ECE 351 - Section 53

Block Diagrams and System Stability

# Lab 7

*Submitted By :*
Chadwick Goodall

# Contents

# 1 Introduction

The objective of this lab was to familiarize oneself with Laplace-domain block diagrams as well as using the factored form of a transfer function in order to just the stability of a system.

The code for this project can be found at the below github repository.

GitHub: `https://github.com/Chadwick-g`

# 2 Equations

Part 1 task 1 equations:

$$G(s) = \frac{S+9}{(S+4)(S+2)(s-8)} \tag{1}$$

$$A(s) = \frac{s+4}{(S+1)(S+3)} \tag{2}$$

$$B(s) = (s+12)(s+14) \tag{3}$$

Part 1 task 3 equation (Open loop transfer function):

$$H(s) = \frac{(s+9)(s+4)}{(S+4)(S+2)(s-8)(S+1)(S+3)} \tag{4}$$

Part 2 task 1 equation (Symbolic closed loop transfer function):

$$H(s) = \frac{NumG * NumA}{DenG * DenA + NumG * DenA * B} \tag{5}$$

Part 2 task 2 equation (Scipy generated transfer function):

$$H(s) = \frac{s^2 + 13s + 36}{2s^5 + 41s^4 + 500s^3 + 2995s^2 + 6878s + 4344} \tag{6}$$

# 3 Methodology

In order to complete the first portion of this lab, by way of hand analysis of the block diagram system, I utilized the scipy.signal.tf2zpk alongside the numpy.roots function to calculate and verify my hand factorization of the A(s), G(s), and B(s) equations with the following code. The objective of this was to confirm that I had properly factored these equations to then find the open loop transfer function.

```
gNum = [1,9]
gDen = [1,-2,-40,-64]
aNum = [1,4]
aDen = [1,4,3]
b = [1,26,168]

gz, gp, gk = sig.tf2zpk(gNum, gDen)
az, ap, ak = sig.tf2zpk(aNum, aDen)
bOut = np.roots(b)

print("zeros of g: " + str(gz) + " poles of g: " + str(gp) + " gain of g
print("zeros of a: " + str(az) + " poles of a: " + str(ap) + " gain of a
print("roots of b: " + str(bOut))
```

The resulting output of the above code snippet can be seen in the appendix section.

The results that I received from this output did in fact end up verifying that I had correctly factored the equations necessary for writing the open loop transfer function of the block diagram system by hand. These hand made equations can be reference in the equations section (equations 1, 2, and 3). The next step in this process was to then write out the complete transfer function (equation 4).

The expression found for the open loop response in the case of this system, however, is not stable as it has a pole in the right half plane. The result of this can easily be observed once the system is plotted graphically, as can be seen in the results section. The system quickly diverges, demonstrating its unstable nature. This, of course, confirms my previous conclusion.

For the second portion of this lab I then proceeded to hand calculate the closed loop transfer function of the system using symbolic algebra. Using the resulting equation (eq. 5) along with the scypy.signal.convolve and tf2zpk functions, I was then able to calculate the exact coefficient values of the associated transfer function for this system (eq. 6). Below is the code involved for calculating the transfer function.

```
# closed loop zeros, poles, and gain
CLNum = sig.convolve(gNum,aNum)
CLDen1 = sig.convolve(gDen,aDen)
CLDen2 = sig.convolve(b,sig.convolve(gNum,aDen))
CLDen =  CLDen1+ CLDen2
#CLDen = sig.convolve(aDen,(gDen + sig.convolve(b, gNum)))
CLz, CLp, CLk = sig.tf2zpk(CLNum, CLDen)

print("CL num: " +  str(CLNum) + "\nCL den: " + str(CLDen))
```

```
#print ("zeros of CL: " + str(CLz) + "\npoles of CL: " + str(CLp) + "\ngo

lti = sig.lti(CLNum, CLDen)
tout, yout = sig.step(lti)
```

As can be seen above I went about calculating the convolutions in a systematic method and finally compiled the results into a single closed loop numerator and denominator (CLNUM, CLDen). I then printed out the numerator and denominator values and recorded these as the typed equation for the transfer function.

The resulting closed loop response ends up being stable as it is now regulated by the feedback loop in the system, ultimately causing it to converge to a horizontal asymptote which can be observed in the closed loop plot in the results section. As such, the graph supports the conclusion that the system is stable.

# 4 Results

Below are the resulting graphs that I constructed for this lab.

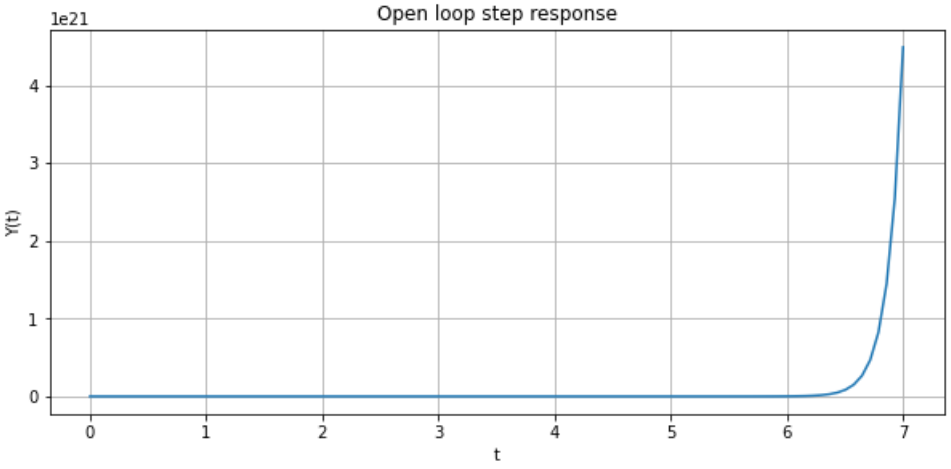


Figure 1: Open Loop Step Response

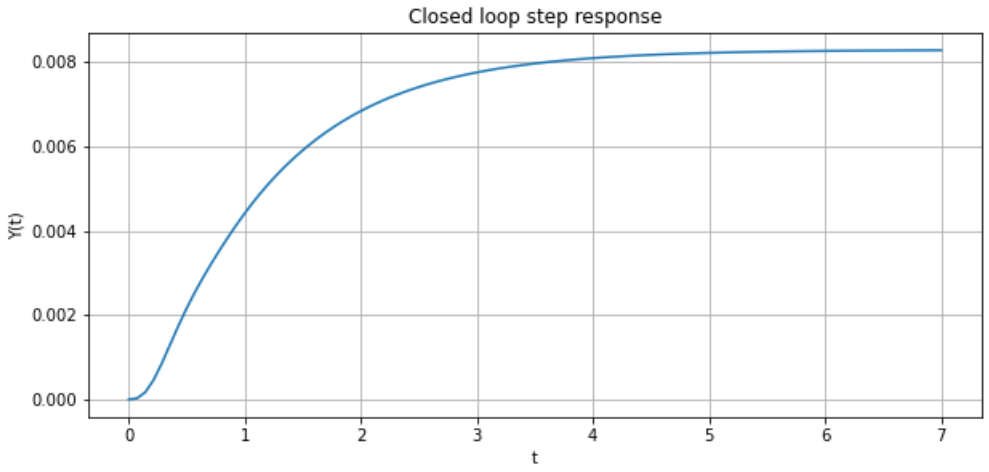


Figure 2: Closed Loop Step Response

The printed outputs of the tf2zpk function for part 1 can be viewed at the end of the document in the appendix section.

# 5 Error Analysis and Difficulties

The most challenging portion of this lab was learning how to use the tf2zpk function and reading its documentation, as well as using the convolve function in order to construct the closed loop transfer function.

# 6 Questions

1. In Part 1 Task 5, why does convolving the factored terms using scipy.signal.convolve() result in the expanded form of the numerator and denominator? Would this work with your user-defined convolution function from Lab 3? Why or why not?

The reason why this results in the expanded form of the numerator and denominator is because convolution is effectively multiplication in the Laplace/S domain. As a result, convolving those terms multiplies out the numerator and denominator. This, however, would not work for the user defined convolution function that I wrote in lab 3 because that is for calculating the convolution in the time domain not the Laplace/S domain.

2. Discuss the difference between the open- and closed-loop systems from Part 1 and Part 2. How does stability differ for each case, and why?

The difference between the open and closed loop system is that, most obviously, the open loop has no feedback and as a result, it is unregulated. This ends up causing the system to be unstable as there is a straight line from the input to the output of the system and the blocks in series that make up the output are unstable. The closed loop system, however, introduces feedback to the system which allows regulation of the output and ultimately brings the system back to a stable state.

3. What is the difference between scipy.signal.residue() used in Lab 6 and scipy.signal.tf2zpk() used in this lab?

4. Is it possible for an open-loop system to be stable? What about for a closed-loop system to be unstable? Explain how or how not for each.

It is possible for both an open loop system to be stable and a closed loop system to be unstable. It is entirely dependent on the configuration of the blocks and their associated functions. For an open loop system, if the blocks in series do not have any poles in the RHP then the system will be stable. Conversely, if the closed loop system introduces and blocks with poles in the RHP then the system will become unstable.

5. Leave any feedback on the clarity/usefulness of the purpose, deliverables, and expectations for this lab.

The code example really helped illustrate how to use the convolve function to expand a factored transfer function. Additionally I'm finding it really helpful to have all of the required deliverables for a section to be outlined before the section.

# 7 Conclusion

Concluding this lab I feel that I now solidified my understanding of block diagrams and their usefulness in modeling systems. With plenty of practice in the course as well as the hands on experience of this lab I feel that I can now analyze any reasonable block diagram.

# 8 Appendix

Below are the recorded results of the tf2zpk function for part 1.

zeros of g: [-9.] poles of g: [ 8. -4. -2.] gain of g: 1.0

zeros of a: [-4.] poles of a: [-3. -1.] gain of a: 1.0

roots of b: [-14. -12.]