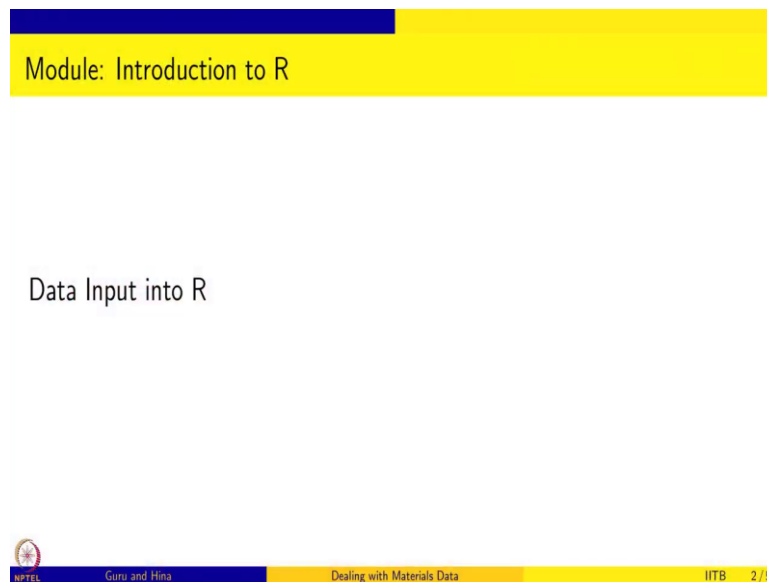


Dealing with Materials Data: Collection, Analysis and Interpretation
Professor M P Gururajan
Professor Hina A Gokhale
Department of Metallurgical Engineering and Material Science
Indian Institute of Technology, Bombay
Lecture No. 16
Importing and Plotting Data

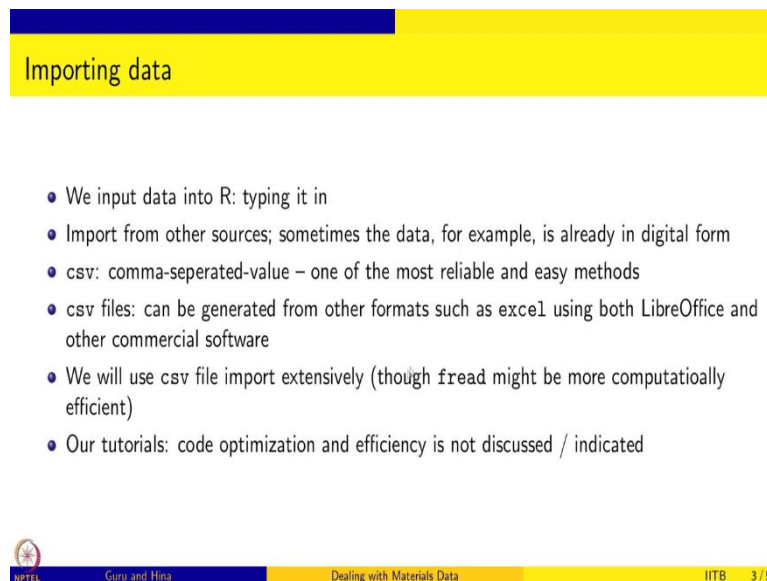
Welcome to dealing with materials data, in this course we are going to look at how to collect analysis and interpret materials data, and this is the first module, this is the module for introduction to the R programming language, and we have been working with R in the past few sessions and we have been looking at how to inter data into R and then work with it, manipulated, plotted and so on and on so forth.

(Refer Slide Time: 00:49)



So, in this session we are going to learn how to import data into R, so data input can be done into R like we learnt earlier the by typing it in, in the R console, but sometimes that is not the best way to inter data into R.

(Refer Slide Time: 01:05)



Importing data

- We input data into R: typing it in
- Import from other sources; sometimes the data, for example, is already in digital form
- csv: comma-separated-value – one of the most reliable and easy methods
- csv files: can be generated from other formats such as excel using both LibreOffice and other commercial software
- We will use csv file import extensively (though fread might be more computationally efficient)
- Our tutorials: code optimization and efficiency is not discussed / indicated

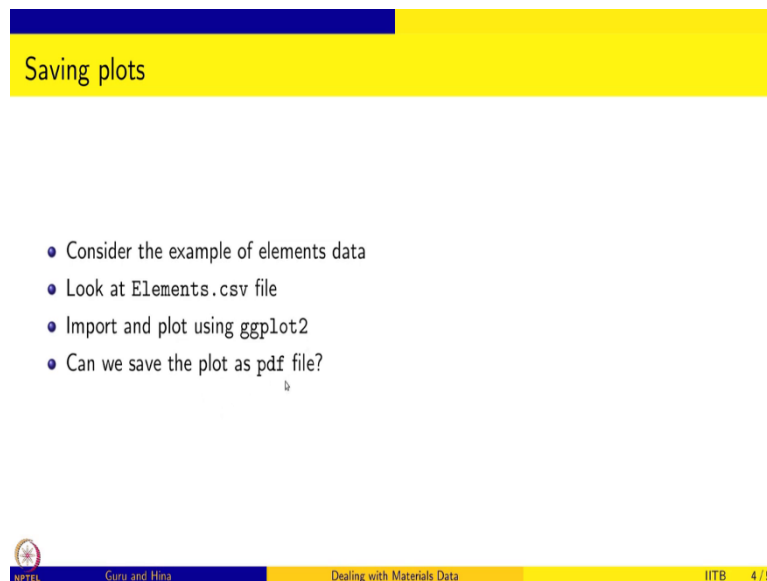
NPTEL Guru and Hina Dealing with Materials Data IITB 3 / 5

And in those cases we can import from other sources, this becomes important because sometimes the data for example might be already in digital form. You might have done an experiment and the data comes and it is stored in the computer in digital form and so you want to take that data and work with that and you do not want to be manually entering all the data and sometimes that data could be very big, so manual entry is time consuming, also it is prone to errors.

So, the most reliable and easy method to enter data from such sources is the csv file format, csv stands for comma-separated-value, and you can take the data which is generated and stored in other formats into csv using either excel or using LibreOffice or any other commercial software for example.

And we will use csv file import extensively, again there is an optimization point which is that reading and writing csv files might be costlier, it might be easier and much more faster to use F read and F write, this again is an optimization point regarding the R programming which we are going to not discuss in this course, but it is better to aware and if you do anything at much larger scale, and you find that the speed of execution becomes a bottleneck for you, I think these are some of the places where you should look and improve your code. So, in these tutorials we are not going to discuss code optimization and efficiency, if possible I will indicate, but we will not do too much of it.

(Refer Slide Time: 03:15)



The slide has a yellow header with the title "Saving plots" in blue. Below the header is a white area containing a bulleted list. At the bottom of the slide is a footer with a logo on the left, the text "Guru and Hina" in the center, "Dealing with Materials Data" on the right, and "IITB 4 / 5" on the far right.

- Consider the example of elements data
- Look at Elements.csv file
- Import and plot using ggplot2
- Can we save the plot as pdf file?

So, we are going to now consider the elements data and I have saved it as a csv file and we are going to import and plot using ggplot2, up to this we have done. But what has not been done is that can be save the plot as the pdf files? So, that is also something that we are going to learn in this session. So, we are going to look at csv file data and we are going to import it into R and we are going to use ggplot2 to plot it and then we are going to save the data, the plot that we made using ggplot as a pdf files, so that is the exercise.

(Refer Slide Time: 03:45)



The slide has a yellow header with the title "Saving plot as a pdf file" in blue. Below the header is a white area containing R code. At the bottom of the slide is a footer with a logo on the left, the text "Guru and Hina" in the center, "Dealing with Materials Data" on the right, and "IITB 5 / 5" on the far right.

```
pdf("Figures/DensityVsMeltingPointA.pdf")
(... plot commands ...)
dev.off()
```

And saving plot as a pdf file is rather straight forward so you have to say that okay, you want to save as a pdf file and you have to tell, where and what is the name of the pdf and then you can give all your plot commands, and then after that is complete you switch off this device that

pdf you close, so that you can get back your regular console to look at the plot, so that is what we are going to do in this session. So, I am going to start by showing you the element strategies in csv.

(Refer Slide Time: 04:27)

The screenshot shows a Linux desktop environment. In the background, a file manager window displays a directory containing several CSV files: 'CuOfFree', 'alCoCond', 'Subly.csv', 'Elements.csv', 'Elements2.csv', 'Elements3.csv', 'Elements4.csv', 'EPCoCond', 'subly.csv', 'GrainSD', and 'at561.csv'. A LibreOffice window is open, displaying a 'Text Import' dialog box for the file 'Elements.csv'. The dialog box has the following settings:

- Character set: Unicode (UTF-8)
- Language: Default - English (en-US)
- From row: 1
- Separator Options: Field with, Separated by, Tab, Comma, Semicolon, Space, Other
- Other Options: Merge delimiters, String delimiter, Format quoted field as text, Detect special numbers
- Fields: A table with columns for Element, Crystal Structure, Density, and Melting Point.

The table in the 'Fields' section is as follows:

Standard	Standard	Standard	Standard
Element	Crystal Structure	Density	Melting Point
Aluminium	FCC	2700	600
Nickel	FCC	8900	1453
Gold	FCC	19300	2983
Silver	FCC	10490	961
Copper	FCC	8960	2030
Iron	BCC	7870	2535
Chromium	BCC	7190	2175

Below the dialog box, a snippet of R code is visible:

```
X <- read.csv("../Data/Elements.csv")
library("ggplot2")
ggplot(X, aes(x = Density, y = Melting.Point, color = as.factor(Crystal.Structure))) +
  xlim(0, 23000) +
  ylim(0, 3600) +
  geom_point() +
  geom_label(data=X, aes(label=X$Element), hjust=-0.2, vjust=-0.1)
```

The top screenshot shows a LibreOffice Calc spreadsheet with the following data:

Element	Crystal Structure	Density	Melting Point
Aluminium	FCC	2700	660
Nickel	FCC	8900	1453
Gold	FCC	19320	1063
Silver	FCC	10490	961
Copper	FCC	8960	1083
Iron	BCC	7870	1535
Chromium	BCC	7190	1875
Molybdenum	BCC	10220	2610
Vanadium	BCC	6100	1900
Tungsten	BCC	19300	3410
Magnesium	HCP	1740	650
Beryllium	HCP	1850	1277
Zinc	HCP	7130	420
Cadmium	HCP	8650	321
Titanium	HCP	4510	1668

The middle screenshot shows the same data in a table format:

Element	Crystal Structure	Density	Melting Point
Aluminium	FCC	2700	660
Nickel	FCC	8900	1453
Gold	FCC	19320	1063
Silver	FCC	10490	961
Copper	FCC	8960	1083
Iron	BCC	7870	1535
Chromium	BCC	7190	1875
Molybdenum	BCC	10220	2610
Vanadium	BCC	6100	1900
Tungsten	BCC	19300	3410
Magnesium	HCP	1740	650
Beryllium	HCP	1850	1277
Zinc	HCP	7130	420
Cadmium	HCP	8650	321
Titanium	HCP	4510	1668

The bottom screenshot shows an RStudio console session with the following code and output:

```

> getwd()
[1] "/home/guru/Desktop/DealingWithMaterialsData"
> X <- read.csv("Data/Elements.csv")
> str(X)
'data.frame': 15 obs. of 4 variables:
 $ Element      : Factor w/ 15 levels "Aluminium","Beryllium",...: 1 10 6 11 5 7 4 9 14 13 ...
 $ Crystal.Structure: Factor w/ 3 levels "BCC","FCC","HCP": 2 2 2 2 1 1 1 1 1 ...
 $ Density      : int  2700 8900 19320 10490 8960 7870 7190 10220 6100 19300 ...
 $ Melting.Point : int  660 1453 1063 961 1083 1535 1875 2610 1900 3410 ...

```

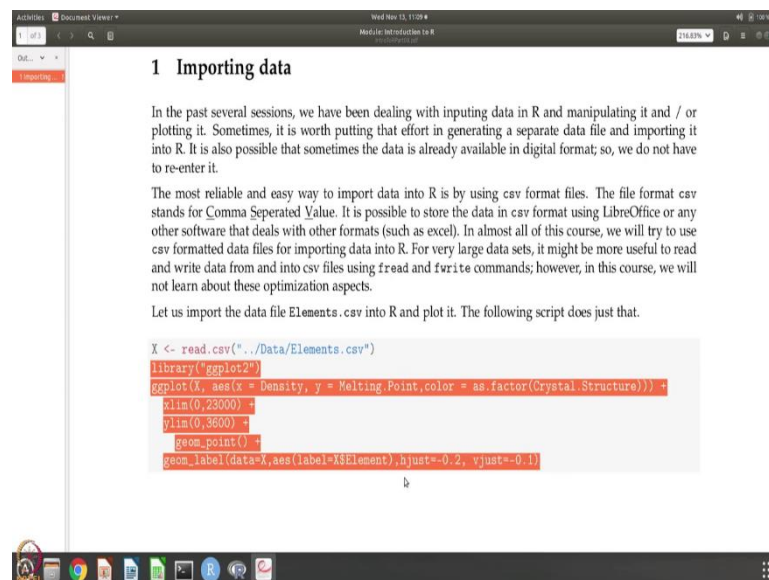
The right sidebar shows the ggplot2 package documentation, including a description and a list of authors.

So, like I said it is csv files can be opened in LibreOffice, so I do not know the data is very clear, so, let us format text, I do not know how increase the size, okay, it might be here. So,

this is how the data looks like, so it is the same data element crystal structure density melting point, and the elements are listed five FCC, five BCC and five HCP, so this is the data and it is stored as elements dot csv.

So, what we are going to do, is to first import data into R and for doing that, so I need to know which directory are there, so I need to see, so, now I am dealing with materials data, so I need to import the data, the way we import is as follows. So, I am going to say read into this variable x or elements and read dot csv, so that is the command, from where? From the data directory and which file? Element.csv . So, it has read, as we can see x is 15 observations of 4 variables. So, we now know that we got the right data. So, it is the same element crystal structure density melting point, okay.

(Refer Slide Time: 06:30)



The screenshot shows a presentation slide titled "1 Importing data". The slide contains text explaining the process of importing data into R and plotting it. The text states that the most reliable and easy way to import data into R is by using csv format files. It also mentions that the data file is named "Elements.csv" and is located in the "Data" directory. The slide includes the following R code:

```
X <- read.csv("../Data/Elements.csv")
library("ggplot2")
ggplot(X, aes(x = Density, y = Melting.Point, color = as.factor(Crystal.Structure))) +
  xlim(0, 23000) +
  ylim(0, 3600) +
  geom_point() +
  geom_label(data=X, aes(label=X$Element), hjust=-0.2, vjust=-0.1)
```

The top screenshot shows the RStudio interface with the following code in the console:

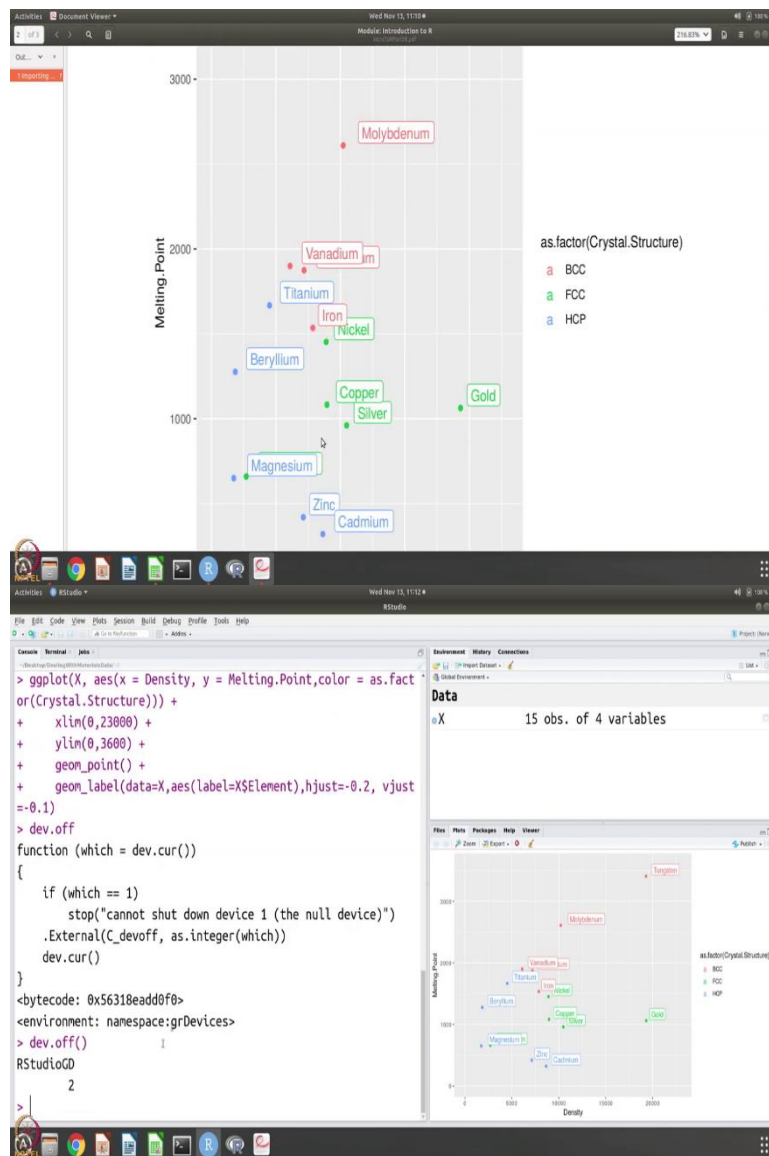
```
> getwd()
[1] "/home/guru/Desktop/DealingWithMaterialsData"
> X <- read.csv("Data/Elements.csv")
> str(X)
'data.frame': 15 obs. of 4 variables:
 $ Element      : Factor w/ 15 levels "Aluminium","Beryllium",...: 1 10 6 11 5 7 4 9 14 13 ...
 $ Crystal.Structure: Factor w/ 3 levels "BCC","FCC","HCP": 2 2 2 2 1 1 1 1 1 1 ...
 $ Density       : int  2700 8900 19320 10490 8960 7870 7190 10220 6100 19300 ...
 $ Melting.Point : int  660 1453 1063 961 1083 1535 1875 2610 1900 3410 ...
> library("ggplot2")
ggplot(X, aes(x = Density, y = Melting.Point, color = as.factor(Crystal.Structure))) +
  xlim(0,23000) +
  ylim(0,3600) +
  geom_point() +
  geom_label(data=X,aes(label=X$Element),hjust=-0.2, vjust=-0.1)]
```

The bottom screenshot shows the same code with the plot rendered. The plot displays Melting Point (y-axis) versus Density (x-axis) for 15 elements. The points are colored according to their crystal structure: BCC (blue), FCC (green), and HCP (red). The plot includes a legend for the crystal structures and labels for each element.

Now, let us do the plotting. So we can go, so we invoke the library ggplot2 and we say okay, x is the data and the aesthetics is x is density, y is melting point, color should be according to crystal structure and we changed the x limit and y limit and the geometry is point. So, you can see several layers I am building, so this is the first one, then the next layer is the for the x range, next layer is for y range, next layer is for the plots points, the geometry of the plotting and then the next layer is for the label and label is again from x data using the element information, and this is how the label is put, so, this be now or comfortable with.

So, we have it, the density verses melting point and you can see all the BCC, FCC, HCP etc. are colored differently and the points and the writing everything is in the corresponding color, so you can get the plot.

(Refer Slide Time: 07:49)



Now, we want to save this file, so for saving it what do we do? We are going to, so I need to know there is a directory called figures here. So, let me see, okay there is no directory called figures, so it is better to make a directory called figures here, okay. So, now I am going to save, so I am going to save pdf, equal to, okay so figures is the directory where I want to save this figure, so it is elements data, so let us say Elementsdataplot.pdf, let say that is the pdf file, so that is the pdf, so we are going to say this is the pdf file we want to generate, okay.

(Refer Slide Time: 09:14)

The image displays three sequential screenshots from an RStudio session, illustrating the process of plotting melting points against density for various metals and their crystal structures.

Top Screenshot: Shows a scatter plot titled "Graphics Output" with "Melting Point" on the y-axis and "Density" on the x-axis. Two points are highlighted: "Tungsten" (red dot) and "Molybdenum" (red dot). The plot is titled "as.factor(Crystal.Structure)".

Middle Screenshot: Shows the RStudio interface with a file explorer window open, displaying folders like "Data", "Figures", "Notes", "Scripts", and "Slides". The console window shows the following R code:

```
<bytecode: 0x56318eadd0f0>  
<environment: namespace:grDevices>  
> dev.off()  
RStudioGD  
2
```

Bottom Screenshot: Shows a zoomed-in view of the scatter plot. The y-axis is labeled "Melting Point" and has tick marks at 2000 and 3000. The x-axis is labeled "Density" and has tick marks at 0, 1000, 1500, 2000, and 2500. Points are labeled: "Tungsten" (red), "Molybdenum" (red), "Vanadium" (red), "Titanium" (blue), "Iron" (green), "Copper" (green), "Silver" (green), "Zinc" (blue), and "Cadmium" (blue). A legend titled "as.factor(Crystal.Structure)" is visible in the bottom right corner, with entries: "a BCC", "a FCC", and "a HCP". A zoom menu is open on the right side of the plot, showing options from 50% to 400%.

Now, we are going to put this command like we did earlier. So we are going to say, okay, so this is the, and then we are going to say device start off, okay so I have to give, okay. So, now let us go check if we have made this figure, yes, there is this pdf file that is available now, and you can open it and you can see that the figure is stored. So, we can, so you can see density verses melting and all the data points and differently colored, so everything is nicely, now stored as a pdf file, so which you can then use for other purposes, okay. So, this is how the saving is done and so this is to import data and to plot data. We will have some more exercises in importing and plotting data in the next session. Thank You.