Assignment No 10

Subhankar Chakraborty

April 10, 2019

1 Convolution

One of the main uses of DFT is to implement convolution. The convolution sum is defined as

$$y[n] = \sum_{k=-\infty}^{\infty} x[n-k]h[k]$$

In fourier space this becomes

$$Y[m] = X[m]H[m]$$

which is a major simplification of the formula.

2 Extraction the filter coefficients

We use the loadtxt module from numpy to extract the filter coefficients. The coefficients correspond to those of an FIR filter. The following piece of code does it.

```
filter_coeffs = np.asanyarray(np.loadtxt('h.csv').\\
reshape(1,-1))[0].reshape(-1,1)
```

3 Analysing the filter

We plot the magnitude and phase plots of the filter. The *freqz* function from the signal module helps in doing this.

```
w,h = sp.freqz(filter_coeffs,whole=True)
```

Now we plot and analyse the filter. figure-1 gives the magnitude plots.

We can deduce the following things from the plot.

- The magnitude is 1 around frequencies 0 and 2π and 0 around π. So this is like a low pass filter.
- Apart from the discontinuities, the phase is linear. So, the filter has a constant group delay and hence, the entire signal is delayed by some constant which may or may not be an integer.

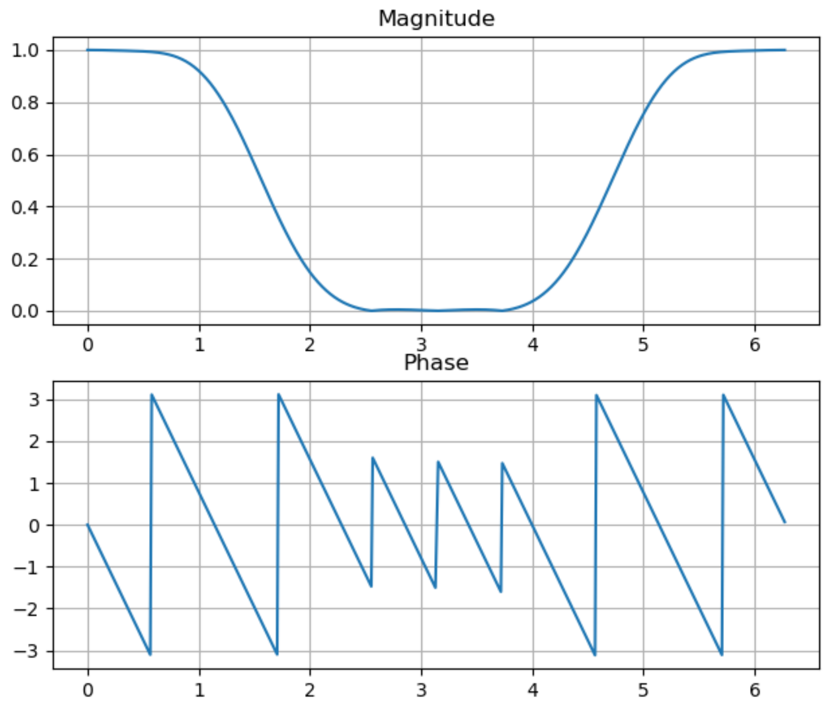

Figure 1: Response of the filter given

4 The signal $x[n] = cos(0.2\pi n) + cos(0.85\pi n)$

We plot x[n] where n varies from 1 to 2^{10}. Note that the signal is plotted only upto values 100 to get an idea about the shape of the signal.

We observe the signal is a superposition of 2 signals, one of with digital frequency 0.2π and the other with digital frequency 0.85π.

5 x[n] through the filter h[n]

We pass x[n] through the filter h[n]. As h[n] is basically a low pass filter, we expect the component of x[n] with digital frequency 0.85π to be cutoff completely. The following code does it.

```
y = convolve2d(x,filter_coeffs, mode='full')
```

Plotting the signal, we see in figure-3

As expected, what is left now is only the low frequency component of x[n], ie, the one with digital frequency 0.2π is only left.

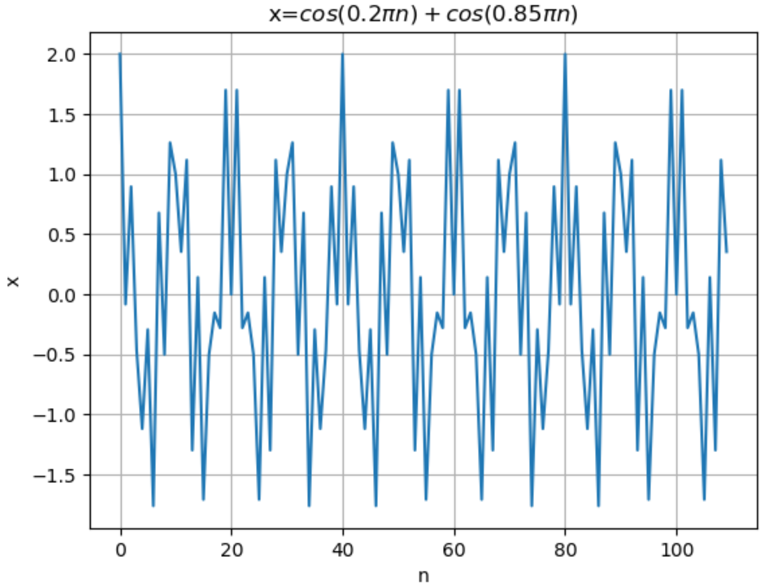

Figure 2: $x[n] = cos(0.2\pi n) + cos(0.85\pi n)$

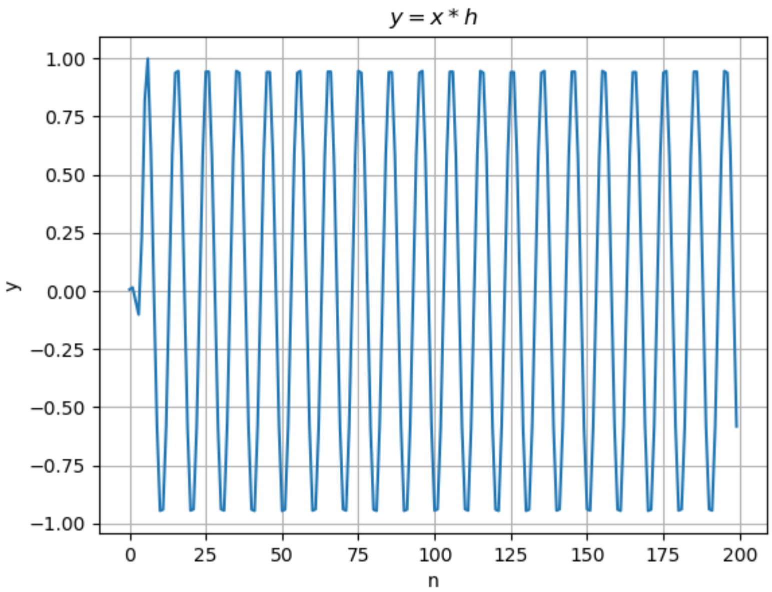

Figure 3: $x[n]$ convolved with $h[n]$

6 Convolution using DFT

We can use the equation

$$Y[m] = X[m]H[m]$$

where A[m] is the DFT of a[n] and then calculate the IDFT to find the convolution sum. Note that we need to zero pad x or h accordingly to make the DTS's multiplyable. The following code does it.

```
Y=np.concatenate((filter_coeffs,np.zeros((len(x)-len(filter_coeffs)
,1))))
temp2 = Y[:,0]
temp1 = x[:,0]
y1=np.fft.ifft(np.fft.fft(temp1)*np.fft.fft(temp2))
```

The plot is shown in figure-4

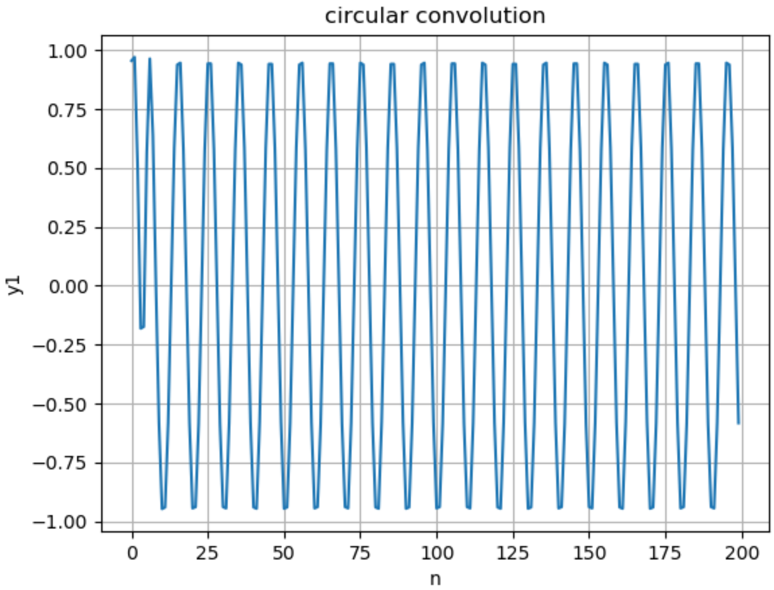

Figure 4: $x[n]$ circular convolved with $h[n]$

There is some error at the beginning from the case of the direct linear convolution. The steady state values are however, exactly similar.

7 Linear convolution using circular convolution

Using the algorithm mentioned in the assignment, we implement linear convolution using circular convolution. The following function does it.

```
def circular_conv(x,h):
P = len(h)
n_ = int(ceil(log2(P)))
h_ = np.concatenate((h,np.zeros(int(2**n_)-P)))
```

```
P = len(h_)
n1 = int(ceil(len(x)/2**n_))
x_ = np.concatenate((x,np.zeros(n1*(int(2**n_))-len(x))))
y = np.zeros(len(x_)+len(h_)-1)
for i in range(n1):
    temp = np.concatenate((x_[i*P:(i+1)*P],np.zeros(P-1)))
    y[i*P:(i+1)*P+P-1] += np.fft.ifft(np.fft.fft(temp)*np.fft.fft(np.
    concatenate((h_,np.zeros(len(temp)-len(h_)))))).real
return y
```

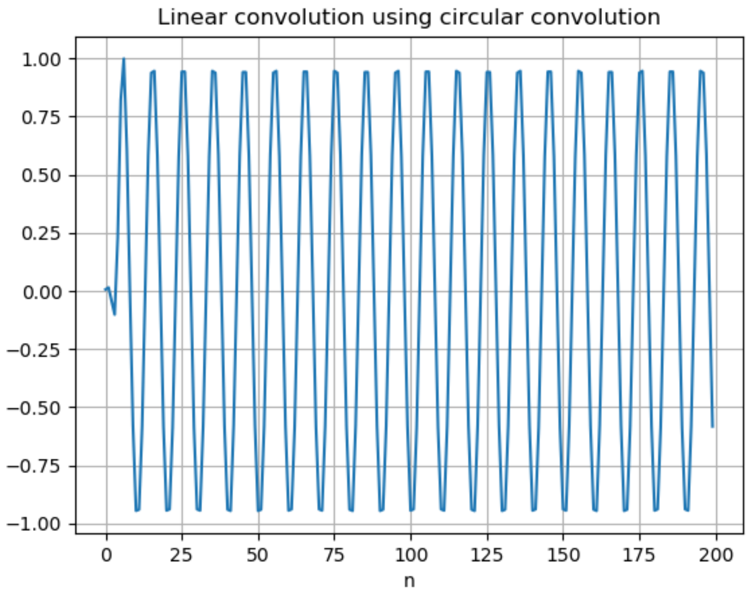

Figure 5: $x[n]$ linear convolved with $h[n]$

We use this function to find the linear convolution of x[n] and h[n].The result is shown in figure-5. We see that this is exactly equal to the value obtained from the linear convolution.

8 Zadoff-Chu sequence

8.1 Reading the coefficients from the file

The following code is used to read the values of the Zadoff-chu coefficients.

```
f = open('x1.csv','r')
raw_data =f.read().splitlines()
```

```
for p in range(len(raw_data)):
    raw_data[p] = complex(raw_data[p].replace('i','j'))
```

8.2 Auto-correlation with a shifted version.

We take the auto-correlation of the sequence with a shifted version of itself. We should expect a peak at n = 5 and zero everywhere else, from the properties of the Zadoff-Chu sequence. figure-6 shows the plot.

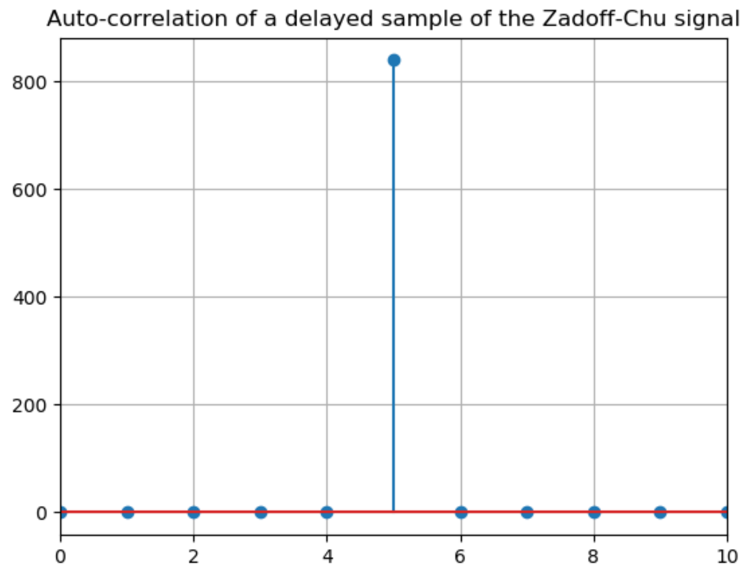

Figure 6: Auto-correlation of circular shifted Zadoff-Chu Sequence

As expected, the peak occurs at n = 5 and is zero everywhere else. The code for doing the above is

```
y2 = np.fft.ifft((np.fft.fft((np.roll(ZADOFF_CHU_COEFFICIENTS[:,0],5)))
*np.conj(np.fft.fft(ZADOFF_CHU_COEFFICIENTS[:,0]))))
```