

Time Dependent Quantum Chemistry
Professor. Atanu Bhattacharya
Department of Inorganic and Physical Chemistry
Indian Institute of Science, Bengaluru
Module 04, Lecture 30
Python Tutorial 4 (Eigenvalue and Eigenfunction)

Welcome to python tutorial 4 of the course Time Dependent Quantum Chemistry. In this tutorial, Python tutorial, we will learn how to represent a matrix in Python programming and then subsequently how to calculate eigenvalues and eigenvector corresponding to that matrix.

With the help of this tutorial, we will be able to learn how to numerically calculate stationary states of quantum systems, stationary state wave function and stationary state energies for a quantum system. Technically, we are actually exploring the, we will be exploring the eigenvectors and eigenvalues associated with a particular Hamiltonian. So, we are solving Time Independent Schrodinger Equation.

(Refer Slide Time: 01:25)

Time Dependent Quantum Chemistry

Python Tutorial 4: Constructing Matrix and Finding its Eigenvalues and Eigenvectors

Quantum Dynamics TDSE

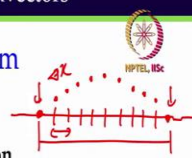
Computational Chemistry Quantum Chemistry

Python

Atanu Bhattacharya
Assistant Professor, Department of Inorganic and Physical Chemistry,
Indian Institute of Science, Bangalore, e-mail: atanub@isc.ac.in

Mathematical Language of Quantum Mechanics is Linear Algebra

Using Grid Representation of the Wavefunction and using Finite Difference Method, within the boundary condition.



$$\frac{\hbar^2}{2m} \frac{d^2}{dx^2} \approx \frac{\hbar^2}{2m} \frac{1}{\Delta x^2} \begin{pmatrix} -2 & 1 & 0 & 0 & 0 & \dots & \dots & \dots \\ 1 & -2 & 1 & 0 & 0 & \dots & \dots & \dots \\ 0 & 1 & -2 & 1 & 0 & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & 0 & \dots & 0 & 1 & -2 \end{pmatrix}_{(N-2) \times (N-2)}$$

$\hat{H} = (\hat{T} + \hat{V})$

$$V(x) = \begin{pmatrix} V_1 & 0 & 0 & \dots \\ 0 & V_2 & 0 & \dots \\ 0 & 0 & V_3 & \dots \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \end{pmatrix}_{N-2 \times (N-2)}$$

So, we will take a look at it, what is the procedure to represent our matrix in Python programming. In module 4 of this course, we have already realized that mathematical language of quantum mechanics is linear algebra that we have already realized. And in addition to that, in the same module, where we have shown the connection between the quantum mechanics and linear algebra, we have presented the grid representation of the wave function.

So, we said that the entire space can be divided into equally spaced grid and the wave function will be represented as a discretized function on this grid like this. So, within this grid presentation and using finite difference method and within the boundary condition, we have to use the boundary condition, boundary condition it means that the wavefunction value would be 0 at these boundaries, boundaries of the grid, so where the grid will be terminated.

Theoretically, the wave function should be extending from minus infinity to plus infinity, but we cannot take such an infinite boundary grid in computation. So, we make it a finite boundary assuming that at the boundary the wave function takes 0 values. So, we use this boundary condition then finite difference method for the derivative and the discretized wave function, then what we get is that this is the kinetic energy operator of the, kinetic energy part of the Hamiltonian.

$$\frac{\hbar^2}{8\pi^2 m} \frac{d^2}{dx^2} = \frac{\hbar^2}{8\pi^2 m \Delta x^2} \begin{bmatrix} -2 & 1 & 0 & 0 & 0 & \cdot & \cdot & \cdot \\ 1 & -2 & 1 & 0 & 0 & \cdot & \cdot & \cdot \\ 0 & 1 & -2 & 1 & 0 & 0 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & 0 & 1 & -2 \end{bmatrix}$$

That kinetic energy part of the Hamiltonian can be expressed in terms of a matrix form like this, this is a tridiagonal matrix that is we have already seen it. And this Δx is the difference between the grid points, this is the adjacent grid points and \hbar cut m comes spontaneously directly from the definition of the kinetic energy operator. On the other hand, the potential energy part what we have seen that it construct the, it can also be represented in the matrix form and it gives me the diagonal matrix.

$$V(x) = \begin{bmatrix} V_1 & 0 & 0 & \cdot & \cdot & & & \\ 0 & V_2 & 0 & \cdot & \cdot & & & \\ 0 & 0 & V_3 & \cdot & \cdot & & & \\ \cdot & \cdot & \cdot & \cdot & \cdot & & & \\ \cdot & \cdot & \cdot & \cdot & \cdot & & & \\ & & & & & \cdot & \cdot & \\ & & & & & \cdot & V_{N-2} & \end{bmatrix}$$

So, potential energy is a diagonal matrix, kinetic energy is a tridiagonal matrix. And if we add them together, we get the total Hamiltonian, the Hamiltonian is nothing but the potential part and the kinetic energy part that we have seen. So, if I have the Hamiltonian and that can be represented in terms of matrix form, then we can immediately get the eigenvalue and eigenvector from that matrix.

(Refer Slide Time: 04:52)

Mathematical Language of Quantum Mechanics is Linear Algebra

Eigenvalues and eigenvectors of a quantum system can be easily obtained by diagonalizing the Hamiltonian matrix

$$U^{-1}AU = \lambda$$

Target of Present Tutorial: Numerically Obtain Eigenvalues and Eigenvectors using scipy.linalg submodule

The diagram includes a handwritten note 'Array' pointing to matrix A, and a graph of energy levels labeled H with arrows indicating transitions.

So, we have also realized in the same module, module 4 of this course, where we have shown the connection between linear algebra and quantum mechanics, we have also realized that the eigenvalues and eigenvectors of a quantum system can be easily obtained by diagonalizing the Hamiltonian matrix.

So, I have shown how to get the Hamiltonian matrix. Hamiltonian matrix will be obtained from adding these two matrices. Once we get this total Hamiltonian matrix in the grid presentation, we are following the grid presentation which is nothing but pseudo spectral representation of the wave function. And so, once we get the Hamiltonian matrix, then we can diagonalize the Hamiltonian matrix to obtain the eigenvalue and eigenvectors.

And eigenvalues and eigenvectors of the spectrum of the quantum system. So, if it is a one-dimensional box problem, then I have this, this is ground state, then excited state and then all the states can be explored and each state are actually corresponding to eigenvector and eigenvalues of the particular Hamiltonian representing the quantum system.

So, we get the entire spectrum or we get the stationary states of the quantum system. That is why diagonalization process is very important in quantum mechanics. And how do you diagonalize it? In module 4, we have shown that diagonalization of a matrix, let us say I have a matrix A. So, in general operator is shown like this way with a hat on it and matrix is with the double line.

So, if I have a matrix A like Hamiltonian matrix, then I can perform a linear transformation like this

$$U^{-1}AU = \lambda$$

with the help of this unitary operator, this unitary operator has to be found. And if we perform this, then what will happen the role of this unitary operator is that it transforms the matrix into a new diagonal form, this is a diagonal form of the matrix in which the diagonal elements represent the eigenvalues of the matrix A and column of the matrix U represents the eigenvector of the matrix A.

So, all eigenvectors will be clubbed inside this unitary matrix U and corresponding eigenvalues or eigen state energies will be given in this diagonal matrix which is λ . So, all information will be just clubbed together arranged together in these two matrices and if we perform this unitary matrix, what is the origin of this unitary matrix we have studied already.

So, if we perform this unitary transformation linear transformation then we get this diagonal form or diagonalize the matrix and we get eigenvalues and eigenvectors. As an example, we have shown that we took an matrix A like this

$$A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

this was the let us say as an example, we took it, this was the matrix and 2 * 2 matrix.

And for this matrix in order to diagonalize this matrix we actually got this matrix, U matrix, unitary matrix as

$$U = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$$

So, this was the unitary matrix. So, if we diagonalize it immediately

$$\lambda = \begin{bmatrix} 1 & 0 \\ 0 & 3 \end{bmatrix}$$

So, one of the eigenvalues would be 1, another eigenvalue up would be 3 and corresponding eigenvectors this column and another eigenvector will be this column.

So, this is the way we get that. And we have analytically shown how to get that but in this tutorial, we will check it numerically using Python programming. In general, the data structure which is used to present a mathematical matrix in computer programming is called an array. So, in order to represent this matrix in computer programming, we need to use this matrix if we want to represent then we need to use an array we need to put them in an array.

So, this is the structure, we will use array structure to represent a matrix. An array can be n dimensional, but 2-dimensional array is called a matrix and a 1-dimensional array called a vector. So, these are, if we represent each column, then this is going to be a vector each column. And two-dimensional array which is $(n \times n)$ matrix we are presenting here, this is 2 by 2 matrix. So, it is two-dimensional array is called the matrix.

Python does not have any intrinsic or building functionality to deal with arrays. So, Python cannot be directly used, we have what we have to use we have to use its sub module that is called `scipy`, which is scientific, which is prepared. which is built to perform scientific computation with Python. So, we have to use that sub module to deal with the array and different numerical linear algebra techniques or routines we have to use from the functionalities of `scipy`.

So, we have to write down to represent this matrix in Python, we have to import `scipy` and then represent in terms of array and remaining eigenvalue problem solving for the to find out the eigenvalues and eigenvectors, we have to use the `scipy.linear algebra` sub module, which is expressed as `scipy.linalg`. So, this sub module will be used to find out them eigenvalues and eigenvectors.

All features of `scipy linear algebra` sub module one can actually look up the features in the `scipy` documentation. And all the routines are available in the `scipy linear algebra` sub module. This `scipy dot linear algebra` sub module is optimized linear algebra routine, then NumPy linear algebra sub module and that is why we are not using NumPy module here we will just keep using this `scipy` sub module of Python.

And one more information can be useful to note down here is that the linear algebra sub module of `scipy`, it has a direct interface with the Fortran `lapack` library, which is very efficient and faster and most optimized linear algebra package developed in Fortran for several decades. So, that has an interface and that is why it is more efficient. So, we will come, we will use this linear algebra sub module of `scipy` to represent the matrix and to perform this, to calculate this eigenvalue and eigenvectors.

(Refer Slide Time: 13:45)

Python Tutorial 4: Matrix, Eigenvalues and Eigenvectors

Steps to Find Eigenvalues and Eigenvectors

$\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} = A$

First, create an (NxN) square null matrix,

then replace (reassign) the elements by desired value to create the final matrix

In the end, find out eigenvalues and eigenvectors using eig functionality of scipy.linalg submodule

$\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}_{2 \times 2}$

$E, V = \text{eig}(A)$

Time dependent Quantum Chemistry

So, how to find out eigenvalues and eigenvectors? First, we have to represent them matrix. So, there are several approaches available currently in scipy linear algebra sub module to find out eigenvalues and eigenvectors of a square matrix. However, here we will follow these steps. There are many other ways one can do that, but we will consistently use the step 2 to represent matrix and then find out the eigenvalue eigenvectors of that matrix.

So, what are the steps we are going to follow? First, we have to create an n-by-n square null matrix, this is something which you should remember. First, we will create the null matrix. Null matrix is the matrix where we have all elements 0, so this is I am writing a 3 by 3 matrix where all elements are 0. So, we will prepare the null matrix first, so we will prepare the null array first of the same dimension 3 by 3,

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

then we will replace or reassign the elements by desired value to create the final matrix.

So, if I have to create this matrix, let us say

$$\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

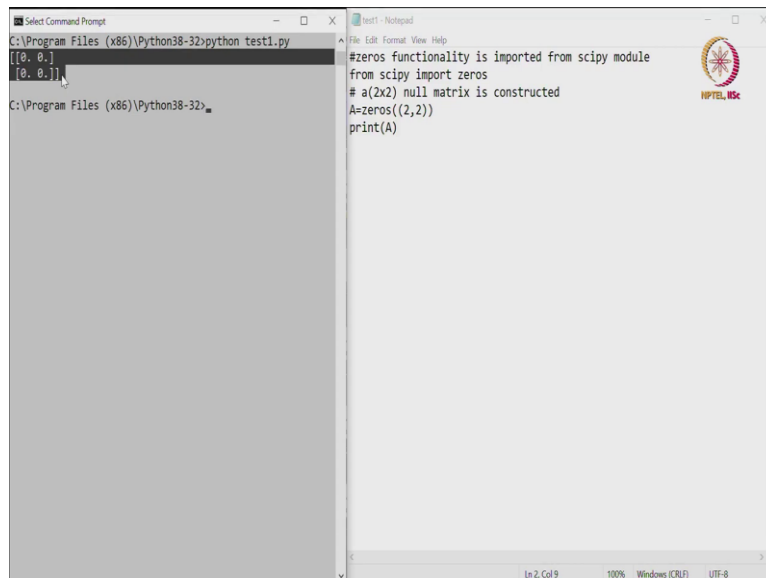
This is the matrix we have to create. So, then first we have to do 2*2 null matrix, we will prepare the null matrix first 2*2. And then we will replace each element value. So, we will

instruct the program to replace the individual element like this way after creating the null matrix, 1 and then this is going to be 2.

And then once we have constructed the desired matrix, we can find out the eigenvalues and eigenvectors by simply this functionality, eig functionality. What we do in eig functionality? We can just simply write down eig within bracket, if I name this matrix to be A, the array to be A, then I will get this. So, e comma v, this is the energy and the eigenvector, eigenvalues and eigenvectors will be given within this construct, it is a very simple way to do.

In the background, linear algebra sub package or sub module of scipy will be performing all the numerical task for me, and then it will give me finally, these values and the vectors using this functionality, is very simple to do. So, we will take a look at it how to represent it.

(Refer Slide Time: 16:43)



```
Select Command Prompt
C:\Program Files (x86)\Python38-32>python test1.py
[[0. 0.]
 [0. 0.]]
C:\Program Files (x86)\Python38-32>

test1 - Notepad
#zeros functionality is imported from scipy module
from scipy import zeros
# a(2x2) null matrix is constructed
A=zeros((2,2))
print(A)
```

Python Tutorial 4: Matrix, Eigenvalues and Eigenvectors

Steps to Find Eigenvalues and Eigenvectors

Example 1: Find eigenvalues and eigenvectors of $\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$

(a) Step 1: Construct the Null Matrix of the Same Dimension

$\text{zeros}(N,N, \text{dtype})$

```
#Import zeros functionality from scipy
from scipy import zeros
#Construct (2x2) null matrix
A=zeros((2,2)) ← ✓
#print the matrix
print(A)
```

```
[[0. 0.]
 [0. 0.]]
```

$\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$

We will go back to the laptop right now. And we will try to construct the matrix. So, first, as I have mentioned, I have to first construct the null matrix. And to construct the null matrix, what we have to do is that from scipy, we will import one functionality called zeros. So, zeros will be producing the array with every elements will be 0.

And then I will name A as zeros within bracket. Now, I have to define the matrix dimension, I want to perform matrix 2*2. So, 2 comma 2 is my matrix dimension. So, what it does, it immediately creates, so if I go back to the slide, so it constructs 2*2 null matrix first, so 2*2 matrix, it means that this zeros will give me 0 0 0 0. That is exactly what it is going to create.

So, we will prove that it is what is going to create, I will go back to the laptop and I will now print this A, so I have created an all matrix and I will write down some instructions here which will be helpful with a hash character, a 2 *2 null matrix is constructed. Because the final matrix which I am going to create is going to be 2 by 2. And here I will write down zeros functionality is imported from scipy module.

And I saved that with the name test 1 dot py. It is already saved. And now I would like to run the program. If I run the program, what we see here is that this is the way Python will show the the matrix or the array. In python programming, we will call it array, but when in mathematical language it is the matrix. So, in computer language we do not call it matrix. So, what we are seeing here is that we can create, if we go back to the slide right now, we can create this 0 0 0 0 matrix very easily with the help of these zeros functionality.

In the zeros functionality, remember that we are using a comma here to separate the, to show the dimension and we have double bracket here one bracket for the zeros and another bracket for to show the dimension of the matrix. So, these are the two things we should remember, we should not get confused by the notation.

An array data structure can be created using this built-in functionality of scipy and Python's modules scipy has several functionalities to define an array of any dimension. For an example, these zeros, these zeros more specifically, the actual functionality is like the zeros, then within bracket N comma N, then comma dtype that is called data type. This is the actual functionality where we have not used this, this option datatype option because this datatype option, d type option is optional.

And that is why we would not use it. But if we use that, then it is actually defining what kind of data type I need is a floating data type or integer data type, we can define it, but I am not

using it. And that is why because it is optional functionality, I am not using it here. Next, what we will do now are according to the steps we have given once we have created the null matrix of the same dimension, so this is the matrix we would like to represent and then find out its eigenvalues and eigenvectors.

(Refer Slide Time: 22:08)

Python Tutorial 4: Matrix, Eigenvalues and Eigenvectors

Steps to Find Eigenvalues and Eigenvectors

Example 1: Find eigenvalues and eigenvectors of $\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$ $0, 1, 2, \dots$

(b) Step 2: Replace Elements to Construct the Target Matrix

```
#Import zeros functionality from scipy
from scipy import zeros
#Construct (2x2) null matrix ✓
→ A=zeros(2,2)
A[0,0]=2 #reassign the element in the 1st row and 1st column
A[0,1]=1 #reassign the element in the 1st row and 2nd column
A[1,0]=1 #reassign the element in the 2nd row and 1st column
A[1,1]=2 #reassign the element in the 2nd row and 2nd column
print(A)

[[2. 1.]
 [1. 2.]]
```

$A = (N \times N)$ array
 $A[i,j] \rightarrow$ element of i th row and j th column
 $A[0,0] = 1$ st row 1st column
 $A[1,2] = 2$ nd row 3rd column

Time dependent Quantum Chemistry

Next step would be to replace the elements to construct the target matrix. So, we have constructed the null matrix first and then we have to replace it. For the replacement when A represents, if A represents a $N \times N$ array. If A representing a $N \times N$ array, then A this construct A i comma j , this construct represents the element of i th row and j th column of that array.

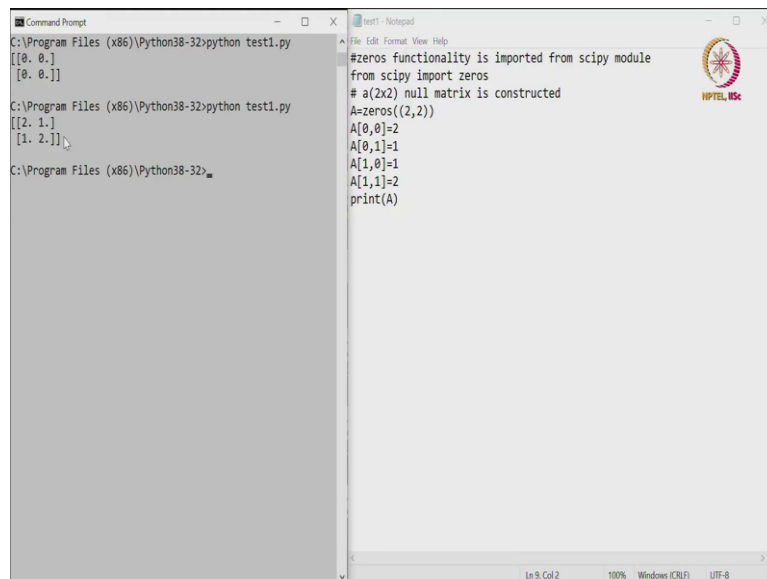
So, if it is so, then one can take some examples, let us say and these are indices and we know that indices will start from 0, it will start from 0 then 1 then 2 like this way it will go. So, A $[0, 0]$ this construct will give me, it will return the element in the first row and first column which means first row first column element is this one. So, that one I am going to assign then if I write down A $[1, 2]$, it means it is first row and second column.

So, this is the first row and second column. So, this is going to be this one, and so on. So, that is the way we are going to assign. So, if we look at this is very simple to do. And similar assignment we have used for them list indexing, we have already seen that the python's built-in list indexing is similar kind. But here we have to just show the entire dimension for this because it is a two-dimensional so you have to give each element, each index corresponding to that particular element.

So, what we are doing here is that first we have constructed the zero matrix null matrix and then individual element are reassigned now. So, A $[0, 0]$ is corresponding to the first row first column, A $[0, 1]$ is the first row and sorry this is A $[1, 2]$, it means it is going to be second row and third column, $[0,0]$ is the first row first column, $[1, 2]$ is the second row and third column.

So, similarly, A [0, 1] now it is going to be first row, second column, so that is going to be first row and second column, column is here so it is going to be here, this one. Then [1, 0] second row, first column, this one. And A[1, 1] is going to be second row second column this value. And that is the way we have assigned, we are assigning, reassigning it. And if we print, it will be able to see what we have assigned. So, let us go back to the laptop and find out how to do that.

(Refer Slide Time: 26:14)



```
Command Prompt
C:\Program Files (x86)\Python38-32>python test1.py
[[0. 0.]
 [0. 0.]]

C:\Program Files (x86)\Python38-32>python test1.py
[[2. 1.]
 [1. 2.]]

C:\Program Files (x86)\Python38-32>

test1 - Notepad
#zeros functionality is imported from scipy module
from scipy import zeros
# a(2x2) null matrix is constructed
A=zeros((2,2))
A[0,0]=2
A[0,1]=1
A[1,0]=1
A[1,1]=2
print(A)
```

So, I am going to now assign individual values, I am going to assign it like this way, I have created the zero null matrix, and then A, then 0 comma 0 is going to be now 1, sorry, 2, then I have 0, 1 is going to be 1, then 1, 0, this is going to be 1. And then 1, 1 is going to be 2. And if I print A, I will be able to see what I have constructed. What I have constructed is 2 1 1 2. That is the matrix which we have constructed. So, I will go back to the slide.

(Refer Slide Time: 27:18)

Python Tutorial 4: Matrix, Eigenvalues and Eigenvectors

Steps to Find Eigenvalues and Eigenvectors

Example 1: Find eigenvalues and eigenvectors of $\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$ 0, 1, 2, ...

(b) Step 2: Replace Elements to Construct the Target Matrix

```
#Import zeros functionality from scipy
from scipy import zeros
#Construct (2x2) null matrix ✓
→ A=zeros(2,2)
A[0,0]=2 #reassign the element in the 1st row and 1st column
A[0,1]=1 #reassign the element in the 1st row and 2nd column
A[1,0]=1 #reassign the element in the 2nd row and 1st column
A[1,1]=2 #reassign the element in the 2nd row and 2nd column
print(A)
```

$A = (N \times N)$ array
 $A[i,j] \rightarrow$ element of i th row and j th column

$A[0,0] =$ 1st row 1st column
 $A[1,2] =$ 2nd row 3rd column

$A[:,0]$ $\begin{pmatrix} 2 \\ 1 \end{pmatrix}$ $A[0,:]$ entire row $(2 \ 1)$

$\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$

Time dependent Quantum Chemistry

So, I have constructed now, this entire matrix


$$\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$$

and the way Python shows it with the help of this arrays, this matrix will be represented in Python with the help of this double brackets. And we have constructed the matrix right now. On this note, I will just mention one thing, that if I write down $A[:,j]$. Or if I write down $A[I;j]$ it means that this will return, this will return me the entire j th column.

So, if it is, let us say, 0, then I will get back the entire column and the entire column is going to be now 2, 1. Similarly, if this is going to give me an entire row. And if it is 0 comma colon, it means that I am going to get back the entire row and that is going to be 2, 1, the first row. And this is first column because zeroth index is indicating the first column. So, these are the ways one can collect the particular enter row and integer column of a matrix which has been represented in Python already.

(Refer Slide Time: 29:29)

Python Tutorial 4: Matrix, Eigenvalues and Eigenvectors



Steps to Find Eigenvalues and Eigenvectors

Example 1: Find eigenvalues and eigenvectors of $\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$

(c) Step 3: Finding Eigenvalues and Eigenvectors

```
from scipy import zeros
from scipy.linalg import eig #Import eig functionality from scipy.linalg
A=zeros((2,2))
A[0,0]=2
A[0,1]=1
A[1,0]=1
A[1,1]=2
E,V=eig(A) #E comprises eigenvalues and V comprises right eigenvectors
print(E)
Print(V)
```

```
[3.+0.j 1.+0.j]

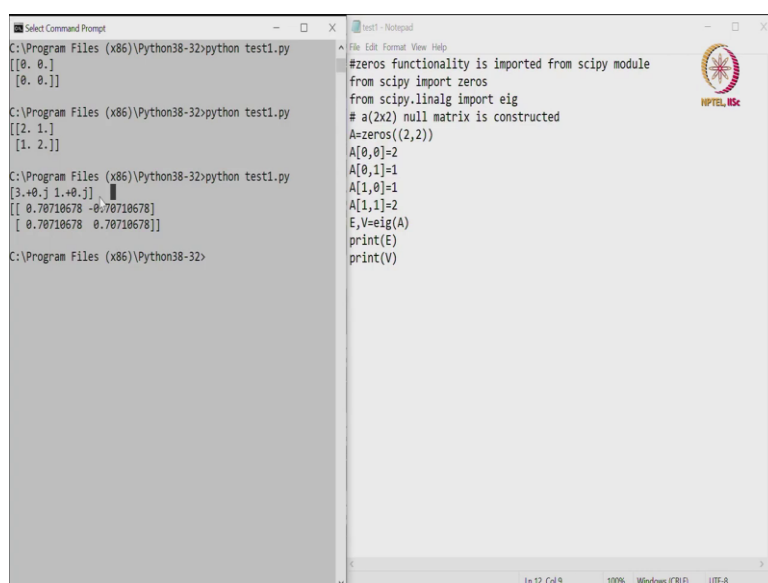
[[ 0.70710678 -0.70710678]
 [ 0.70710678  0.70710678]]
```

Time dependent Quantum Chemistry

So, we will move on and we will go to the third step which is very important step, finding the eigenvalues and eigenvectors of that created matrix. So, as we see that, from here to here is now known, I have created the matrix. Then after that I am giving a name, the first one is the eigenvalue, second one is the eigenvector.

And if I keep the name separated by a comma, then it calculates the eigenvalue eigenvector of the matrix which has been formed. Here, a matrix has been constructed with the help of this eig bracket construct. So, we will print then E and V, and we will see how they are printing. So, we have already constructed we will go back to the laptop right now.

(Refer Slide Time: 30:17)




```
Select Command Prompt
C:\Program Files (x86)\Python38-32>python test1.py
[[0. 0.]
 [0. 0.]]

C:\Program Files (x86)\Python38-32>python test1.py
[[2. 1.]
 [1. 2.]]

C:\Program Files (x86)\Python38-32>python test1.py
[3.+0.j 1.+0.j]
[[ 0.70710678 -0.70710678]
 [ 0.70710678  0.70710678]]

C:\Program Files (x86)\Python38-32>
```

```
test1 - Notepad
File Edit Format View Help
#zeros functionality is imported from scipy module
from scipy import zeros
from scipy.linalg import eig
# a(2x2) null matrix is constructed
A=zeros((2,2))
A[0,0]=2
A[0,1]=1
A[1,0]=1
A[1,1]=2
E,V=eig(A)
print(E)
print(V)
```



Ln 12, Col 9 100% Windows (Ctrl) UTF-8

And we have already constructed A. Now, we will do one thing, I do not need to print it again because I know what I have created here constructed here, I will just go ahead with this E comma V. One can change the name and make it anything, eigenvalues, eigenvalues and one can name it eigenvector.

So, any name is possible, but both has to be separated by comma and the first one has to be eigenvalue and the second one has to be eigenvector then this construct eig A, this construct eig within bracket A, this construct is indicating that Python has to find out the eigenvalues and eigenvectors.

But before we do so, I have to import this feature from scipy linear algebra sub module, it is not available with scipy directly it is available with scipy linear algebra sub module because it is a linear algebra routine which we will be following. So, it is going to be scipy dot Linear alg. I am going to import and then I am going to print E and V, print E, print V, do not use capital letter in print because Python will not be able to recognize it is sensitive to the letters.

So, now I will run this program. So, if I run the program, what I see is that the first it is printing two different arrays, it is printing to different arrays, the first array is this one and the second one is printing like this. So, E the eigenvalues are given in an array, the first eigenvalue, I will go back to the slide.

(Refer Slide Time: 32:43)

Python Tutorial 4: Matrix, Eigenvalues and Eigenvectors

Steps to Find Eigenvalues and Eigenvectors

Example 1: Find eigenvalues and eigenvectors of $\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$ $\left. \begin{array}{l} E_1 = 3 \\ E_2 = 1 \end{array} \right\}$

(c) Step 3: Finding Eigenvalues and Eigenvectors

$\hat{A}\Psi = \lambda\Psi$
 $\Psi\hat{A} = \lambda\Psi$

↑
left eigenvector

↑
right eigenvector

```

from scipy import zeros
from scipy.linalg import eig #Import eig functionality from scipy.linalg
A=zeros((2,2))
A[0,0]=2
A[0,1]=1
A[1,0]=1
A[1,1]=2
E,V=eig(A) #E comprises eigenvalues and V comprises right eigenvectors
print(E)
Print(V)

```

$E, V = \text{eig}(A)$ normalized right eigen vectors

$E, V = \text{eig}(A, \text{left}=\text{false}, \text{right}=\text{true})$

$E_1 = 3$
 $E_2 = 1$

$[3.+0.j 1.+0.j]$

$\begin{bmatrix} 0.70710678 & -0.70710678 \\ 0.70710678 & 0.70710678 \end{bmatrix}$

Time dependent Quantum Chemistry

Steps to Find Eigenvalues and Eigenvectors

Example 1: Find eigenvalues and eigenvectors of $\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$ $\left. \begin{matrix} E_1 = 3 \\ E_2 = 1 \end{matrix} \right\}$

(c) Step 3: Finding Eigenvalues and Eigenvectors

right eigenvector

 $\hat{A}\Psi = \lambda\Psi$

left eigenvector

 $\Psi\hat{A} = \lambda\Psi$

```

from scipy import zeros
from scipy.linalg import eig #Import eig functionality from scipy.linalg
A=zeros((2,2))
A[0,0]=2
A[0,1]=1
A[1,0]=1
A[1,1]=2
E,V=eig(A) #E comprises eigenvalues and V comprises right eigenvectors
print(E)
Print(V)
                    
```

$E, V = \text{eig}(A)$ *normalized right eigen vectors*

$E[j]$ <i>eigen value</i>	$V[:,j]$
------------------------------	----------

$(3+0.j, 1+0.j)$

$\begin{pmatrix} 0.70710678 & -0.70710678 \\ 0.70710678 & 0.70710678 \end{pmatrix}$

The first eigenvalue is given by this, this is j is the complex number, what we express is i in mathematics, Python will express as j . So, and zero multiplied by j means it is a real number it is actually showing, by default it is showing in the complex form. So, the first value eigenvalue is going to be 3 which means that E_1 is going to be 3 and E_2 another eigenvalue is going to be 1, this is going to be 1.

So, these are the two eigenvalues we get and this is exactly we have found analytically, when we did this analytical solution, we have found it that this matrix has two different eigenvalues one is 3 and another one was 1. And we have discussed these things in module 4 of this course, already. So, numerical we are finding the same results and corresponding to the eigenvalues what we have found, we are seeing that this eigenvector is presented in terms of, in the again array form, this is another array which has been given.

So, these vectors are given in the array and what it does actually this construct E comma V equals $\text{eig}(A)$, this construct computes the eigenvalues and normalized, it gives me normalized right eigenvectors, it gives me normalized right eigenvectors. So, right eigenvectors what does it mean? I have an operator acting on Ψ I get eigenvalue and multiply by the eigen function, the eigenvector. So, this is called right eigenvector.

Another expression I can have where A is acting from the right, this can also give me an eigenvalue equation. When it does that then this is called left eigenvector. So, by default if I do not mention anything. So, actual construct of this evaluating eigenvalues is following, this is going to be, this should be eig , then name of the matrix, then we have to write left equals false and right equals true, that we have to write down false or true.

So, generally, by default it is taking right as true. If I need to get specifically left eigenvector, then I have to write down left equals true, then all I will be able to get that, if I do not mention anything by default is taking giving me the right eigenvector and it is normalized form. So, normalized form and what we have got here is the normalized, this is this is one vector and another vector is this one.

So, there are two eigenvectors we have which is clubbed together in this array. So, now this construct actually giving me normalized line right eigenvectors. And one can access the values of the eigenvectors like this because it is an array, and one can also access there are normalized right eigenvectors in the matrix form in which jth column corresponds to the normalized right eigenvector associated with jth eigenvalue.

So, if the jth eigenvalue is this one then to get the eigenvector, so this is the eigenvalue and corresponding eigenvector is going to be, it has to be represented like this way colon comma j. So, actually I am taking the column. So, this particular column is associated with this eigenvalue and this particular column is associated with this eigenvalue.

This is the way they are representing. So, if I need to pick up the eigenvector corresponding to a particular eigenvalue, then I have to pick up following this way, V has all the information hidden, I have to just pull up the particular column corresponding to that particular eigenvalue.

(Refer Slide Time: 39:17)

Python Tutorial 4: Matrix, Eigenvalues and Eigenvectors

Steps to Find Eigenvalues and Eigenvectors

Example 1: Find eigenvalues and eigenvectors of $\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$

(c) Step 3: Finding Eigenvalues and Eigenvectors

```
from scipy import zeros
from scipy.linalg import eig
A=zeros(2,2)
A[0,0]=2
A[0,1]=1
A[1,0]=1
A[1,1]=2
E,V=eig(A)
print(E[0])
print(V[:,0])
print(E[1])
print(V[:,1])

(3+0j)
[0.70710678 0.70710678]

(1+0j)
[-0.70710678 0.70710678]
```

Time dependent Quantum Chemistry

```
Select Command Prompt
C:\Program Files (x86)\Python38-32>python test1.py
[[0. 0.]
 [0. 0.]]

C:\Program Files (x86)\Python38-32>python test1.py
[[2. 1.]
 [1. 2.]]

C:\Program Files (x86)\Python38-32>python test1.py
[[3.+0.j 1.+0.j]
 [ 0.70710678 -0.70710678]
 [ 0.70710678  0.70710678]]

C:\Program Files (x86)\Python38-32>

test1 - Notepad
#zeros functionality is imported from scipy module
from scipy import zeros
from scipy.linalg import eig
# a(2x2) null matrix is constructed
A=zeros((2,2))
A[0,0]=2
A[0,1]=1
A[1,0]=1
A[1,1]=2
E,V=eig(A)
print(E)
print(V)
```

We will now move on and we will check what we have said right now, particularly. So, we have already calculated these things. So, what we will do we will continue this session in the next class.