EE2703 : Applied Programming Lab
Assignment 8
The Digital Fourier Transform

Aditya Nanda Kishore
EE20B062

April 14, 2022

Introduction

In this assignment, we are going to explore the Discrete Fourier Transform(DFT) of various signals using **NumPy** commands

Importing Necessary Libraries

```
import numpy as np
import matplotlib.pyplot as plt
```

DFT spectrum function

We first define a dictionary of functions, whose keys are strings and values corresponding are the actual mathematical functions defined using lambda function

```
func_list = {'sin(5x)': lambda x : np.sin(5*x),
'(1+0.1cost)*cos(10t)': lambda x : (1+0.1*np.cos(x))*np.cos(10*x),
'cos^3' : lambda x : np.cos(x)**3,
'sin^3' : lambda x : np.sin(x)**3,
'fm' : lambda x : np.cos(20*x+5*np.cos(x)),
'gauss' : lambda x : np.exp(-x**2/2) }
```

Since this assignment only asks us to plot the transform function for various functions with different time arrays and sampling rates, I have created a spectrum function to make my job easier. Our function's arguments are

- **func**- A String which maps to the required function in the dictionary defined above.
- **start**- Start of the time array.
- **end**- End of the time array.
- **N**- Sampling Rate. This is the number of samples of the time , frequencies and function in both time and frequency domain.This has to be varied accordingly for better accuracy.
- **w** - Frequency limit for frequency domain. This is later modified into an array of frequencies on which our plot is plotted.
- **phase-split-plot**- A boolean. If False, it has to plot the phase plot in red dots.
- **magnitude-split-plot** and **mag-min**- A boolean and a number. If True, It has to plot green dots in the phase plot at the points wherever magnitude is greater than *mag-min*.If false, It only plots the red dots as above.
- **xlim**- Limits for the x-axis in frequency domain
- **plot-name**- String that sets the title of the plot
- **fig-name**- Saves our final plot.png as this string.

We write the code for the same using **numpy.fft.fft** function. We define a time array and map a function to the time array. Then, the **numpy.fft.fft** maps it to frequency domain by finding it's fourier transform. But we have to take care of magnitude and phase plots by making a few modifications in the arguments of the function like

- Peaks need not be present at where we expect them to be. This has to be fixed by **fftshift** function.
- Peaks may have a different magnitude from what we expect. This is because of the scaling up of the fourier transform by a certain number.
- Phase at peaks may vary.

These can be fixed by choosing appropriately scaled Transform function while plotting as well as choosing proper time arrays and sampling rate.

So, the function would look something like this

```
def spectrum(func,start, end, N, w,mag_plot, magnitude_split_plot,
phase_split_plot, mag_min, xlim, plot_name, fig_name):
    t= np.linspace(start,end, N+1)[:-1]
    w= np.linspace(-w,w,N+1)[:-1]
    y= func_list[func](t)
    if func == 'gauss' :
        Y = np.fft.fftshift(np.fft.fft(y))/N
        Y = Y *  np.sqrt(2*np.pi)/max(Y)
    else:
        Y= np.fft.fftshift(np.fft.fft(y))/N
    plt.figure()
    plt.subplot(2,1,1)
    plt.plot(w,abs(Y), lw = 2)
    plt.xlim([-xlim, xlim])
    plt.ylabel(r"$|Y|$",size=16)
    plt.title(plot_name)
    plt.grid(True)
    plt.subplot(2,1,2)
    if phase_split_plot == False:
        plt.plot(w,np.angle(Y),'ro',lw=2)
    if (magnitude_split_plot == True):
        ii= np.where(abs(Y)>mag_min)
        plt.plot(w[ii],np.angle(Y[ii]),'go',lw=2)
    plt.xlim([-xlim, xlim])
    plt.ylim([-2, 2])
    plt.ylabel(r"Phase of $Y$",size=16)
    plt.xlabel(r"$k$",size=16)
    plt.grid(True)
    plt.savefig(fig_name)
    plt.show()
    return Y,w
```

In further sections, we are going to use this function with changing arguments again and again to get required plots.

DFT of a sinusoid

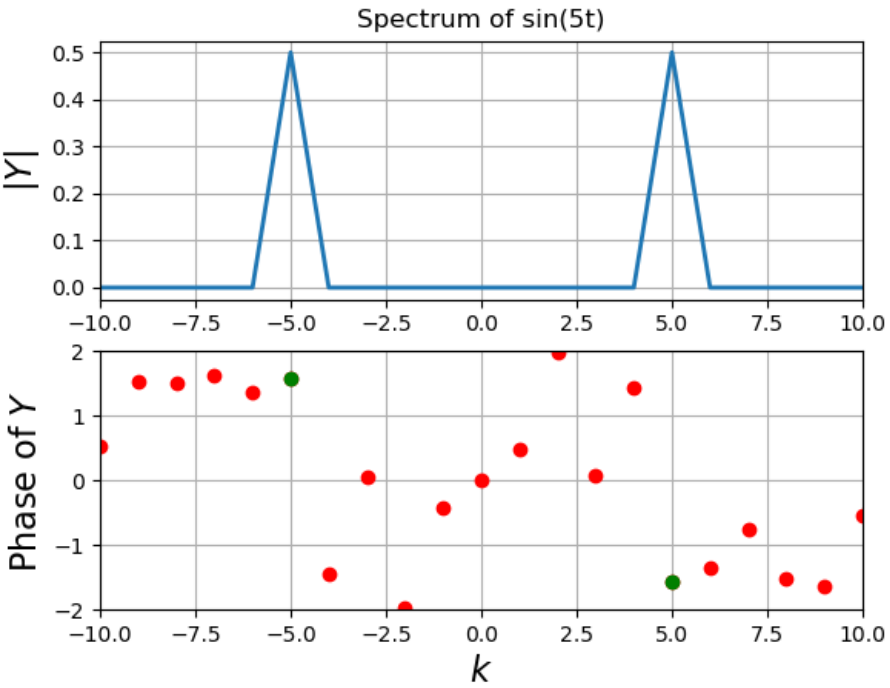
Now Let's consider a sinusoid function

$$f(x) = sin(5x) = \frac{e^{j5x} - e^{-j5x}}{2j} \tag{1}$$

The fourier transform of this function is

$$\mathcal{F}(f(x))(\omega) = \frac{\delta(\omega - 5) - \delta(\omega + 5)}{2j} \tag{2}$$

For sinusoid, let's take a sampling rate of 128 and time array goes from $[0, 2\pi)$.

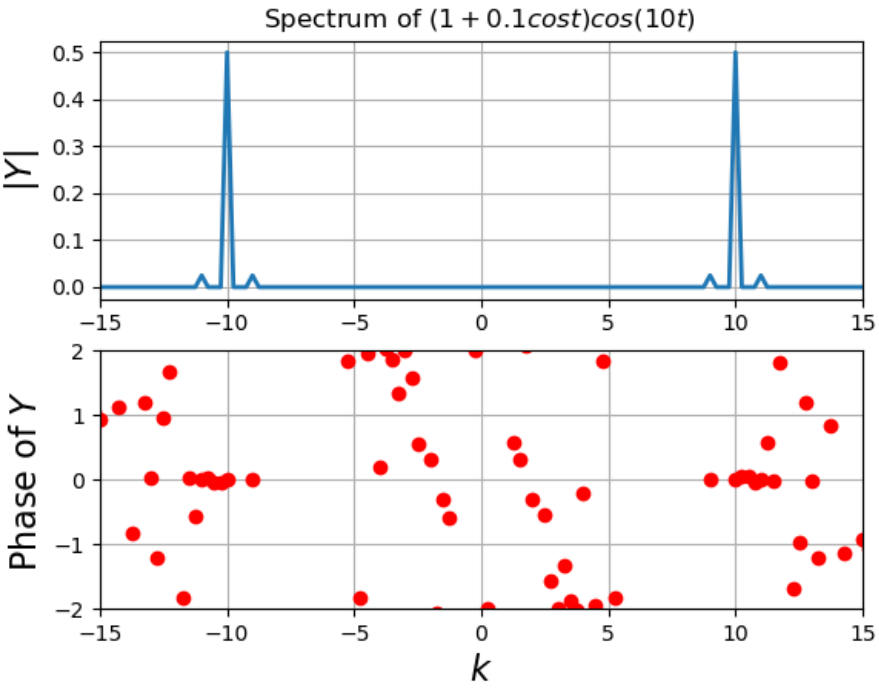


Amplitude Modulation

Let's consider a function

$$f(x) = (1 + 0.1cosx)cos(10x) \tag{3}$$

For this , we need a higher sampling rate as peaks are closer and transform changes rapidly.We take it to be 512.



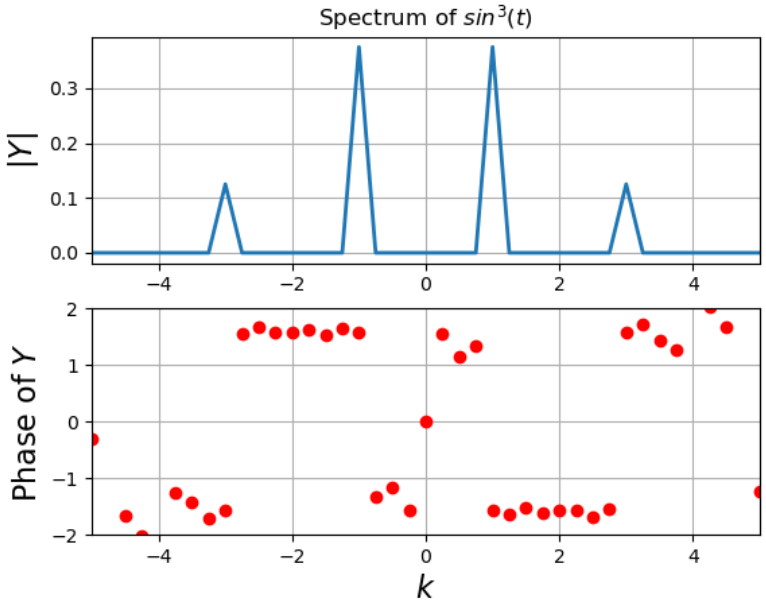
DFT of Cubic Sinusoids

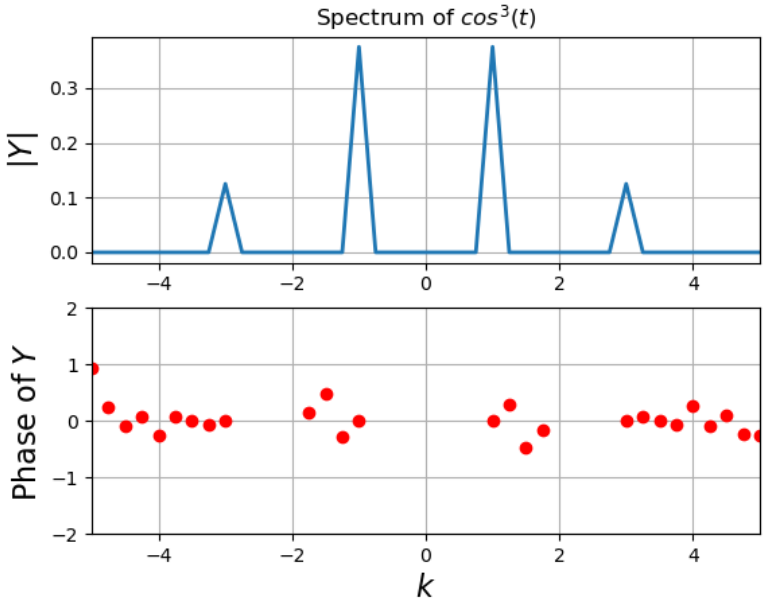
Given the functions sin^3x and cos^3x.We know that,

$$sin^3x = \frac{3sinx - sin3x}{4} \tag{4}$$

$$cos^3x = \frac{cos3x + 3cosx}{4} \tag{5}$$

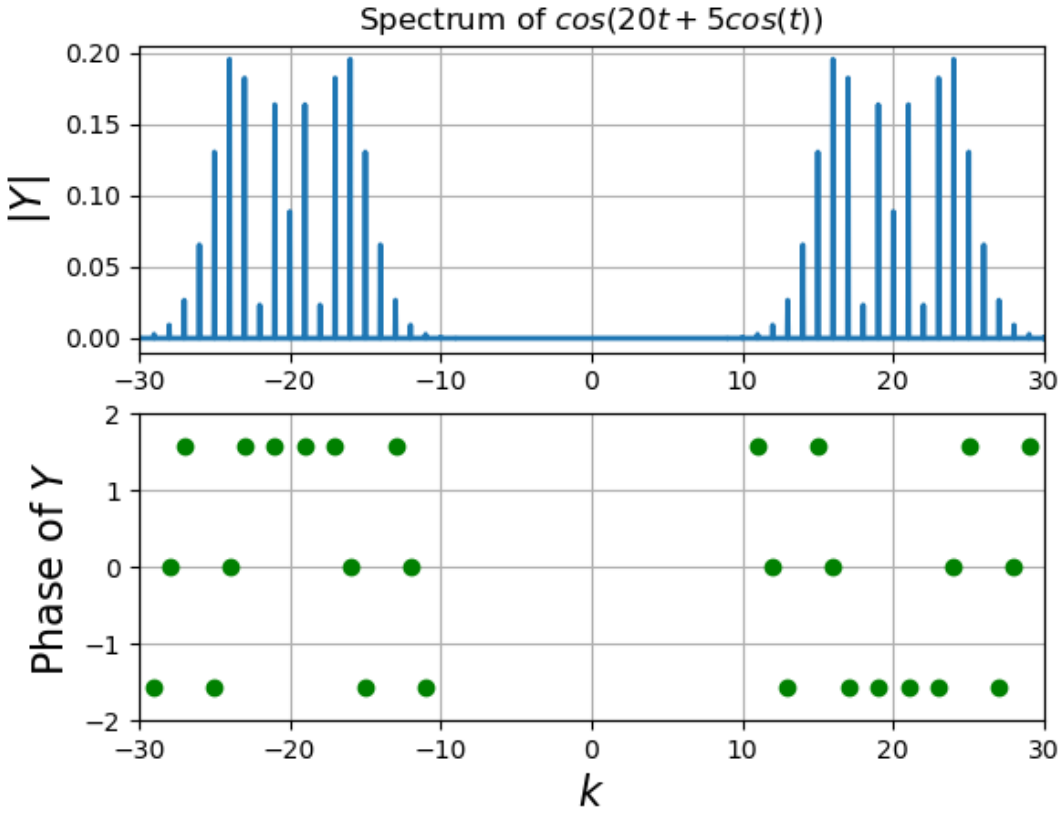
So, We expect two peaks at frequencies = 1 and 3 and phase difference to be π for sin^3x at peaks and phase to be zero at peaks for cos^3x. Using $N = 512$ for both, we would get





Frequency Modulation

For frequency Modulation, we will consider the function $cos(20t + 5cos(t))$. For this we can't take a low N as the energy in the sidebands of the peaks is comparable to the signal. If we take time from $[-32\pi, 32\pi)$ and $N = 16384$ and using the same spectrum function as above, we get



Gaussian Distribution

This is a special case. Frequency Transform of a gaussian series is also a scaled gaussian function.It is also not "bandlimited" in frequency, which means the spectrum that we are gonna depends on the ω_{lim} we choose and also it doesn't decay even for larger frequencies.Let's first analyse the fourier transform of a guassian.
Fourier Transform of the given CT signal $e^{-t^2/2}$ is given by,

$$\int_{-\infty}^{\infty} e^{-t^2/2} e^{-jwt}\, dt = \frac{1}{\sqrt{2\pi}} e^{-\omega^2/2} \tag{6}$$

Clearly, It is real. So, The phase plot should be at zero for all ω and I normalised the Y for Guassian to avoid confusion. The fourier transform's properties still remain the same irrespective of the normalising factor, so it's valid. The only thing we have to take care of now is error.We have to play around with T and N. For that I referred this pdf

https://nicholasdwork.com/tutorials/approxDFT.pdf
We can now write CTFT as

$$X(\omega) = \int_{-T/2}^{T/2} x(t) e^{-jwt}\, dt \tag{7}$$

which as Reimann sum can be expressed using small steps $\Delta t = T/N$ as

$$X(\omega) = \frac{\Delta t}{2\pi} \sum_{n=-N/2}^{N/2-1} x(n\Delta t) e^{-j\omega n \Delta t}$$

Now on substituting $\Delta\omega = 2\pi/T$ and rearranging terms , we will get

$$X(k\Delta\omega) = \frac{\Delta t}{2\pi} \sum_{n=-N/2}^{N/2-1} x(n\Delta t) e^{-j\frac{2\pi}{N} nk}$$

$$X(k\Delta\omega) = \frac{\Delta t}{2\pi} DFT(x(n\Delta t) \tag{10}$$

So for improving accuracy, We have to make T larger as LHS becomes closer to actual function on doing that and also We have to increase N , so that Higher changes in the function are not ignored. Hence, We have to keep increasing T and N till we get $\epsilon_{max} < 10^{-3}$. If we start with T = 2π and N = 128, we would get an error of $\epsilon_{max} = 0.002597659954699405$, which is greater than 0.001. We can also see that phase is always zero.
So, we have to double T and N.

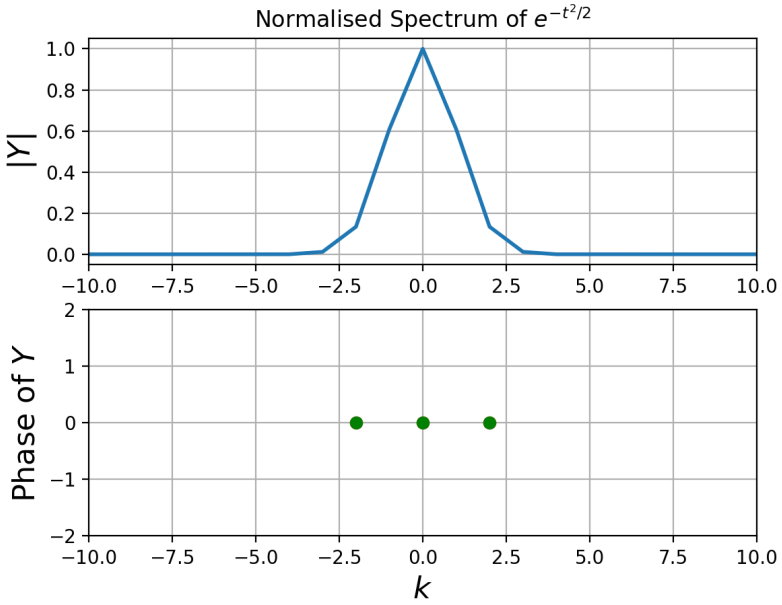


Figure 1: T = 2π , N = 128, $\epsilon_{max} = 0.002597659954699405$

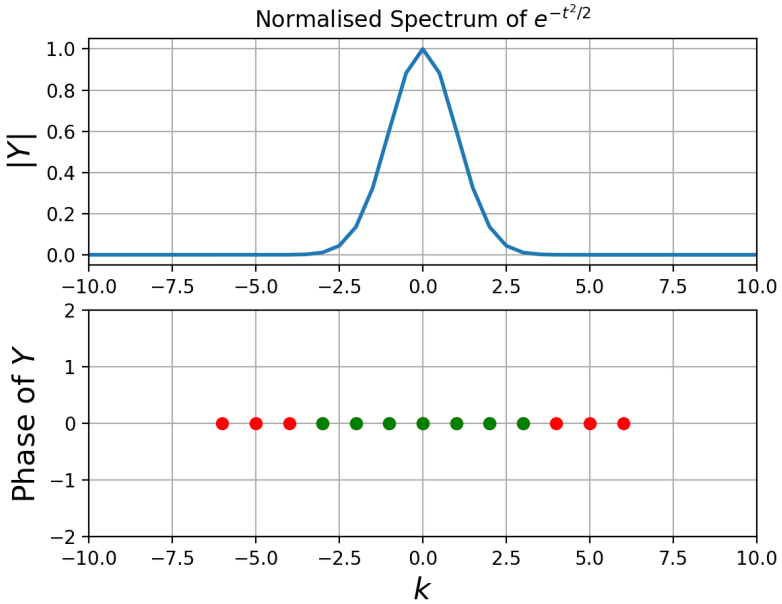


Figure 2: T = 4π , N = 256 , $\epsilon_{max} = 6.27631169258791 * 10^{-10}$

As we can see, Error has been decreased by a lot but the curve is not yet smooth.We can make it smoother by increasing T and N further

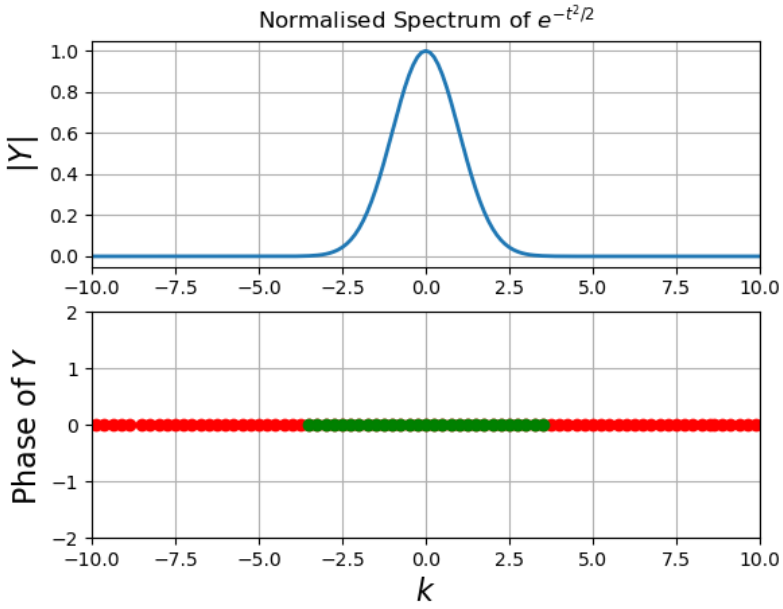


Figure 3: T = 16π , N = 1024 , $\epsilon_{max} = 3.8582503464623026 * 10^{-16}$

Conclusion

- NumPy's **fft** library has great tools to help us understand the fourier transform better. We can also understand how python works through the fourier transform and how can we improve code's accuracy using **fftshift**.
- DFT's of various sinusoids gave us an idea of where peaks lie and how the phase of peaks should vary. We also examined AM and FM signals to understand how N and T can affect the transform plot.
- I also learnt how to use time ranges and sampling improve the accuracy of fourier transform of a discretised continuous time signal.