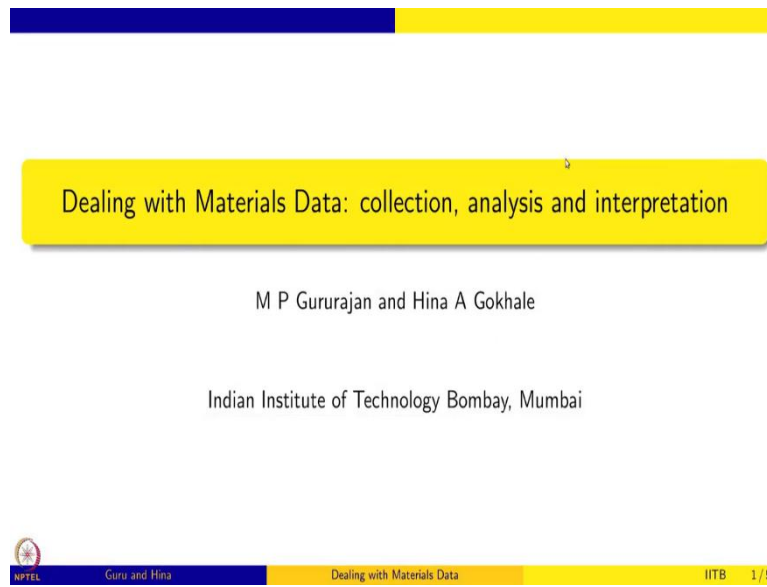


**Dealing with Materials Data: Collection, Analysis and interpretation**  
**Professor M P Gururajan**  
**Department of Metallurgical Engineering and Materials Science**  
**Indian Institute of Technology, Bombay**  
**Lecture 13**  
**Tabular Data in R: Alternate Methodology**

(Refer Slide Time: 01:10)



Dealing with Materials Data: collection, analysis and interpretation

M P Gururajan and Hina A Gokhale

Indian Institute of Technology Bombay, Mumbai

NPTEL Guru and Hina Dealing with Materials Data IITB 1/5

Welcome to Dealing with Materials Data, we are going to look at the collection, analysis and interpretation of data and we are in the first module. The module is interpretation to R and we have looked at how to enter data into R and do analysis with the data plot the data and things like that, so we are going to continue, we have been using one data set which is about certain elements, their crystal structures, densities and melting point.

So we are going to continue with that and in one of the earlier session I showed one way of entering the data as a table and working with it and I also gave the problem or modification to what we did by adding the name of the element in the table as part of the data and that is what we are going to do in today's session.

(Refer Slide Time: 01:26)

Module: Introduction to R

Tabulated data: second version

NPTEL Guru and Hina Dealing with Materials Data IITB 2 / 5

Elements and their properties

- Consider some elements
- Let us consider their crystal structure (fcc, bcc, and hcp), density (in  $\text{kg}/\text{m}^3$ ), and melting point  $^{\circ}\text{C}$  (as reported in Raghavan, Materials Science and Engineering)
- FCC: Aluminium (2700, 660), nickel (8900, 1453), gold (19320,1063), silver (10490,961), and copper (8960,1083)
- BCC: Iron (7870, 1535), chromium (7190,1875), molybdenum (10220,2610), vanadium (6100,1900) and tungsten (19300,3410)
- HCP: Magnesium (1740,650), beryllium (1850,1277), zinc (7130,420), cadmium (8650,321) and titanium (4510, 1668)

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


So let us look at, so we are going to look at tabulated data the second version and just to remind you this are the information that is in the data. So we are going to consider some elements and we are going to consider their crystal structure whether they are FCC, BCC or HCP their density given in kilo grams per meter cube and melting point which is reported in degree Celsius. This data is taken from Raghavan's material science and engineering text book.

So for FCC we have chosen aluminium, nickel, gold, silver and copper and the two numbers are the densities and meting points respectively. So, and for BCC we are going to consider Iron, Chromium, Molybdenum, Vanadium and Tungsten and for HCP Magnesium, Beryllium, Zinc, Cadmium and titanium. So this is the data and we want to enter this data and work with it and plot it and so on so forth.

(Refer Slide Time: 02:24)

Table of data: second version

- Slight modification to earlier version: include element names
- Rows: data – no longer labelled
- Columns: labelled!!!

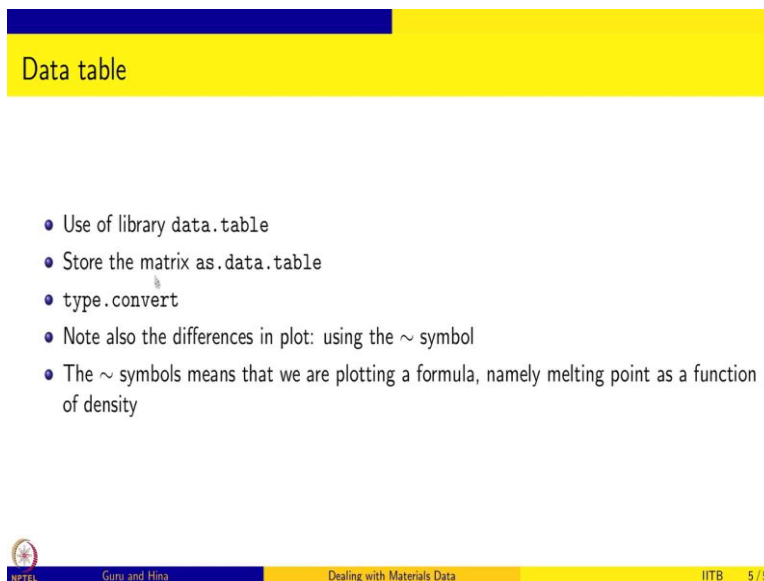


NPTEL      Guru and Hina      Dealing with Materials Data      IITB 4 / 5

So this is the slight modification to the earlier version. In the earlier version the data of the melting points and densities along with the crystal structure was given and the rows were labelled according to the element and the columns were named as to whether it is the melting point or density or crystal structure etc.

What we want to do in this case is to remove the labelling of the rows instead make the name of the element as part of the data. So we will now have only columns to be labelled and the columns will be elements, crystal structure, density and melting point.

(Refer Slide Time: 03:07)



The slide features a yellow header with the title "Data table" in dark blue. Below the header, a list of five bullet points is centered on the slide. At the bottom, there is a footer with a logo on the left, the text "Guru and Hima" in the middle, and "IITB 5/5" on the right.

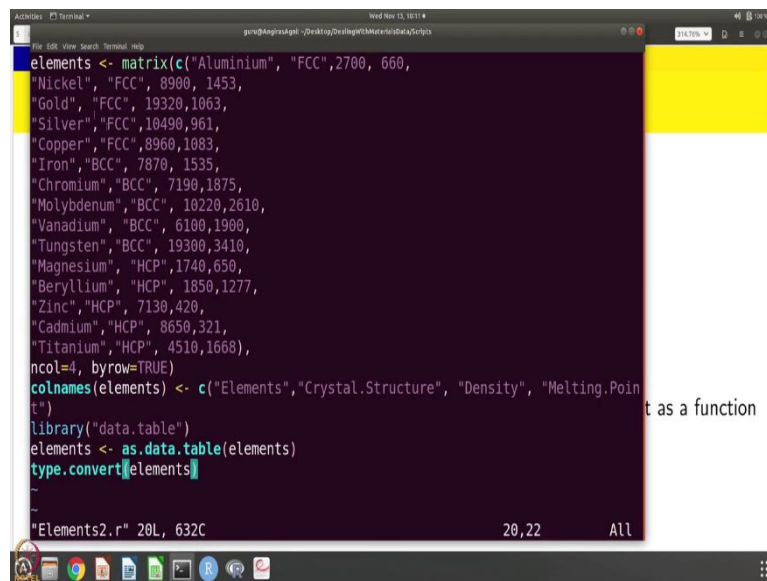
- Use of library `data.table`
- Store the matrix as `data.table`
- `type.convert`
- Note also the differences in plot: using the `~` symbol
- The `~` symbols means that we are plotting a formula, namely melting point as a function of density

NPTEL    Guru and Hima    Dealing with Materials Data    IITB 5/5

And rest of it would be data and we are going to enter the data in R and we are going to use a library called data dot tables and we are going to store the matrix as data table and we are also going to use type convert command and there are also going to be some minor differences in the way in which we are going to do the plotting.

And so we are going to use the tilde symbol for doing the plotting and the tilde symbol means that we are plotting a formula. So we are going to say that melting point as a function of density, so it should be consider like a functional relationship and the plotting should be done. So that is what we are going to do.

(Refer Slide Time: 04:00)



```
elements <- matrix(c("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),
ncol=4, byrow=TRUE)
colnames(elements) <- c("Elements", "Crystal.Structure", "Density", "Melting.Point")
library("data.table")
elements <- as.data.table(elements)
type.convert(elements)
```

So let us do that lets first look at the way to enter the data and here is the script that does that so as we did earlier so we are going to say elements is a matrix and this are the data that is entered into the matrix. So you can see that the name of the element, the crystal structure, the density and the melting point.

And then there is nickel FCC 8900, 1453, gold FCC 19320, 1063 and so on. So we have listed for all of them. So there are 5 FCC and 5 BCC and 5 HCP, so number of columns is 4 so and the data is stored by row. So what we mean by row equal to true is that the data is as aluminium FCC 2700, 660 so that is one row.

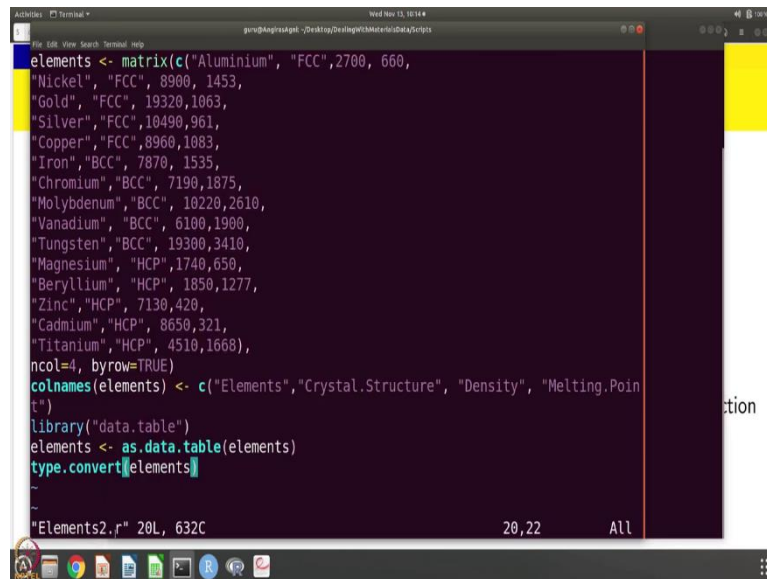
And number of columns is 4, so it has to take 4 data points and next 4 data point will form next row and so on so forth. So this is how this matrix is stored, this matrix has 4 columns and it stored by row and there are 15 rows. So we are going to give names for this columns, so that is what column names does.

So for this matrix which is stored in the variable elements and we are going to give the column names and the column name are elements, crystal structure, density and melting point and then we invoke the library data dot tables and so that library helps us stored the elements matrix as a data table and then we are going to use type dot convert because as you can see there are strings and there are numbers and so this type conversion helps us make sure that the data is stored appropriately, so this is what this script does. And so let us open R and run the script and see what happens.

(Refer Slide Time: 06:05)

```
elements <
"Nickel", R version 3.6.1 (2019-07-05) -- "Action of the Toes"
"Gold", "F"Copyright (C) 2019 The R Foundation for Statistical Computing
"Silver", "Platform: x86_64-pc-linux-gnu (64-bit)
"Copper", "I
"Iron", "BCR is free software and comes with ABSOLUTELY NO WARRANTY.
"Chromium" You are welcome to redistribute it under certain conditions.
"Molybdenum"Type 'license()' or 'licence()' for distribution details.
"Vanadium"
"Tungsten" Natural language support but running in an English locale
"Magnesium
"Beryllium" R is a collaborative project with many contributors.
"Zinc", "HCl"Type 'contributors()' for more information and
"Cadmium", 'citation()' on how to cite R or R packages in publications.
"Titanium"
ncol=4, byType 'demo()' for some demos, 'help()' for on-line help, or
colnames(e'help.start()' for an HTML browser interface to help.
t") Type 'q()' to quit R.
library("d.
elements <>
type.conve
~
"Elements2
```

```
elements <You are welcome to redistribute it under certain conditions.
"Nickel", Type 'license()' or 'licence()' for distribution details.
"Gold", "F
"Silver", "I Natural language support but running in an English locale
"Copper", "I
"Iron", "BCR is a collaborative project with many contributors.
"Chromium" Type 'contributors()' for more information and
"Molybdenum" 'citation()' on how to cite R or R packages in publications.
"Vanadium"
"Tungsten" Type 'demo()' for some demos, 'help()' for on-line help, or
"Magnesium" 'help.start()' for an HTML browser interface to help.
"Beryllium" Type 'q()' to quit R.
"Zinc", "HCl
"Cadmium", > getwd()
"Titanium" [1] "/home/guru"
ncol=4, by> setwd("Desktop/DealingWithMaterialsData/Scripts/")
colnames(e> getwd
t") function ()
library("d.Internal(getwd())
elements <<-bytecode: 0x55bdc5eaf890>
type.conve<environment: namespace:base>
> getwd()
[1] "/home/guru/Desktop/DealingWithMaterialsData/Scripts"
"Elements2>
```



```
elements <- matrix(c("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),
ncol=4, byrow=TRUE)
colnames(elements) <- c("Elements", "Crystal.Structure", "Density", "Melting.Point")
library("data.table")
elements <- as.data.table(elements)
type.convert(elements)
```

So as usual we open R and first thing we have to check is that we have the right version, so R version is 3.6.1 action of the toes so this is what we have been using and I have to know which directory I am in? So that is done by the command `getwd` that work for get working directory, so that is what is stands for and it says I am in home guru, so I have to go to the working directory which his in a desktop and it called dealing with materials data, I need to go to the scripts directory in there.

So we set, so now if you say get working directory a you (sorry) get working directory and you get the answer that I am in home guru desktop dealing with material data scripts, and this script that I showed you is stored as `elements2 dot r` there.

(Refer Slide Time:07:07)

```
elements <- Internal(getwd())
"Nickel", <-bytecode: 0x55bdc5eaf890>
"Gold", "F<environment: namespace:base>
"Silver", "> getwd()
"Copper", "[1] "/home/guru/Desktop/DealingWithMaterialsData/Scripts"
"Iron", "BC" > source("Elements2.r")
"Chromium" data.table 1.12.6 using 2 threads (see ?getDTthreads). Latest news: r-datatable
"Molybdenum.com
"Vanadium" > str("elements")
"Tungsten" chr "elements"
"Magnesium" > class("elements")
"Beryllium" [1] "character"
"Zinc", "HCl" > list.dir()
"Cadmium", Error in list.dir() : could not find function "list.dir"
"Titanium" > list.files()
ncol=4, by [1] "Archive" "Elements2.r"
colnames(e [3] "ElementsDF.r" "ElementsImportPlot.r"
t") [5] "Figures" "PlottingElementsDFGGPlot.r"
library("d: [7] "SpecificModulus.r" "SpecificModulusVsTm.r"
elements < [9] "SpecificStrength.r"
type.conve > source("Elements2.r")
> str("elements")
chr "elements"
"Elements2" > elements
```

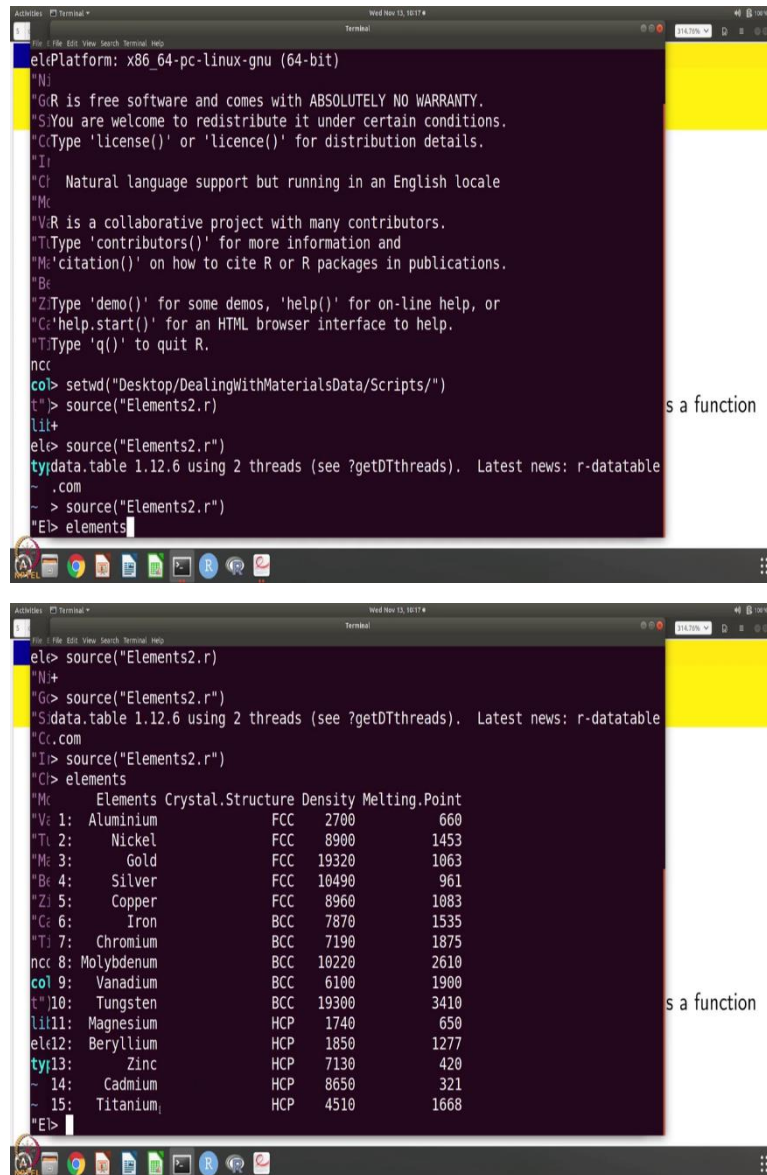
```
elements < [5] "Figures" "PlottingElementsDFGGPlot.r"
"Nickel", [7] "SpecificModulus.r" "SpecificModulusVsTm.r"
"Gold", "F [9] "SpecificStrength.r"
"Silver", "> source("Elements2.r")
"Copper", "> str("elements")
"Iron", "BC" chr "elements"
"Chromium" > elements
"Molybdenum Elements Crystal.Structure Density Melting.Point
"Vanadium" 1: Aluminium FCC 2700 660
"Tungsten" 2: Nickel FCC 8900 1453
"Magnesium 3: Gold FCC 19320 1063
"Beryllium 4: Silver FCC 10490 961
"Zinc", "HCl 5: Copper FCC 8960 1083
"Cadmium", 6: Iron BCC 7870 1535
"Titanium" 7: Chromium BCC 7190 1875
ncol=4, by 8: Molybdenum BCC 10220 2610
colnames(e 9: Vanadium BCC 6100 1900
t") 10: Tungsten BCC 19300 3410
library("d:11: Magnesium HCP 1740 650
elements <12: Beryllium HCP 1850 1277
type.conve13: Zinc HCP 7130 420
14: Cadmium HCP 8650 321
15: Titanium HCP 4510 1668
"Elements2" >
```

So we need to source the script, so that is what we are going to do elements2 dot r and when we source okay, so there is some information that is given data dot table is using two threads and so you can get information about it. Anyway, so to know whether our data is stored properly let us just check the structure of elements and so we get and what is the class elements? So okay, so list okay.





(Refer Slide Time: 09:01)



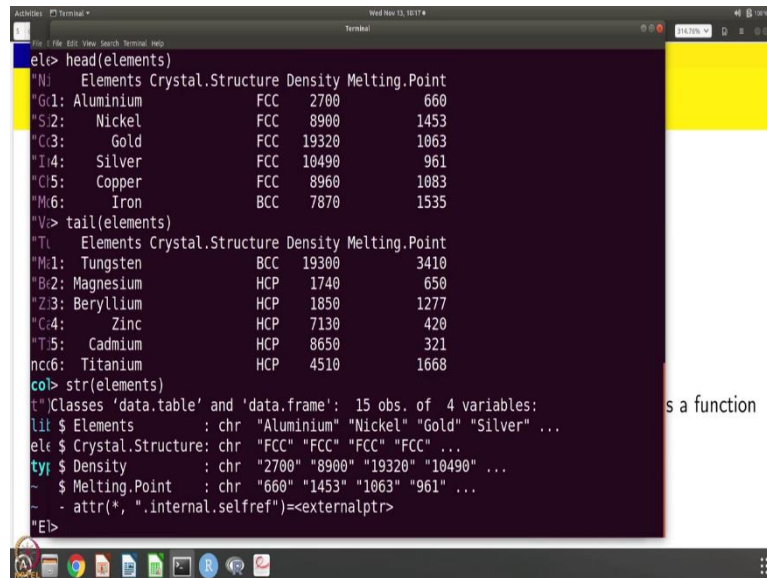
```
el@Platform: x86_64-pc-linux-gnu (64-bit)
"R is free software and comes with ABSOLUTELY NO WARRANTY.
" You are welcome to redistribute it under certain conditions.
" C Type 'license()' or 'licence()' for distribution details.
" I
" C Natural language support but running in an English locale
" M
" V R is a collaborative project with many contributors.
" T Type 'contributors()' for more information and
" M 'citation()' on how to cite R or R packages in publications.
" B
" Z Type 'demo()' for some demos, 'help()' for on-line help, or
" C 'help.start()' for an HTML browser interface to help.
" T Type 'q()' to quit R.
ncc
co> setwd("Desktop/DealingWithMaterialsData/Scripts/")
t"> source("Elements2.r")
li|+
el@> source("Elements2.r")
ty|data.table 1.12.6 using 2 threads (see ?getDTthreads). Latest news: r-datatable
~ .com
- > source("Elements2.r")
"E|> elements

el@> source("Elements2.r")
"N|+
" G> source("Elements2.r")
" S data.table 1.12.6 using 2 threads (see ?getDTthreads). Latest news: r-datatable
" C .com
" I> source("Elements2.r")
" C> elements
" M Elements Crystal.Structure Density Melting.Point
" V 1: Aluminium FCC 2700 660
" T 2: Nickel FCC 8900 1453
" M 3: Gold FCC 19320 1063
" B 4: Silver FCC 10490 961
" Z 5: Copper FCC 8960 1083
" C 6: Iron BCC 7870 1535
" T 7: Chromium BCC 7190 1875
ncc 8: Molybdenum BCC 10220 2610
co| 9: Vanadium BCC 6100 1900
t" 10: Tungsten BCC 19300 3410
li|11: Magnesium HCP 1740 650
el|12: Beryllium HCP 1850 1277
ty|13: Zinc HCP 7130 420
- 14: Cadmium HCP 8650 321
- 15: Titanium HCP 4510 1668
"E|>
```

So let us source the elements2 dot r so that is a okay, so there is a mistake. So let us source the table, okay so we have sourced it and we can check the data element so now we can see that the element is stored as a data table and it has elements, crystal structure, density and melting point and the elements are named and their crystal structure is given, density is given and melting point is given.



(Refer Slide Time: 09:52)



```
el> head(elements)
"Elements Crystal.Structure Density Melting.Point
"Cr1: Aluminium FCC 2700 660
"Cr2: Nickel FCC 8900 1453
"Cr3: Gold FCC 19320 1063
"Cr4: Silver FCC 10490 961
"Cr5: Copper FCC 8960 1083
"Cr6: Iron BCC 7870 1535

"V> tail(elements)
"Elements Crystal.Structure Density Melting.Point
"Cr1: Tungsten BCC 19300 3410
"Cr2: Magnesium HCP 1740 650
"Cr3: Beryllium HCP 1850 1277
"Cr4: Zinc HCP 7130 420
"Cr5: Cadmium HCP 8650 321
"Cr6: Titanium HCP 4510 1668

co> str(elements)
t"Classes 'data.table' and 'data.frame': 15 obs. of 4 variables:
li $ Elements : chr "Aluminium" "Nickel" "Gold" "Silver" ...
el $ Crystal.Structure: chr "FCC" "FCC" "FCC" "FCC" ...
ty $ Density : chr "2700" "8900" "19320" "10490" ...
~ $ Melting.Point : chr "660" "1453" "1063" "961" ...
- - attr(*, ".internal.selfref")=<externalptr>
"El>
```

So, we can also use the head which will give you the top 6 data point and tail which will give you the bottom data points, And of course you can also find out the information so its we have a data table and data frame and it consists of elements, crystal structure density, melting point and so on, so you can also use the command class we already know that its going to be data table and data frame,

So at this point I want to mention that we stored things as table or we can also store as data frame which is what the next session we are going to do and it is sometime favourable to store data as a table specially if it is a very large data that is related to the optimization of the R code and that is one aspect that we are not going to spend too much time in this course. In fact we are going to do very little of code optimization.

So this is more of a tutorial version just to learn R and how to work with it, But if you want to make more efficient code you may have to spend more time and learn some of this other aspects, But we are not going to do it in these sessions. Now that we have the data so we can try to plot the data.

(Refer Slide Time: 11:34)

The top screenshot shows the R console output for creating a data frame from a table of elements and their properties. The output is as follows:

```

elements 4:      Zinc      HCP      7130      420
"Nickel", 5:  Cadmium    HCP      8650      321
"Gold", 6:   Titanium    HCP      4510      1668
"Silver",> str(elements)
"Copper",Classes 'data.table' and 'data.frame': 15 obs. of  4 variables:
 "Iron", "B $ Elements : chr "Aluminium" "Nickel" "Gold" "Silver" ...
 "Chromium $ Crystal.Structure: chr "FCC" "FCC" "FCC" "FCC" ...
 "Molybden $ Density : chr "2700" "8900" "19320" "10490" ...
 "Vanadium $ Melting.Point : chr "660" "1453" "1063" "961" ...
 "Tungsten - attr(*, ".internal.selfref")=<externalptr>
"Magnesium> class(elements)
"Beryllium [1] "data.table" "data.frame"
"Zinc", "H" > X <- elements$Density
"Cadmium" > X
"Titanium [1] "2700" "8900" "19320" "10490" "8960" "7870" "7190" "10220" "6100"
ncol=4, b[10] "19300" "1740" "1850" "7130" "8650" "4510"
colnames > Y <- elements$Melting.point
t")
> Y
library("NULL
elements > Y <- elements$Melting.Point
type.conv > Y
[1] "660" "1453" "1063" "961" "1083" "1535" "1875" "2610" "1900" "3410"
[11] "650" "1277" "420" "321" "1668"
"Elements> plot(Y~X)

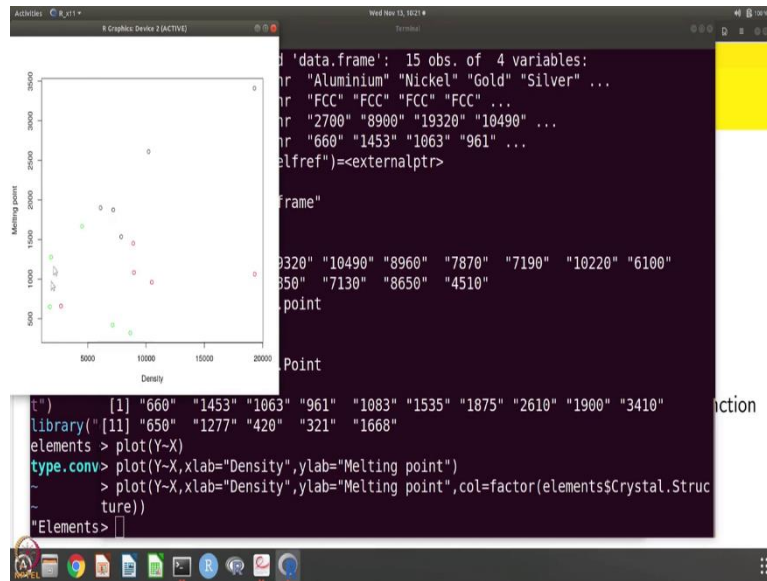
```

The bottom screenshot shows the same R console output with a scatter plot overlaid on the left side. The plot shows density on the X-axis and melting point on the Y-axis. The data points are colored according to their crystal structure: FCC (red), BCC (green), and HCP (blue).

So to plot the data we are going to say I am going to say that X is a so let us just say X is form the elements the density okay so you can see that X has chosen all the densities and let us say that Y is nothing but the melting points okay. So Y is the melting point. Okay so it is null because I have made a mistake, so let us look at it now Y has.

So we are going to say plot we are going to say Y as a function of X is what it has to plot, so you can do this then you will get a plot we have got the plot. Ofcourse we know how to label the X. The X label should be density and Y label should be melting point. So okay well we have got density versus melting point. We also want to colour let us say the points according to their crystal structure because it is not clear now which is FCC, which is BCC, which is HPC.

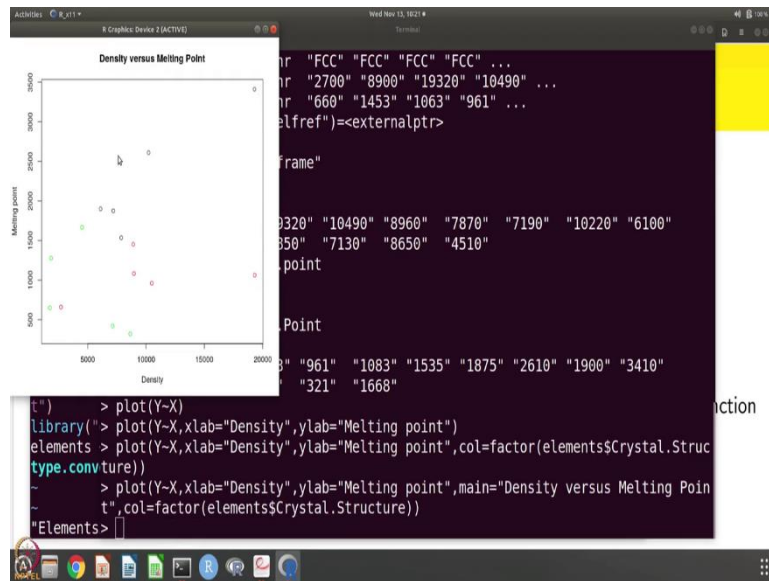
(Refer Slide Time: 13:07)



So to do that I am going to use command colour and that is according to elements okay, so that is, now you can see that there are 3 different colours there are greens and then reds and then blacks and they correspond to the 3 different crystal structures.

So this is similar to what we have been doing so you can load the data and in this case we loaded the data as data table in that sense also it is slightly different from what we have done earlier and then you can take individual columns and you can plot them and you can give the name to the XY axis. Off course there is one more thing that we did last time so we can do that here also.

(Refer Time Slide: 14:04)



We can give a name to the plot, so we can give a name and as you can see density versus melting point name is given. So you can do all this things and we have also colour coded the points in the scatter plot to show the different crystal structure.

So there are 3 different crystal structure, so you see 3 different colours, so this is another way of doing it and ofcourse there is another one more way which is what we are going to do is to store the data as a data frame and work with it. So that we will do in the next session, thank you.