# ASSIGNMENT 10 - EE17B114

April 10, 2019

In this experiment we will be trying to do convolution operation on various signals and using both methods of linear convolution and circular convolution.In the theory class we have analyzed the advantages of using circular convolution over using linear convolution when we are recieving continuos samples of inputs.We will also be analysing the effect of passing the signal x = cos(0.2∗pi∗n) + cos(0.85∗pi∗n) through a given filter.At last we will be analysing the cross-correllation output of the Zadoff–Chu sequence.

**Importing necessary packages**

```
In [2]: from scipy import signal
        import csv
        import matplotlib.pyplot as plt
        import numpy as np
```

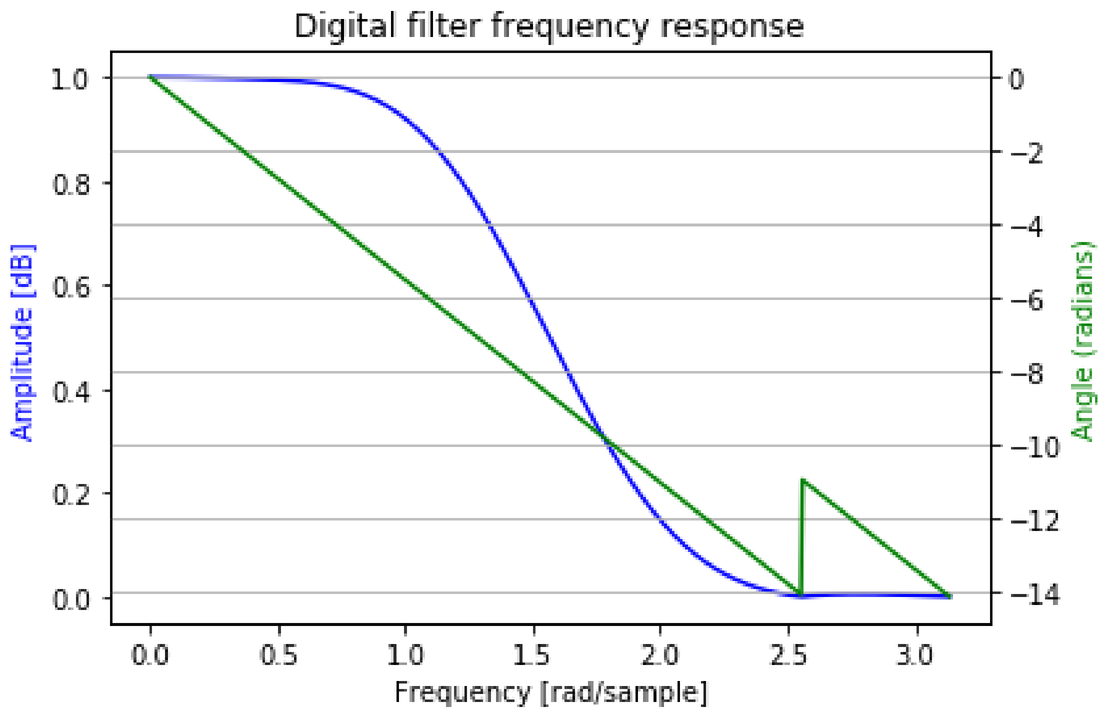
**Importing the filter sequence**

```
In [3]: with open('/home/pruthvirg/Downloads/h.csv', 'r') as f:
          reader = csv.reader(f)
          your_list = list(reader)
```

Now we will use the signal.freqz function to visuaize the filter in frequency domain.

```
In [4]: w, h = signal.freqz(your_list)
```

```
In [5]: fig, ax1 = plt.subplots()
        ax1.set_title('Digital filter frequency response')
        ax1.plot(w, abs(h), 'b')
        ax1.set_ylabel('Amplitude [dB]', color='b')
        ax1.set_xlabel('Frequency [rad/sample]')
        ax2 = ax1.twinx()
        angles = np.unwrap(np.angle(h))
        ax2.plot(w, angles, 'g')
        ax2.set_ylabel('Angle (radians)', color='g')
        ax2.grid()
        ax2.axis('tight')
        plt.show()
```

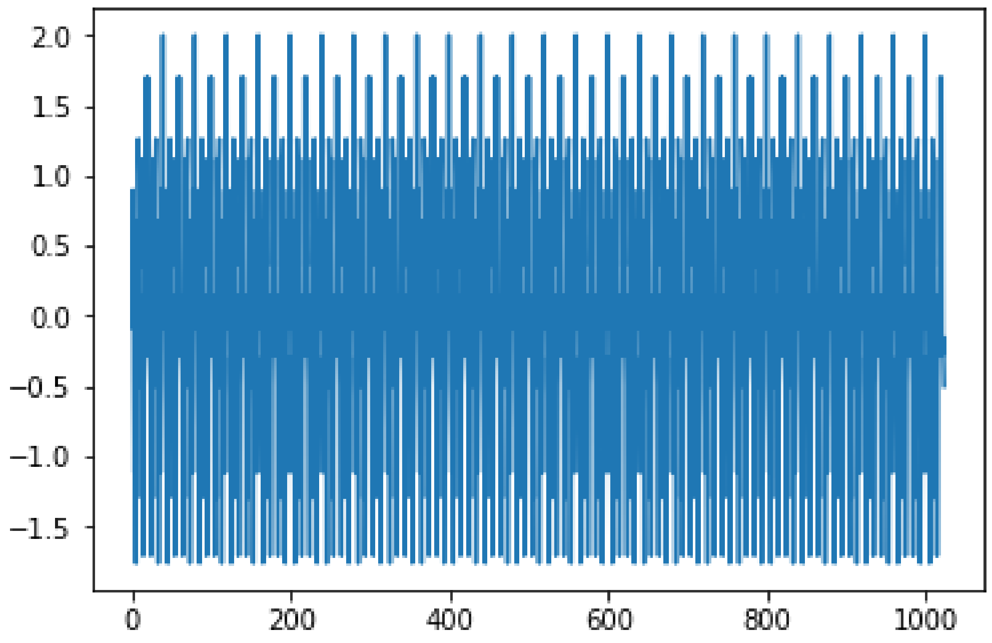


Here I have plotted both the magnitude and phase response of the filter in the appropriate frequency range. It is clear from the plot that the given filter is a low-pass filter with a cutoff frequency of about 0.75 rad/s.

```
In [6]: x = np.linspace(1,2**10,2**10)
        y = np.cos(0.2*(np.pi)*x) + np.cos(0.85*(np.pi)*x)
```

Now let us consider a sequence and pass it through the given filter and analyze the output of the fiter.

```
In [7]: plt.plot(y)

Out[7]: [<matplotlib.lines.Line2D at 0x7f062e4037b8>]
```



Clearly the input seqence has frequency components of 0.2pi = 0.628 rad/s and 0.85pi = 2.669 rad/s.

The given list is in form of strings hence we will convert them into floats before proceeding.

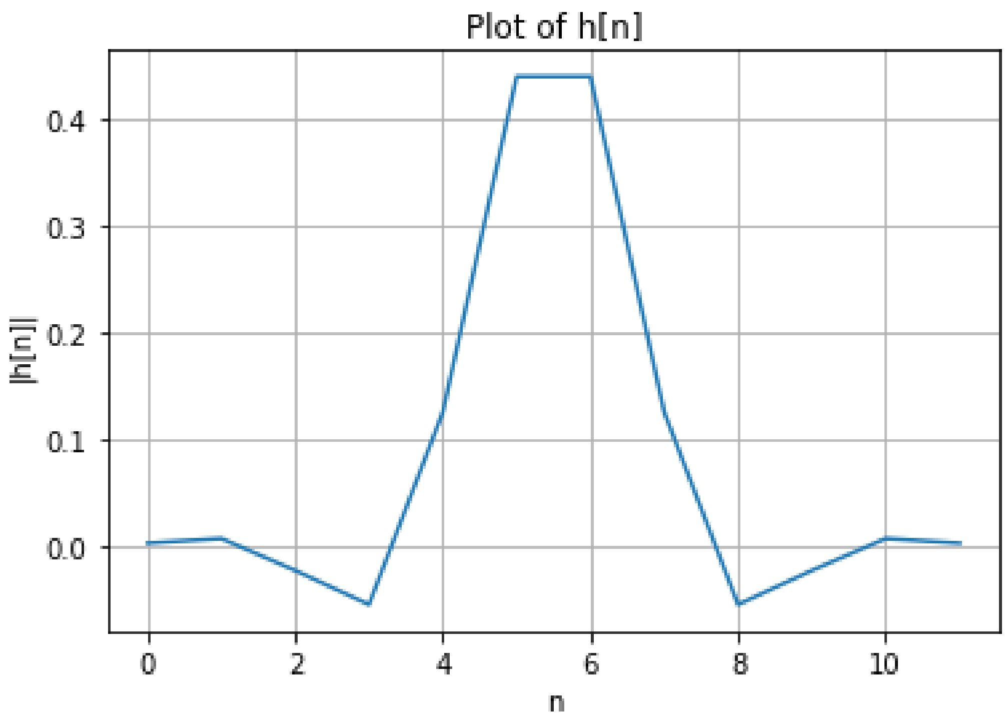
```
In [8]: your_list_float = [float(i[0]) for i in your_list]
```

```
In [9]: your_list_float
```

```
Out[9]: [0.003261,
         0.0076237,
         -0.022349,
         -0.054296,
         0.12573,
         0.44003,
         0.44003,
         0.12573,
         -0.054296,
         -0.022349,
         0.0076237,
         0.003261]
```

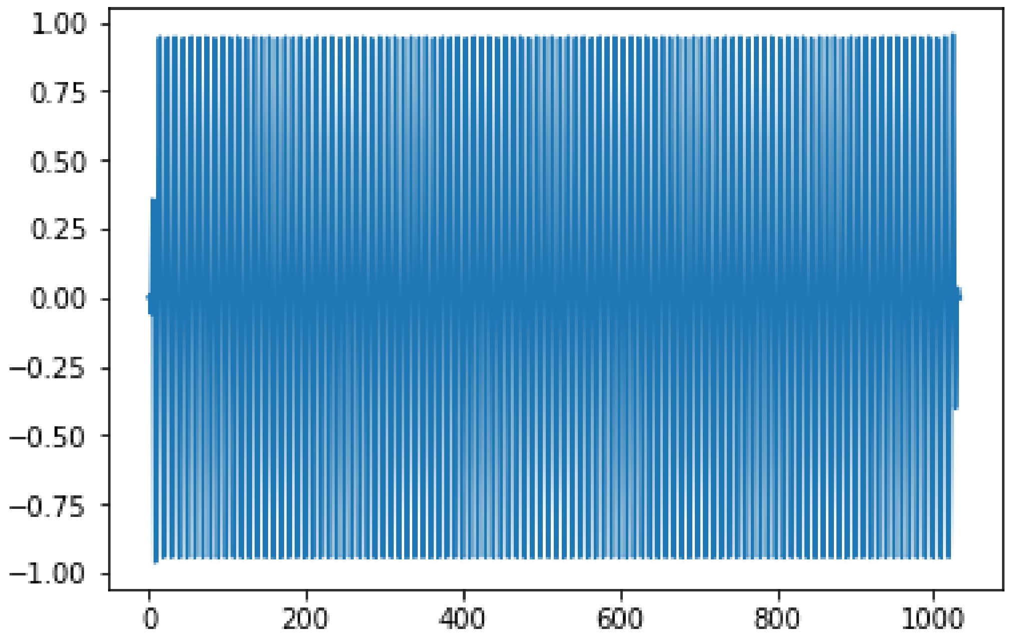
Let us plot the given filter points

```
In [35]: plt.plot(your_list_float)
         plt.title('Plot of h[n]')
         plt.xlabel('n')
         plt.ylabel('|h[n]|')
         plt.grid(True)
```



Let us find-out the convolution result between the given signal and the given filter.

```
In [11]: z = np.convolve(your_list_float,y)
         plt.plot(z)

Out[11]: [<matplotlib.lines.Line2D at 0x7f062e3385c0>]
```



Here we are clearly able to see that the higher frequency of 0.85pi has been rejected as it is a low-pass system.Hence the output has only one frequency component.

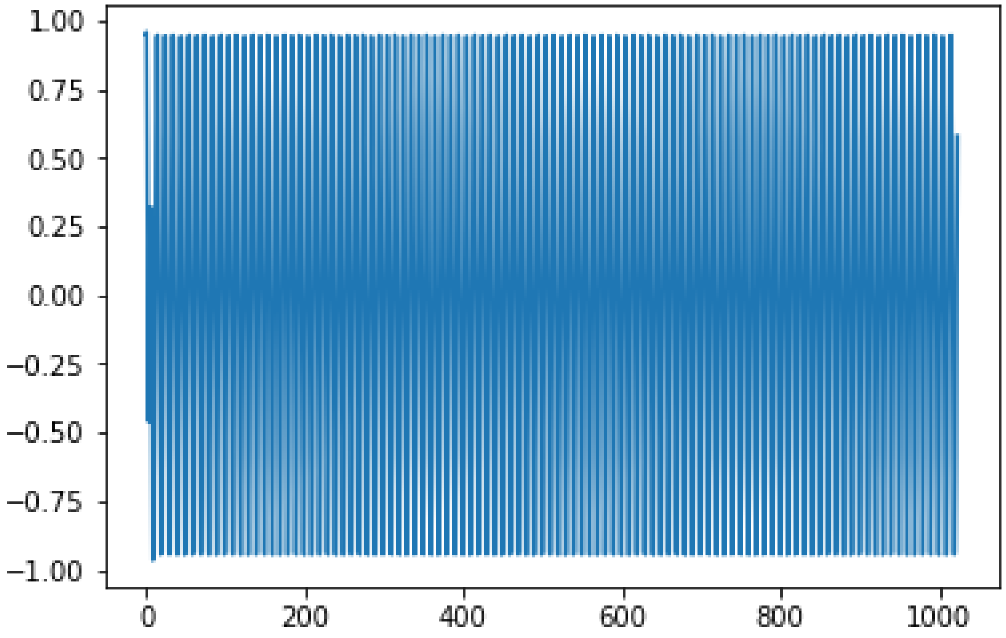
```
In [12]: a_1 = np.concatenate([ your_list_float,np.zeros(len(x)-len(your_list_float)) ])
```

```
In [13]: y1=np.fft.ifft(np.fft.fft(y) * np.fft.fft(a_1))
```

```
In [14]: plt.plot(y1)
```

```
/usr/local/lib/python3.6/dist-packages/numpy/core/numeric.py:492: ComplexWarning: Casting comple
  return array(a, dtype, copy=False, order=order)
```

```
Out[14]: [<matplotlib.lines.Line2D at 0x7f062e31d630>]
```



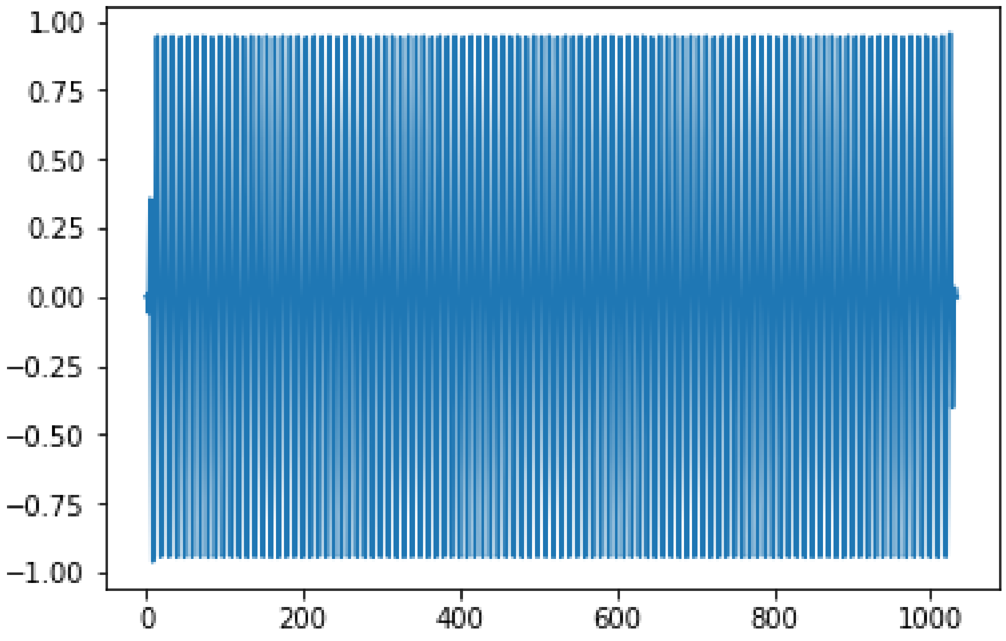
Inorder to compute the output using circular-convolution I am imitially padding my signals in to avoid overlapping of the output over itself.By doing this we will be getting the output seqence just like linear convolution.

```
In [15]: N = len(x)+len(your_list_float)-1
```

```
In [16]: fil = np.concatenate([your_list_float,np.zeros(N-len(your_list_float))])
```

```
In [17]: y_new = np.concatenate([y,np.zeros(N-len(y))])
```

```
In [18]: y2=np.fft.ifft(np.fft.fft(y_new) * np.fft.fft(fil))
In [19]: plt.plot(y2)

/usr/local/lib/python3.6/dist-packages/numpy/core/numeric.py:492: ComplexWarning: Casting comple
  return array(a, dtype, copy=False, order=order)


Out[19]: [<matplotlib.lines.Line2D at 0x7f062e232a20>]
```



We clearly see that the output is exactly similar to the one which we got by linear convolution.Hence it is seen that by padding the sequence appropriately we will be able to achieve the output using circular convolution.

Now we are asked to find out the correlation of the Zadoff–Chu sequence using circular convolution.

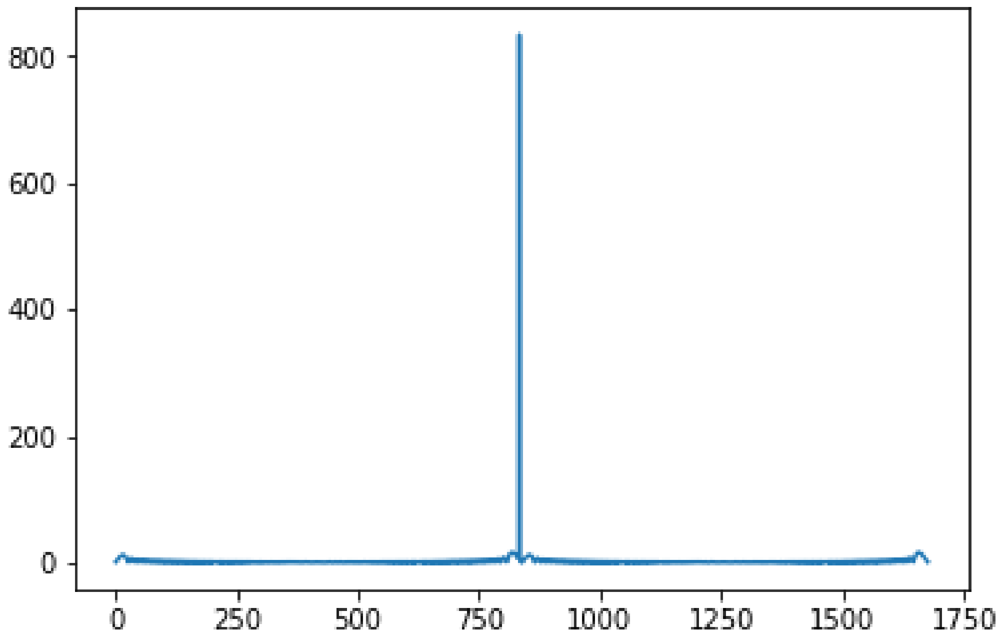
```
In [20]: with open('/home/pruthvirg/Downloads/x1.csv', 'r') as f:
             reader = csv.reader(f)
             z_seq = list(reader)
```

Rotating the signal in clockwise manner as required in the question.

```
In [23]: z_seq_shifted = z_seq[-5:]+ z_seq[:-5]

In [24]: z_seq_shifted = [complex((i[0].replace("i", "j")).strip()) for i in z_seq_shifted]

In [25]: z_seq = [complex((i[0].replace("i", "j")).strip()) for i in z_seq]

In [27]: z_1 = np.correlate(z_seq,z_seq_shifted,"full")
In [28]: z_1_mag = [abs(i) for i in z_1]
```

```
In [29]: plt.plot(z_1_mag)
```

```
Out[29]: [<matplotlib.lines.Line2D at 0x7f062e22be80>]
```



The output of the cross-correlation functio is obtained as above.

CONCLUSION:

Hence through this assignment we are able to take the output of a system for a given signal using convolution. We approached convolution using linear method and circular method.We use padding to make the filter of appropriate size before we do the circular convolution.Later we analysed the crosscorrelation function of Zadoff–Chu sequence with its circularly shifted version of itself.We observe a sharp peak in appropriate location according to the circular shift done.