ECE 351 - 52

Signals and Systems 1

Lab 4

Submitted By :
Owen Blair

Contents

1 Important Notes 2
2 Part 1 Deliverables 2
3 Part 2 Deliverables 2
4 Equations 3
5 Results 4
6 Listings 5
7 Questions 7
7.1 Question 1 7

1 Important Notes

It is important to note that plt.show() is needed at the end of the python code to properly show the plots of each function. It was not included in every section of code that plots a function(s) because it is assumed that its included at the end the code.

The web address to the GitHub where LaTeX code is stored is here:
https://github.com/Blairis123/ECE351_Reports

The web address to the GitHub where the Python code is here:
https://github.com/Blairis123/ECE351_Code

2 Part 1 Deliverables

Part 1 is about using the step function from the previous labs to make a transfer function. This starts with three user defined functions:

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

$$h_2(t) = u(t-2) - u(t-6)$$

$$h_3(t) = cos(\omega_0 t)u(t)$$

The functions above can be seen in Figure 1 in the Results section and the code to create and plot the functions can be seen in the Code Listings section as Listing 1. The function $h_1(t)$ is an exponential function that starts when $t = 0$ and ends when $t = 3$. This behavior is governed by the $[u(t) - u(t-3)]$ part of the function. This part is equal to one when t is between and equal to zero and three. Function $h_2(t)$ is a simple pulse with a height of one between the values of $t = 2$ and $t = 6$. The last function, $h_3(t)$ is a cosine function that starts at $t = 0$ and goes on forever. It should also be noted that $f_0 = 0.25$Hz is the cosine frequency (in radians, the frequency is 0.5π). All of these functions are plotted with a time domain from negative ten seconds to ten seconds.

3 Part 2 Deliverables

Part 2 was about finding the step response of the three transfer functions defined in part 1. Using the convolution function created in lab 3, the step response for each transfer function was found. The code generated step response of each function can be seen in Figure 2 and the code used to make and plot the convolutions can be found in Listing 2. These can be compared to the had calculated convolutions equations seen the Equations section as Equation 1, Equation 2, and Equation 3 respectively.

4 Equations

Equation 1

$$(0.5 - 0.5e^{-2t} * u(t) - (0.5 - 0.5e^{-2(t-3)} * u(t-2)$$

Equation 2

$$r(t-2) - r(t-6)$$

Equation 3

$$\frac{180}{\pi\omega_0} sin(\omega_0 t)u(t)$$

5 Results

Figure 1: $h_1(t), h_2(t) and h_3(t)$

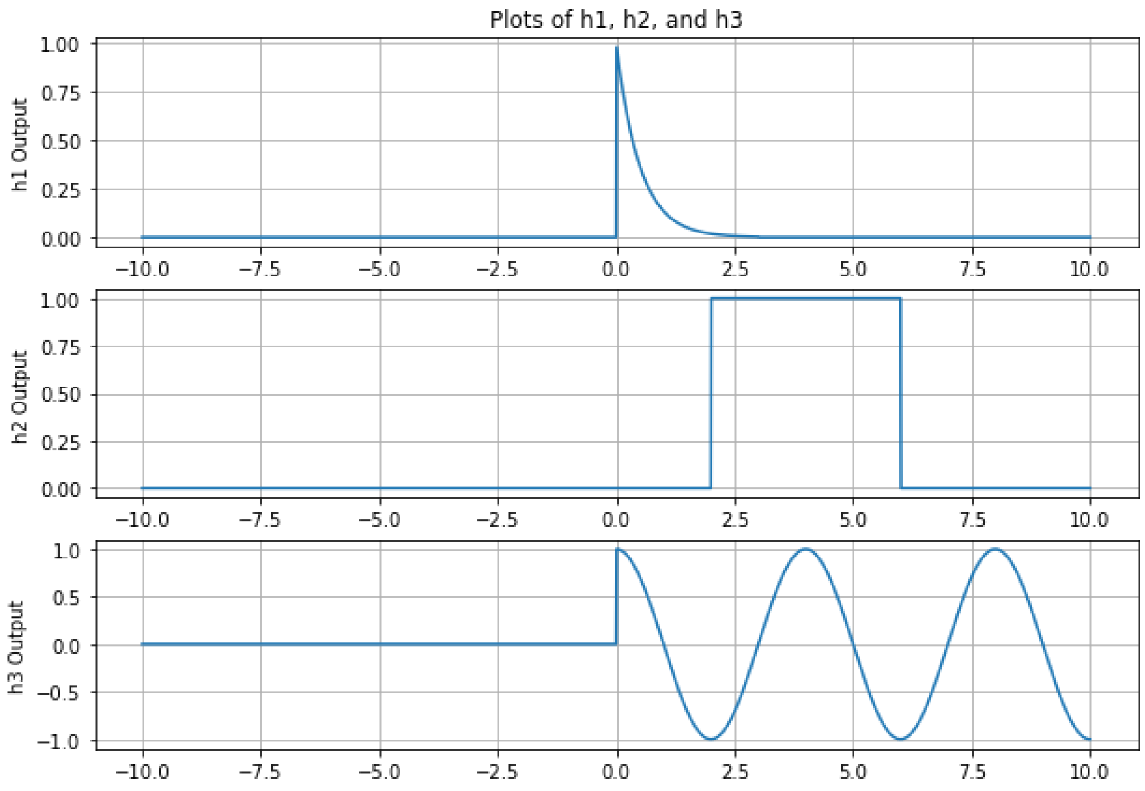
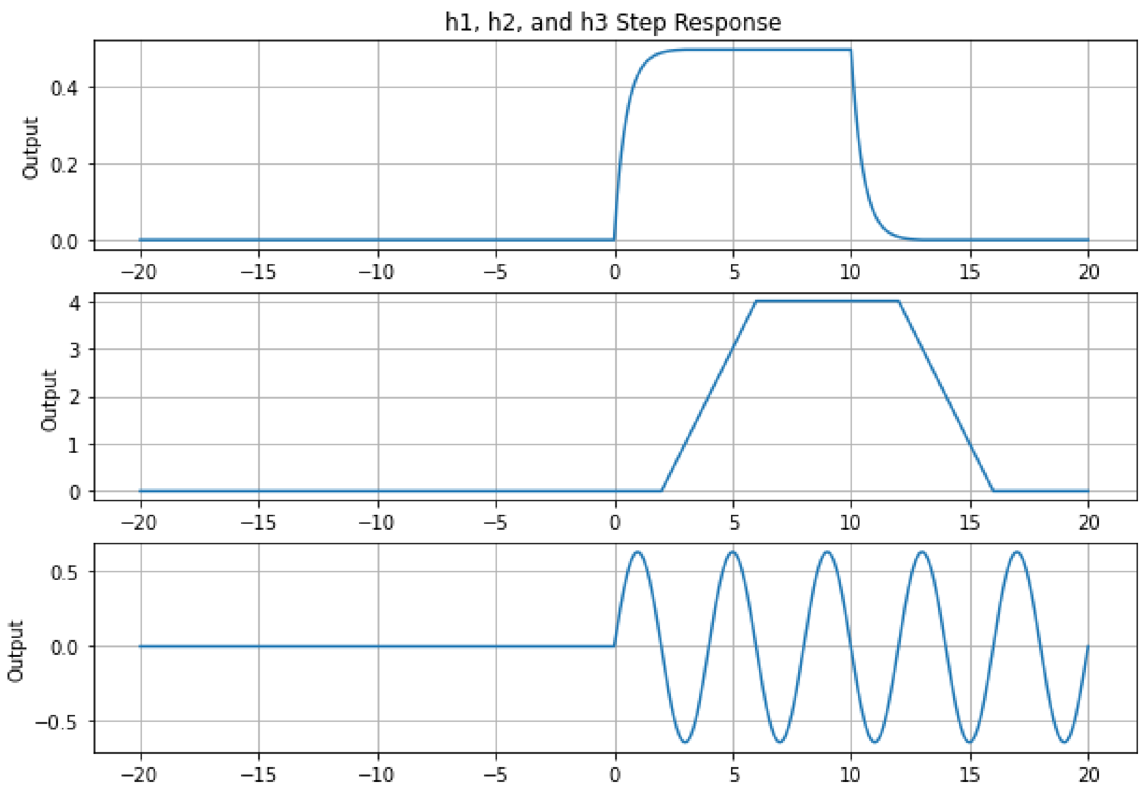


Figure 2:Code Generated Step Response



6 Listings

Listing 1: Make and Plotting of h_1, h_2, h_3

```
#---------------------PART 1----------------------------#

    #Define step size
steps = 1e-2

    #t for part 1
start = -10
stop = 10
    #Define a range of t_pt1. Start @ 0 go to 20 (+a step) w/
    #a stepsize of step
t_pt1 = np.arange(start, stop + steps, steps)


    #Make h1(t) = e^(-2t)[u(t)-u(t-3)]
```

```
h1 = eExpo(t_pt1, 1, -2) * (stepFunc(t_pt1, 0, 1) - stepFunc(t_pt1,
    3, 1))

    #Make h2(t) = u(t-2) - u(t-6)
h2 = stepFunc(t_pt1, 2, 1) - stepFunc(t_pt1, 6, 1)

    #Make h3(t) = cos(wt)*u(t) --> frequency = 0.25
    #This means that w is about 1.5707963 (0.25 * 2 * pi)
h3 = cosine(1.5707963 * t_pt1) * stepFunc(t_pt1, 0, 1)

#-----MAKE PLOTS--------#
    #Make plot and then show it
    #Make plots for Part 1-----------#
plt.figure(figsize=(10,7))
plt.subplot(3,1,1)
plt.plot(t_pt1,h1)
plt.grid()
plt.ylabel('h1 Output')
plt.title('Plots of h1, h2, and h3')

plt.subplot(3,1,2)
plt.plot(t_pt1,h2)
plt.grid()
plt.ylabel('h2 Output')

plt.subplot(3,1,3)
plt.plot(t_pt1,h3)
plt.grid()
plt.ylabel('h3 Output')
```

[language=Python] Listing 2: Code Generated Convolutions

```

#---------------------PART 2-----------------------------#
    #t for part 2
start = -10
stop = 10
    #Define a range of t_pt1. Start @ -10 go to 10 (+a step) w/
    #a stepsize of step
t_pt2 = np.arange(start, stop + steps, steps)


    #Make h1(t) = e^(-2t)[u(t)-u(t-3)]
h1 = eExpo(t_pt2, 1, -2) * (stepFunc(t_pt2, 0, 1) - stepFunc(t_pt2,
    3, 1))

    #Make h2(t) = u(t-2) - u(t-6)
h2 = stepFunc(t_pt2, 2, 1) - stepFunc(t_pt2, 6, 1)
```

```

    #Make h3(t) = cos(wt)*u(t) --> frequency = 0.25
    #This means that w is about 1.5707963 (0.25 * 2 * pi)
h3 = cosine(1.5707963 * t_pt2) * stepFunc(t_pt2, 0, 1)

    #Make step response to convolve with --> f(t) = u(t)
forceFunc = stepFunc(t_pt1, 0, 1)

    #Make convolutions for step response!
h1StepResponse = convolve(h1, forceFunc) * steps

h2StepResponse = convolve(h2, forceFunc) * steps

h3StepResponse = convolve(h3, forceFunc) * steps

#Make a tConv range to pot the convolve functions!
#This should be the same as the size for all convolutions for this
    lab
tConv = np.arange(0, ((len(h3StepResponse) -1) * steps) + steps,
    steps) -20

#-----MAKE PLOTS--------#
    #Make plot and then show it
    #Make plots for Part 2-----------#
plt.figure(figsize=(10,7))
plt.subplot(3,1,1)
plt.plot(tConv,h1StepResponse)
plt.grid()
plt.ylabel('Output')
plt.title('h1, h2, and h3 Step Response')

plt.subplot(3,1,2)
plt.plot(tConv,h2StepResponse)
plt.grid()
plt.ylabel('Output')

plt.subplot(3,1,3)
plt.plot(tConv,h3StepResponse)
plt.grid()
plt.ylabel('Output')
```

7 Questions

7.1 Question 1

This lab was straight forward and easy to understand. I had no misconceptions about what was expected and most of the lab was just getting the time intervals correct for *the convolution.*