EE599 Homework #1

Yifan Wang
wang608@usc.edu #3038184983

January 15, 2019

Problem 1

The $hdf5$ file required is named:

$$Yifan\_Wang\_hw1P1.hdf5$$

Problem 2

1 Specify equation in α

For the first order AR Filter the difference equation is:

$$y[n] = b_0 * x[n] + b_1 * x[n-1] - a_0 * y[n] - a_1 * y[n-1] \tag{1}$$

The frequency response is:

$$H(z) = \frac{b_0 + b_1 * z^{-1}}{1 + a_1 * z^{-1}} \tag{2}$$

where:

$$z = e^{j*\pi*v}$$

It is a AR filter which implies that $b_1 = 0$. Since it has unit gain at DC, which means $\omega = 0, z = 1$; When $n = 0$ the difference equation become:

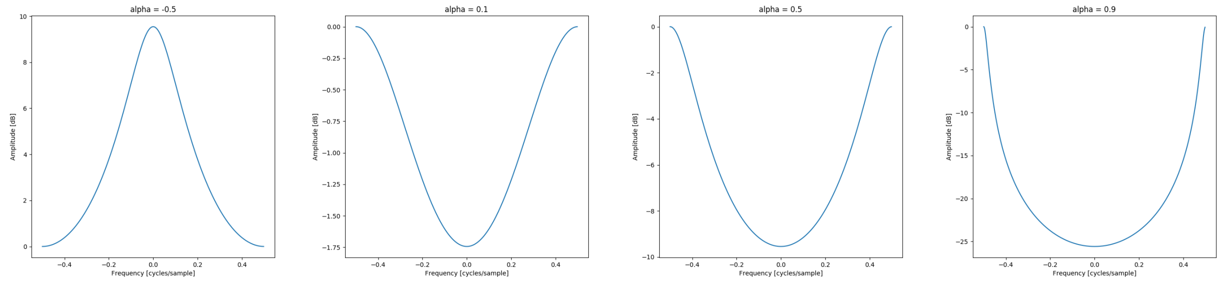
$$H(1) = \frac{b_0}{1 + a_1} \tag{3}$$

So that $b_0 = 1 + a_1$
From these information, the frequency response can be written as:

$$H(z) = \frac{1 + a_1}{1 + a_1 * z^{-1}} \tag{4}$$

It's pole point is when $1 + a_1 * z^{-1} = 0$, $p = -a_1$ which is the α given; so that $a_1 = -\alpha$.

$$H(z) = \frac{1 - \alpha}{1 - \alpha * z^{-1}} \tag{5}$$



2 20% decay point

The following code are used to generate the figures in the four α provided.

```
import scipy.signal as signal
import numpy as np
import matplotlib.pyplot as plt
pi = 3.1415926535

alpha = np.array([-0.5, 0.1, 0.5, 0.9])
for index in range (0,4):
    a = np.array([1.0, -alpha[index]])
    b = np.array([1-alpha[index], 0])
    #save figure separatly
    w, h= signal.freqz(b,a,whole='true')
    fig = plt.figure(figsize = (5,5))
    ax1 = plt.subplot()
    ax1.set_title('alpha = '+str(alpha[index]))
    ax1.plot((w-pi)/(2*pi), 20 * np.log10(abs(h)))
    ax1.set_ylabel('Amplitude [dB]')
    ax1.set_xlabel('Frequency [cycles/sample]')
    plt.savefig('result_'+str(alpha[index])+'.png')
```

Apply inverse z transform on (4), we can get

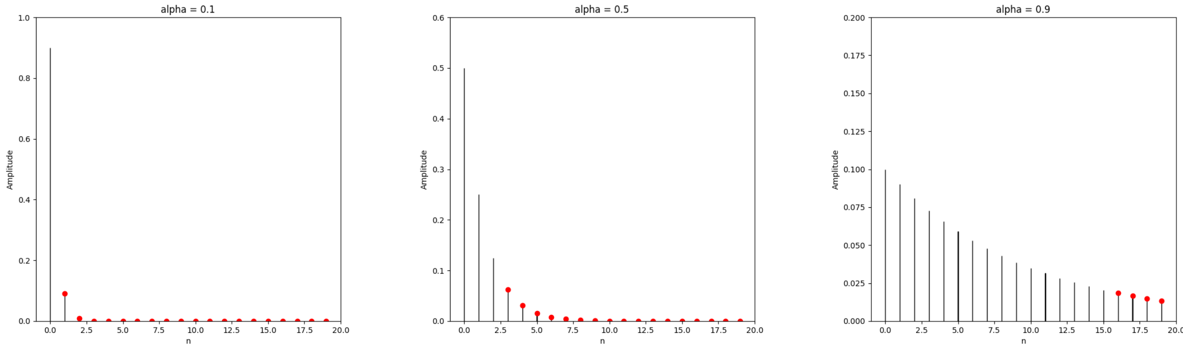
$$h(t) = (1 - \alpha) * \alpha^n$$

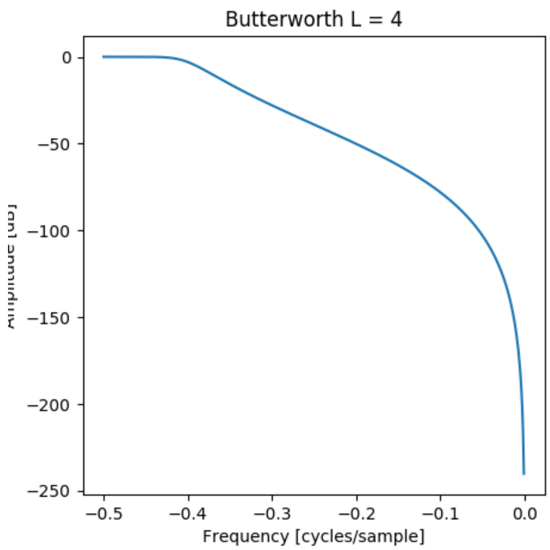
$n = \lceil log_a(0.2) \rceil$ is the number of samples required.

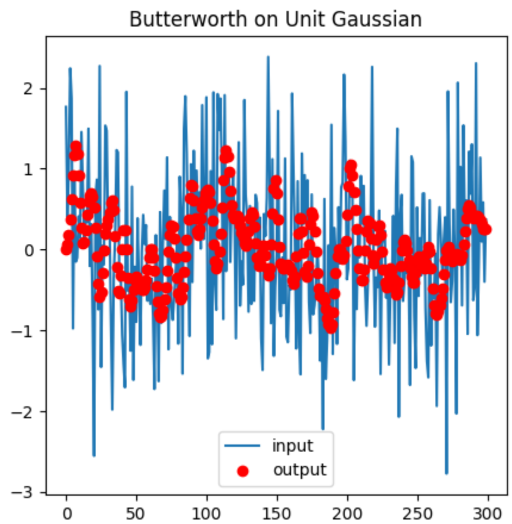
```
for index in range (1,4):
    x = np.arange(0,20)
    y = (1-alpha[index])*alpha[index]**x
    m = np.where(y < 0.2*y.max())
    print('alpha = ',alpha[index], '20% decay point is', x[m[0][0]])
    fig = plt.figure(figsize = (7,7))
    ax2 = plt.axes()
    ax2.set_title('alpha = '+str(alpha[index]))
    for i in range (0,20):
        ax2.arrow(x[i], 0, 0, y[i], head_width=0, head_length=0)
    ax2.plot(x[m[0]], y[m[0]], 'ro')
    ax2.set_ylim(0,1.1-alpha[index])
    ax2.set_xlim(-1,20)
    ax2.set_ylabel('Amplitude')
    ax2.set_xlabel('n')
    plt.savefig('result_1'+str(alpha[index])+'.png')
```

For the α given, they are:

```
('alpha = ', 0.1, '20% decay point is', 1)
('alpha = ', 0.5, '20% decay point is', 3)
('alpha = ', 0.9, '20% decay point is', 16)
```







3 Butterworth Filter $L = 4$

The general frequency response for a $L = 4$ Butterworth Filter with AR and MA coeffieients is generated by $scipy.signal.butter$

```
import scipy.signal as signal
import numpy as np
import matplotlib.pyplot as plt
pi = 3.1415926535
#generate coefficient for L=4 Butterworth Filter
b, a = signal.butter(4, 0.2)
w, h = signal.freqz(b, a)

fig = plt.figure(figsize = (5,5))
ax1 = plt.subplot()
ax1.set_title('Butterworth L = 4')
ax1.plot((w-pi)/(2*pi), 20 * np.log10(abs(h)))
ax1.set_ylabel('Amplitude [dB]')
ax1.set_xlabel('Frequency [cycles/sample]')
plt.savefig('butter.png')

print('a: ',a)
print('b: ',b)
#generate noise
```

```
np.random.seed(0)
s = np.random.normal(0, 1, 300)
#filtering
y = signal.lfilter(b,a,s)
fig = plt.figure(figsize = (5,5))
ax2 = plt.subplot()
ax2.set_title('Butterworth on Unit Gaussian')
ax2.plot(np.arange(300), s, label='input')
ax2.plot(np.arange(300), y, "ro",label='output')
ax2.legend()
plt.savefig('butterresult.png')
```

The one possible coefficient is:

```
AlexdeMacBook-Pro-2:ee599h1 alex$ python ./butterworthL4.py
('a: ', array([ 1.        , -2.36951301,  2.31398841, -1.05466541,
    0.18737949]))
('b: ', array([0.00482434, 0.01929737, 0.02894606, 0.01929737, 0.00482434])
    )
```