ECE 351 - Lab 7

Khoi Nguyen
https://github.com/3khoin

21 October 2021

Contents

1 Introduction

The goal of this lab was to understand Laplace-domain block diagrams and implementation of the resulting transfer functions in Python, as well as to judge the stability of a system using the factored form of the transfer function.

2 Equations

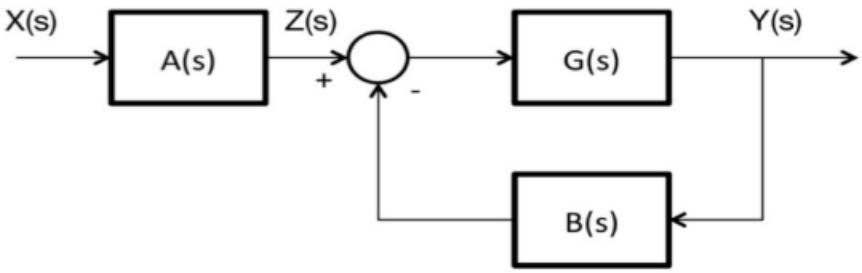
$$G(s) = \frac{s+9}{(s^2-6s-16)(s+4)}$$

$$A(s) = \frac{s+4}{s^2+4s+3}$$

$$B(s) = s^2 + 26s + 168$$

3 Methodology

The following block diagram was used to calculate the open-loop and closed-loop transfer functions.



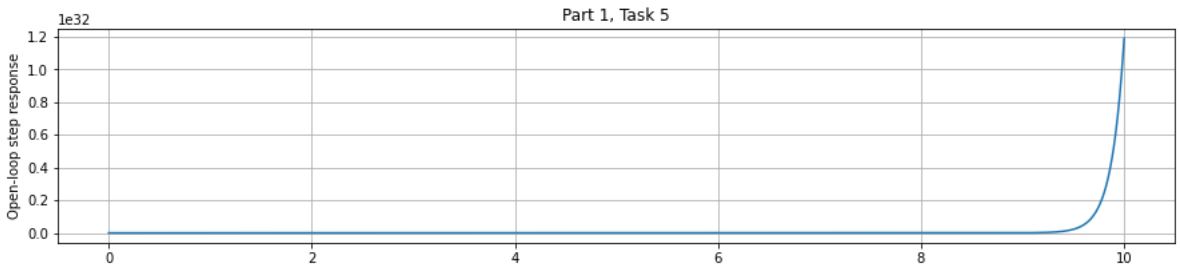
We began by factoring and identifying the poles and zeros for the equations G(s), A(s), and B(s), then verified our results with the scipy.signal.tf2zpk() function (and the numpy.roots() function for B(s)). We then found the open-loop transfer function for the block diagram using the provided equations. We plotted the step response of the resulting transfer function, with the numerator and denominator being put together with multiple uses of scipy.signal.convolve().

We then symbolically assembled the closed-loop transfer function for the block diagram using the numerator and denominator for each block. The numerical values for the transfer function were then calculated using scipy.signal.convolve(). Afterwards, the step response of the transfer function was plotted using scipy.signal.step().

4 Results

$$H_{OL}(s) = \frac{(s+9)(s+4)}{(s-8)(s+4)(s+3)(s+2)(s+1)}$$

The open-loop response is not stable since one of the poles (s = 8) is in the right half of the s-plane, so it will not converge.

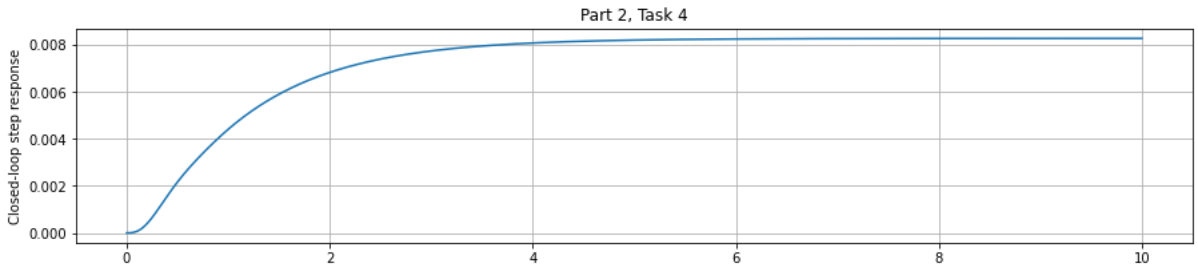


As seen in the plot, the function diverges to infinity.

$$H_{CL}(s) = \frac{numA * numG}{demA * demG + demA * numG * numB}$$

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

The closed-loop response is stable since the feedback loop means that some of its poles are on the left half of the s-plane, which is enough to keep it from diverging.



As seen in the plot, the function converges to a value as it approaches infinity.

5 Error Analysis

No errors in particular were found through the lab.

6 Questions

1. According to the convolution theorem, convolution in the time domain is equivalent to multiplication in the s domain. The polynomials are in the s-domain, so convolution in the time domain will result in their multiplication in the s-domain. Our user-defined convolution function from Lab 3 would not work for this purpose, as it is meant for continuous function representations, not discrete convolution required for that between polynomial coefficients.

2. The open-loop system was unstable because it contained a pole in the right half of the s-plane, which meant that the system would reach a value of infinity as time went to infinity. The closed-loop system behaved oppositely; it approached a vertical asymptote as time approached infinity, meaning it had a stable value over time.

3. The scipy.signal.residue() function found coefficients for partial expansion, whereas scipy.signal.tf2zpk() found the zeroes for the numerator of the transfer function.

4. Open-loop systems can be stable if absolutely all of the poles of their intermediate functions are in the left side of the s-plane. For closed-loop systems, they will be unstable if every pole of their intermediate functions are in the right side of the s-plane.

5. The lab specifications were standard and easily comprehensible. Learning about block diagrams and closed and open loop systems in this lab was particularly useful.

7 Conclusion

In this lab, we were able to utilize scipy.signal.tf2zpk() as well as scipy.signal.convolve() alongside our own calculations to understand and implement transfer functions stemming from a block diagram. Using block diagrams as a method of organization proved very useful for the calculation of more complex systems.