EE2703: Applied Programming Lab

Assignment # 10

Sourav Sahoo, EE17B040

April 10, 2019

Contents

1 Introduction

In mathematics (in particular, functional analysis) convolution is a mathematical operation on two functions (f and g) to produce a third function that expresses how the shape of one is modified by the other. The term convolution refers to both the result function and to the process of computing it. Some features of convolution are similar to cross-correlation: for real-valued functions, of a continuous or discrete variable, it differs from cross-correlation only in that either $f(x)$ or $g(x)$ is reflected about the y-axis; thus it is a cross-correlation of $f(x)$ and $g(x)$, or $f(x)$ and $g(x)$. Convolution has applications that include probability, statistics, computer vision, natural language processing, image and signal processing, engineering, and differential equations.[1]

1.1 Some Important Formulae

$$y[n] = (f * g)[n] = \sum_{m=-\infty}^{\infty} f[m]g[n-m] \tag{1}$$

$$Y[m] = F[m]G[m] \tag{2}$$

$$(f * g_N)[n] = \sum_{m=0}^{N-1} f[m]g[(n-m)mod\ N] \tag{3}$$

2 Problem 1

Import the following libraries.

```python
import numpy as np
import scipy.signal as sp
import matplotlib.pyplot as plt
import csv
import string
```

Now we read from the `h.csv` file and form a NumPy array.

```python
filter = []
with open('h.csv','r') as csvfile:
    reader = csv.reader(csvfile,delimiter=',')
    for row in reader:
        filter.append(row)

filter = np.reshape(np.asarray(filter),(-1,))
filter = np.asarray(filter,dtype=np.float32)
w,h = sp.freqz(filter,1,whole=True)
h = np.fft.fftshift(h)
```

Next, the magnitude and phase response is found out using `scipy.signal.freqz()` and plotted.We see that the filter is a Low Pass Filter.

```python
plt.figure(figsize=(16,8))
plt.subplot(2,1,1)
plt.grid()
plt.title(r'Magnitude Response of Digital Filter')
plt.xlabel(r'$f\rightarrow$')
plt.ylabel(r'$|H(j\omega)|\rightarrow$')
plt.plot(w-np.pi,abs(h))

plt.subplot(2,1,2)
plt.grid()
plt.title(r'Phase Response of Digital Filter')
```

[1]https://en.wikipedia.org/wiki/Convolution

```
plt.xlabel(r'$f\rightarrow$')
plt.ylabel(r'$\angle H(j\omega)\rightarrow$')
plt.plot(w-np.pi,np.angle(h))
plt.show()
```

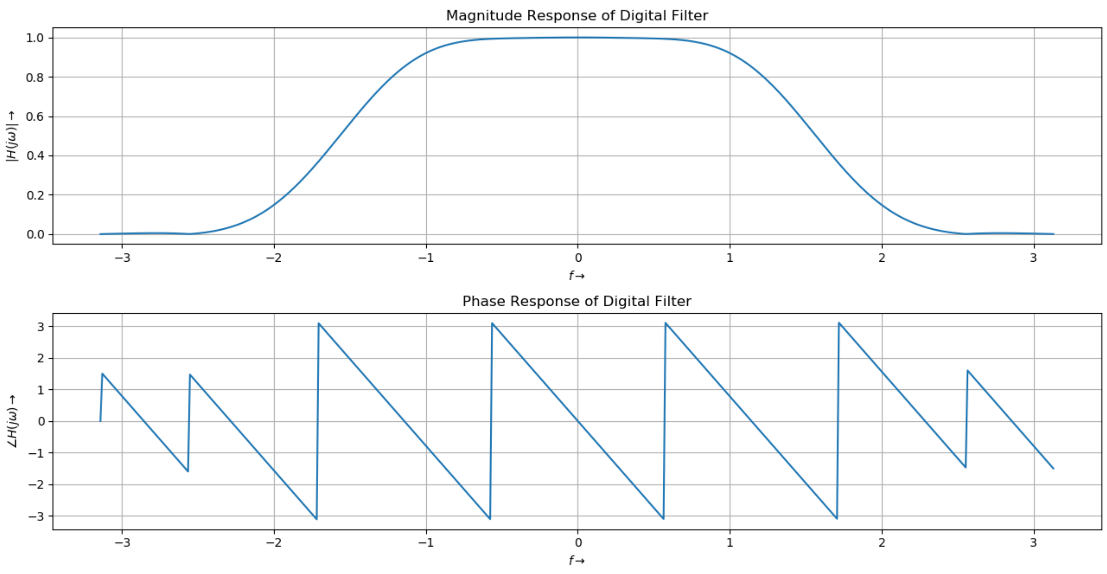

Figure 1: Magnitude and Phase response of the LPF

3 Problem 2

The input signal is :

$$x[n] = cos(0.2\pi n) + cos(0.85\pi n)\ for\ n = 1, 2, 3 \ldots 1024$$

```
n = np.arange(1,1025)
x = np.cos(0.2*np.pi*n)
x1 = np.cos(0.85*np.pi*n)

plt.figure(figsize=(16,8))
plt.subplot(3,1,1)
plt.grid()
plt.title(r'cos(0.2$\pi$t)')
plt.xlabel(r'$n\rightarrow$')
plt.ylabel(r'$x\rightarrow$')
plt.xlim([0,100])
plt.plot(n,x,'g-.')
plt.legend([r'cos(0.2$\pi$t)'])

plt.subplot(3,1,2)
plt.grid()
plt.title(r'cos(0.85$\pi$t)')
plt.xlabel(r'$n\rightarrow$')
plt.ylabel(r'$x\rightarrow$')
plt.xlim([0,100])
plt.plot(n,x1,'r-.')
plt.legend([r'cos(0.85$\pi$t)'],loc=1)

plt.subplot(3,1,3)
plt.grid()
plt.title(r'cos(0.2$\pi$t)+cos(0.85$\pi$t)')
plt.xlabel(r'$n\rightarrow$')
plt.ylabel(r'$x\rightarrow$')
plt.xlim([0,100])
plt.plot(n,x+x1)
plt.legend([r'cos(0.2$\pi$t)+cos(0.85$\pi$t)'])
plt.show()
```

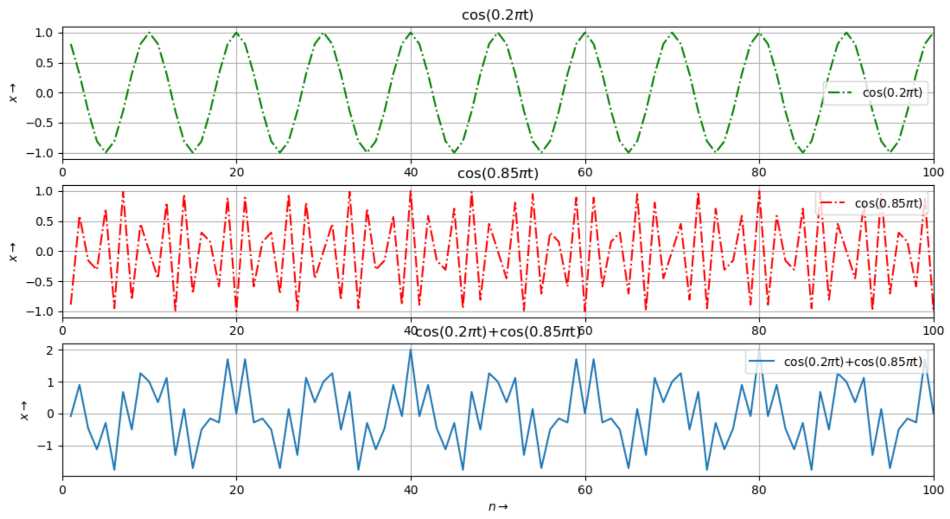

Figure 2: Components of the input to the LPF

4 Problem 3

Now, the input is superimposition of two sinusoids of different frequencies. We see that the low frequency signal passed through the filter where as the high frequency signal is blocked.

```
y = sp.fftconvolve(x,filter,mode='full')
plt.figure(figsize=(16,8))
plt.grid()
plt.title(r'Output Signal')
plt.xlabel(r'$n\rightarrow$')
plt.ylabel(r'$y\rightarrow$')
plt.xlim([0,200])
plt.plot(y)
plt.show()
```

5 Problem 4

Now, circular convolution is employed to find the output of the filter. We observe that the initial values are different for linear and circular convolution but the steady state values are identical.

```
padded_filter = np.zeros_like(x)
padded_filter[:len(filter)] = filter
```

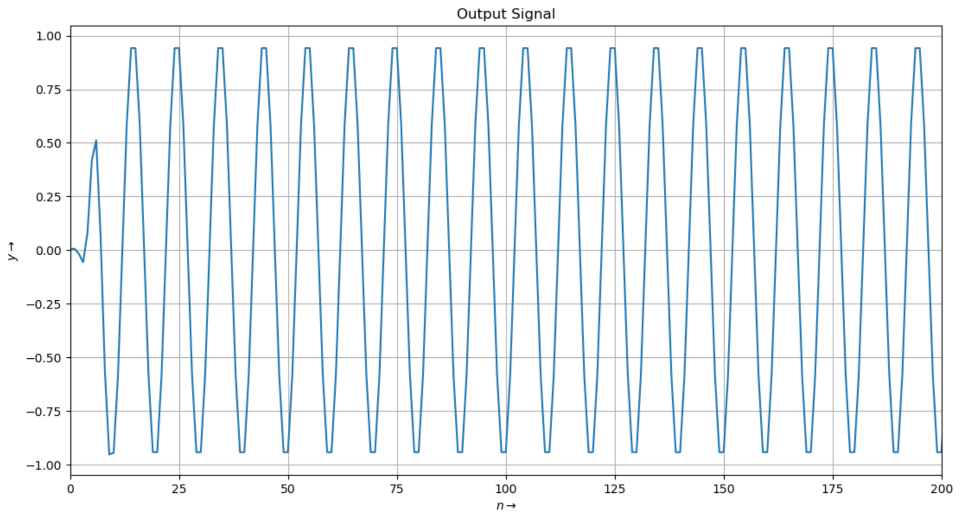

Figure 3: Output of the LPF using Linear convolution

```
y1 = np.fft.ifft(np.fft.fft(x)*np.fft.fft(padded_filter))
plt.figure(figsize=(16,8))
plt.grid()
plt.title(r'Output Signal')
plt.xlabel(r'$n\rightarrow$')
plt.ylabel(r'$y\rightarrow$')
plt.xlim([0,200])
plt.plot(np.real(y1))
plt.show()
```

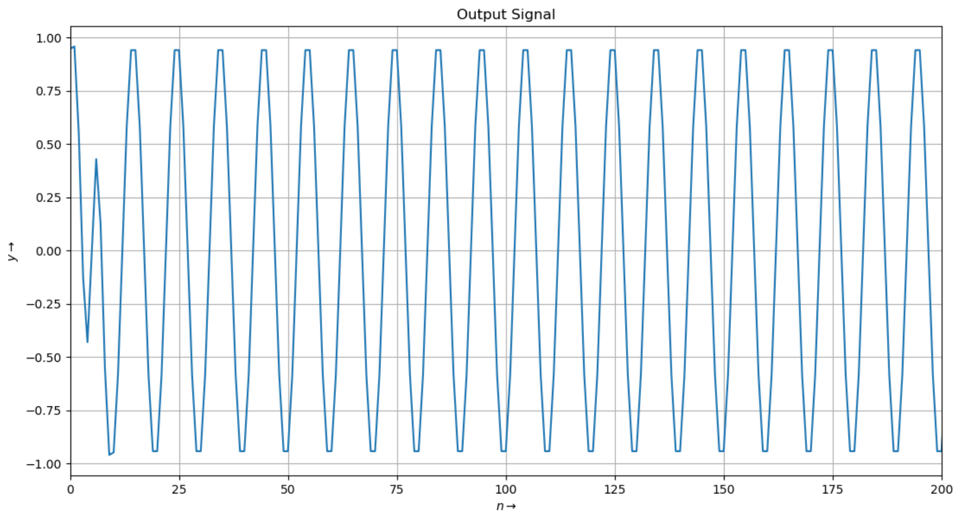

Figure 4: Output signal of LPF using Circular Convolution

6 Problem 5

In this problem, we implement linear convolution using circular convolution by following the algorithm mentioned in the problem sheet and then plot the output.

```
P = 2**4
N = 2**5 - 1
pad_filter = np.zeros(N)
pad_filter[:len(filter)] = filter
X = np.reshape(x,(-1,P))
Y = np.zeros(len(x)+len(filter)-1,dtype=np.complex)
for i in range(64):
    x_t = X[i]
    x_t_pad = np.pad(x_t,(0,N-P),'constant',constant_values=(0))
    y_t = np.fft.ifft(np.fft.fft(x_t_pad)*np.fft.fft(pad_filter))
    if i < 63:
        Y[16*i:16*i + N] += y_t
    else:
        Y[16*i:] += y_t[:27]

plt.figure(figsize=(16,8))
plt.grid()
plt.title(r'Output Signal by Linear Convolution using Circular Convolution')
plt.xlabel(r'$n\rightarrow$')
plt.ylabel(r'$y\rightarrow$')
plt.xlim([0,200])
plt.plot(np.real(Y))
plt.show()
```

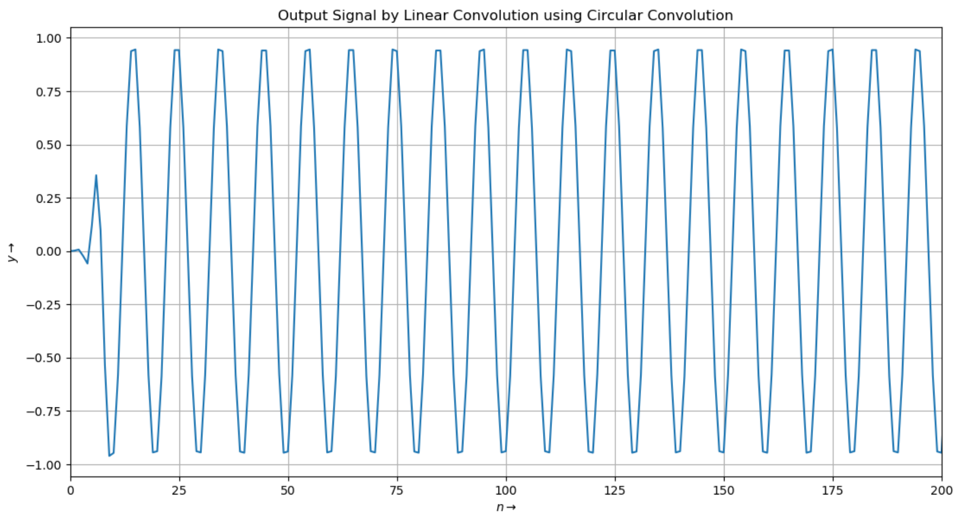

Figure 5: Linear convolution using Circular Convolution

We observe that even if there are differences in the beginning in the three cases, the steady state values of the outputs are identical.

7 Problem 6

We plot the magnitude and the phase of the Zadoff-Chu sequence reading from .csv file.We see that it is a constant amplitude signal.

```
zc_filter_raw,zc_filter = [],[]
with open('x1.csv','r') as csvfile:
    reader = csv.reader(csvfile,delimiter=',')
    for row in reader:
        zc_filter_raw.append(row)

for x in zc_filter_raw:
    for obj in x:
        zc_filter.append(complex(obj.replace('i','j')))

zc_filter = np.asarray(zc_filter,dtype=np.complex)

plt.figure(figsize=(16,8))
plt.subplot(2,1,1)
plt.grid()
plt.title(r'Magnitude of Zadoff-Chu Filter')
plt.xlabel(r'$n\rightarrow$')
plt.ylabel(r'$y\rightarrow$')
plt.xlim([0,100])
plt.stem(np.abs(zc_filter),markerfmt='.')

plt.subplot(2,1,2)
plt.grid()
plt.title(r'Phase of Zadoff-Chu Filter')
plt.xlabel(r'$n\rightarrow$')
plt.ylabel(r'$y\rightarrow$')
plt.xlim([0,100])
plt.stem(np.angle(zc_filter),markerfmt='.')
plt.show()
```

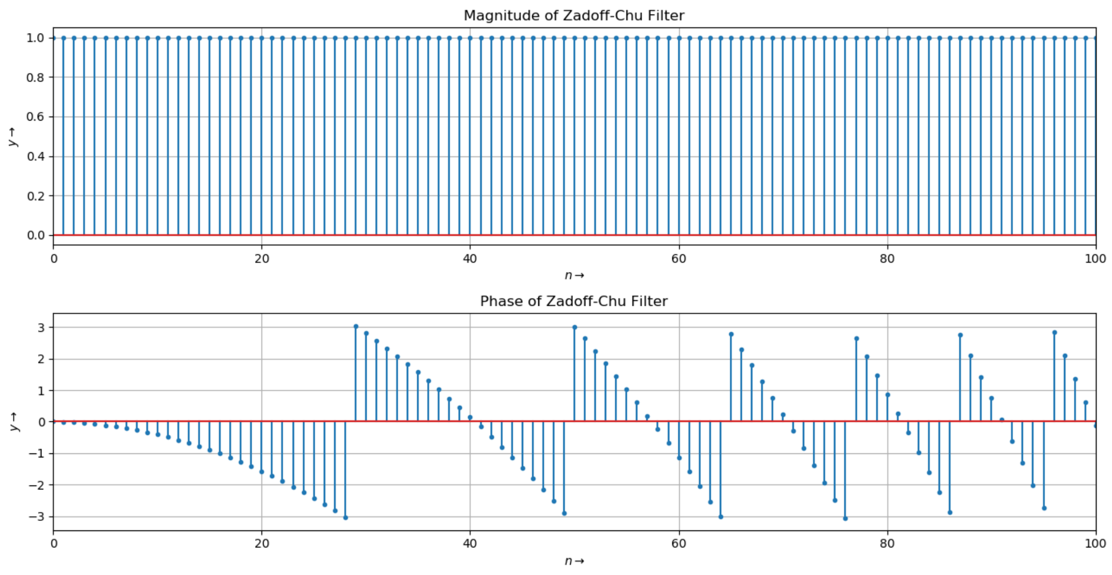

Figure 6: Magnitude and Phase response of the Zadoff-Chu Sequence

7.1 Circular Correlation

In this problem we do the auto correlation of the signal with a shifted version of itself and get a peak at the magnitude of the delay.

```
y1 = np.fft.ifft(np.fft.fft(np.roll(zc_filter,5))*np.conj(np.fft.fft(zc_filter)))
plt.figure(figsize=(16,8))
plt.grid()
plt.title(r'Auto correlation of shifted Zadoff-Chu Filter')
plt.xlabel(r'$n\rightarrow$')
plt.ylabel(r'$y\rightarrow$')
plt.xlim([-5,100])
plt.stem(abs(y1),markerfmt='.')
plt.show()
```

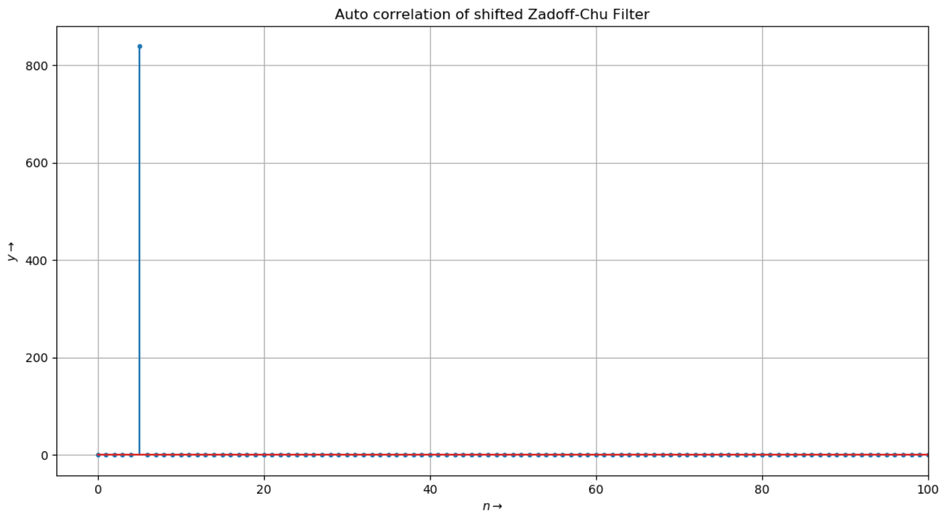

Figure 7: Auto correlation of shifted Zadoff-Chu Sequence

8 Conclusion

In this assignment, we implement convolution in time domain as well as in Frequency domain by finding the DFT.