Assignment No. 10

Srivenkat A (EE18B038)

June 3, 2020

Introduction

The assignment focuses on using numpy.fft() library in python to calculate the Discrete fourier transforms of two signals and in turn using them to computationally realise linear convolution.

Low Pass FIR Filter

We are given the coefficients of a FIR Low Pass Filter in the **h.csv** file. We read the coefficients and using the **scipy.freqz** function, we compute the frequency response of the FIR filter and plot the corresponding bode plot:

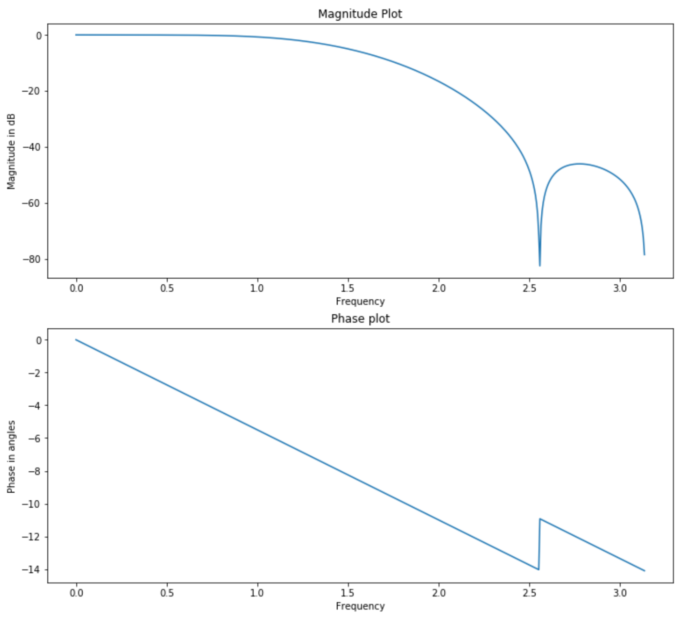

Figure 1: Caption

The filter has all real coefficients in its impulse response and has a stop band at around 2.5 rad/s.

Mixed Frequency Input Signal

We create a 1024 element vector of the function;

$$x(t) = cos(0.2\pi t) + cos(0.85\pi t)$$

On plotting the input signal, we get The input signal has frequency compo-

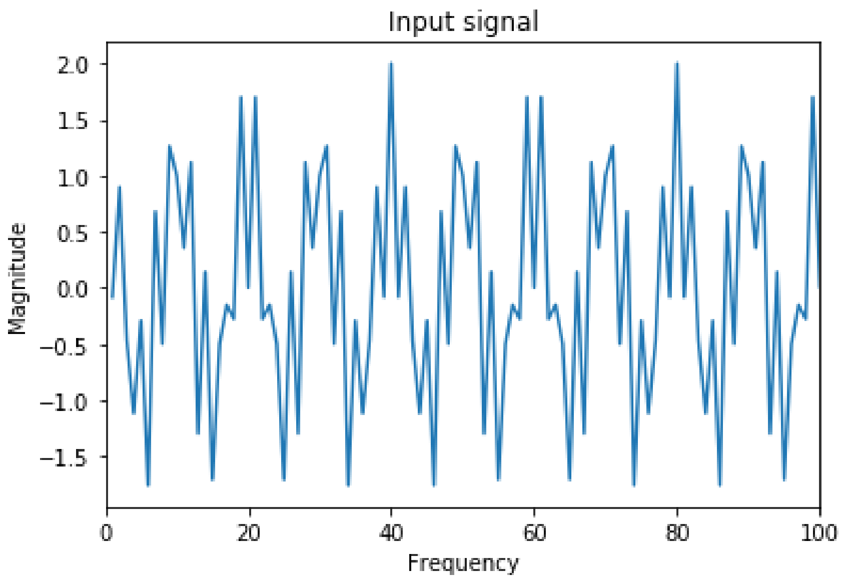

Figure 2: Caption

nents at around 0.63 rad/s and 2.670 rad/s. Therefore, by passing it through a matching low pass filter, we can extract the 0.2π component alone

Linear Convolution

To get the output of passing the input signal into the FIR filter, we can convolve the input signal and the impulse response of the filter using the **numpy.convolve** function.

We get the convolution output as

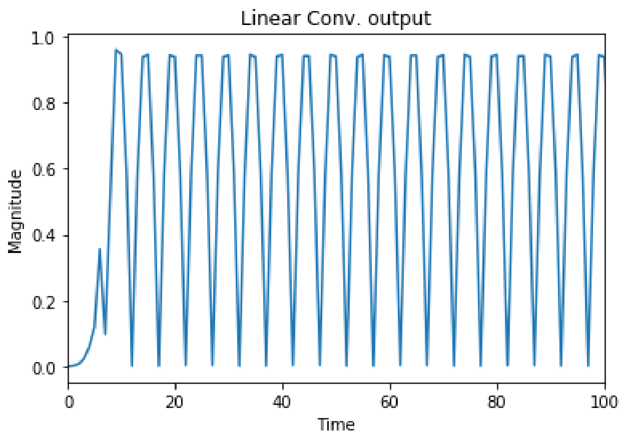

Figure 3: Caption

It matches the expected sinusoid at a single frequency.

Circular Convolution

Calculating the linear convolution in time domain in computationally very expensive. So, we can find their corresponding fourier transforms, multiply them and find the corr. inverse fourier transform to get the convolution output. Using the FFT algorithm to calculate the discrete fourier transform is computationally very efficient.

The convolution thus obtained is the circular convolution. Its plot is:

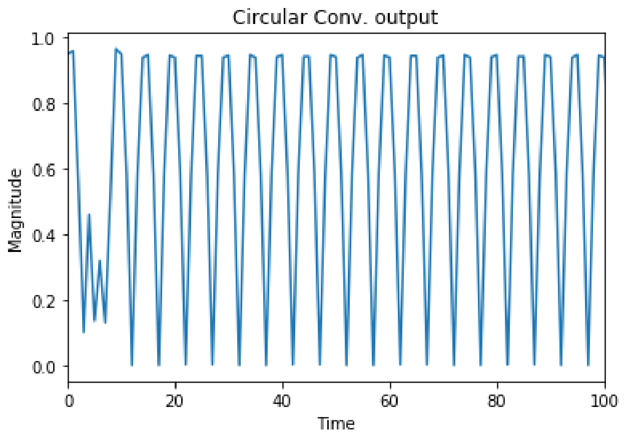

Figure 4: Caption

Using DFTs for Linear Convolution

By dividing the input signal into smaller blocks and zero padding the input signal appropriately to remove the overlaps, we can get the linear convolution output in a computationally efficient way.

The output thus obtained is:

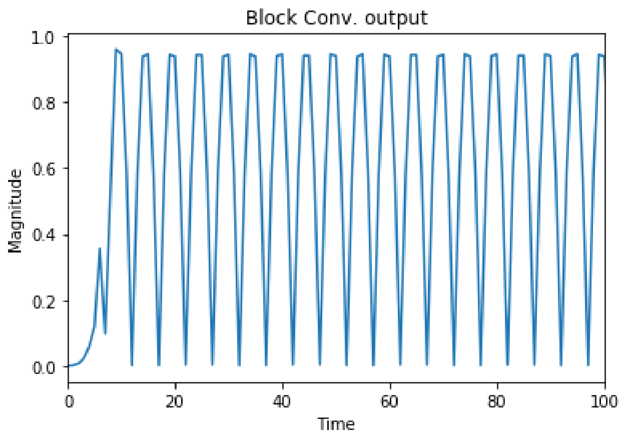

Figure 5: Caption

As expected, it is identical to the linear convolution output. Since the circular convolution is calculated for a particular block of the input signal, it is also called block convolution.

Correlation

Correlation can be defiend as:

$$R(jw) = A(jw) * conj.(B(jw))$$

We calculate the correlation of the Zadoff–Chu sequence with a cyclically shifted version of itself.

The Zadoff-Chu sequence has the following properties:

- It is a complex sequence.
- It is a constant amplitude sequence.
- The auto correlation of a Zadoff–Chu sequence with a cyclically shifted version of itself is zero.
- Correlation of Zadoff–Chu sequence with the delayed version of itself will give a peak at that delay.

It is highly used in communication. We try to calculate the correlation of the sequence with a cyclic 5-right shifted version of itself. The observed plot is:

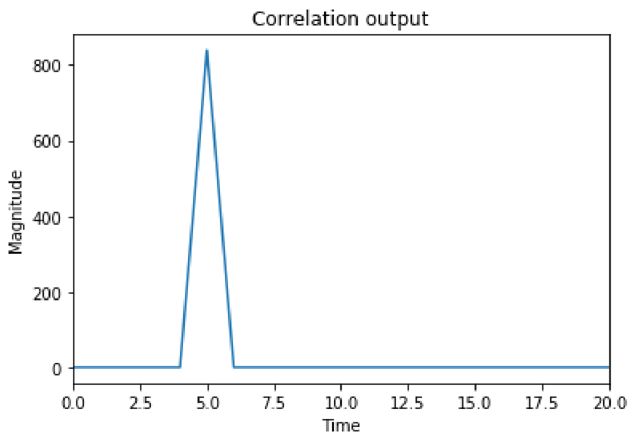

Figure 6: Caption

As expected, the correlation output gets a peak at +5 and is 0 everywhere else.

Conclusion

To convolve two signals of length n each, the time complexity is of the order of n^2 and thus is computationally very expensive. Instead, optimising using the FFT algorithm, we get the complexity to reduce to $nlog(n)$. We make use of this optimisation in calculating the linear convolution output using block convolution.
For the Zadoff-Chu sequence, the correlation of the sequence with the cyclic shifted version of itself has a peak at the point equal to the shift.