Electronic Properties of the Materials: Computational Approach
Prof. Somnath Bhowmick
Department of Materials and Engineering
Indian Institute of Technology - Kanpur

Lecture: 05
Visualization Using Python (Part 1)

Hello friends. So, far we have learned some basic topics in quantum mechanics. In this lecture we are going to visualize some of the most important things discussed. So, far using python in case you have never done any python programming before I shall provide some basic training however since this is not a course on python I shall discuss on the bare minimum required for our purpose.

(Video Starts: 00:43)

Throughout the course Google collaboratory is going to be used for writing and running the python codes. You do not need to install any additional software all you need is a Google account and you can write and run your codes in your web browser we start our discussion on list and array because they are needed to deal with vectors and matrices. First we import the necessary libraries.

We need numpy library for mathematical operations and matplotlib dot pi plot for plotting. Let us create a vector with three components v is equals to 1 2 and 3. Now we can calculate the length of the vector we have to take the first component and calculate its square. So, this is how you calculate the square and similarly for the second and third component you calculate the square and then finally you add all of them and you calculate the square root.

And for calculating the square root we need numpy library. Note that since we are using the function square root from numpy library we have to use this word np to call it. Any function from numpy library will be called with this word np. We can do several operations with list like finding the maximum value finding the minimum value finding the number of elements in the list and finding the sum of all the elements in the list etc.

Now these print statements will print several messages as required by us. Let us run the code and see the output. So, to run the code you just have to click this button which shows the arrow. Now we have the output the first line says that this is a list and this is an outcome of this line

this print command. And the second line just prints the vector and this is an outcome of this print command. The third line prints the magnitude of the vector and this is an outcome of this print command.

The fourth line paints the maximum of all the elements and that is an outcome of this print command. The fifth line prints the minimum value and that is an outcome of this print command. Then we are printing uh the total number of elements and that is outcome of this command and finally sum of all elements and that is outcome of this print command. For mathematical operations arrays are more convenient than list.

However arrays can store only one type of data this is why when we create an array we have to mention the data type for example it can be an integer or it can be a float. In this example we create a 1d array containing three elements and the data type is integer. This is a vector with three elements. Next we create a two cross two matrix we cannot do this using a list and we have to use array for this purpose.

We are using numpy library for doing this. In the previous case also we used numpy library for doing this. Next we can create a two class two matrix from two different lists for example first we define two lists r1 and r2 and then again with the help of a function in the numpy library we can convert these two arrays to two different rows in the matrix let us run the code and see the output.

So, the first line says that this is an array and this is an outcome of this print command the next line prints the elements of the array and all of them are 0 because that is what we have defined here np dot 0's that means it will create a vector where all the elements are 0's. So, the next line is just printing all the elements of the array and this is outcome of this print command. Next uh the third line again it prints the class as array and this is outcome of this print command.

And then ah we are just printing the elements of the matrix and this is what we have done np dot once. So, that means it will create a 2 cos 2 matrix with all the elements equal to 1 and that is what it is printing and this is an outcome of this print command. And then we print another matrix and this is an outcome of this print command. Just now we saw an example of creating an array from multiple lists.

We can also create multiple lists from an array. For example first we create an array like this, this is a 3 cross 3 array and then we can convert each row of this matrix to a list using this command. So, let us run and see the output. Now you see that it has converted the first row of the matrix to a list we can do it for the second row as well for to do that we have to make these changes run the code again and then you see that the second row has been converted to a list.

Next we learn to plot a function this is going to be one of the most important examples and I shall demonstrate it very carefully. Let us assume I want to plot f x is equals to sine x from x equal to 0 to x equal to 2 pi. First we create an equally spaced grid having 100 points using this command. And we are using utilities from the numpy library for doing this. So, this will create a grid of 100 points between 0 to 2 pi.

Note that we also take the value of pi from numpy library. Then we define the functions which needs to be plotted. So, for example we define y is equals to cos x and y is equals to sine x and see that we are using numpy even for defining the functions. Then we plot using utilities from matplotlib dot pi plot and this is the command where we are plotting the functions. And then these are the rest of the things like for marking the x level marking the y level and then you can also like level the plots and then finally you can plot it.

So, let us run the code and see the output. Let us do this step by step first we define the range it is from 0 to 2 pi and we want 100 read points between 0 and 2 pi. Next we define the functions to be plotted cosine of x and then sine of x next we plot the function. So, x versus y and we can also label it this is cos of x. Now let us run the code, finally we have enough skills to plot something we have studied in the course.

We start with probability density of particle in a box. The eigen function is given by psi n of x is equals to square root of 2 by L sine n pi x by L. Now we take L equal to pi such that the eigen function is psi n of x is equals to square root of 2 y pi sine nx. We want to plot mod psi n square for any given value of n. In this section the user has to input the value of n and then we create equally spaced grid point.

Now in this part we define the functions and then in this part we plot the functions. So, let us do it portion of the code is already written let me enter one of the eigen functions. So, this is square root of 2 divided by pi sine n1 x and the eigen function for n2 is already given here.

Now let us run the code and see the output. So, it is asking us to enter the index of the first wave function. So, let us enter some small value.

Then it is asking for the index of the second in function let us enter a bigger value and finally you can see the plot of the probability density for n1 equal to 2 and for n2 equal to 10. Let us calculate the position uncertainty for a given eigen function. To calculate the position noise uncertainty first we have to calculate the expectation value of x. Next we have to calculate the expectation value of x square and from that we can calculate the position uncertainty.

The expectation value of x is given by integral 0 to pi x psi n square x dx the expectation value of x square is given by integral 0 to pi x square psi n square x dx. Remember that we are taking the box length to be equal to pi. Now we are going to evaluate these integrals by using some numerical technique. We are going to use trapezoidal rule to evaluate the integrals numerically and we are going to use sci-fi library for that purpose.

So, first the user has to give the index of the eigen function then equally space grid is created and then we define the eigen function here and this is same as this right we are using L equal to pi. Now in this step we evaluate the first integral x psi n square x dx. In the next step we calculate the second integral x square psi n square dates and then in this step we calculate this value and then the code writes the calculated value.

Let us run the code. So, we have to enter the index of the eigen function. So, let us enter 1 and then you see the position uncertainties this. Let us try one more value and let us try 2 and position uncertainty is given by this. Before proceeding further let us enhance our coding skill a little bit we have to learn about user defined function for loop and if. We are going to write a code to verify whether some function has a root in some given range.

Let us understand the algorithm first. If we plot the function f x is equals to sine x by x this is how it looks like. A root exists at every point where the function is equal to 0. Note that fx changes psi from positive to negative or negative to positive as it crosses the point where it has a root thus if we take the value of the function just before and just after the root and multiply I should get a negative number.

For example if I take the value of the function at this point and if I take the value of the function at this point and multiply these two values I should get a negative number. This is the algorithm for finding the root. Now let us understand the code. The range needs to be defined here then we define a function such that it returns the value sine x by x for a given value of x then we create equally spaced grid points and then we run a for loop to check whether the product of two consecutive values of the function is less than 0 or not.

So, here we take two consecutive values of the function f at point I and point I plus 1 and then check whether the product that is calculated here whether that is less than 0 or not if it is less than 0 then we know that there is a root. So, at that point the loop stops and the value of the root is printed. So, let us run the code. So, here at this point we have to define the function sine x by x the function is to return this value.

So, once we have defined the function then we create the equally spaced grid points between x1 and x2 and then we run for loop and then we have to call the user defined function that we have defined here f of x. So, this is how you call the function f and then you give the value of x and what we want to check here whether the product of two consecutive values of the function is less than zero or not. So, we just do this right.

So, this is the value of the function at a point x of i and this is the value of the function at a point x of i plus 1 and we have to check whether the product is less than 0 or not. And for that we use the if statement and if the condition is true right for example if the product is less than 0 in that case I just print the value of the root and then the break statement tells to stop the for loop once we have found the correct. So, let us now run the code.

Before we do that let me just show you the plot of uh this sine x by x this is how it looks like. So, for example if we check a point between like 2 and 5 then we surely know that there is a root. So, let us just check that. The lower bound is 2 the upper bound is 5 and the root is at 3.14. Let us check one more point and again we see that if we check between 5 and 7 then we surely have a root.

So, let us do that lower bound is 5 and the upper bound is 7 and the root is at this point. Let us now calculate the momentum on uncertainty for eigen functions of particle in a box. We know that this is the momentum operator and expectation value of p is given by this integral

essentially we have to calculate the integral of psi of x times psi dash of x in the limit 0 to pi because the length of the box is taken to be equal to pi.

Psi of x is given by sine of nx times square root of 2 by pi and similarly psi n dash of x is cos of nx times the appropriate constant. Now this is the square of the momentum operator and to calculate the expectation value of p square we have to evaluate. This integral this is an integral of psi x times psi 2 dash of x dx from 0 to pi and psi 2 dash of x is sine of nx times some constant. Once we calculate the expectation value of e square and expectation value of p then we can calculate the value of momentum uncertainty.

Now let us look at the code. So, in this portion we import the necessary libraries next we have some user defined function which calculates the value of delta of t. So, in this step we define the wave function psi this is sine of nx times some appropriate constant this step we calculate psi dash and in the third step we calculate psi 2 dash. And then here we calculate the first integral psi times psi dashed dx.

And in this step we calculate the second integral psi times psi 2 dash dx and finally we calculate the value of the momentum uncertainty and this value is returned. Now in this code we are calculating the expectation value for a range of values of n in this example we vary n and find out the expectation value. For example we vary n from 1 to 100 where n equal to 1 is the ground state and then we also calculate the x the momentum uncertainty for excited steps up to a state n equal to 100.

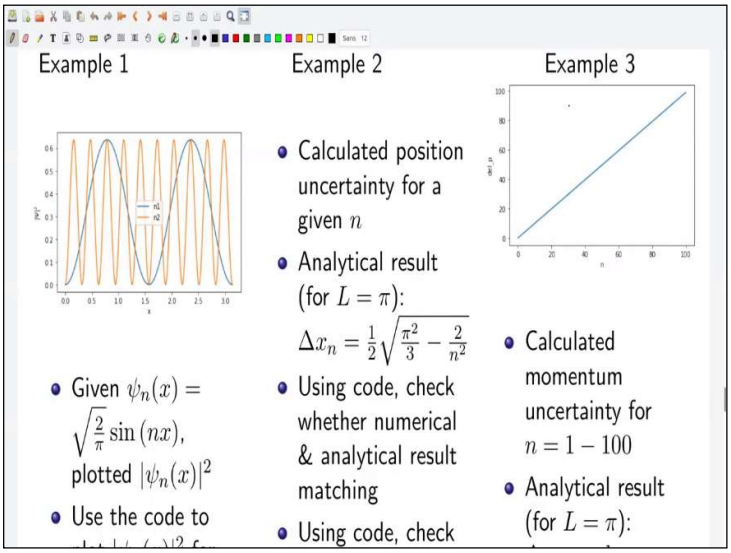
So, here we run a for loop and we get the value of momentum uncertainty for various different values of n. So, here we call the function f of i and then this is done for n equal to 1 to n equal to 100 and then we store the values of the momentum uncertainty in this array and after that we plot the value of momentum uncertainty as a function of n. Let us see the code. So, this is where the libraries are important.

And then this is the user defined function that will calculate the value of momentum uncertainty for a given value of n and this is where we run a for loop and get the value of momentum uncertainty for various different values of n and this is where the values are plotted as a function of n. So, let us run the code and see the output. Thus we got the plot of momentum uncertainty as a function of n.

And we see that the momentum uncertainty is increasing linearly with the value of n. You should match the result with whatever we have learned in the theory part.

(Video Ends: 24:57)

(Refer Slide Time: 24:58)



Let me quickly summarize what we have done so, far and also tell you what you are supposed to do with the codes shown in the lecture. First we have plotted the probability density for eigen functions in particle in a box. You can use the code to plot mod psi nx square for any given n next we calculated position uncertainty for a given n we derived the analytical result during the theory lectures and using the code you can check whether the numerical and analytical result is matching or not.

Also you can use the code to check for the asymptotic value of delta x what do I mean by asymptotic value in the equation of delta x n if you put n equal to infinity that will give you the asymptotic value. Next we calculated the momentum uncertainty starting from the ground state to some excited state with n equal to 100. Again you can match with the analytical result derived in the theory lecture and that turns out to be delta p n is equals to n h bar.

And you can match this with the plot we got numerically and then you see that it perfectly matches with the analytical value.