# Project Report for ECE 351

*Lab 11: Z - Transform Operations*

Isaias Munoz

November 13, 2022

University of Idaho

# Contents

# 1 Introduction

The purpose of this lab is analyze discrete systems using Python's build -in-functions and a function developed by Christopher Felton.

# 2 Methodology

## 2.1 Part 1

The first task that began with solving for the transfer function of the $y[k]$ and to begin to do so we need to take the z-transform of it and then solve for $Y[Z]/X[Z]$ which is output over input but in the z-domain. This way we can get the transfer function. Then we can take the inverse transform and go back to $h[k]$ and using partial fraction find the coefficients.

$$Y(k) = 2x[k] - 40x[k-1] + 10y[k-1] - 16[k-2]$$

I then take $y[k]$ function to the z domain, $Y[Z]$.

$$Y(Z) = 2X[Z] - 40Z^{-1}X[Z] + 10Y[Z]Z^{-1} - 16Z^{-2}Y[Z]$$

Doing algebra and rearranging for $Y[Z]/X[Z]$ or the transfer function I obtain.

$$H[Z] = \frac{Y[Z]}{X[Z]} = \frac{3Z^2 - 40Z}{Z^2 - 10Z + 16}$$

Now to find $h[k]$ by partial fraction expansion I factor out a Z from the numerator and factor everything that could be factor.

$$\frac{H[Z]}{Z} = \frac{2(z-20)}{(Z-8)(Z-2)} = \frac{A}{Z-8} + \frac{B}{Z-2}$$

Doing algebra once again I obtained A=-4, and B =6. Plugging them back in I obtained the following H(Z).

$$\frac{H[Z]}{Z} = \frac{-4}{Z-8} + \frac{6}{Z-2}$$

Lastly taking the inverse of the z transform to obtain h[k] I get this function:

$$h[k] = (-4 * 8^k + 6 * 2^k) * u[k]$$

To find the explicitly algebra and steps I have attached a by hand solution that show the steps taken more in detail to obtain the following functions. They are under the result section of this lab report.

Moving onto task 3 of the lab it was to verify our partial fraction expansion with *scipy.signal.residuez*() below is the code done to do so.

```
a=[2,-40]
b=[1,-10,16]


rout, pout,kout= sig.residue(a,b)


print (' this is the Root solutions to the long DEQ')
print(rout)
print (' this is the Poles solutions to the long DEQ')
print(pout)
# for i in range (len(apout)):
#     print( apout[i])
print (' this is the K (residue of a given term) solutions to the long DEQ')
print(kout)

```

Figure 1: Code for verifying the poles, roots, and k values of the transfer function

As can be seen making an array of the coefficients only and using *sig.residue* to verify was straightforward since we have done so in other previous labs. The verification numbers can be seen in the result section of this lab report. Task 5 was to use the function *zplane*() given to verify our partial fraction expansion that was done. Since the zplane() function was given all we needed to do was call it and have three letters that would take the return of *Zplane*() and then print those outputs. Again a very similar approach to the previous task since we are verifying the pole-zeroe plot the verification is under the resul section along with the zplot graphs.

```python
def zplane(b,a,filename=None):
    """Plot the complex z-plane given a transfer function.
    """
    import numpy as np
    import matplotlib.pyplot as plt
    from matplotlib import patches
    # get a figure/plot
    ax = plt.subplot(111)
    # create the unit circle
    uc = patches.Circle((0,0), radius=1, fill=False,
    color='black', ls='dashed')
    ax.add_patch(uc)
    # The coefficients are less than 1, normalize the coeficients
    if np.max(b) > 1:
        kn = np.max(b)
        b = np.array(b)/float(kn)
    else:
        kn = 1
    if np.max(a) > 1:
        kd = np.max(a)
        a = np.array(a)/float(kd)
    else:
        kd = 1
    # Get the poles and zeros
    p = np.roots(a)
    z = np.roots(b)
    k = kn/float(kd)
    # Plot the zeros and set marker properties
    t1 = plt.plot(z.real, z.imag, 'o', ms=10,label='Zeros')
    plt.setp( t1, markersize=10.0, markeredgewidth=1.0)
    # Plot the poles and set marker properties
    t2 = plt.plot(p.real, p.imag, 'x', ms=10,label='Poles')
    plt.setp( t2, markersize=12.0, markeredgewidth=3.0)
    ax.spines['left'].set_position('center')
    ax.spines['bottom'].set_position('center')
    ax.spines['right'].set_visible(False)
    ax.spines['top'].set_visible(False)
    plt.legend()
    # set the ticks
    # r = 1.5; plt.axis('scaled'); plt.axis([-r, r, -r, r])
    # ticks = [-1, -.5, .5, 1]; plt.xticks(ticks); plt.yticks(ticks)
    if filename is None:
        plt.show()
    else:
        plt.savefig(filename)
    return z, p, k

az=[2,-40,0]
bz=[1,-10,16]
Z,P,K=zplane(az, bz)
print (' this is the Root solutions to the Long DEQ in z plane')
print(Z)
print (' this is the Poles solutions to the Long DEQ in z plane')
print(P)
# for i in range (len(apout)):
#     print( apout[i])
print (' this is the K (residue of a given term) solutions to the Long DEQ i
print(K)
```

Figure 2: Code for verifying the poles, roots, and k values of the transfer function using the *zplane*() function

Laslty we needed to use *scipy.signal.freqz*() to plot the magnitude and phase response of $H(Z)$.

```

# Use scipy.signal.freqz() to plot the magnitude and phase responses of H(z). Note:
# You must set whole = True within the scipy.signal.freqz() command. (See function
# documentation for details).
wfreq, mag = sig.freqz(az, bz, whole = True)

#wfreq= radians
wfreqnew=wfreq/2*np.pi
magnitude=20*np.log10(abs(mag))
plt . figure ( figsize = (6 , 2) )
plt . plot (wfreqnew,  magnitude )
plt . ylabel ('Y output dB')
plt . xlabel ('w rads/second')
plt . title ('magnitude of transfer function task 4 ')
plt . grid ()


angle=np.angle(mag)*180/(np.pi)
# magnitude=20*np.log10(abs(angle))
plt . figure ( figsize = (6 , 2) )
plt . plot (wfreqnew, angle )
plt . ylabel ('phase otput')
plt . xlabel ('w rads/second')
plt . title ('phase of transfer function task 5 ')
plt . grid ()

```

Figure 3: Code for using *scipy.signal.freqz*()

Here the in house function returns the frequency in radians per second and the magnitude complex. transforming it to a frequency of Hz because this is America, and dB scale of magnitude we obtain the following graphs in the result section.

# 3 Results

## 3.1 Part 1

```
this is the Root solutions to the long DEQ
[ 6. -4.]
this is the Poles solutions to the long DEQ
[2. 8.]
this is the K (residue of a given term) solutions to the long DEQ
[]
```

Figure 4: Verifying the partial fraction coefficients

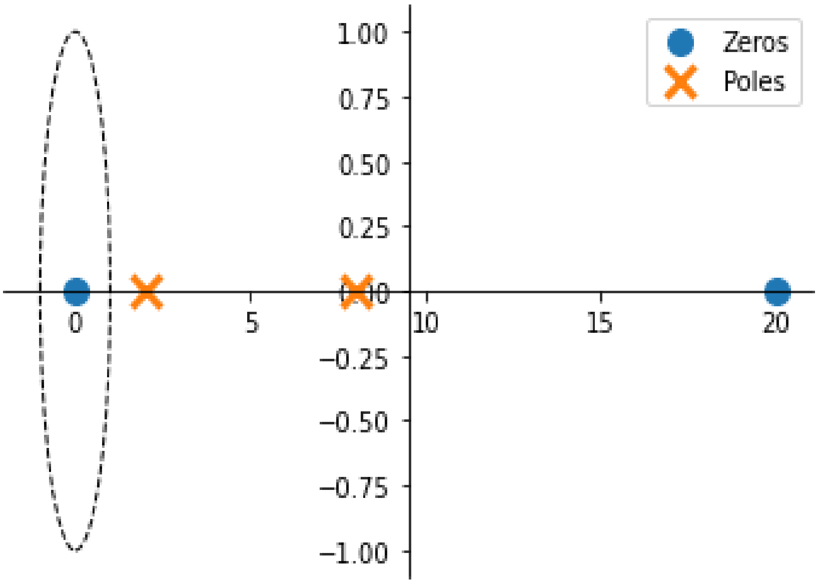
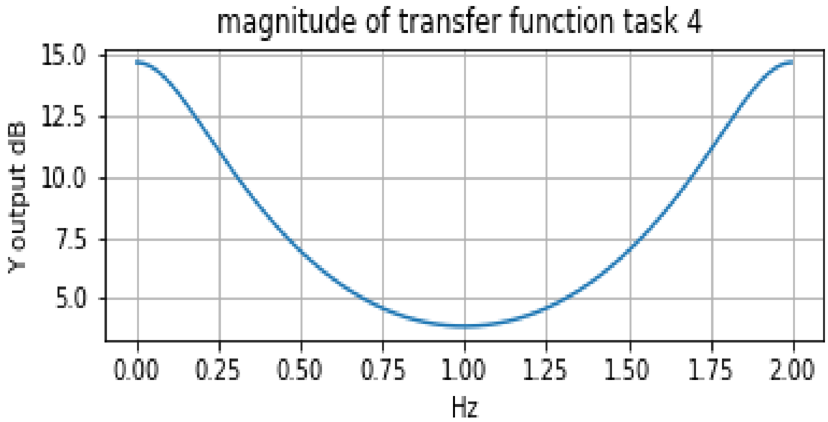



Figure 5: pole-zero plot for $H(z)$

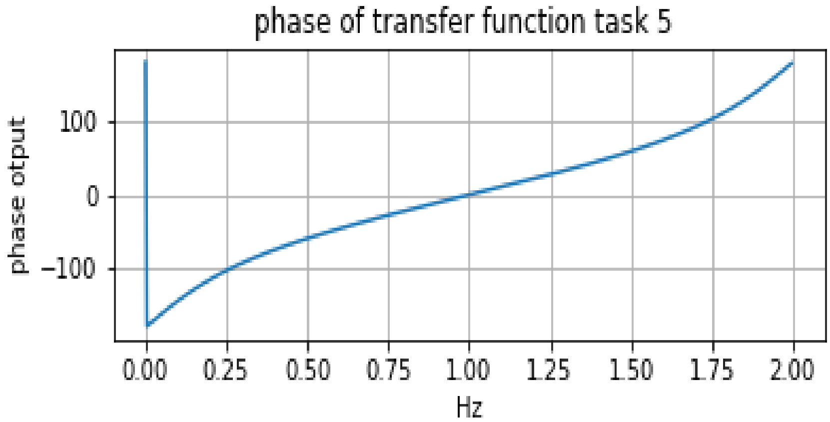
```
this is the Root solutions to the long DEQ in z plane
[20.  0.]
this is the Poles solutions to the long DEQ in z plane
[8. 2.]
this is the K (residue of a given term) solutions to the long DEQ in z plane
0.125
```

Figure 6: Actual zero confirmations of $H(z)$



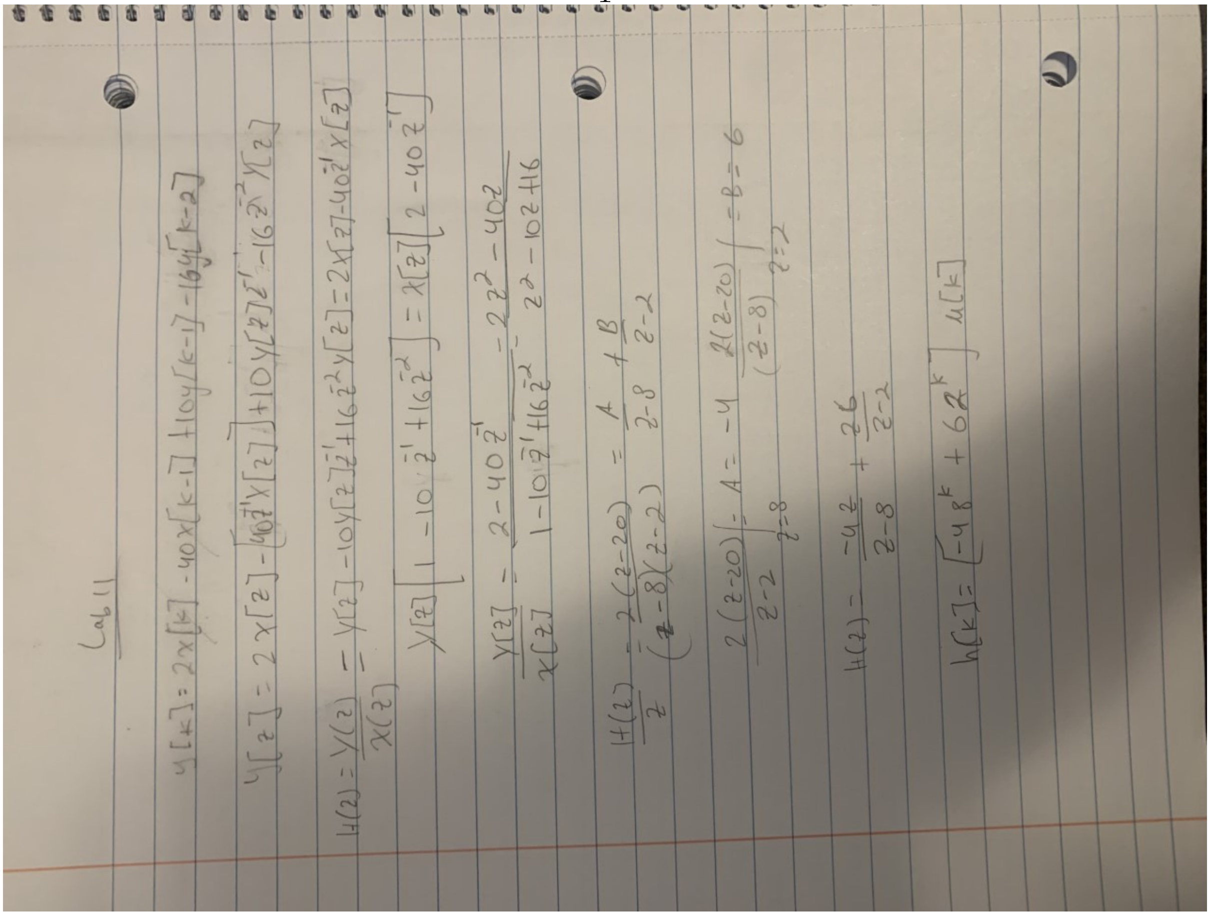

1

Figure 7: magnitude vs HZ




1

Figure 8: phase vs HZ



Lab 11

$$y[k] = 2x[k] - 40x[k-1] + 10y[k-1] - 16y[k-2]$$

$$Y[z] = 2X[z] - 40z^{-1}X[z] + 10Y[z]z^{-1} - 16z^{-2}Y[z]$$

$$H(z) = \frac{Y(z)}{X(z)} \quad Y[z] - 10Y[z]z^{-1} + 16z^{-2}Y[z] = 2X[z] - 40z^{-1}X[z]$$

$$Y[z]\left[1 - 10z^{-1} + 16z^{-2}\right] = X[z]\left[2 - 40z^{-1}\right]$$

$$\frac{Y[z]}{X[z]} = \frac{2 - 40z^{-1}}{1 - 10z^{-1} + 16z^{-2}} = \frac{2z^2 - 40z}{z^2 - 10z + 16}$$

$$\frac{H(z)}{z} = \frac{2(z-20)}{(z-8)(z-2)} = \frac{A}{z-8} + \frac{B}{z-2}$$

$$\left.\frac{2(z-20)}{z-2}\right|_{z=8} = A = -4 \quad \left.\frac{2(z-20)}{(z-8)}\right|_{z=2} = B = 6$$

$$H(z) = \frac{-4z}{z-8} + \frac{6z}{z-2}$$

$$h[k] = \left[-4\,8^k + 6\,2^k\right] u[k]$$

Figure 9: Derivation of z transforms task 1-2

# 4 Questions

1. Looking at the plot generated in Task 4, is H(z) stable? Explain why or why not.

   It is not stable because there is a zero in the positive side of the graph.

2. Leave any feedback on the clarity of lab tasks, expectations, and deliverables.

It was straightforward I think.