University of Idaho

ECE 351 - Section #53

Lab #4 Report

Step Response Using Convolution

Submitted To:
Kate Antonov
University of Idaho
kantonov@uidaho.edu

Submitted By :
Macallyster S. Edmondson
University of Idaho
edmo7033@vandals.uidaho.edu
github.com/mac-edmondson

15th February, 2022

Contents

1 Introduction

The goal of this weeks lab was to become familiar with using convolution to compute a systems step response. This is something we have definitely seen in class, but we have not yet performed it using discrete convolution in Python. As will be seen throughout this report, the discrete step response is not a perfect/exact calculation of the step response found through theoretical mathematics. Yet again, this lab was completed using *Python* through the *Spyder-IDE*. The packages used in the completion of this lab were `numpy` for definitions of mathematical functions, `matplotlib.pyplot` to plot outputs of functions, and `scipy.signal` to perform the discrete convolutions calculated in this lab.

All code for this lab, including this report, can be found on my Github.

2 Equations

The equations used within this lab are shown in this section. The equations will be referenced by number throughout the rest of the report.

$$h_1(t) = e^{-2t}[u(t) - u(t-3)] \tag{1}$$

$$h_2(t) = u(t-2) - u(t-6) \tag{2}$$

$$h_3(t) = cos(\omega_0 t)u(t) \quad \text{for } f_0 = 0.25 \text{ Hz} \tag{3}$$

$$h_{1\,SR}(t) = \frac{1}{2}[e^{-2(t-3)} - 1]u(t-3) - \frac{1}{2}[e^{-2t} - 1]u(t) \tag{4}$$

$$h_{2\,SR}(t) = (t-2)u(t-2) - (t-6)u(t-6) \tag{5}$$

$$h_{3\,SR}(t) = \frac{1}{\omega_0} sin(\omega_0 t)u(t) \tag{6}$$

3 Methodology

3.1 Lab: Part 1

Part 1 fo this lab was very simple. All that was necessary was to implement and plot, from $0 \leq t \leq 20$, the functions in Equations (1), (2), & (3). All three equations were very simple to implement and the code for their implementation can be seen below. Additionally, the plots generated for these functions can be seen in Figure 1.

```
1  #PART 1
2  #1
3
4  def h1(t) :
```

```
    out = np.exp(-2*t) * (ss.u(t) - ss.u(t-3))
    return out

    def h2(t) :
        out = ss.u(t-2) - ss.u(t-6)
        return out

    def h3(t) :
        f0 = 0.25 #Hz
        w0 = 2*np.pi*f0
        out = np.cos(w0 * t) * ss.u(t)
        return out

    #Plot...
```

3.2 Lab: Part 2

In Part 2 of this lab, the goal was to plot and compare the differences in the discrete and hand-calculated step responses of the functions in Equations (1), (2), & (3). The discrete convolution was generated using the `scipy.convolve` function, while the hand calculated function was plugged in directly. The work for the hand-calculated step-responses can be seen in the Attachments Section of this report, while the equations themselves can be seen in the Equations Section of this report in Equations (4), (5), & (6). The plots for the outputs of the different convolutions can be seen in Figures 2 & 3. Below, can be seen the code for the implementation of Part 2 of this lab.

```
# PART 2

# 1

bound = 10
t = np.arange(-10, bound + step_size, step_size)
f1 = h1(t)
f2 = h2(t)
f3 = h3(t)
u = ss.u(t)
f1p = spsig.convolve(f1, u, mode='full')
f2p = spsig.convolve(f2, u, mode='full')
f3p = spsig.convolve(f3, u, mode='full')
t = np.arange(-10 - bound, bound*2 + .5*step_size, step_size)
#Plot...

# 2

t = np.arange(-20, 20 + step_size, step_size)
w0 = 2*np.pi*0.25
```

```
21  f1 = ((1/2) * (np.exp(-2*(t-3)) - 1) * ss.u(t-3)) - ((1/2) * (np.exp
      (-2*(t)) - 1) * ss.u(t))
22  f2 = ((t-2) * ss.u(t-2)) - ((t-6) * ss.u(t-6))
23  f3 = (1/w0) * np.sin(w0 * t) * ss.u(t)
24  #Plot ...
```

4 Results

The results of this lab are very straightforward. The implementation of all functions worked as expected and the results are as expected after some analysis. You would expect the plots of Figures 2 & 3 to look identical but, as seen, they are not. The reason for this is discussed in the Error Analysis Section.

The deliverables for Parts 1 & 2 of this lab can be seen in Figures 1, 2, & 3, below.

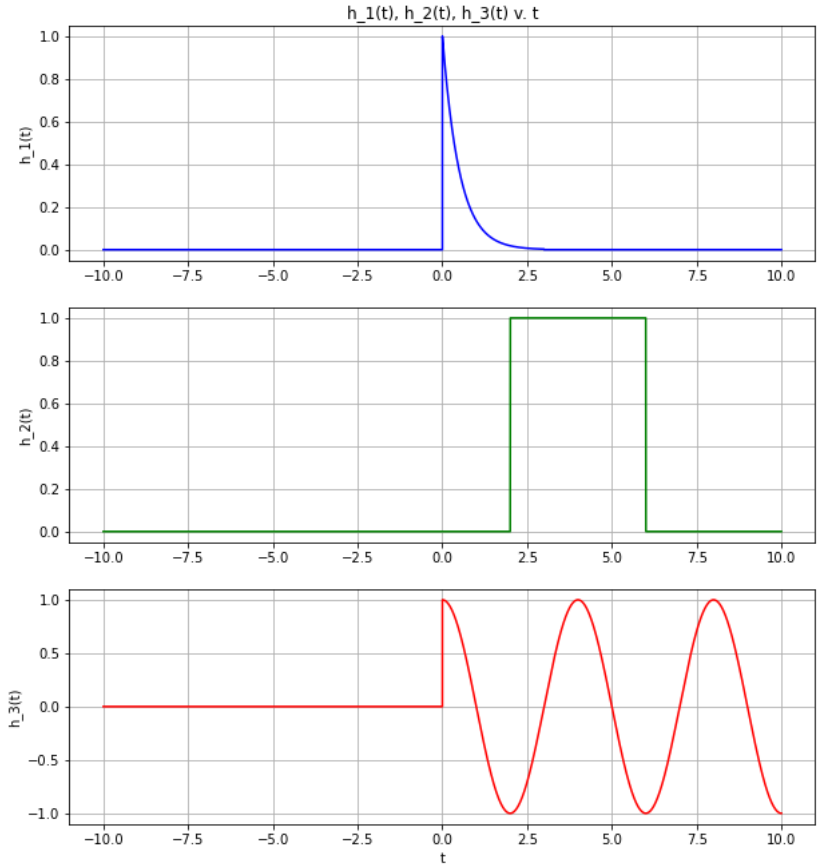

Figure 1: Part 1, Task 2 - Plots of Python Implementations of (1), (2), (3)

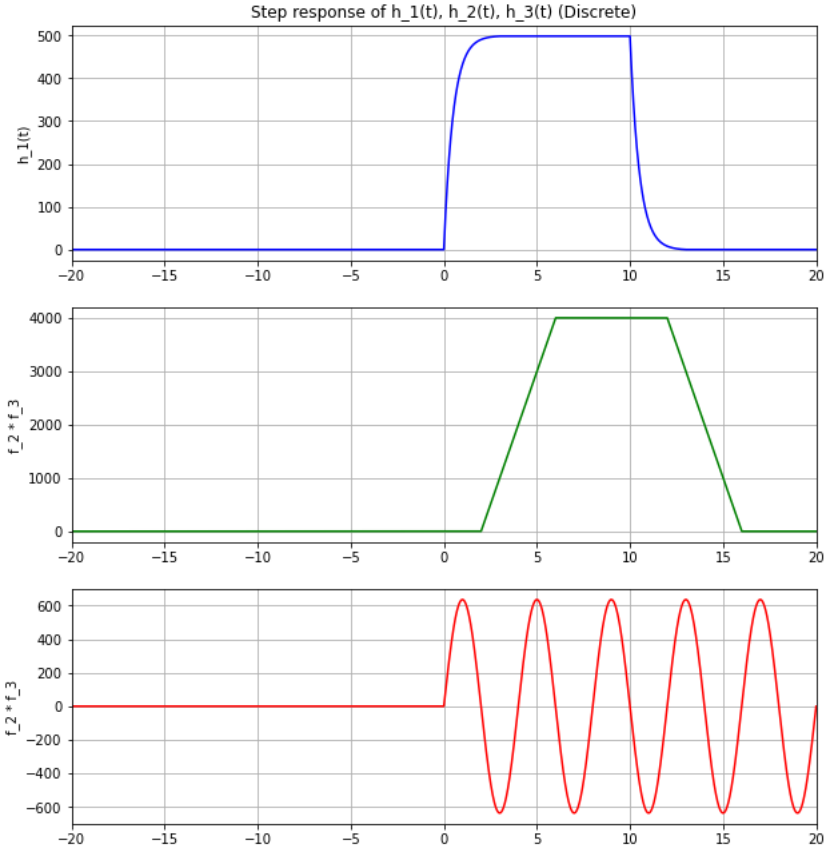

Figure 2: Part 2, Task 1 - Discrete Step Response of Eq. (1), (2), (3)

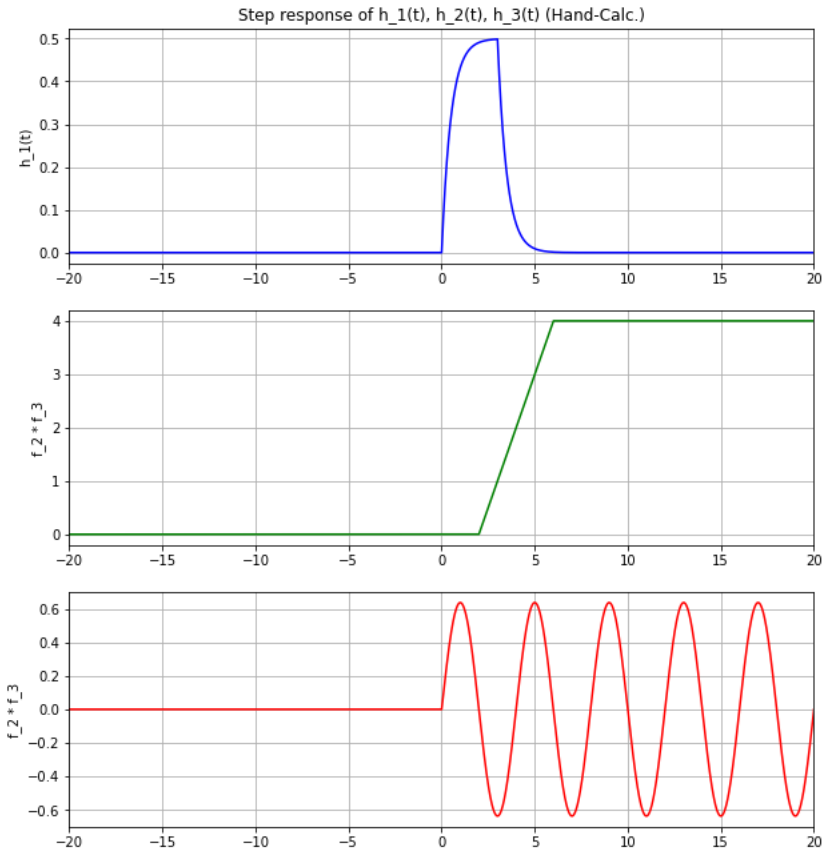

Figure 3: Part 2, Task 2 - Calculated Step Response of Eq. (1), (2), (3)

5 Error Analysis

As can be seen, Figures 2 & 3 do not look the same. This begs the question, which is correct and why? Assuming a mistake wasn't made in the hand calculated versions of the step response equations (there wasn't), the reason the sets of plots looks different comes down to the way the convolution is performed computationally. As the function in `scipy` used to perform the convolution uses summation, the summation will continue for the entire length of each function only over the interval given. This is why the plots found computationally can appear stretched or shrunk with respect to the "true"/calculated step response.

6 Questions

- This lab and its tasks were very concise in what is expected for deliverables.

7 Conclusion

In conclusion, I feel this lab was very successful. The implementation of the code in this lab was quite simple and I really enjoyed seeing the difference in the hand calculated and discrete step-response's. This was another useful exercise to show how discrete mathematics can differ from theoretical mathematics. All in all, I am very satisfied with what this lab has taught me and feel it was an excellent use of time.

8 Attachments

$$h_1 * v(t) = \int_0^t e^{-2\tau}\left(v(t-\tau) - v(t-3-\tau)\right) d\tau$$

$$\int_0^t e^{-2\tau} d\tau = -\frac{1}{2}\left[e^{-2t} - 1\right]$$

$$\Rightarrow \boxed{h_1 * v(t) = \frac{1}{2}\left[e^{-2(t-3)} - 1\right] v(t-3) - \frac{1}{2}\left[e^{-2t} - 1\right] \cdot v(t)}$$

$$h_2 * v(t) = \int_{-\infty}^{\infty} \left[v(\tau-2) - v(\tau-6)\right] \cdot v(t-\tau)\, d\tau$$

$$= \int_2^t v(\tau-2)\, d\tau - \int_6^t v(\tau-6)\, d\tau$$

$$\boxed{h_2 * v(t) = (t-2)\cdot v(t-2) - (t-6)\cdot v(t-6)}$$

$$h_3 * v(t) = \int_{-\infty}^{\infty} \cos(\omega_0 \tau) \cdot v(\tau) \cdot v(t-\tau)\, d\tau$$

$$= \int_0^t \cos(\omega_0 \tau)\, d\tau = \boxed{h_3 * v(t) = \frac{1}{\omega_0} \cdot \sin(\omega_0 t) \cdot v(t)}$$