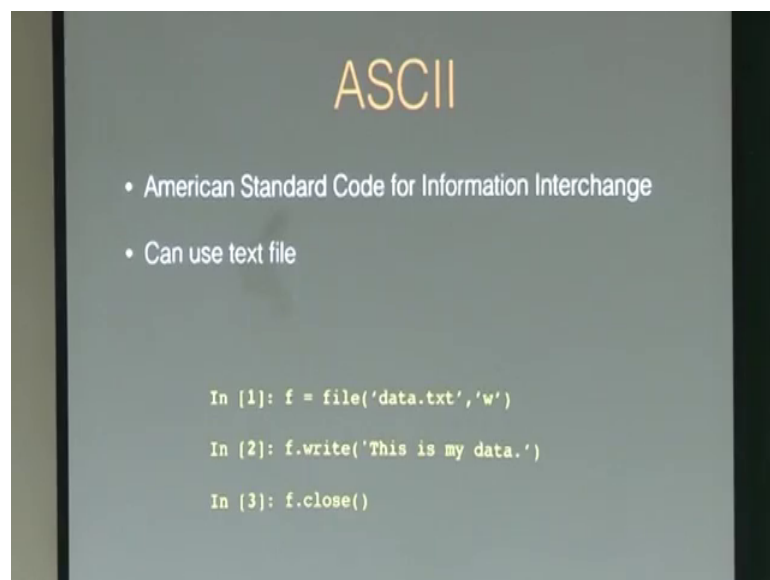


Computational Science and Engineering using Python
Prof. Mahendra K. Verma
Department of Physics
Indian Institute of Technology, Kanpur

Lecture - 08
Data Input/ Output and Mayavi

So, so far you have done some computation using python, but you have not read some data or written some data using python. So, if you are doing some simulation or if you solve a ordinary differential equation and if it takes a longer time, then you would not be able to analyze that on the fly. So, what you need to you need to write those data in some format, and then at later time you can analyze those data.

(Refer Slide Time: 00:46)



So, I will discuss these things today. So, there is one format called ASCII format, which is the American standard code for information interchange. So, you can use a text file to write a data and that is. So, this is ASCII there are different format of ASCII. So, text file is one of them. So, I will discuss about this. So, for that you need to create an object f and this is a function file. So, you write a you keep a name of a data dot txt. So, data will be saved in this file and w stands for writing.

So, next line is f dot write, and I have written this is my data. So, you can write this string will be written in that txt file, and next is to just say f dot close. So, this is the simplest example to write something whatever you want write.

(Refer Slide Time: 01:45)

```
In [1]: x = linspace(-0.9,0.9,100)
In [2]: y = x**6 - x**4 + 0.2*x**2
In [3]: f = file('data.txt','w')
In [4]: f.write('#This is my data.\n')
In [5]: for i in range(len(x)):
....:     f.write('%f \t %f \n'%(x[i], y[i]))
In [6]: f.close()

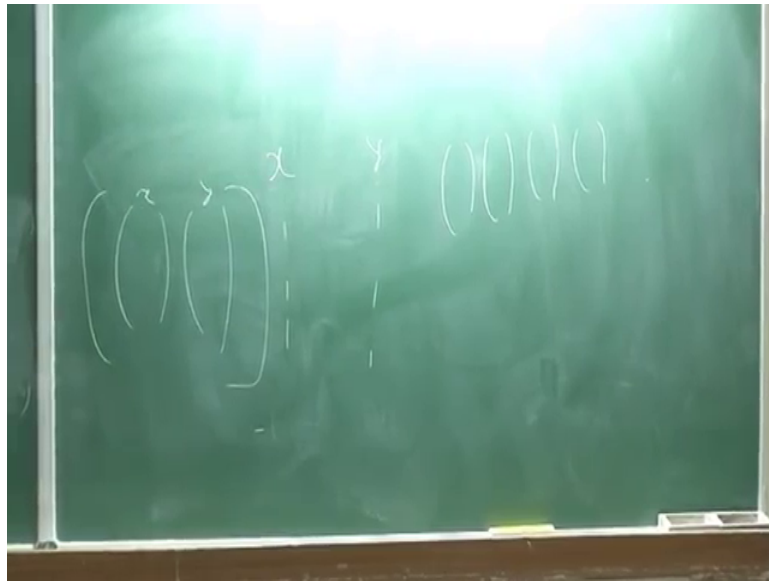
In [1]: data = np.loadtxt('data.txt')
In [2]: x = data[:,0]
```

Now, if you want to write some array. So, how you will do that and a please stop me if you have any problem in understanding. So, for that I have a x s linspace everyone knows linspace. So, it goes from minus 0.9 to 0.10 with 100 datasets, and why is this some complicated function. So, this it is a simple reference, but it could be a very complicated you can solve ordinary differential equation and you get some y. Then again I have created an object f and here is the data dot txt and I have written this is my data. So, hash is the comment. So, if you write something in python with hash that is taken as a comment would python will ignore that, and slash and for the a change of line. So, when you get those data you can write some comment about that data.

Then I have run a loop for I in range len x, len x is the length of your x array and write percent f. Percent f is the value of that data and for x i, then slash t is the tab then again percent f for this y i and then slash n change in line. So, what how loop will work. So, for x is equal to 0, it will write the x 0 value then a tab then y 0 value then change of line then again the loop will be using and you will get all set of it. I will demonstrate this.

And next is to close that file and if you want to read that data, then you just have some name data I have data and p load txt and data dot txt. So, it will take whole 2d array.

(Refer Slide Time: 03:21)



And now you can. So, if you have if you have x and y and some data. So, it will. So, these two vectors will be inside one array. So, now, you can use x colon zeros. So, it will take this x data and next will be. So, one will take y data. So, I will demonstrate this, it is visible.

Student: Almost.

(Refer Slide Time: 04:01)

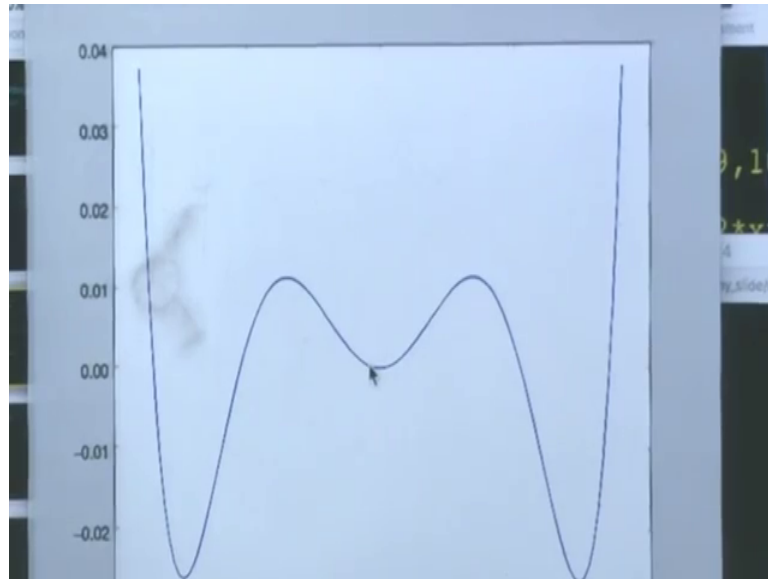
```

In [1]: x = linspace(-0.9,0.9,100)
In [2]: y = x**6 - x**4 + 0.2*x**2
In [3]: plot(x,y)
Out[3]: [<matplotlib.lines.Line2D at 0x108447710>]
In [4]: f = file('data.txt','w')
In [5]: f.write("#This is my data\n")
In [6]: for i in range(len(x)):
|         f.write('%f \t%f \n'%(x[i],y[i]))
|
|
In [7]: f.close()
In [8]: data = loadtxt('data.txt')
In [9]: data[:].0

```

So, space 0.9 9 let us y is x 6 minus x 4 plus 1 2 x, x square. So, if I want to plot this.

(Refer Slide Time: 04:25)



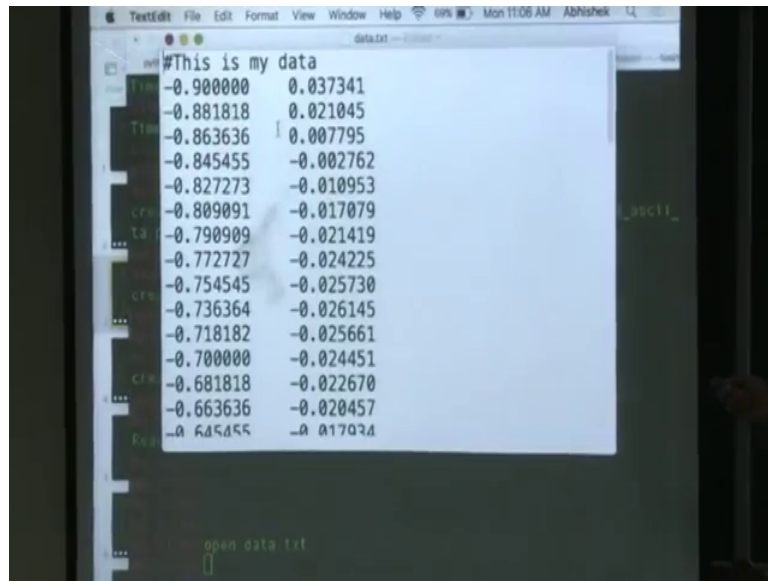
So, you have x comma y. So, now, I will write this data and read it and will test that is whatever I have read is exactly gives this function or not. So, I created object f. So, I can write some comment.

(Refer Slide Time: 05:47)

```
Time Taken (in Seconds) = 0.124115943909
example/python load_h5_data.py
Time Taken (in Seconds) = 0.114464044571
example/
example/
example/ls
... create_data.py data.txt data_set.h5 load_asci_
ta.py load_h5_data.py
example/rm data.txt data_set.h5
example/ls
... create_data.py load_asci_data.py load_h5_data.py
example/
example/
example/ls
... create_data.py load_asci_data.py load_h5_data.py
example/cd ..
my_slide ls
Reading_wri key data.txt
my_slide/
my_slide/
my_slide/
my_slide/
... my_slide/open dat
```

So, now it has written the x and y values. So, if I want to read this. So, data disc dot txt has this is my data and then x values and y values, I want to load this.

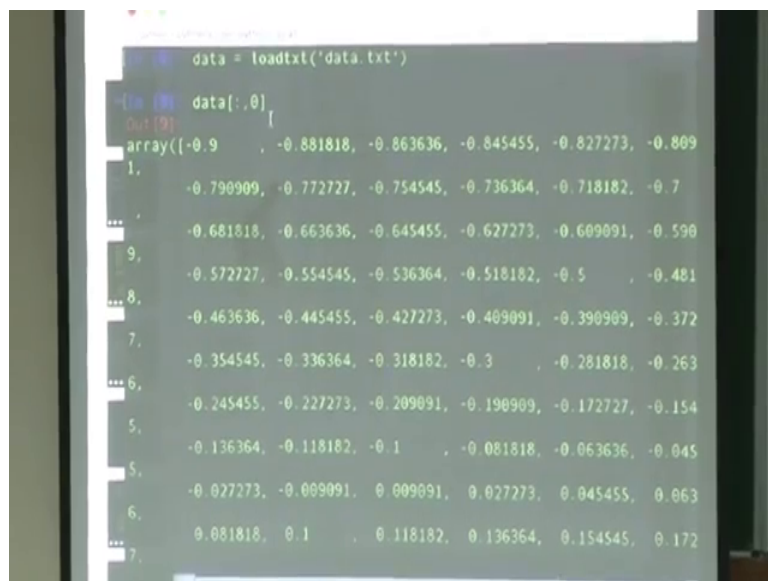
(Refer Slide Time: 05:53)



```
data.txt
#This is my data
-0.900000 0.037341
-0.881818 0.021045
-0.863636 0.007795
-0.845455 -0.002762
-0.827273 -0.010953
-0.809091 -0.017079
-0.790909 -0.021419
-0.772727 -0.024225
-0.754545 -0.025730
-0.736364 -0.026145
-0.718182 -0.025661
-0.700000 -0.024451
-0.681818 -0.022670
-0.663636 -0.020457
-0.645455 -0.017024
```

So, let us call data and equal to load and if I want to read what is x; so colon 0.

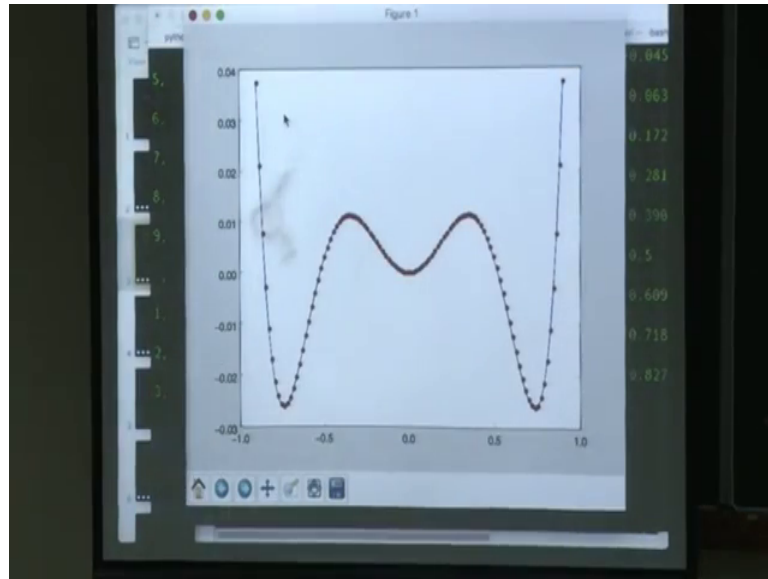
(Refer Slide Time: 06:22)



```
data = loadtxt('data.txt')
data[0]
array([-0.9, -0.881818, -0.863636, -0.845455, -0.827273, -0.809091,
        -0.790909, -0.772727, -0.754545, -0.736364, -0.718182, -0.7,
        ...,
        -0.681818, -0.663636, -0.645455, -0.627273, -0.609091, -0.590909,
        ...,
        -0.572727, -0.554545, -0.536364, -0.518182, -0.5, -0.481818,
        ...,
        -0.463636, -0.445455, -0.427273, -0.409091, -0.390909, -0.372727,
        ...,
        -0.354545, -0.336364, -0.318182, -0.3, -0.281818, -0.263636,
        ...,
        -0.245455, -0.227273, -0.209091, -0.190909, -0.172727, -0.154545,
        ...,
        -0.136364, -0.118182, -0.1, -0.081818, -0.063636, -0.045455,
        ...,
        -0.027273, -0.009091, 0.009091, 0.027273, 0.045455, 0.063636,
        ...,
        0.081818, 0.1, 0.118182, 0.136364, 0.154545, 0.172727])
```

So, it will give the x value. So, from 0.9 to minus 0.9 to point let us plot this.

(Refer Slide Time: 06:53)



So, this is x and this is y and this plotted with our p, so that building. So, it retrieves the same data.

Student: (Refer Time: 06:59).

R p is. So, r is stands for red and p is that point. So, my first curve was the blue curve original curve, and then if I would have put r then you would not be able to see any difference. So, it would overlap.

(Refer Slide Time: 07:28)

Binary File

- Dump whole data in 1 and 0
- Keeps full information
- .npy format
- .hdf5 format

```
In [1]: data_array = np.vstack((x,y))
In [2]: np.save('my_data', data_array)

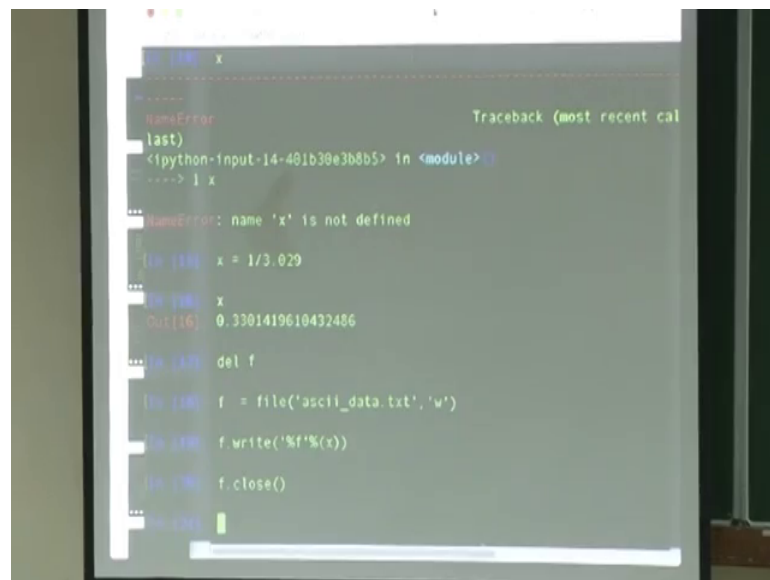
In [1]: data = np.load('my_data.npy')
```

So, next is about binary file. So, another format is to write in binary file. So, in binary it dumps the whole data in ones and zero. So, computers understand the binary system. So, it writes everything in ones and zeros. So, it gives the full information on the data. So, I will give you an example to show how it gives the full information. So, how to write in binary?

So, there are two there are many formats. So, I explain the dot npy format, which is comes from numpy and another is very advanced which is HDF 5 format. So, I will cover these two. So, first let us look at npy which is very simple. So, let us have a data array. So, this v stack means. So, if you have array x and array y then it will stack those vertically. So, if you x comma y comma z it will stay all these arrays. So, it forms a two d array from x and y then save. So, this is my data. So, you can keep any name comma data array. So, it will save this data array and that is it, and to load this data you need to say data and below my data and np. So, this is the extension is dot npy.

So, just dot npy it will take the array. So, it is very easy then the ASCII dot txt file. So, I will give you example how nice it gives the full information.

(Refer Slide Time: 09:05)



```
In [148]: x
-----
NameError                                Traceback (most recent call
last)
<ipython-input-14-481b38e3b8b5> in <module>
----> | x
...

NameError: name 'x' is not defined

In [149]: x = 1/3.029
...

In [150]: x
Out[150]: 0.3301419610432486

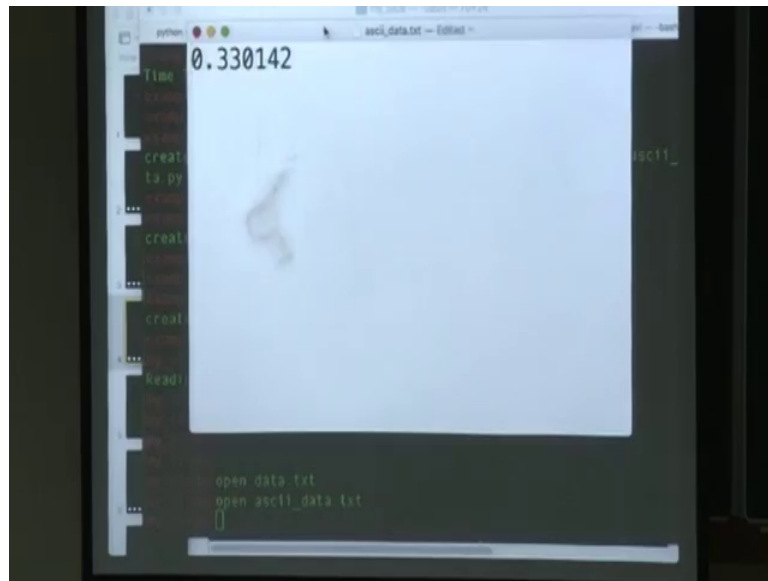
In [151]: del f
...

In [152]: f = file('ascii_data.txt', 'w')
In [153]: f.write('%f'%x)
In [154]: f.close()
...

In [155]:
```

So, let us delete x and y let call x is 1.31 by 3029. So, axis this.

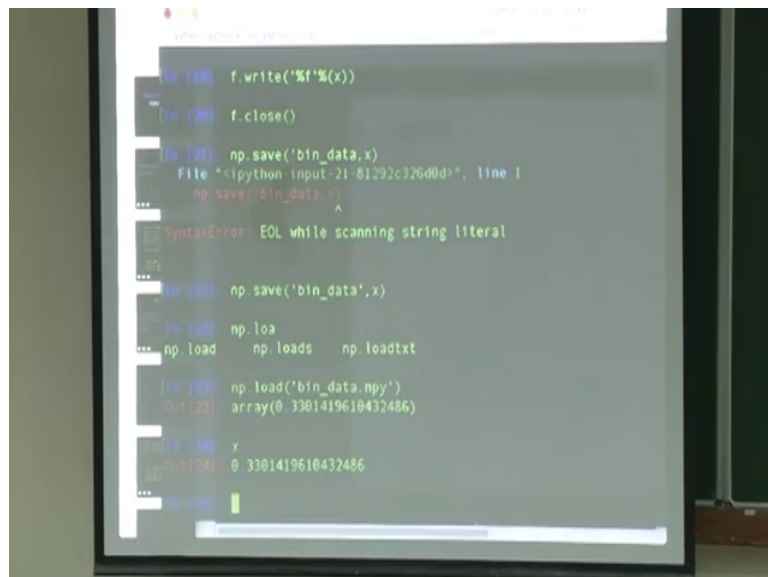
(Refer Slide Time: 09:53)



```
python
0.330142
Time
creat
ta.py
creat
Read
open data.txt
open ascii_data.txt
```

So, let us save this in ASCII. So, this much is reduced it has truncated up to 142. So, it is keeping how much and one is point and 0 see these are 8 characters. So, size of this is 8 byte. So, you can check. So, just say ls minus lh. So, size is 8 byte and my information is I do not have the full information it has not written the whole thing.

(Refer Slide Time: 10:31)



```
python
In [180]: f.write('%f'%(x))
In [181]: f.close()
In [182]: np.save('bin_data',x)
File "<ipython input 21-81292c326d0d>", line 1
      np.save('bin_data',x)
      ^
SyntaxError: EOL while scanning string literal
In [183]: np.save('bin_data',x)
In [184]: np.loa
*** np.load np.loads np.loadtxt
In [185]: np.load('bin_data.npy')
Out[185]: array(0.3301419610432486)
In [186]: x
Out[186]: 0.3301419610432486
```

So, it is truncated up to this point. So, if I save in binary. So, np dot save and if I want to read this I cannot read. So, its question mark and question mark. So, it is written in binary, I cannot read it in doing cat or anything else. So, I need to read in python.

So, np dot load and it returns the whole information. So, where is x; so it keeping whole x. So, you have nothing is lost. So, if you do a some simulation say of a 1000 cube data and you have 1000 cube differential equations. So, when you read this that initial condition in with the ASCII and binary you will see the difference and you know that about chaos.

So, if the initial condition will be sensitive then it will reflect later. So, that is why for doing a bigger simulation one should use binary file system rather than an ASCII file system, where you keep the full information and nothing is lost. So, is it cleared up to this point. So, next is HDF5. So, HDF5 is a very advanced version.

(Refer Slide Time: 12:04)

```
In [1]: write_file = h5py.File('data_set.h5')
In [2]: write_file['data_1'] = data_array
In [3]: write_file.close()

In [1]: read_file = h5py.File('data_set.h5')
In [2]: data = read_file['/data_1']
In [3]: data = np.array(data)
In [4]: write_file.close()
```

So, let us look how to write in HDF5. So, first you need to install the h 5 py package. So, if you are using anaconda. So, you need to say conda install h 5 py. So, it will install that packet. So, this h this file is the object of h 5 py. So, I create right file h 5 py file data set. So, as HDF 5 gives it is a file system. So, you can say many arrays inside that. So, I say write file data underscore one. So, it will create space for one array and data array.

So, data array was that a stack x and y, I could have also done data underscore two data underscore three and as many as I want then write file dot close. So, it will just write those that data. So, this is how I write the data not to read. So, I create read file and same data set then I read the data y read file slash data underscore one. So, if I have other data then different underscore two three whatever according to the name of that. Then I need

to convert this data to numpy array because in python we use numpy and then write file dot close. So, it will close the HDF a bit. So, we will do one example for this.

(Refer Slide Time: 13:23)

```
1 import numpy as np
2 import h5py
3
4 x = np.arange(0,5000,0.001)
5 y = np.sin(x)
6
7 f = file('data.txt', 'w')
8
9 for i in range(len(x)):
10     f.write('%f \t %f \n'%(x[i], y[i]))
11 f.close()
12
13 data_array = np.vstack((x,y))
14 write_file = h5py.File('data_set.h5')
15 write_file['data_1'] = data_array
16 write_file.close()
17
18
```

So, I have written one code to create data this is a code where numpy import library numpy, then another library h5py and I write from 0 to 5000 this step point 0.001 and y is just sin x, and again I create f for data dot txt and write it in this loop which I have explained earlier. Next what I do the same data is written as HDF 5. So, I stack this x and y write them in data set dot h5 data underscore one this array is inserted over here write dot close. So, same data is returned in ASCII as data dot txt as well as it hdf.

So, let us run this code. So, we will see that is there any compression in saving ASCII and compared to HDF 5. So, size of data dot txt is 116 mega byte, data dot set s 576 mega byte.

(Refer Slide Time: 14:21)

```
1
2
3
4 example/open create_data.py
5 example/open create_data.py
6 example/open create_data.py
7 example/python create_data.py
8 example/ls -lh
9 total 393376
10 -rw-r--r--@ 1 abhishek staff 296B Jan 31 20:28 create_data.py
11 -rw-r--r-- 1 abhishek staff 116M Feb  1 11:19 data.txt
12 -rw-r--r-- 1 abhishek staff 76M Feb  1 11:19 data_set.h5
13 -rw-r--r--@ 1 abhishek staff 201B Jan 31 17:02 load_ascii_data.
14 -rw-r--r--@ 1 abhishek staff 270B Jan 31 17:02 load_h5_data.py
15 example/open load_ascii_data.py
16
17
18
```

So, it is reduced and now I will load this data. So, it is ASCII. So, we will see how much time does it take while reading ASCII data and while reading HDF 5 data.

(Refer Slide Time: 14:42)

```
1 import numpy as np
2 import timeit
3
4
5 start = timeit.default_timer()
6 data = np.loadtxt('data.txt')
7 t = data[:,0]
8 x = data[:,1]
9 stop = timeit.default_timer()
10 print "Time Taken (in Seconds) = ",stop-start
11
```

So, I will introduce one import time it. So, time is the library. So, you do not need to install that it is already installed it comes with the package of python.

So, here I write written start the timer. So, it will just start the clock and here a data is loaded with np load dot txt and t, and x is same thing was I have described earlier and then it is stopped.

(Refer Slide Time: 15:17)

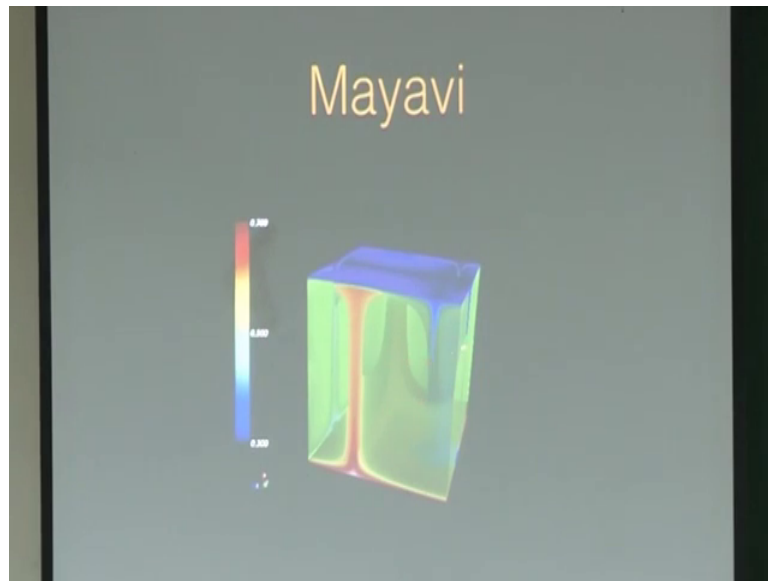
```
1 in
2 in
3 in
4 example/open create_data.py
5 st example/open create_data.py
6 re example/open create_data.py
7 da example/python create_data.py
8 da example/ls -lh
9 t total 393376
10 x -rw-r--r--@ 1 abhishek staff 296B Jan 31 20:28 create_data.py
11 -rw-r--r-- 1 abhishek staff 116M Feb 1 11:19 data.txt
12 st -rw-r--r-- 1 abhishek staff 76M Feb 1 11:19 data_set.h5
13 pr -rw-r--r--@ 1 abhishek staff 201B Jan 31 17:02 load_ascii_data.
14 -rw-r--r--@ 1 abhishek staff 270B Jan 31 17:02 load_h5_data.py
example/open load_ascii_data.py
example/python load_ascii_data.py
Time Taken (in Seconds) = 39.278646946
example/open load_h5_data.py
example/python load_h5_data.py
Time Taken (in Seconds) = 0.125559091568
example/
```

So, and I will compute the time taken in this operation is I stop minus start will give me the amount of time taken in seconds and the mean yes. So, it start. So, its take 39.27 seconds, and I will now load with HDF 5. So, this is the code for HDF 5. So, this is the code for HDF 5. So, I introduce one more library h5 py again I start then read file this data set data underscore one, convert this data to numpy array then t x stop and it will print. So, it will read the same data ASCII must take 39.2 second. So, any guess how much time will it will take with as HDF 5.

Student: (Refer Time: 15:54) half.

Half 0.12 second; so when you will do some projects in this course later time, I had to save in HDF 5. So, that will save your time, do not save in ASCII and then read with ASCII. So, you that would not be so beneficial, while doing your project you will save some big data. So, you will you will get about say 200 mb or something then save it in HDF 5 do not use ASCII for saving bigger data.

(Refer Slide Time: 16:39)



Now, I will do a little bit of visualization. So, when you have written those data then you will analyze them. So, for analyzing you can use Mayavi. So, Mayavi we can create some density plot and i. So, contours. So, this was made by the Indian person IIT Bombay a programmer prabhuramachandran he is the creator of this Mayavi package. So, this is one of my research problem. So, I will just demonstrate this. So, how to create? So, this is the hot fluid which is trying to go up and the cold fluid who is trying to come down, these are the plume. So, if you have this data and you want if you want to create here I have the ISO contours and also the density plot.

(Refer Slide Time: 17:14)

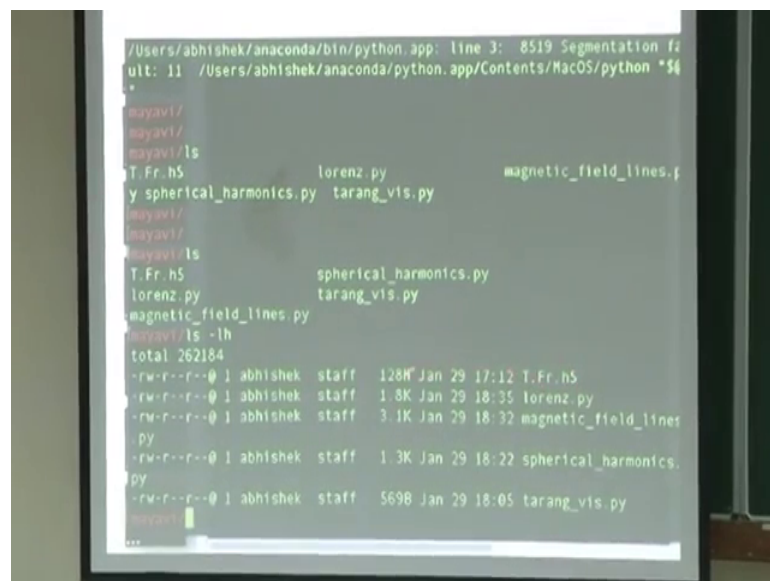
```
maya.figure(bgcolor=(0, 0, 0),size=(800, 600))
maya.contour3d(T, colormap='jet')
src = maya.pipeline.scalar_field(T)
maya.pipeline.surface(src, colormap='spectral',opacity=0.7)
maya.colorbar(orientation='vertical', nb_labels=3)
maya.show()
```

So, how to use what you will do it is same something similar to what you use in matplotlib. So, you just say Maya dot figure, this is the background color and the size 800 in comma 600 pixels and I have an array t. So, it is that is temperature. So, you could have different array according to your problem and I set a color map say jet.

So, there are different color maps. So, you will be able to see later, this is for the contour 3d. Now if I want to plot the density plot the source this scalar field t and I write this surface src color mass spectrum, and opacity is 0.7. So, what does opacity is move from 0 to 1. So, if you decrease the opacity you can see inside fine inside the structures, and then you said the color bar. So, this is similar thing you have already done with your contour plots of matplotlib.

So, just any this is orientation. So, you can change your orientation and number labels. So, I keep three and then Maya dot show similar to plt dot.

(Refer Slide Time: 18:13)



```
/Users/abhishek/anaconda/bin/python.app: line 3: 8519 Segmentation fa
ult: 11 /Users/abhishek/anaconda/python.app/Contents/MacOS/python *$@
*
mayavi/
mayavi/
mayavi/ls
T.Fr.h5          lorenz.py          magnetic_field_lines.py
y spherical_harmonics.py  tarang_vis.py
mayavi/
mayavi/
mayavi/ls
T.Fr.h5          spherical_harmonics.py
lorenz.py        tarang_vis.py
magnetic_field_lines.py
mayavi/ls -lh
total 262184
-rw-r--r--@ 1 abhishek  staff  128K Jan 29 17:12 T.Fr.h5
-rw-r--r--@ 1 abhishek  staff   1.8K Jan 29 18:35 lorenz.py
-rw-r--r--@ 1 abhishek  staff   3.1K Jan 29 18:32 magnetic_field_lines
py
-rw-r--r--@ 1 abhishek  staff   1.3K Jan 29 18:22 spherical_harmonics.
py
-rw-r--r--@ 1 abhishek  staff   569B Jan 29 18:05 tarang_vis.py
mayavi/
...
```

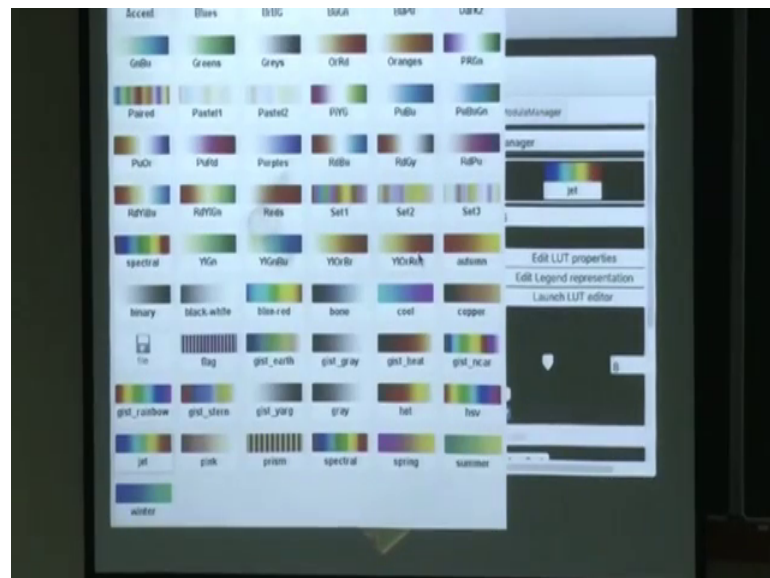
So, I have a data not t f r dot h file which of 128 mb and let us do this I show you therefore,. So, forget about this. So, you should just take this Mayavi package. So, you need to install Mayavi also. So, you need to say conda installed Mayavi then you will be able to put Mayavi library, and just ignore this much. So, I take theta and I convert theta to t and this is same what I have shown in the slides.

(Refer Slide Time: 18:21)

```
1 from __future__ import division
2 import h5py
3 import numpy as np
4 import mayavi.mlab as mlab
5
6 N = 256
7
8 x = np.linspace(0,1,N)
9 X = np.zeros((N,N,N))
10
11 for i in range(0,N):
12     X[i,:,:] = x[i]
13
14
15 theta_data = h5py.File('T.Pr.h5')
16 theta = theta_data['T.Pr']
17 theta = np.array(theta)
18
19 T = 1 -X +theta
20
21
22 mlab.figure(bgcolor=(0, 0, 0),size=(800, 600))
23 mlab.contour3d(T, colormap='jet', contours = 4, vmax = 0.7, vmin = 0)
24 src = mlab.pipeline.scalar_field(T)
25
```

So, let us run this code so. So, now, you can visualize this, you can go inside and see how the structures are formed.

(Refer Slide Time: 18:43)

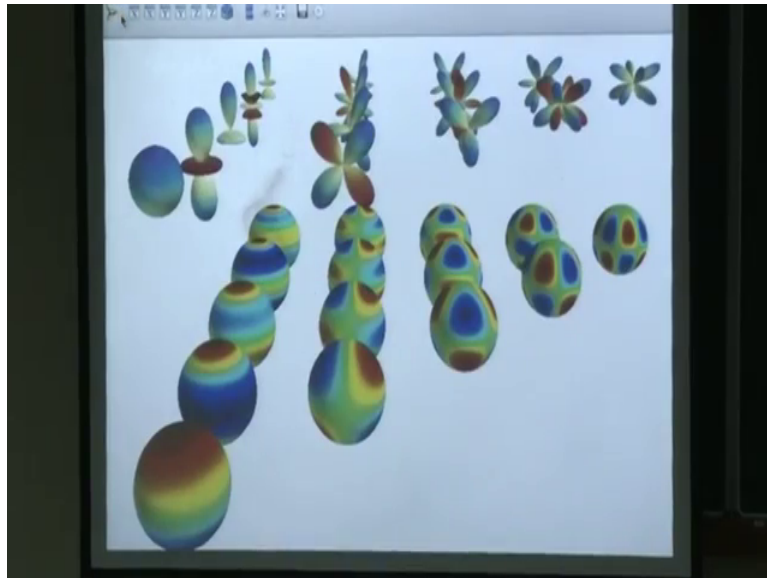


So, when you take your project you may need to do not similar kind of problem, but something close to that and you can produce some very nice plots, and that will be very useful in presentation the colorful plots are always useful. So, this what I wanted to show regarding today's class, I have some other if you want to use

Student: (Refer Time: 19:14) color map.

The color map; so you can. So, you have different color maps. So, you can. So, I put jet you can put these many color maps.

(Refer Slide Time: 19:46)

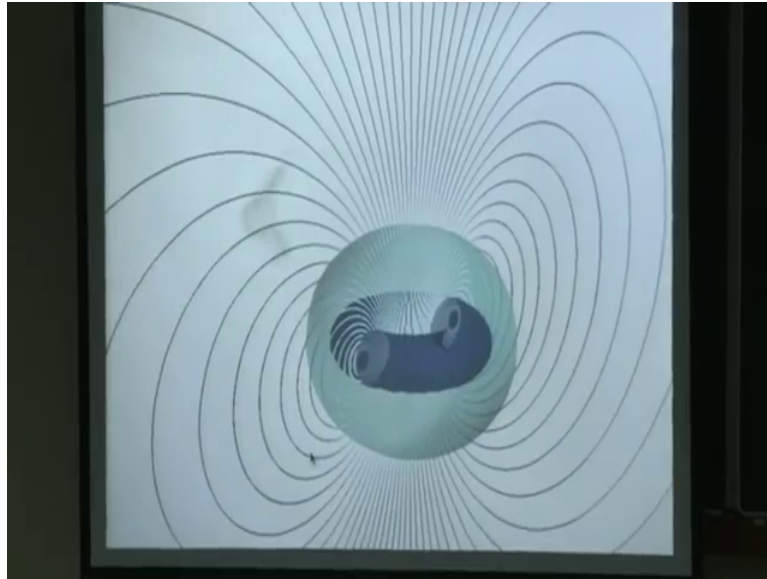


So, with different color maps you can just keep on trying and get the best you can get. So, I have just copied few examples from Mayavi side. So, just I will demonstrate this, then like a spherical harmonics.

Student: (Refer Time: 19:50)

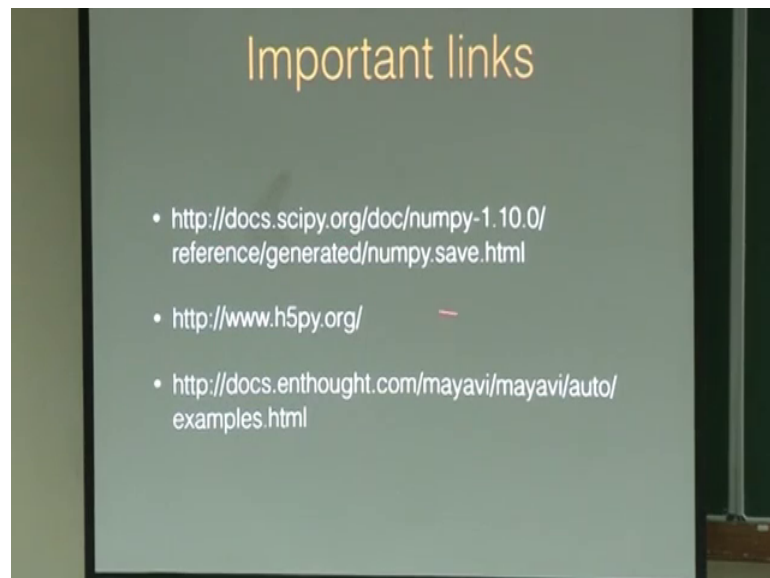
So, you can just see this orbital. So, I think task is there was problem with everything nice. So, you can make as many as and you can just play with these things. So, it has done all the spherical harmonics. So, if you look at the code I will give you these codes. So, it varies n from 1 to 6 with m , with range n and it will just take the spherical harmonics from the from scipy packet.

(Refer Slide Time: 20:44)



So, do not worry about these codes so and another example of the magnetic field lines. So, this is the magnetic field lines from a magnetic dipole.

(Refer Slide Time: 20:45)



So, you can try these at this link you will get the information about; if you want to know more about numpy array and py sorry numpy save. So, that n py file system you can use this link the other link is for h5 py and this is for the mayavi example gary what I have shown just now.