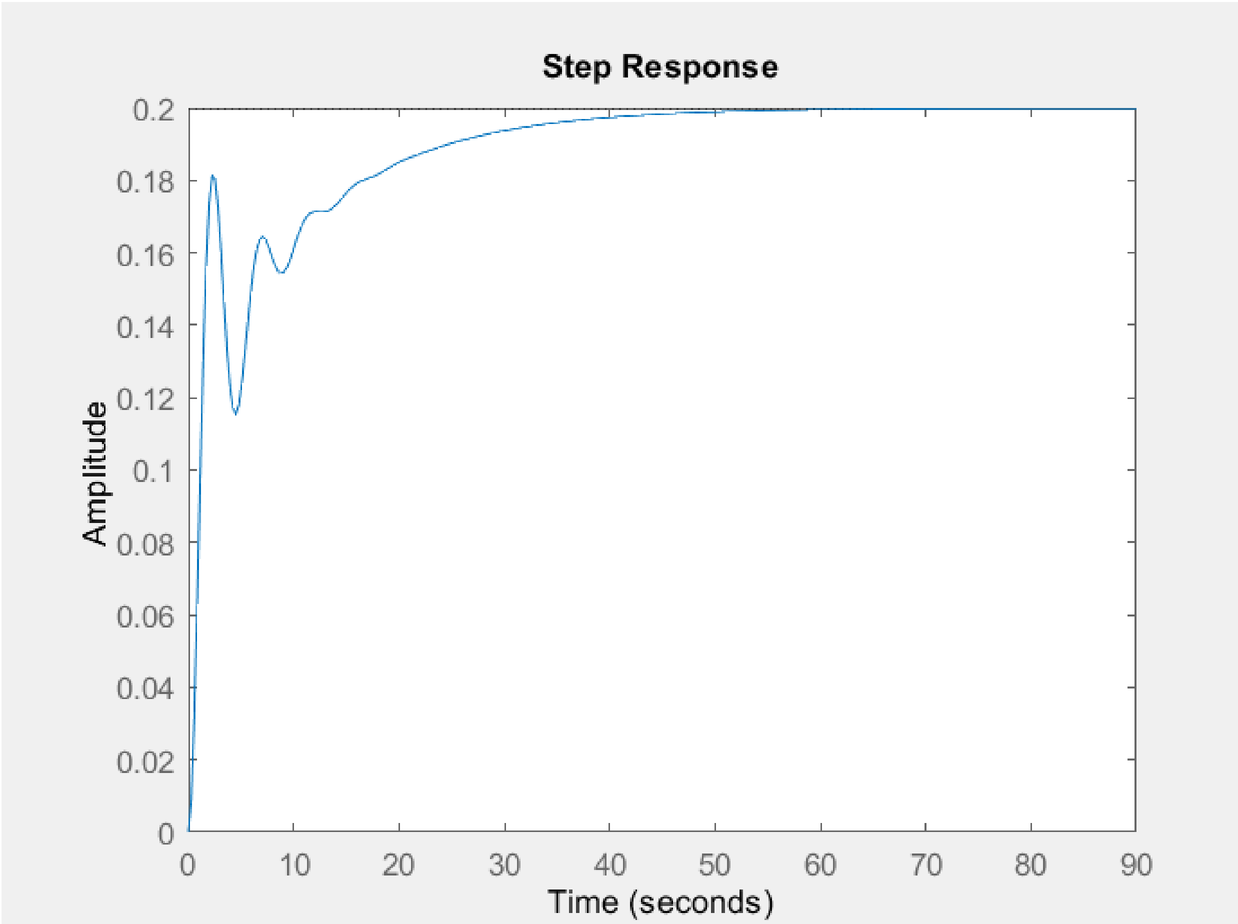

Figure 4: Amplitude vs. Time (s) for closed loop step response

The closed-loop step response should match the explicit function, which it does. Both show an amplitude of 0.2 which is the desired amplitude for this system. The step response graph was found using the feedback function.

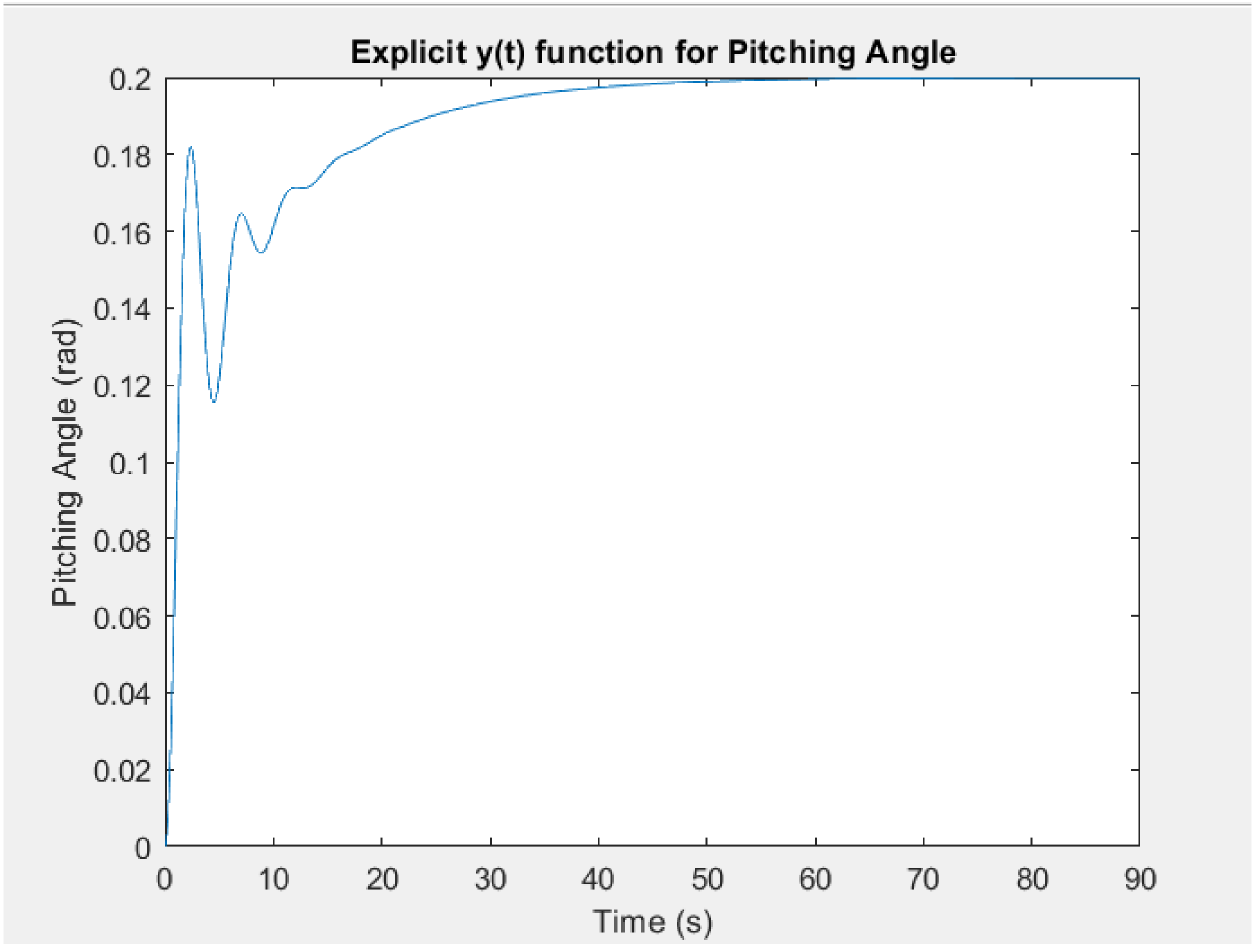

Figure 5: Pitch angle (rad) vs. Time (s) for explicit function

As shown in the figure, the pitching angle reaches a maximum angle of 0.2 radians over a period of 90 seconds. This function was found by taking the inverse Laplace transform of the transfer function. The oscillation accounts for the correction of the system to achieve its desired result of 0.2 radians.

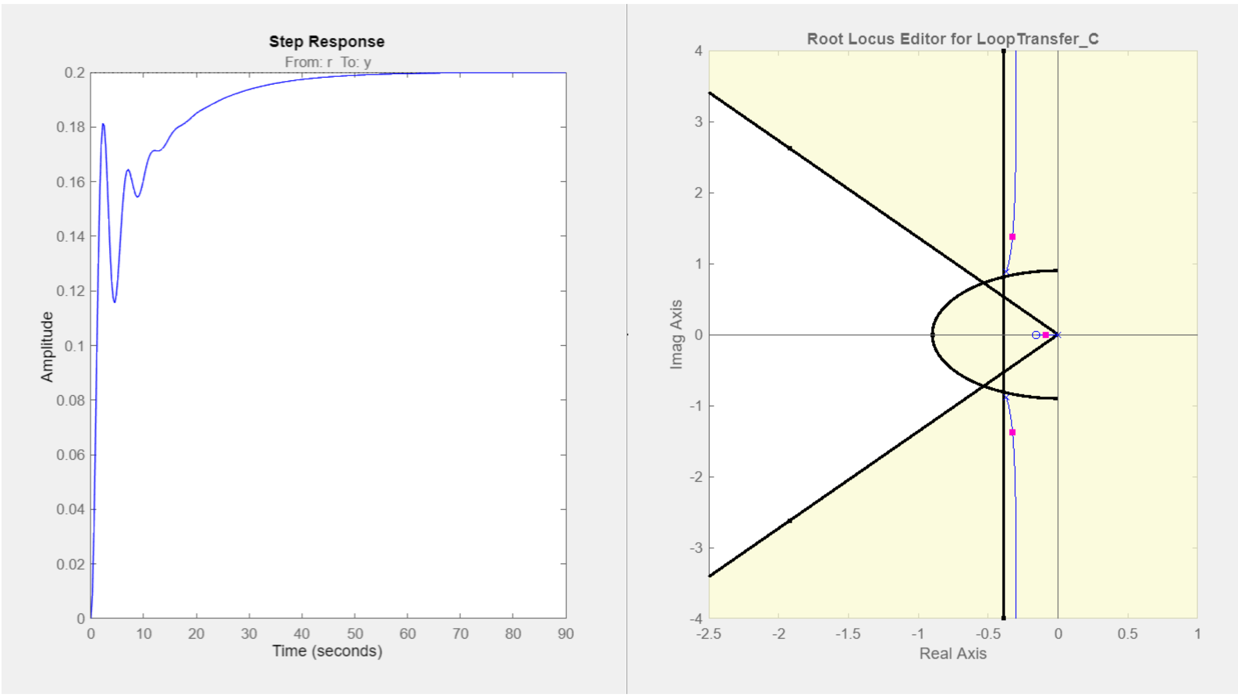


Figure 6: Step response and root locus with K = 0

This plot is the step response and the root locus with no compensator. As shown in the figure, there are no values on the root locus that satisfy the design requirements. The settling time occurs at 35 seconds which is outside of the design requirements of less than 10 seconds. The overshoot is zero percent for this plot, but this is not enough to compensate for the other missing design requirements.

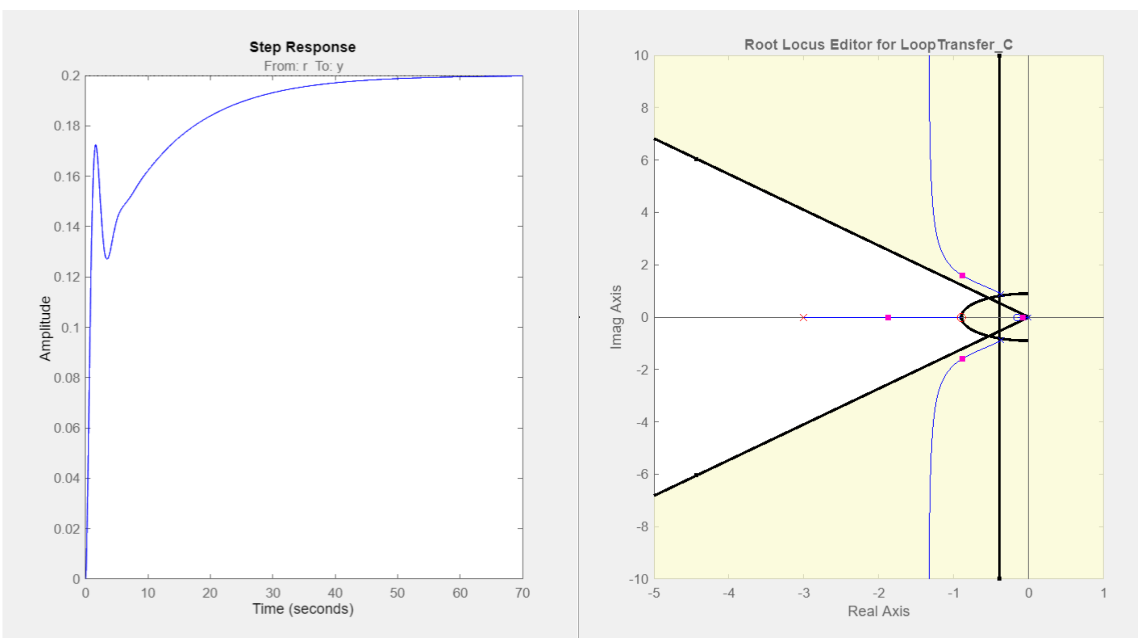


Figure 7: Step response and root locus for K = 3.33, added pole and zero

This is the plot for K=3.33 and the root locus with an addition of a pole at –3 and a zero at –0.9. These additions show that the root locus now exists within the unshaded region, however, it still will not meet the design requirements. This is because the settling time is still too high. The settling time must be under 10 seconds and the settling time for this compensator is approximately 36 seconds. The overshoot percentage remains at 0 percent.

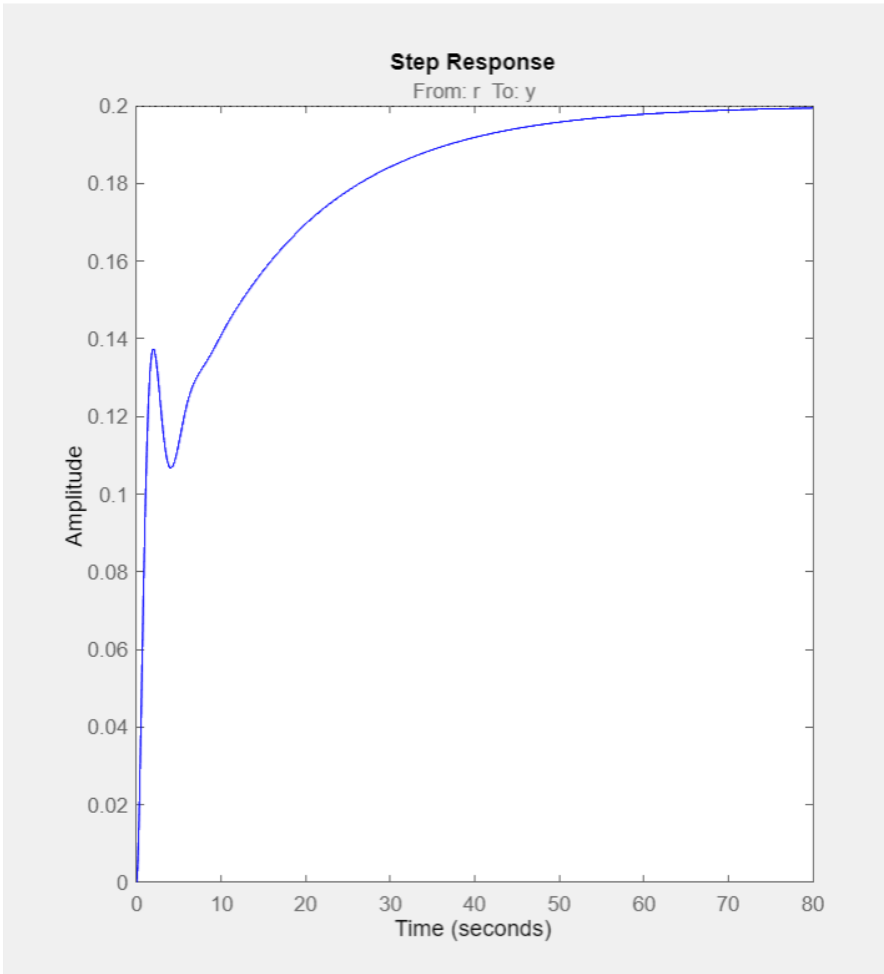


Figure 8: Step response for K = 2

This graph displays the step response for a compensation of K = 2. This is less than the previous compensator mentioned, and the root locus plot is the same. The trend follows suit here with an overshoot percentage of zero percent, but the settling time is still too high to meet the requirements. The settling time for this graph is 50.8 seconds. This compensator also approaches a value of 0.2 in amplitude.

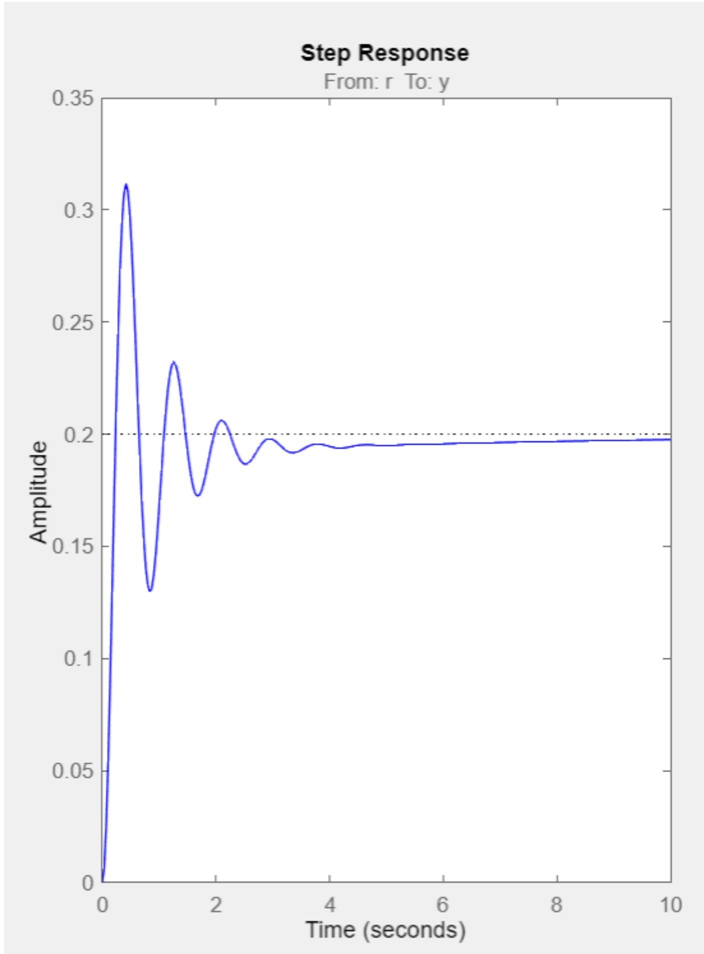


Figure 9: Step response for K = 50

This is the compensator value for K = 50, a significant increase from the previous value of 2. The graph has more of an oscillatory pattern to it and has a clear overshoot. The settling time is now reasonable, being 6.44 seconds, which is less than the design requirements. However, this time the overshoot percentage is too high to satisfy the design requirements. The overshoot percentage is at 55.9% which is well above 10%.

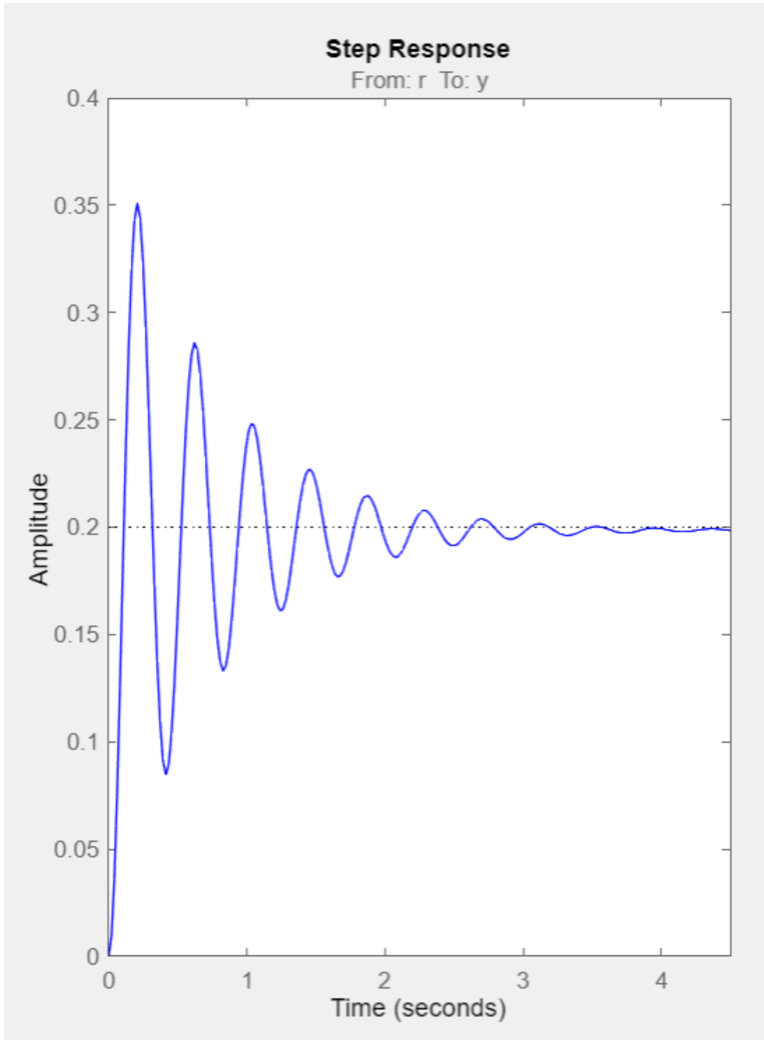

Figure 10: Step response for K = 200

The compensator graph for K = 200 is shown above. Once again, the settling time is reasonable, right around 2.97 seconds. The overshoot percentage is at 75.4% which is well outside of the design requirements range. This shows a clear pattern that as K increases for this system, the overshoot percentage will also increase. It can be concluded that there is no value of K that meets these design requirements.