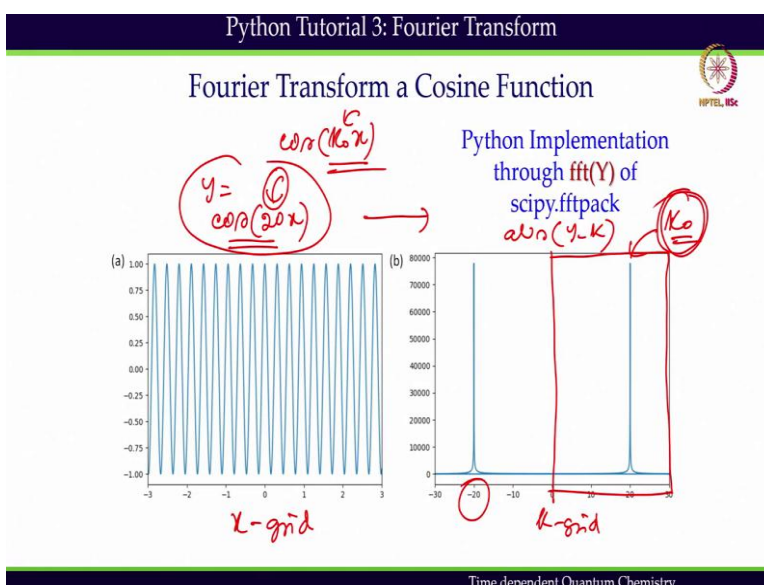


Time Dependent Quantum Chemistry
Professor Atanu Bhattacharya
Department of Inorganic and Physical Chemistry
Indian Institute of Science Bengaluru
Module 03 Lecture 23
Fourier Transform using fft

Welcome back to Python tutorials three, what we have presented is what is the very basic concept behind this Discrete Fourier Transform, which will be implemented in terms of Fast Fourier Transform algorithm in Python, and we have seen how to create the x-grid. So, there are two steps in Fourier transforming a function, first we have to create the k-grid and with the help of `fftfreq` functionality of `scipy.fftpack` and then we have to convert the function with the help of FFT.

(Refer Slide Time: 01:05)



So, what we have seen here is that the cos function. So, this is your x-grid and this is your k-grid. On this k-grid, we are representing it and we had $\cos(20x)$, this was the function and once we have Fourier transforming, this is we have defined as y and once we have Fourier transforming we are taking absolute value of this y_k , plotting the absolute value of y_k because y_k is Fourier transform function.

Once we Fourier transform it is always going to be complex function and to visualize a complex function it is convenient to use the absolute value of that complex function, so that we can get an idea what is going on. What do you see that once we have converted this cos function we know that $\cos(\omega_0 t)$, in the time domain we are very much familiar with it, this is the frequency and frequency is related to the Fourier domain.


So, similarly on the x -axis, this is the relative spatial frequency 20 and that 20 is now showing up here, this k_0 value this is actually $\cos(k_0x)$ and the spatial frequencies defined by this k_0 . So, this k_0 value is not now showing up. So, we have single frequency 20. But it has, it is showing up in the negative regime also.

This negative and positive part is just the consequence of the Fourier transform, it has nothing to do with the practical reality. So, in the practical purpose we can only take note down this positive part of the frequency and we can take a look at it the frequency components we are getting already k_0 , that should be, because \cos function is oscillating over the space. \cos function are oscillating with the spatial frequency 20, and that is showing up after the Fourier transform, we have only one frequency component which is 20 on the x -grid. So, this is the meaning of the Fourier transform of this function.

(Refer Slide Time: 03:30)

Python Tutorial 3: Fourier Transform

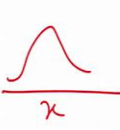
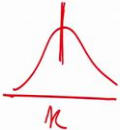
Fourier Transform a Gaussian Function



```

#Importing the Required Libraries
from scipy import arange, pi, cos, exp
from scipy.fftpack import fftfreq, fft
from matplotlib.pyplot import plot, show, xlim
#Creating the x-grid
xmin=-100
xmax=100
dx=0.001
x=arange(xmin, xmax, dx)
#Defining a Cosine Function in Position Space Grid
y=exp(-x**2)*cos(20*x)
#Creating Fourier Grid (or k-grid)
N=len(x)
k=2*pi*fftfreq(N, dx)
y_k=fft(y)
plot(k, abs(y_k))
xlim(-30, 30)
show()
          
```

Python Implementation through $\text{fft}(Y)$ of scipy.fftpack

Time dependent Quantum Chemistry



Fourier Transform a Gaussian Function

Wavepacket

Python Implementation
through $\text{fft}(Y)$ of
scipy.fftpack

```
#Importing the Required Libraries
from scipy import arange,pi,cos,exp
from scipy.fftpack import fftfreq,fft
from matplotlib.pyplot import plot,show,xlim
#Creating the x-grid
xmin=-100
xmax=100
dx=0.001
x=arange(xmin,xmax,dx)
#Defining a Cosine Function in Position Space Grid
y=exp(-x**2)*cos(20*x)
#Creating Fourier Grid (or k-grid)
N=len(x)
k=2*pi*fftfreq(N,dx)
y_k=fft(y)
plot(k,abs(y_k))
xlim(-30,30)
show()
```

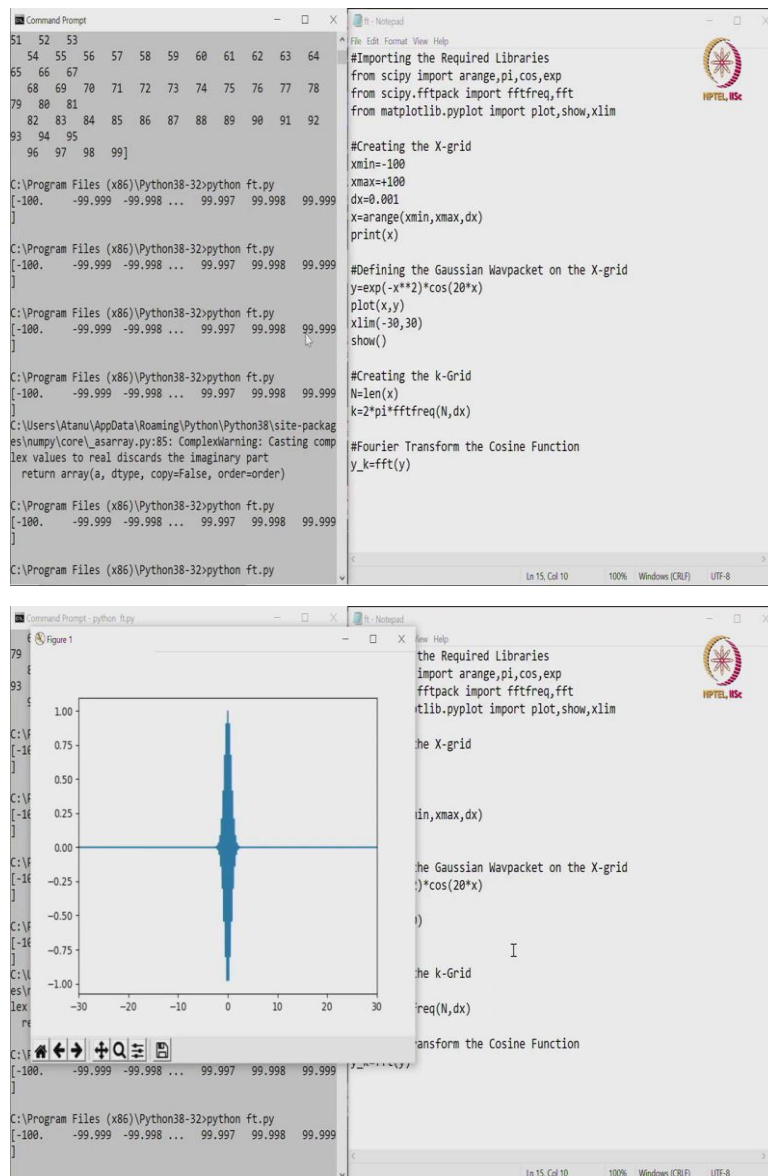
$$e^{-x^2} \cos(20x)$$



We will move on and we will check what is going on with the Gaussian function and we know that Fourier transform of a Gaussian is another Gaussian. So, if I have a Gaussian function in the x domain, then in the k domain I will get another Gaussian function. But then question is at what point the Gaussian function will be centered? That depends on, so this is not a simple Gaussian function it is actually a traveling Gaussian function.

So, it is a Gaussian pulse, let us say. So, it does not look like this, $e^{-x^2} \cos(20x)$, this is a fast-moving component. So, this is more like a Gaussian wave packet. So, we should make it a Gaussian wave packet. So, this is the fast component (cos function), this is the slow component (Gaussian function) which is considered to be an envelope function (see slide). So, how does it look like in the x domain, it should have an envelope function, but it is an oscillatory part. So, this is the x domain function look like this. And we need to find out what is the k domain function looks like. So, we will move to the laptop now.

(Refer Slide Time: 04:58)

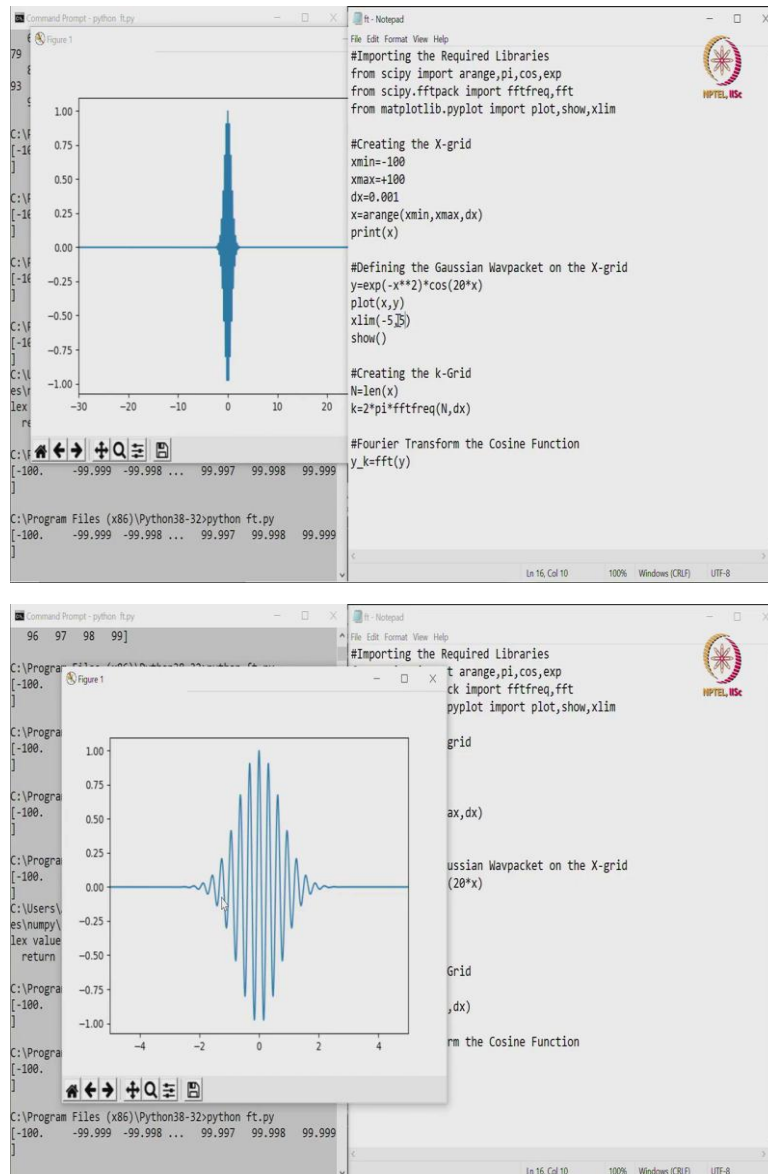


And in the laptop, in the program, we will make some changes, we keep the x-grid to be the same, we do not make changes here, depending on the convenience one can change it, but for the time being we will not change it. We will define, when you will define the Gaussian wave packet on the x-grid, we have to multiply it by exponential function and this exponential function is going to be x^2 , that is the function.

Now, exponential function is not again available with a built-in library in Python. So, we have to import it again from the SciPy and then the k-grid formation is remains to be the same, it does not have any change, then the Fourier transform of the functions remains to be the same and plotting part is remains the same.

But before we plot that, we will try to plot the function itself first. To find out what kind of functions we are considering here. So, plot x versus y and the limit, we have to select, I will check whether this limit will work or not. So, let us run the program. If we run the program, we see that there is a pulse localized it, is not a pulse. So, wave packet which is localized, but the limit which we are selecting -30 to +30, it is not convenient to look at the inspect that wave packet very clearly.

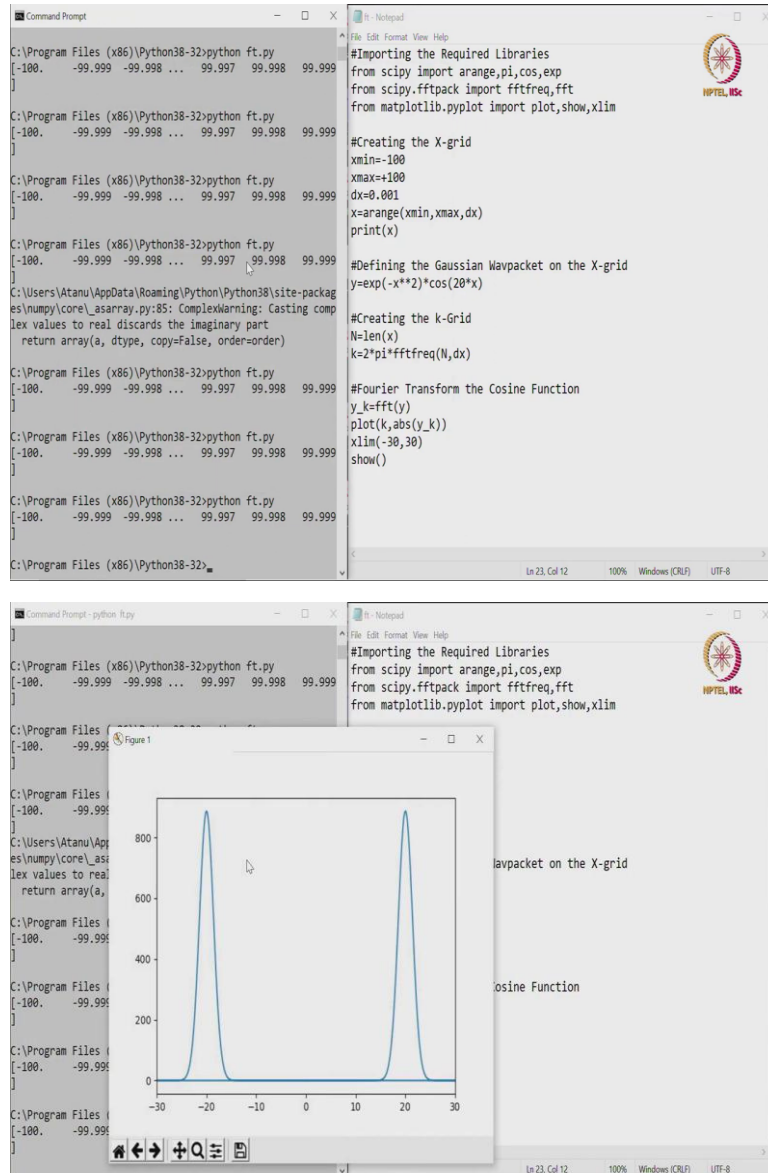
(Refer Slide Time: 06:55)



So, we will set the limit to be -5 to +5. We run the program and we see that it is centered at 0-position and x-position. And it is like a wave packet, the wave packet has been at the position. So, at this time, if the wave packet looks like this, what are the frequency

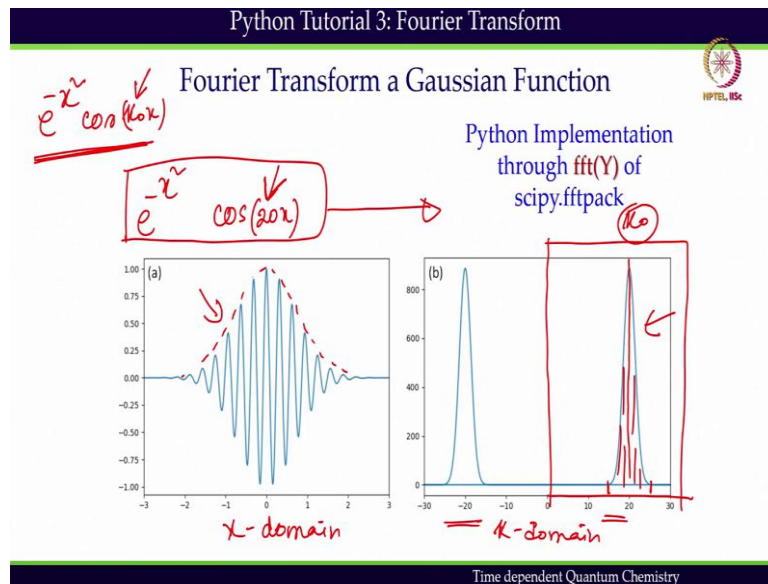
components it is carrying, that is exactly the information we will get from the Fourier transform. So, this entire function will be Fourier transform now.

(Refer Slide Time: 07:34)



And if we do the Fourier transform, we will now plot the Fourier transform function, so that we can get an idea of what we are having here. So, k then absolute values of y_k and limit will change again, because this limit will help us to see everything what we see is that if we run the program, we see that again there are negative and positive part of the spectrum. We go back to the slide now.

(Refer Slide Time: 08:09)



What we see here is that this is the x domain which is representing an wave packet, this is the Gaussian envelope, this is the Gaussian envelope and the fast wearing component is $\cos(20x)$. So, this is the let us say wave function I have and this wave function, if we convert it to the k domain, so this is the k domain.


On the K domain, we see that there is always negative and positive part because the nature of the Fourier transform makes that but for practical purpose, we can just take the positive part of the Fourier transform spectrum. What we see is that the center it is centered at 20 and that should be because we know that from the wave packet discussion if I have an wave packet represent wave Gaussian packet represented by $e^{-x^2} \cos(k_0 x)$

It means that this sent the average component average the spatial frequency component is going to be k_0 . So, the frequency components is distributed around this k_0 . So, I have this k_0 and then around this k_0 all this frequency components, the spatial frequency components are available.

So, that is the way we have understood the Gaussian wave packet before and that is representing here. So, one thing is quite clear from this demonstration that Gaussian, Fourier transform of a Gaussian is another Gaussian and it is centered at k_0 , which is the average frequency component, the spatial frequency component for the wave packet.

(Refer Slide Time: 10:37)

Python Tutorial 3: Fourier Transform



Summary

scipy.fftpack submodule

$$k = 2 * \pi * \text{fftfreq}(N, dx)$$

Fourier Transform `fft(Y)`

Time dependent Quantum Chemistry

So, with this we have come to the end of this Python tutorial, what we have learned from this tutorial is that there are two simple functionalities already given in `scipy.fftpack` submodule, one functionality will help us construct the k-grid and other functionality will help us Fourier transform the discretized wave function from space domain to the Fourier domain.

In the `fftfreq` I need an input, the input is the window length, which is the number of elements in the present in the x-grid and delta x (or dx) is the spacing in the x-grid. And when you are doing Fourier transform the function the discretized function, this function name needs to be given. So, we this we will conclude this tutorial. We will meet again in the next module and tutorial.