Assignment No. 8

Srivenkat A (EE18B038)

May 5, 2020

Introduction

The assignment focuses on using exploring the Discrete fourier transform, the computaional method uses to analyse the spectra of a signal and reconstruction of the signal from the DFT. We use numpy.fft() to calculate the DFT.

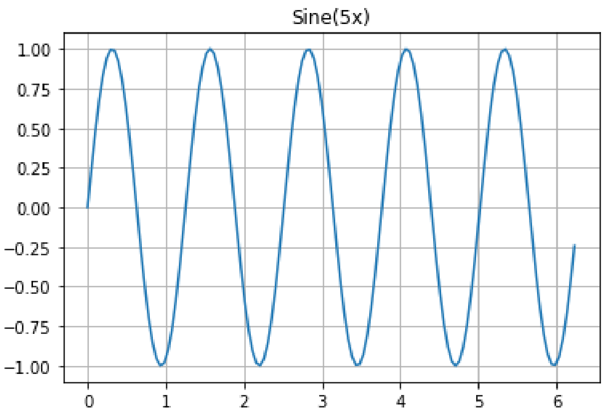
Calculating DFT for known signals

General sinusoid

For the sinusoid **sine(5x)**,

$$sin(5x) = \frac{e^{jx} - e^{-jx}}{2}$$

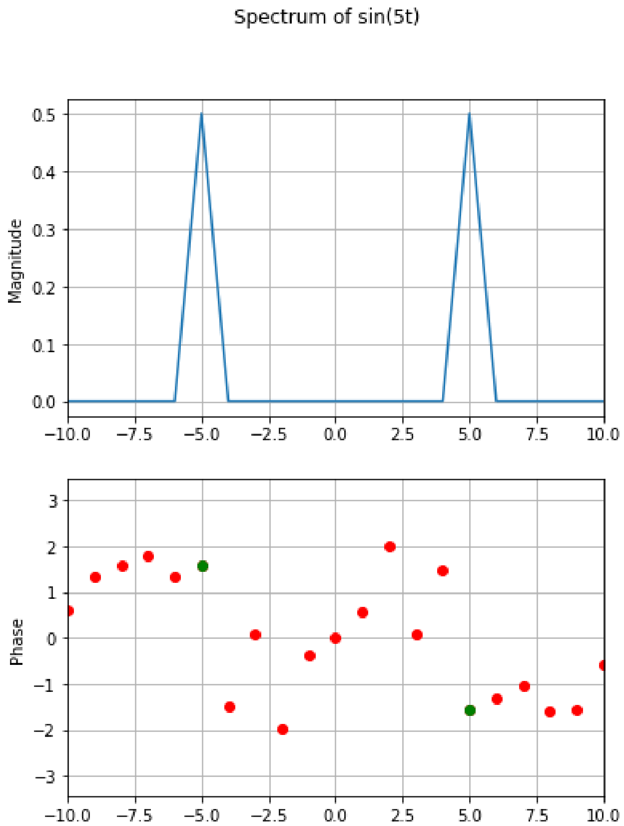
The signal waveform:



Therefore, the expected spectra is

$$Y(w) = \frac{\delta(w-1) - \delta(w+1)}{2j} \tag{1}$$

The DFT is a sampled version of the scaled periodic extension of the continuous time fourier transform.
In the principal period, the only region of analysis in DFT, we expect the DFT to be 2 impulses at +w and -w The 128 point DFT looks like: Because

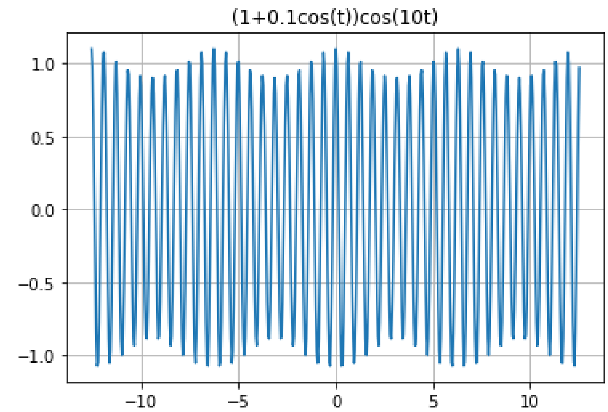


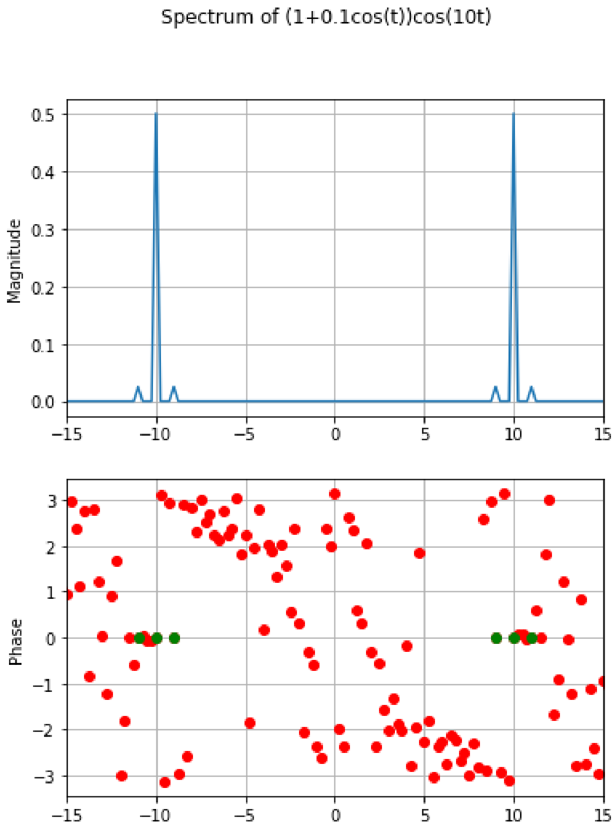
of the limited sampling frequency, instead of 2 sharp impulses, we get 2 triangles, peaking at +w and -w. In the attached magnitude and phase plots, the green points alone show those frequencies where the magnitude is above 0.001

Amplitude modulated sinusoid

For the input sinusoid **(1+0.1cos(t))*cos(10t)**, visualisation:

It ha 3 distinct frequency components with the carrier wave frequency term being the most dominant. So, to observe the modulated frequencies distinctly, we need a high sampling rate, i.e., a longer DFT. So, the 512 point DFT looks like:





Manually calculating the DFT,

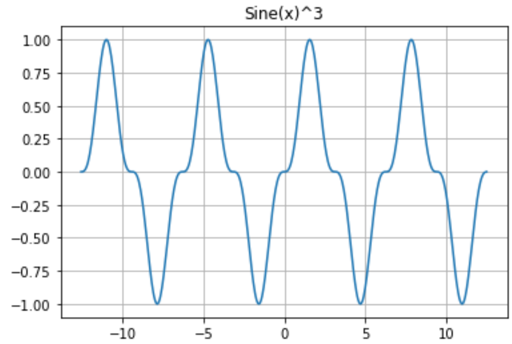
$$(1 + 0.1cosx)cos(10x) = cos(10x) + 0.05(cos(11x) + cos(9x))$$

So, the signal has frequency components of 9Hz,10Hz and 11Hz.
As expected, we get a larger peak at +-10Hz and relatively smaller peaks at $\pm 10 \pm 1Hz$.

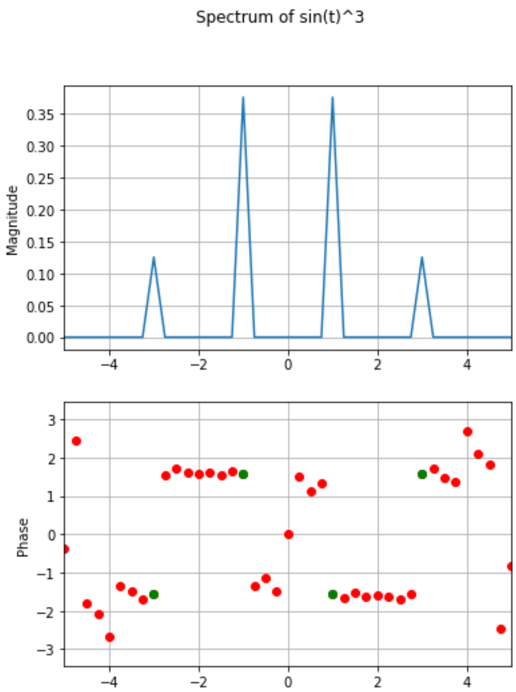
Cube of a sinusoid

For the input signal $sin^3(x)$,

$$sin^3(x) = \frac{3sin(x) - sin(3x)}{4}$$



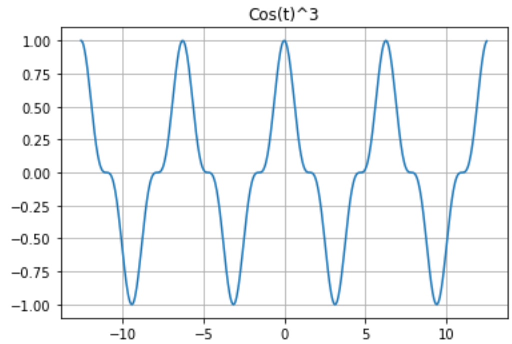
The underlying frequency components are at 1Hz and 3Hz. Therefore we expect to see spikes at $\pm 1Hz$ and $\pm 3Hz$. Since 1Hz has thrice the amplitude of 3Hz, spike at 1Hz should be thrice the height of spike at 3Hz
The corresponding 512 point DFT looks like: So, the DFT matches with



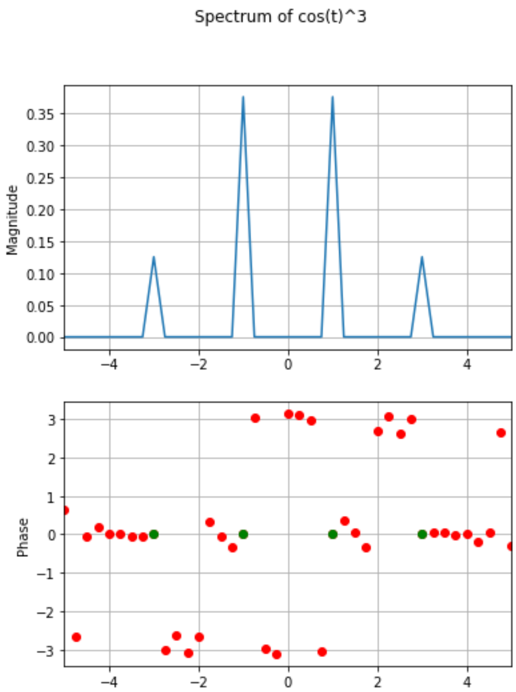
the expectations.

For the input signal $cos^3(x)$,

$$cos^3(x) = \frac{3cos(x) + cos(3x)}{4}$$



The underlying frequency components are at 1Hz and 3Hz. Therefore we expect to see spikes at $\pm 1Hz$ and $\pm 3Hz$. Since 1Hz has thrice the amplitude of 3Hz, spike at 1Hz should be thrice the height of spike at 3Hz.
The corresponding 512 point DFT looks like: So, the DFT matches with



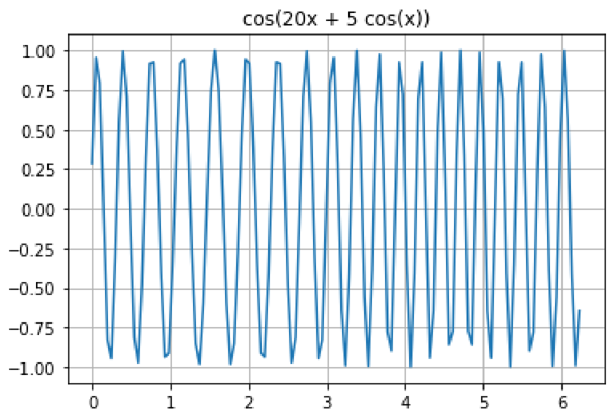
the expectations.

The above DFTs are calculated from the following code snippet:

```
x = np.linspace(-4*np.pi,4*np.pi,513)
x=x[:-1]
y = pow(np.sin(x),3)
Y = fft.fftshift(fft.fft(y))/512
w = np.linspace(-64,64,513)
w=w[:-1]
t = np.linspace(-4*np.pi,4*np.pi,513)
t=t[:-1]
y = pow(np.cos(t),3)
Y = fft.fftshift(fft.fft(y))/512
w = np.linspace(-64,64,513)
w=w[:-1]
```

Frequency Modulated signal

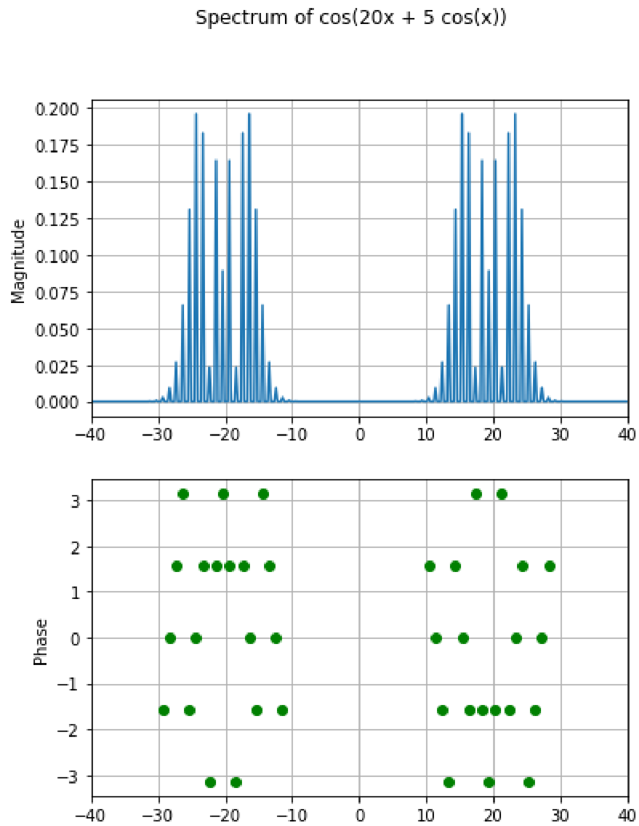
For the input signal cos(20t+5cos(t)), The frequency of the signal, itself has another sinusoidal component in it. So, it is an example of frequency modulation. Input signal: On plotting the DFT, we get: We can observe a set of



impulses, closely packed aroound $\pm 20 Hz. Itisbecause, themodulatedsignalhasitsfrequencycomponen$ $0.001 aredisplayed.$

We get the above graph from this code snippet:

```
x = np.linspace(-4*np.pi,4*np.pi,513)
x=x[:-1]
y = np.cos(20*x + 5*np.cos(x))
Y = fft.fftshift(fft.fft(y))/512
w = np.linspace(-64,63,513)
w=w[:-1]
```

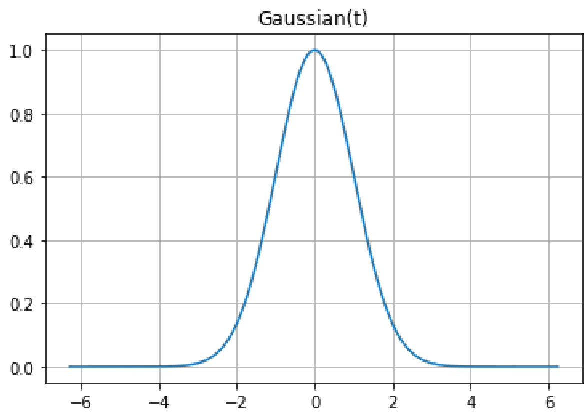


Gaussian signal

The input gaussian signal has the expression:

$$f(t) = e^{-t^2/2}$$

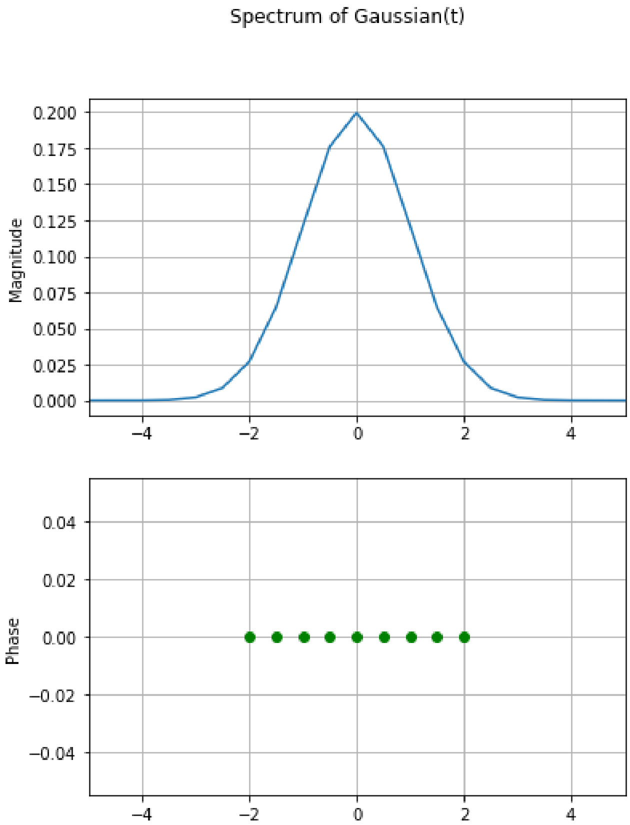
It is of the form: It is not bandlimited, i.e., it has non-negligible frequency



components all along the spectral axis. On calculation, it has the continuous time fourier transform as:

$$F(jw) = \sqrt{2\pi} e^{-w^w/2} \tag{2}$$

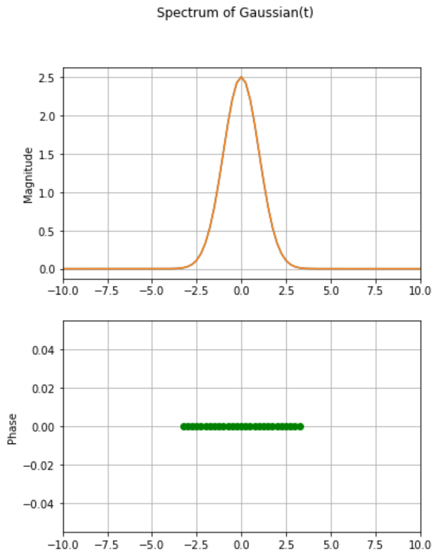
Since the fourier transform is a real function, its phase values should be zero along the spectral axis. On trying with N=256, time range of $(-2\pi to 2\pi)$, we get



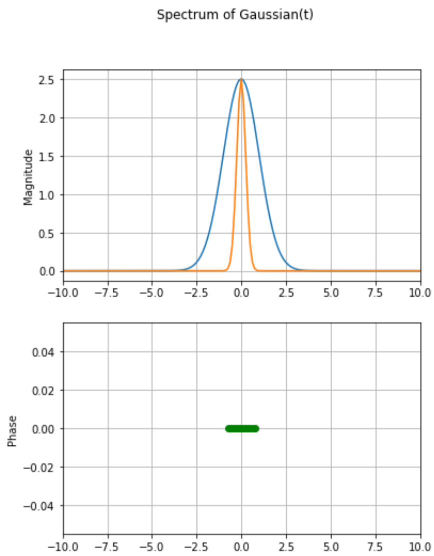
As expected, the plot is choppy because of low sampling rate. So, we try with different possible N values and time ranges to get the best estimate using the following code:

```
t = np.linspace(-4*np.pi,4*np.pi,513)
t=t[:-1]
y = np.exp(-0.5*pow(t,2))
w = np.linspace(-64,64,513)
w=w[:-1]
Y = fft.fftshift(abs(fft.fft(y)))/512
Y = Y*np.sqrt(2*np.pi)/max(Y)
Y2 = np.exp(-w**2/2)*np.sqrt(2*np.pi)
err = abs(Y-Y2).max()
```

N=512 time range:$(-4\pi to 4\pi)$ error: order of 10e-19

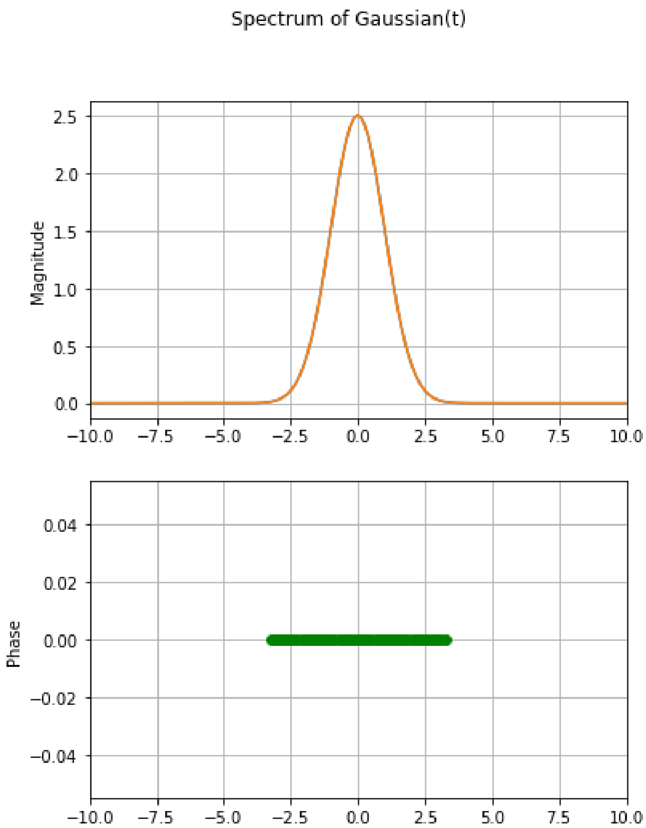


We need to increase the sampling rate and time range proportionally. Else, N=1024 time range=$(-2\pi to 2\pi)$



Though the graph obtained is very smooth because of the high sampling rate, error is of the order of 1-3.

For N=1024, time range=($-8\pi to 8\pi$): The graph is smooth and the error



is of the order of 10e-16. As we increase the N and time range values, we can get a more precise approximation.

Conclusion

The fft() library, used here to calculate the DFT provides an optimised way to calculate the Digital Fourier Transform for a discrete time signal. From the examples, it is observed that:

- We choose the block length as exponet of 2 to make optimal use of FFT algorithm. For Sinusoids, having a higher value of N gives a higher sampling rate and hence, sharper spikes at the component frequencies.
- For amplitude modulated and frequency modulated signals, we get the freq samples of the modulated wave in a band around the carrier wave frequency.
- For gaussian input signal, increasing N and time range gives better estimates