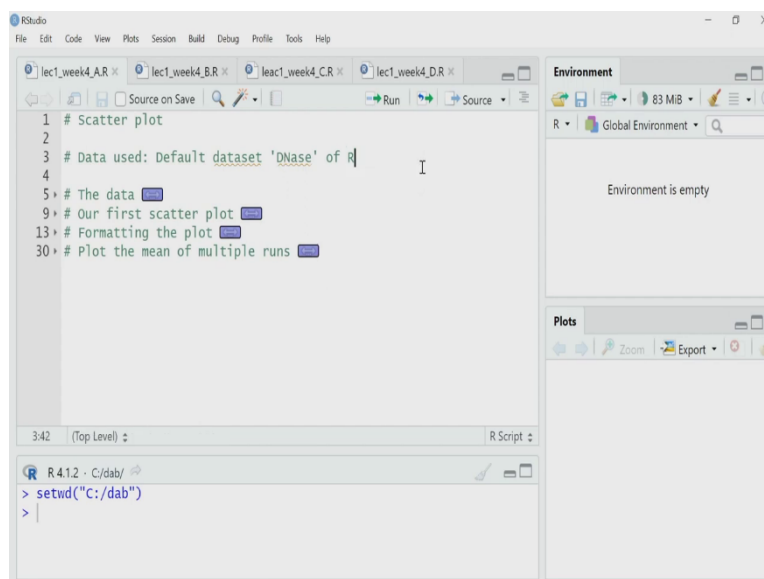


Data Analysis for Biologists
Professor Biplab Bose
Department of Biosciences & Bioengineering,
Mehta Family School of Data Science & Artificial Intelligence
Indian Institute of Technology Guwahati
Lecture 20
Scatter plot, Line plot, Bar plot

Hello everyone, welcome back. In this lecture, we learn how to visualize data as scatter plot, line plot or bar plot. And while doing so, I will use R to make those plots. So, in a way in this lecture, we will learn how to create, when to create scatter plot and bar plot and how to create them using R. Now, you must be knowing that the scatter plot, line plots and bar plots, these are all the most common method of visualizing data and the oldest one also.

So, already you must be creating those using some other software like Excel or something like that. What you will learn here? You will learn how to make those using R. And I will also discuss certain important issues that you have to keep in mind when visualizing data as scatterplot or a bar plot. So, we will start with scatter plot.

(Refer Slide Time: 1:31)



RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

lec1_week4_AR x lec1_week4_BR x lec1_week4_CR x lec1_week4_DR x

Source on Save Run Source

```

1 # Scatter plot
2
3 # Data used: Default dataset 'DNase' of R
4
5 # The data ----
6
7 d <- DNase
8
9 # Our first scatter plot
10
11 # Formatting the plot
12
13 # Plot the mean of multiple runs

```

7:11 The data R Script

R 4.1.2 · C:/dab/

```

> setwd("C:/dab")
> d <- DNase
>

```

Environment

R · Global Environment

Data

d 176 obs. of 3 v...

Plots

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

lec1_week4_AR x d x lec1_week4_BR x lec1_week4_CR x lec1_week4_>>

Filter

Run	conc	density
1	0.04882812	0.017
2	0.04882812	0.018
3	0.19531250	0.121
4	0.19531250	0.124
5	0.39062500	0.206
6	0.39062500	0.215
7	0.78125000	0.377
8	0.78125000	0.374
9	1.56250000	0.614
10	1.56250000	0.609

Showing 1 to 11 of 176 entries, 3 total columns

R 4.1.2 · C:/dab/

```

> setwd("C:/dab")
> d <- DNase
> View(d)
>

```

Environment

R · Global Environment

Data

d 176 obs. of 3 v...

Plots

RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

lec1_week4_AR x d x lec1_week4_BR x lec1_week4_CR x lec1_week4_>>

Filter

Run	conc	density
169	1.56250000	0.704
170	1.56250000	0.684
171	3.12500000	0.994
172	3.12500000	0.980
173	6.25000000	1.421
174	6.25000000	1.385
175	12.50000000	1.715
176	12.50000000	1.721

Showing 168 to 176 of 176 entries, 3 total columns

R 4.1.2 · C:/dab/

```

> setwd("C:/dab")
> d <- DNase
> View(d)
>

```

Environment

R · Global Environment

Data

d 176 obs. of 3 v...

Plots

d ← DNase

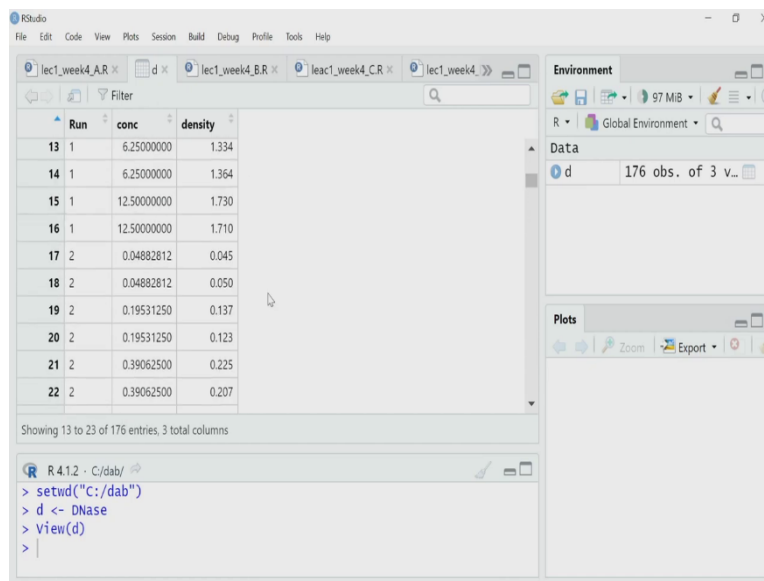
To create a scatter plot, I will be using a inbuilt data set called the DNase data set in R. In this data set what you have? You have a Elisa assay results. So, in this what they are doing? You are, they assaying the enzymatic activity of enzyme called DNase and they are measuring that activity in term through a method called Elisa. So, what you have? You have the data for different concentrations of the enzyme versus the absorbance they are measured at the end of the Elisa.

And not just that they have made the, perform the experiment or repeated the experiment many times. So, all those repeat data are there. Let me first read that data and then I will explain how the data has been arranged by them. So, I will read it, this is a default data set, so, I will store that in d. So, let us open the d, let us see what we have. It has 176 observation, the first column is for run, second column is for concentration and the third one is density, that is the absorbance or optical density actually.

If you look into the first two reading, the run one, the concentration of these two rows are same. So, that means this is run one, the first experiment, they have multiple repeat, those repeats are called run and for each repeat there are around eight concentration of the enzyme used for the assay and for each concentration they have measured the reading, taken the reading twice.

So that is why you have two reading for concentration 0.0482, two reading for concentration 0.195, something like that. And then you if you look go down you can see there are 8, 9 and 10, up to 11 different repeats. So, they are independent runs, independent repeats, to increase the reproducibility of the results they have repeated it so many times.

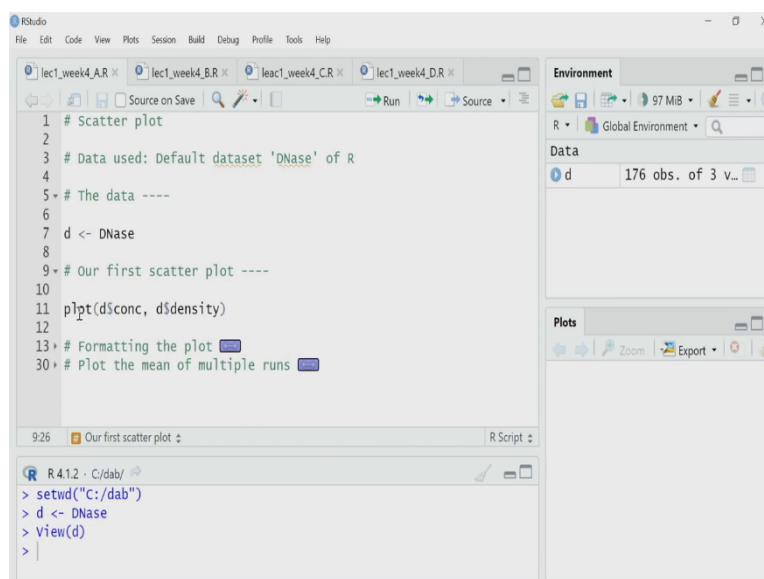
(Refer Slide Time: 3:33)

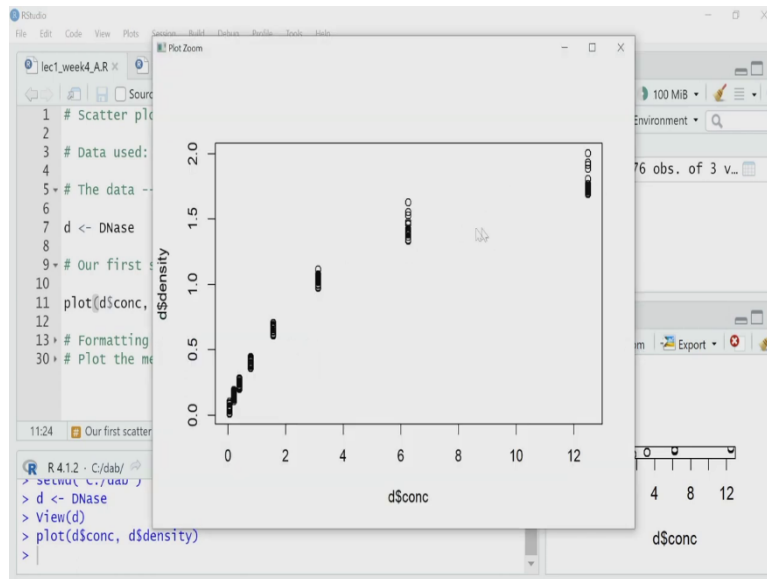


So, what I want? I want to plot this concentration versus density plot, concentration should be in horizontal line, density should be on the vertical line. And as you can see this is a pairwise data, for each value of concentration, I have a reading for absorbance or optical density.

So, if you remember, scatter plot is usually used for visualizing these type of pairwise data, where I have two variables and they are having values are smoothly changing. So, here concentration values are smoothly changing, along with that the optical density of absorbance is also smoothly changing and I have a pairwise data set for that.

(Refer Slide Time: 4:13)





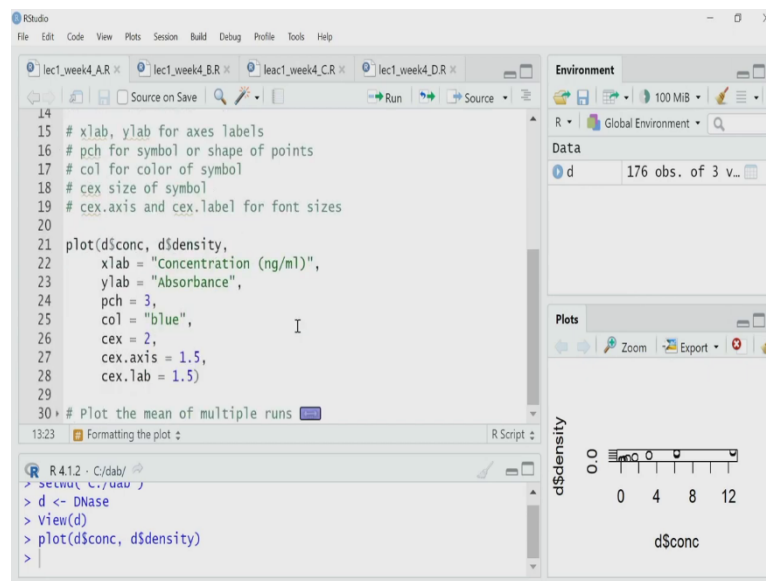
`d <- DNase`

`plot(d$conc, d$density)`

Let me go back to the script. They let us first make a scratch scatterplot. So, what I will use? I will use the plot function. Plot function is a very powerful function in R, it can do lots of thing, it can generate lots of type of plot in different fashion. So, what it will do here? By default, I have given the concentration and density as x and y, the horizontal axis variable and the vertical axis variable here, by putting d dollar sign conc, that is the concentration that is in the second column as the x and density as the second one.

So, what it will do? By default it will create a scatterplot. Let me zoom the figure. So, you can see all the data points has been plotted. You may not be expecting this type of plot, I can understand, maybe you want to plot the mean of each repeated measurement or something like that, but we will go to that in a moment. So, this is the basic plot I have got. What I will do now? I will do some amount of formatting and I will explain what type of formatting you should do while creating a plot and what issues you should always pay attention to.

(Refer Slide Time: 5:31)



`plot(d$conc, d$density,`

`xlab = "Concentration (ng/ml)",`

`ylab = "Absorbance",`

`pch = 3,`

`col = "blue",`

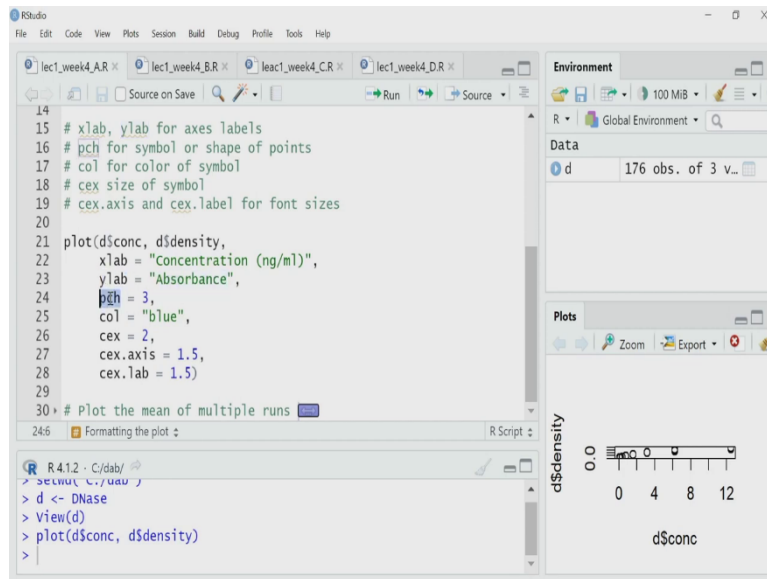
`cex = 2,`

`cex.axis = 1.5,`

`cex.lab = 1.5)`

So, to do formatting I have to use certain arguments. So, plot has lots of arguments that we can use, I am just using few of them. If you want to know all of them, please look into the help file or the documentation of R for all the arguments that you can use to format the graph. So, obviously, x and y is the defined at the very beginning, then the second argument that I use, I use that label the x axis, so, x lab is equal to concentration in nanogram per ml, y label the vertical axis label is y lab and it is absorbance.

(Refer Slide Time: 6:11)



`plot(d$conc, d$density,`

`xlab = "Concentration (ng/ml)",`

`ylab = "Absorbance",`

`pch = 3,`

`col = "blue",`

`cex = 2,`

`cex.axis = 1.5,`

`cex.lab = 1.5)`

Pch is the name of the symbol R has some inbuilt symbol, if you go to the documentation you can get a list for that I am using three. So, it will be a cross or a plus sign in a way you can say. Color of that symbol I want blue, the size of the symbol I have written Cex equal to 2, by default R will consider a size of the symbol, I want to increase it two times. So, I have written Cex equal to 2.

(Refer Slide Time: 6:42)

```

14
15 # xlab, ylab for axes labels
16 # pch for symbol or shape of points
17 # col for color of symbol
18 # cex size of symbol
19 # cex.axis and cex.label for font sizes
20
21 plot(d$conc, d$density,
22      xlab = "Concentration (ng/ml)",
23      ylab = "Absorbance",
24      pch = 3,
25      col = "blue",
26      cex = 2,
27      cex.axis = 1.5,
28      cex.lab = 1.5)
29
30 # Plot the mean of multiple runs

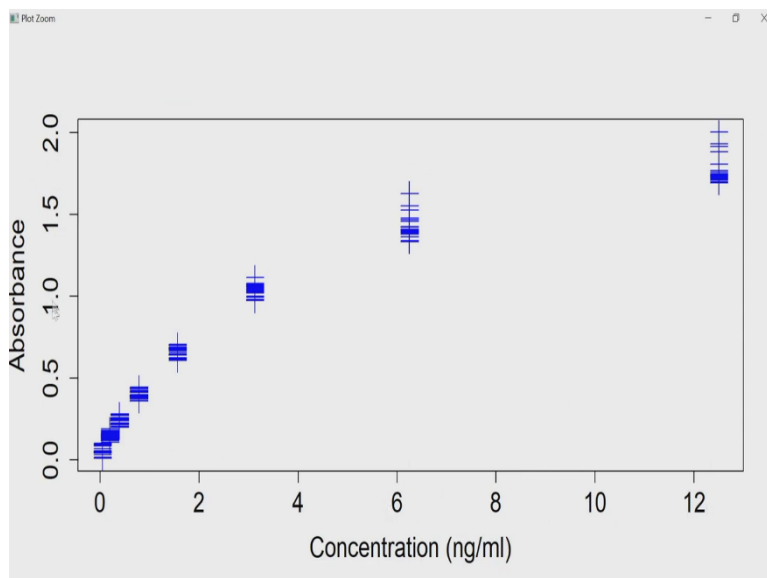
```

Environment: R, Global Environment, 100 MB
 Data: d, 176 obs. of 3 v...
 Plots: d\$density

```

R 4.1.2 · C:/dab/
> setwd("C:/dab")
> d <- DNase
> view(d)
> plot(d$conc, d$density)
>

```



```

plot(d$conc, d$density,
     xlab = "Concentration (ng/ml)",
     ylab = "Absorbance",
     pch = 3,
     col = "blue",
     cex = 2,
     cex.axis = 1.5,
     cex.lab = 1.5)

```

Then I have said cex dot axis equal to 1.5, that means, see in the axis I have the labels, 1, 2, 3, 4 something like that for the concentrations and also the absorbance. So, those are fonts and

R use a default font size for that, I want to change that font size by increasing it to 1.5 times. Similarly, the labels the x label, y label that we are defining here, I want that the size should be 1.5 times of the default one.

Here I plot it, and then I will explain these things again. So, let me zoom out. So, if you can see now, the labels and as well as the tick labels are very legible. This is one point we have to keep in mind, while we are creating a plot that all the text and numbers should be legible to the person who is reading or seeing that.

I should not use very small font that somebody cannot understand what is written there or I should not use a huge font that the font become predominant rather than the symbols and the other aspect of the data. So, judiciously I have to always choose the font size. Second important thing or rather I will say the most important thing, while you are creating a plot is that, please always label your axes a graph without axis is not label is meaningless.

For example, here the horizontal axis must be labeled as concentration and I will insist always put the unit up that concentration used for example, here it is nanogram per ml. Whereas the vertical axis we have written absorbance, I could have written it as an arbitrary unit in the bracket. Usually people know that absorbance is not an absolute reading.

So, they know there is will be arbitrary unit of that. I could have added here the wavelength at which the absorbance has been taken. The next thing is that symbols and color should be chosen in such a way so that I can easily understand the pattern in the data. I should not choose a color or a symbol that obscured the pattern of the data as well as the aesthetic of the visualization. So, I have plotted it.

Now, once you have seen this plot, you must be thinking that usually you do not do it this way, you do not plot all the data point at scatter point and visualize for this type of experimental data, what you prefer is, that you want the mean of each repeated runs, each repeated measurement. So, for each concentration, you have 11 run, and for each run there are two repeats. So, you have so many reading for each of the concentration.

So, you want to take a mean have that and plot that. So to do that, I have to calculate the mean of it. So, how I will do?

(Refer Slide Time: 9:38)

The screenshot shows the RStudio interface with the following R code in the editor:

```
1 # Scatter plot
2
3 # Data used: Default dataset 'DNase' of R
4
5 # The data
9 # Our first scatter plot
13 # Formatting the plot
30 # Plot the mean of multiple runs |---
31
32 # Calculate the mean absorbance for
33 # each concentration using aggregate()
34
35 d.avg <- aggregate(density ~ conc, d, mean)
36
37 # Plot conc vs mean absorbance
38
39 plot(d.avg$conc, d.avg$density,
```

The console shows the execution of the plot command with options:

```
R 4.1.2 - C:/dab/
+ col = "blue",
+ cex = 2,
+ cex.axis = 1.5,
+ cex.lab = 1.5)
>
```

The Environment pane shows a data object 'd' with 176 observations and 3 variables. The Plots pane displays a scatter plot of Absorbance vs Concentration (ng/ml) with blue points and error bars.

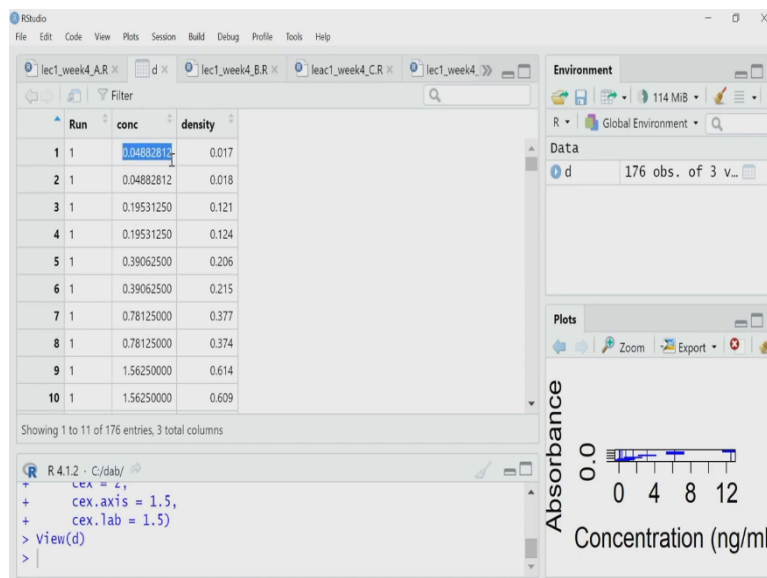
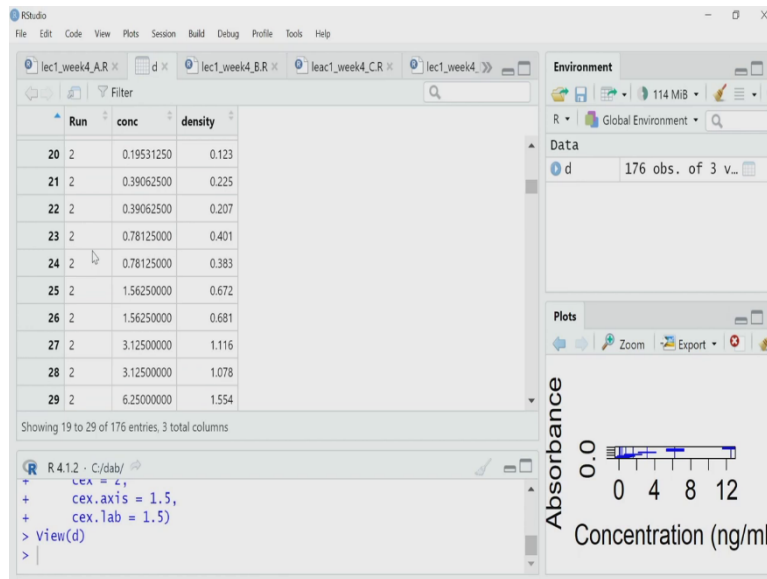
The screenshot shows the RStudio interface with the 'View(d)' window open, displaying a table of the data object 'd':

Run	conc	density
1	0.04882812	0.017
2	0.04882812	0.018
3	0.19531250	0.121
4	0.19531250	0.124
5	0.39062500	0.206
6	0.39062500	0.215
7	0.78125000	0.377
8	0.78125000	0.374
9	1.56250000	0.614
10	1.56250000	0.609

The console shows the command used to view the data:

```
R 4.1.2 - C:/dab/
+ cex = 2,
+ cex.axis = 1.5,
+ cex.lab = 1.5)
> View(d)
>
```

The Environment pane shows the data object 'd' with 176 observations and 3 variables. The Plots pane displays the same scatter plot of Absorbance vs Concentration (ng/ml) as in the previous screenshot.



`d.avg ← aggregate(density ~ conc, d, mean)`

Now, if I go back into the data d, I can see that in this case, the data is stacked based on the run, the first run data is there, then it is you have the second run data and these goes on. And also in each run you have for each concentration you have two readings. So, I have to somehow sort out this and calculate the mean.

(Refer Slide Time: 10:04)

RStudio

```

1 # Scatter plot
2
3 # Data used: Default dataset 'DNase' of R
4
5 # The data
9 # Our first scatter plot
13 # Formatting the plot
30 # Plot the mean of multiple runs ----
31
32 # Calculate the mean absorbance for
33 # each concentration using aggregate()
34
35 d.avg <- aggregate(density ~ conc, d, mean)
36
37 # Plot conc vs mean absorbance
38
39 plot(d.avg$conc, d.avg$density,
35:19 Plot the mean of multiple runs

```

Environment

R • Global Environment

Data

d 176 obs. of 3 v...

Plots

Absorbance

Concentration (ng/ml)

R 4.1.2 · C:/dab/

```

+ cex = 2,
+ cex.axis = 1.5,
+ cex.lab = 1.5)
> View(d)
>

```

RStudio

lec1_week4_AR x d x lec1_week4_BR x lec1_week4_CR x lec1_week4_>

Run	conc	density
1	0.04882812	0.017
2	0.04882812	0.018
3	0.19531250	0.121
4	0.19531250	0.124
5	0.39062500	0.206
6	0.39062500	0.215
7	0.78125000	0.377
8	0.78125000	0.374
9	1.56250000	0.614
10	1.56250000	0.609

Showing 1 to 11 of 176 entries, 3 total columns

Environment

R • Global Environment

Data

d 176 obs. of 3 v...

Plots

Absorbance

Concentration (ng/ml)

R 4.1.2 · C:/dab/

```

+ cex = 2,
+ cex.axis = 1.5,
+ cex.lab = 1.5)
> View(d)
>

```

RStudio

```

1 # Scatter plot
2
3 # Data used: Default dataset 'DNase' of R
4
5 # The data
9 # Our first scatter plot
13 # Formatting the plot
30 # Plot the mean of multiple runs ----
31
32 # calculate the mean absorbance for
33 # each concentration using aggregate()
34
35 d.avg <- aggregate(density ~ conc, d, mean)
36
37 # Plot conc vs mean absorbance
38
39 plot(d.avg$conc, d.avg$density,
35:30 Plot the mean of multiple runs

```

Environment

R • Global Environment

Data

d 176 obs. of 3 v...

Plots

Absorbance

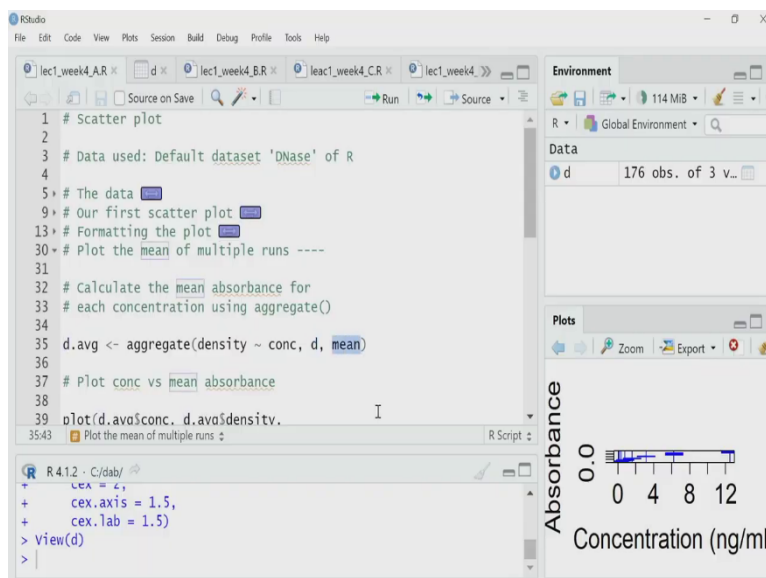
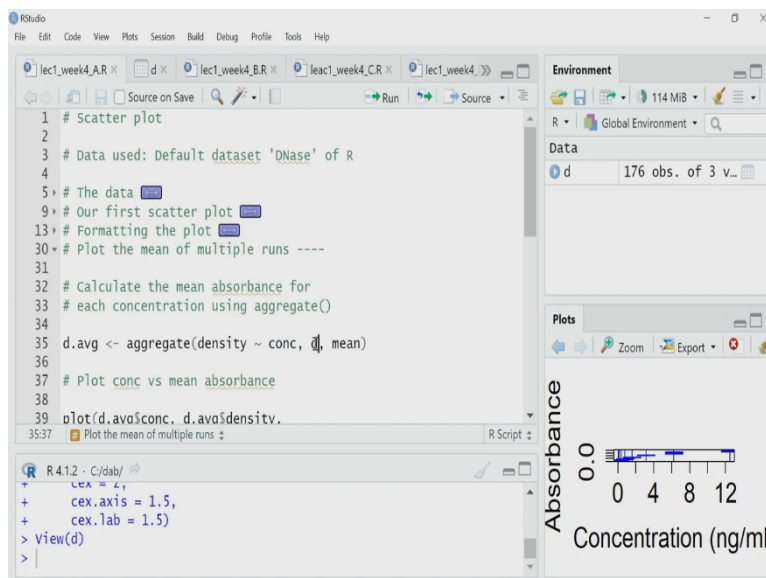
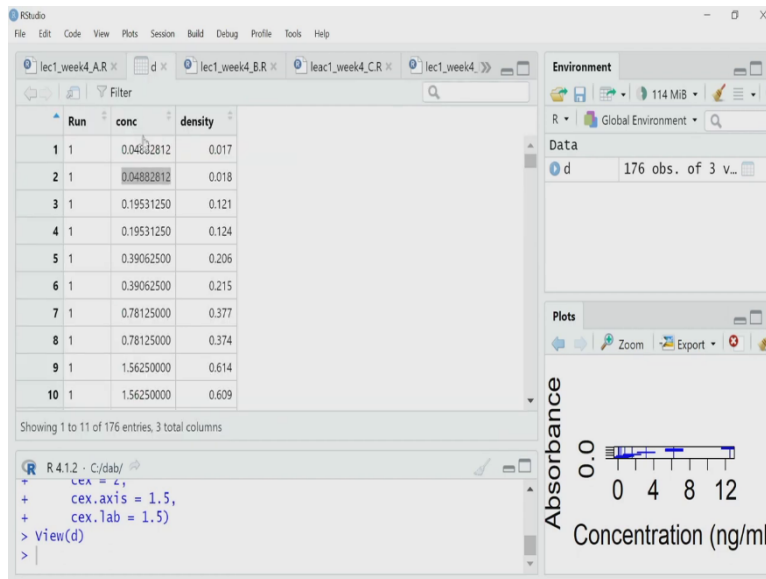
Concentration (ng/ml)

R 4.1.2 · C:/dab/

```

+ cex = 2,
+ cex.axis = 1.5,
+ cex.lab = 1.5)
> View(d)
>

```

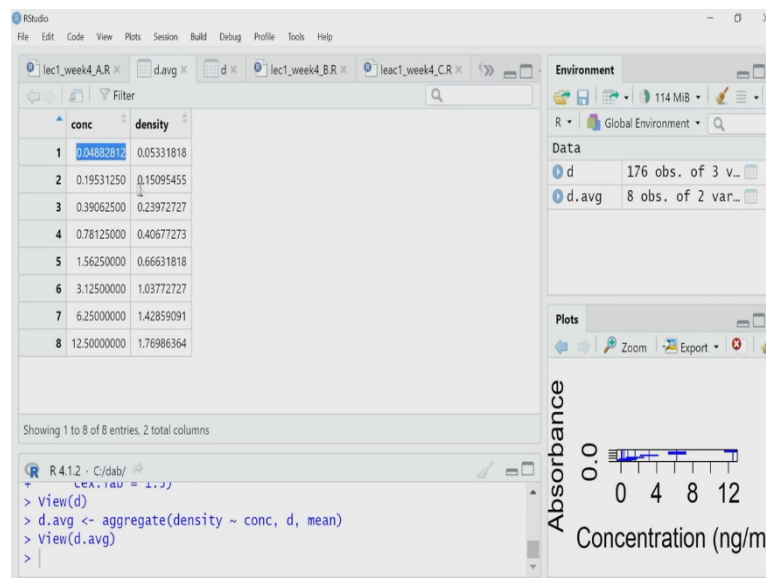
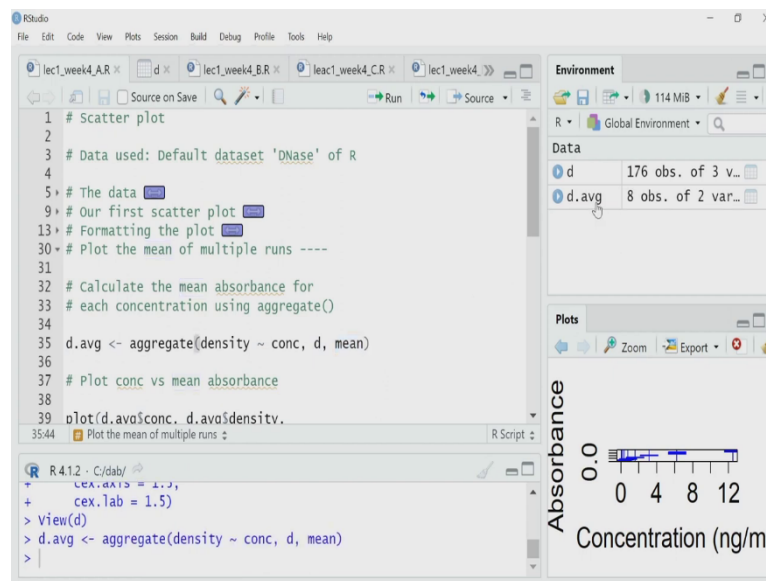



```
d.avg ← aggregate(density ~ conc, d, mean)
```

So, do that, what I am using here, the function called aggregate, I want R to aggregate the data based on what? I want to aggregate the data in density data that second column, density data, based on the value of the concentration conc, which is that the second column, name of the second column a second variable is conc, concentration.

So, I want to aggregate the density data with respect to or based on the concentration data from where from that d variable which has the all the data and while aggregating it will you calculate the mean, that is why the mean is written as the last argument.

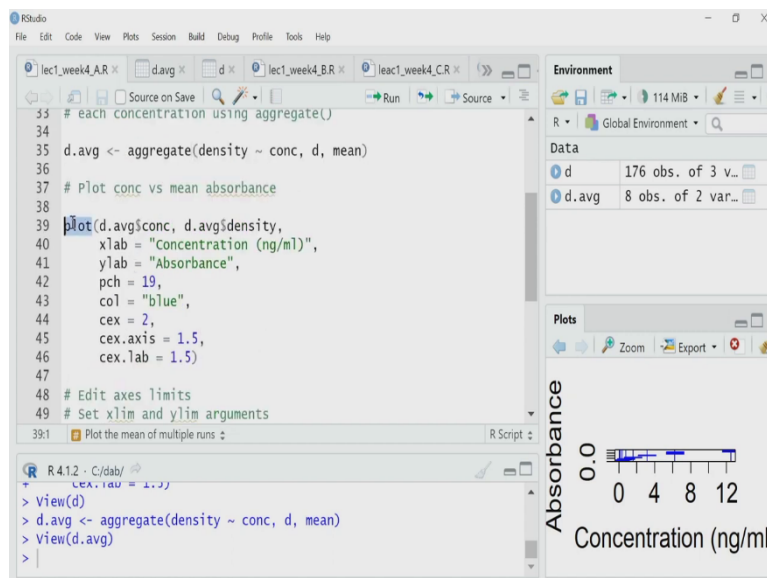
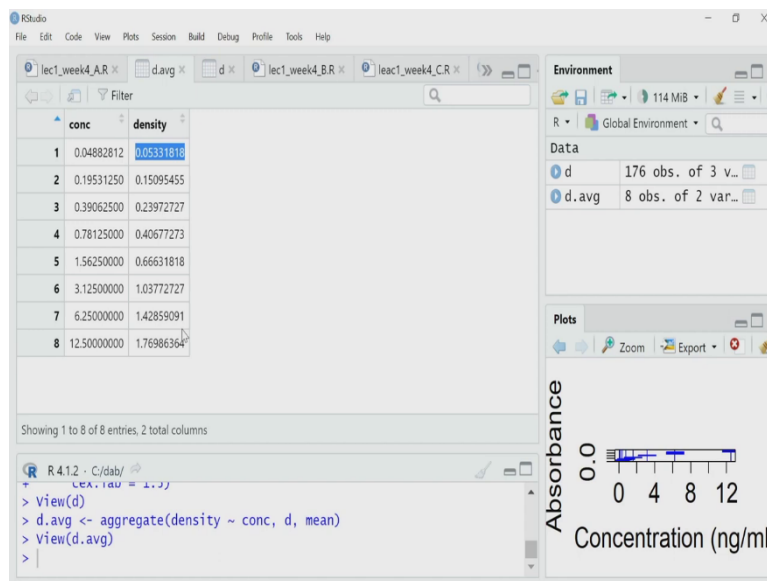
(Refer Slide Time: 10:49)



`d.avg ← aggregate(density ~ conc, d, mean)`

So, if I execute this and then let me see the I have created d average variable, where this aggregated data is stored, let me open that you can now see you have eight data points, because I have 8 concentration in this experiment. So 0.0488, up to 12.5. And the second column is actually now the mean density that we have got through the repeated measurement, the mean of those repeated measurement for that particular concentration in that, is in that column.

(Refer Slide Time:11:23)



`plot(d.avg$conc, d.avg$density,`

`xlab = "Concentration (ng/ml)",`

`ylab = "Absorbance",`

`pch = 19,`

`col = "blue",`

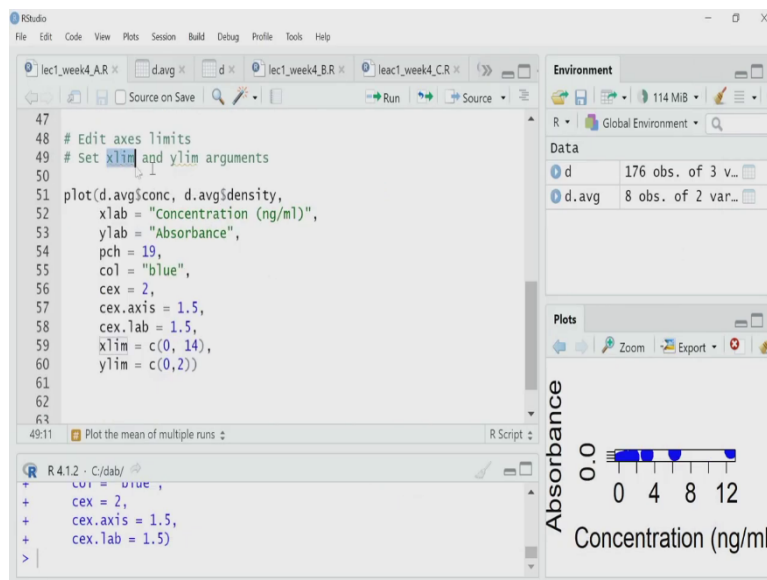
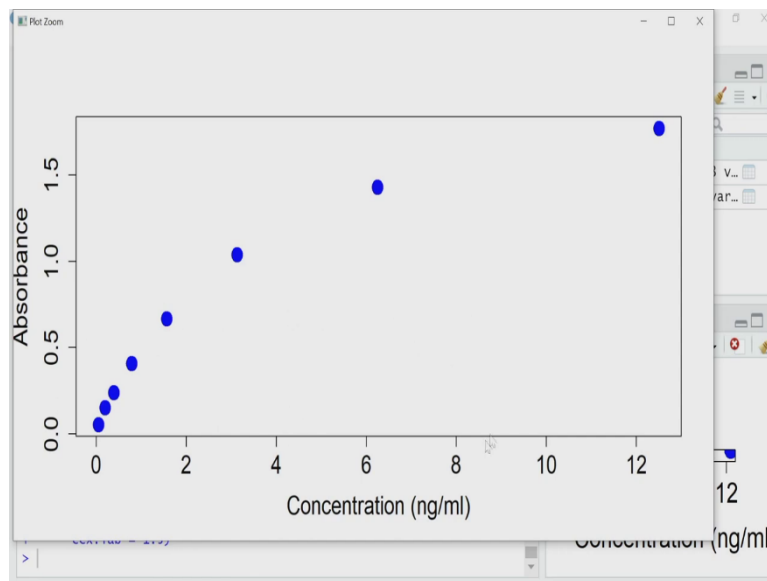
`cex = 2,`

`cex.axis = 1.5,`

`cex.lab = 1.5)`

So, now, I want to plot this concentration versus density plot or concentration versus mean absorbance plot. So, to do that, what I will do, I will again use the plot option, it will be a scatter plot, everything remains same only thing I have done here, now, I have changed the symbol, I am using 19, pch equal to 19 so that it will be a filled circle and the color will be blue.

(Refer Slide Time: 11:53)



`plot(d.avg$conc, d.avg$density,`

`xlab = "Concentration (ng/ml)",`

`ylab = "Absorbance",`

`pch = 19,`

`col = "blue",`

`cex = 2,`

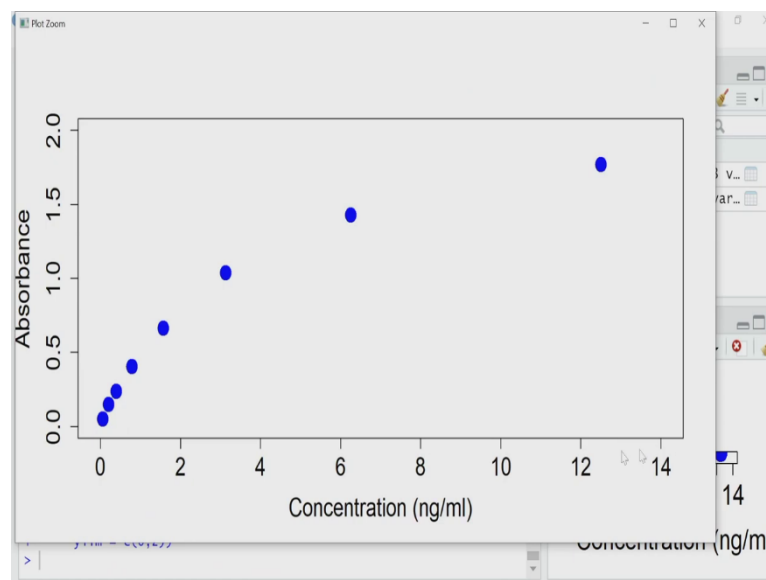
`cex.axis = c(0, 14),`

`cex.lab = c(0, 2))`

So, here is the plot. Now you can see it looks neat, and you must be recognizing it because you may have created this type of plot earlier for your lab work. So, these are showing me the mean absorbance for each of the concentration. Now, if you carefully notice what has happened here, I have some issue with respect to the scale of the axis, we start with the vertical axis, we are starting from 0.0, we have the tick mark up to 1.5. But you have a data which is bigger than 1.5, but I do not have any tick mark for that or I hope that.

Similarly, my reading is slightly more than 12. But I do not have any tick on the right hand side in the horizontal axis. So I want to correct, to do that what I will do, I will use two argument `x lim, y lim`. So, these will limit the scale. So, I am putting `x lim` equal to `c 0 comma 14`, that means I want the ticks to start from 0 up to 14 for the horizontal axis, the concentration axis, whereas `y lim` is equal to `see 0 comma 2`. So, I want it up to 0 to 2.

(Refer Slide Time: 13:11)

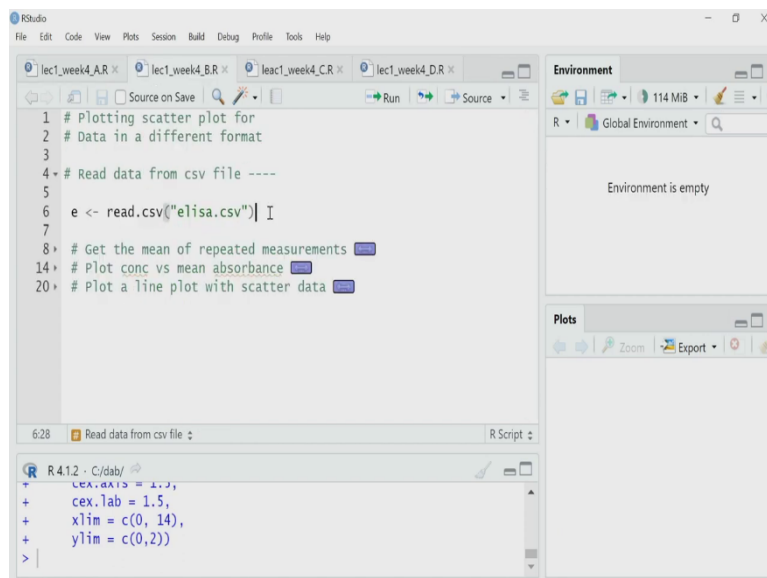
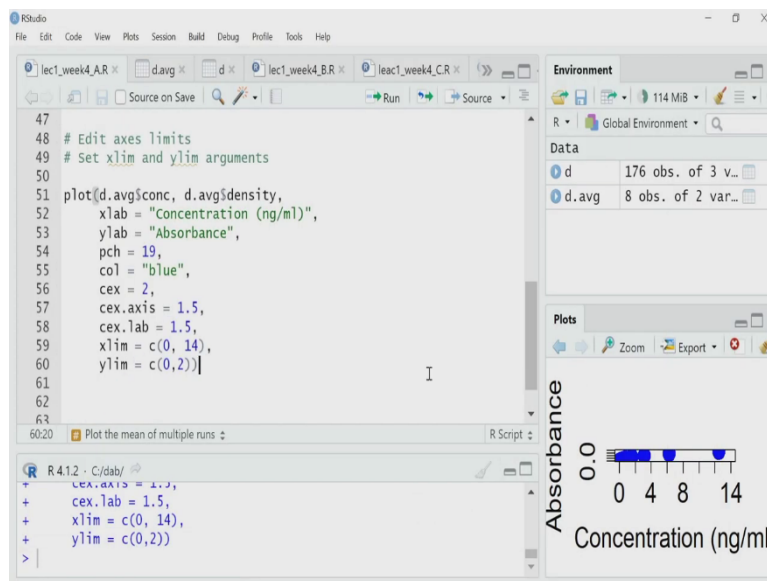


So, if I now plot this one, this one is much better. Now I have all the tick label and all my data are within this scale for both vertical and horizontal axis. So, what I have shown here is, the mean data and you must be wondering that where is the error bar, because every time you plot these type of average data, mean data, we are supposed to put the error bar, which in this case can be standard deviation.

Now, by default, R does not have this plot of function does not have any way to create the error bar. That is quite unfortunate; there is a roundabout for to do that, that is a bit complicated, I will not go into it. In future lectures, we will learn about some advanced function, advanced packages for visualization of data and for creating graphics, which will be

publication quality. So in that case, we will be able to easily put the error bar and we will discuss that at that time.

(Refer Slide Time: 14:07)



	conc	abs.1	abs.2	abs.3	abs.4
1	0.04882812	0.017	0.045	0.011	0.035
2	0.19531250	0.121	0.137	0.118	0.132
3	0.39062500	0.206	0.225	0.200	0.224
4	0.78125000	0.377	0.401	0.364	0.385
5	1.56250000	0.614	0.672	0.620	0.658
6	3.12500000	1.019	1.116	0.979	1.060
7	6.25000000	1.334	1.554	1.424	1.425
8	12.50000000	1.730	1.932	1.740	1.750

```

R 4.1.2 · C:/dab/
+ xlim = c(0, 14),
+ ylim = c(0, 2))
> e <- read.csv("elisa.csv")
> View(e)

```

```

plot(d.avg$conc, d.avg$density,
     xlab = "Concentration (ng/ml)",
     ylab = "Absorbance",
     pch = 19,
     col = "blue",
     cex = 2,
     cex.axis = c(0, 14),
     cex.lab = c(0, 2))

```

Now, I will move into another type of data set. What we have here, I have rearranged the same data in a different way because in some of your experiments, your data may not be arranged that way we have in this particular data file. So, I have created a data file using the DNS data, part of it to show that maybe your own data may be arranged in this particular fashion.

So, that is a Elisa dot csv file, I will read it and then explain what is the data pattern format. So, this may be usually the way you may store data in your excel sheet when you are doing these type of experiments. So, you have the first column is the concentration. So, you have eight concentration and you have taken multiple reading for the same sample or you may have repeated the experiment with the same concentration.

(Refer Slide Time: 14:54)

	conc	abs.1	abs.2	abs.3	abs.4
1	0.04882812	0.017	0.045	0.011	0.035
2	0.1953125	0.121	0.137	0.118	0.15
3	0.390625	0.206	0.225	0.200	0.224
4	0.78125	0.377	0.401	0.364	0.385
5	1.5625	0.614	0.672	0.620	0.658
6	3.125	1.019	1.116	0.979	1.060
7	6.25	1.334	1.554	1.424	1.425
8	12.5	1.730	1.932	1.740	1.750

```

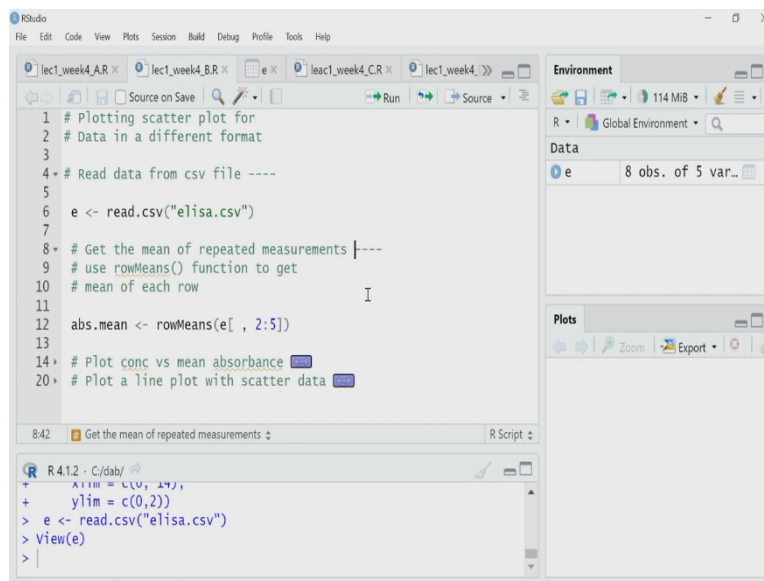
R 4.1.2 · C:/dab/
+ xlim = c(0, 14),
+ ylim = c(0, 2))
> e <- read.csv("elisa.csv")
> View(e)
>

```

`e ← read.csv("elisa.csv")`

So, all those repeats are now in stated columns. So, the first observation is a for second column, second observations are in the second column, and so on. So for each of these concentration, I have four readings. This may be the way you have your data. And you now you want to create the scatterplot. So, again, this is pairwise data. Again, rather than plotting all the data points, what you will prefer to calculate the average of that of these all these observation four observation, and then plot them as a scatterplot.

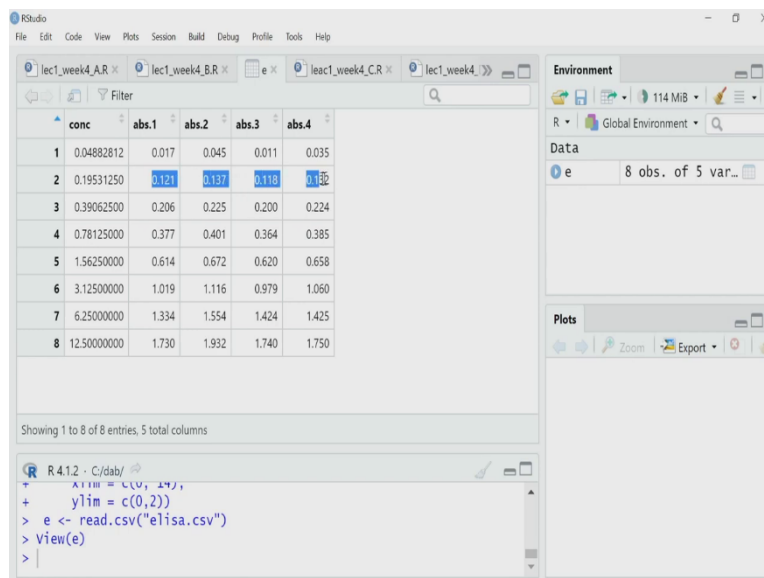
(Refer Slide Time: 15:31)



```
1 # Plotting scatter plot for
2 # Data in a different format
3
4 # Read data from csv file ----
5
6 e <- read.csv("elisa.csv")
7
8 # Get the mean of repeated measurements |----
9 # use rowMeans() function to get
10 # mean of each row
11
12 abs.mean <- rowMeans(e[, 2:5])
13
14 # Plot conc vs mean absorbance
15
20 # Plot a line plot with scatter data
```

Environment: 114 MiB, Global Environment, Data: e (8 obs., 5 var...), Plots: (empty)

```
R 4.1.2 - C:/dab/
+ xlim = c(0, 14),
+ ylim = c(0, 2)
> e <- read.csv("elisa.csv")
> View(e)
>
```



	conc	abs.1	abs.2	abs.3	abs.4
1	0.04882812	0.017	0.045	0.011	0.035
2	0.19531250	0.121	0.137	0.118	0.132
3	0.39062500	0.206	0.225	0.200	0.224
4	0.78125000	0.377	0.401	0.364	0.385
5	1.56250000	0.614	0.672	0.620	0.658
6	3.12500000	1.019	1.116	0.979	1.060
7	6.25000000	1.334	1.554	1.424	1.425
8	12.50000000	1.730	1.932	1.740	1.750

Showing 1 to 8 of 8 entries, 5 total columns

```
R 4.1.2 - C:/dab/
+ xlim = c(0, 14),
+ ylim = c(0, 2)
> e <- read.csv("elisa.csv")
> View(e)
>
```

`e ← read.csv("elisa.csv")`

`abs.mean ← rowMeans(e[, 2:5])`

So, I have to get the mean of these four observations, I have four repeats absorbance one after absorbance three, for each concentration, I have to take these values and calculate the mean of that, to get the mean, what I will do in this case, I will use the rowMeans functions. rowMeans function will take one row at a time and calculate the mean of the data in that row.

(Refer Slide Time: 15:54)

The screenshot shows the RStudio interface with the following code in the editor:

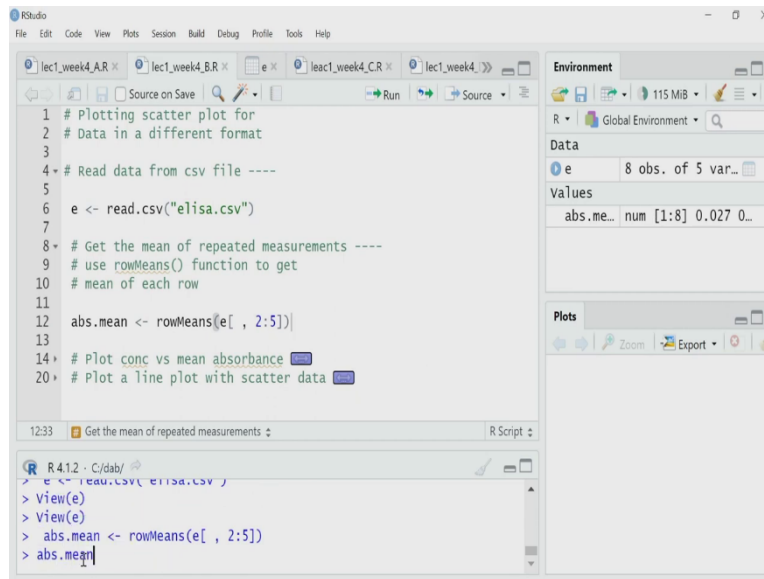
```
1 # Plotting scatter plot for  
2 # Data in a different format  
3  
4 # Read data from csv file ----  
5  
6 e <- read.csv("elisa.csv")  
7  
8 # Get the mean of repeated measurements ----  
9 # use rowMeans() function to get  
10 # mean of each row  
11  
12 abs.mean <- rowMeans(e[, 2:5])  
13  
14 # Plot conc vs mean absorbance  
20 # Plot a line plot with scatter data
```

The Environment pane on the right shows a data object 'e' with 8 observations and 5 variables. The console shows the execution of the code up to the `View(e)` command.

The screenshot shows the RStudio interface with the resulting data table displayed in the Environment pane:

	conc	abs.1	abs.2	abs.3	abs.4
1	0.04882812	0.017	0.045	0.011	0.035
2	0.19531250	0.121	0.137	0.118	0.132
3	0.39062500	0.206	0.225	0.200	0.224
4	0.78125000	0.377	0.401	0.364	0.385
5	1.56250000	0.614	0.672	0.620	0.658
6	3.12500000	1.019	1.116	0.979	1.060
7	6.25000000	1.334	1.554	1.424	1.425
8	12.50000000	1.730	1.932	1.740	1.750

The Environment pane shows the data object 'e' with 8 observations and 5 variables. The console shows the execution of the code up to the second `View(e)` command.



```
e ← read.csv("elisa.csv")
```

```
abs.mean ← rowMeans(e[, 2:5])
```

So, what is the argument for that, I want only the second column up to fifth column data. Because the first column is for concentration, I do not want to use that while calculating the mean. So, I will take these second column to third, fifth column, and all rows from this e and calculate the mean for each of them, each of the rows.

So, to do that, what I have done as the argument I am writing e in the square bracket, the index of row I have left empty, that means I want all in a rows, and then in the column index, I am writing 2 colon 5, that means take from column 2, up to column 5. And if I execute these, I will get a the rowMeans data and I am storing that in a abs dot mean variable. So, I can print that data here the mean values, you can see.

(Refer Slide Time: 16:50)

```

1 # Plotting scatter plot for
2 # Data in a different format
3
4 # Read data from csv file ----
5
6 e <- read.csv("elisa.csv")
7
8 # Get the mean of repeated measurements ----
9 Get the mean of repeated measurements

```

```

R 4.1.2 · C:/dab/
+ pcn = 19,
+ col = "blue",
+ cex = 2,
+ cex.axis = 1.5,
+ cex.lab = 1.5,
+ xlim = c(0, 14),
+ ylim = c(0, 2)
> e <- read.csv("elisa.csv")
> View(e)
> View(e)
> abs.mean <- rowMeans(e[, 2:5])
> abs.mean
[1] 0.02700 0.12700 0.21375 0.38175 0.64100 1.04350 1.43425 1.78800
> |

```

	conc	abs.1	abs.2	abs.3	abs.4
1	0.04882812	0.017	0.045	0.011	0.035
2	0.19531250	0.121	0.137	0.118	0.132
3	0.39062500	0.206	0.225	0.200	0.224
4	0.78125000	0.377	0.401	0.364	0.385
5	1.56250000	0.614	0.672	0.620	0.658
6	3.12500000	1.019	1.116	0.979	1.060
7	6.25000000	1.334	1.554	1.424	1.425
8	12.50000000	1.730	1.932	1.740	1.750

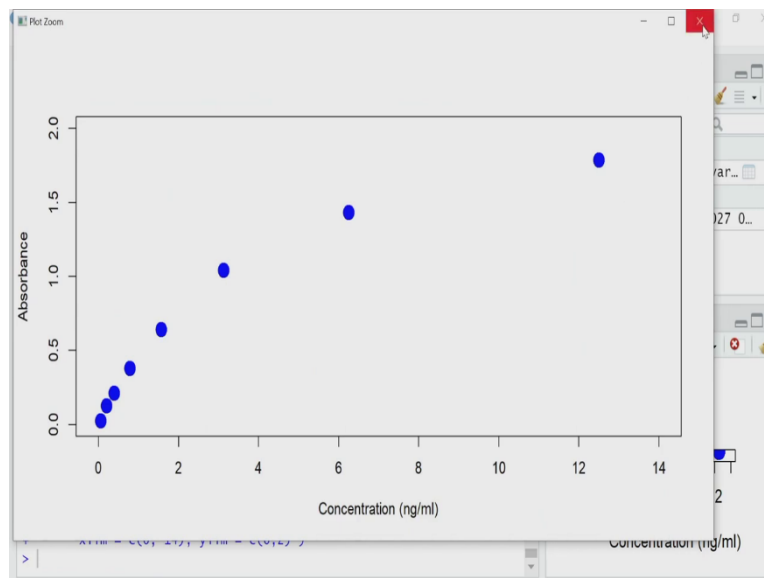
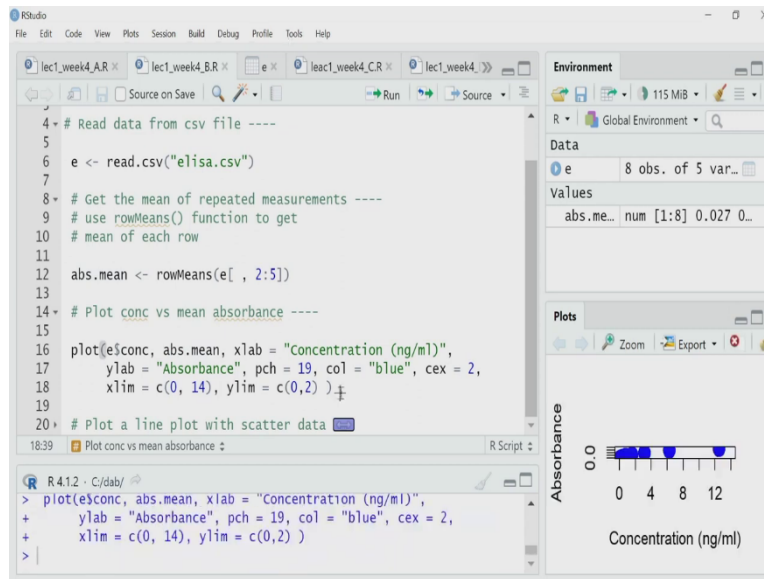
```

R 4.1.2 · C:/dab/
> abs.mean <- rowMeans(e[, 2:5])
> abs.mean
[1] 0.02700 0.12700 0.21375 0.38175 0.64100 1.04350 1.43425 1.78800
> |

```

I have eight values here, because I have eight concentration, I have now got the mean values. So, now, what I will do, I will plot these concentration versus mean absorbance. Again, I am using the same plot for scatter plot I will not go in detail of that, because we have already discussed how to use that and the arguments, only one important thing I have to remember, the x axis will be e dollar conc, because that is the concentration the first column and thus y argument, argument for y is abs dot mean because I have the mean data there.

(Refer Slide Time: 17:34)

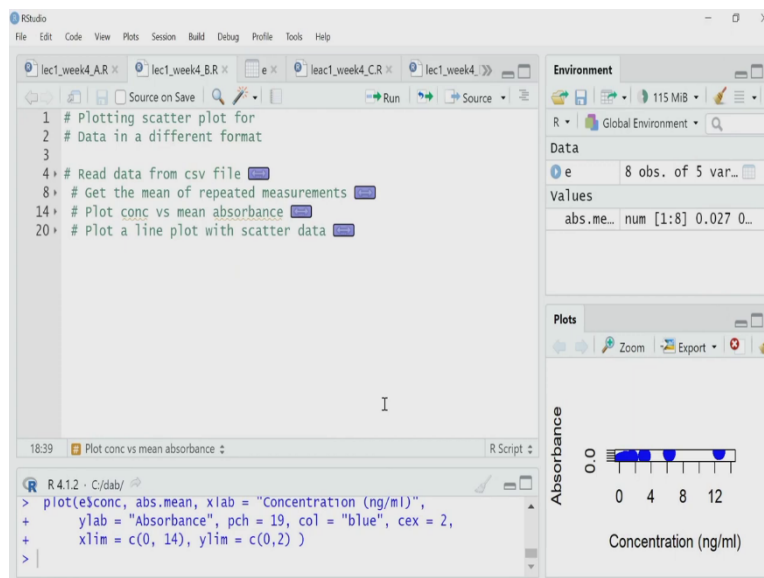
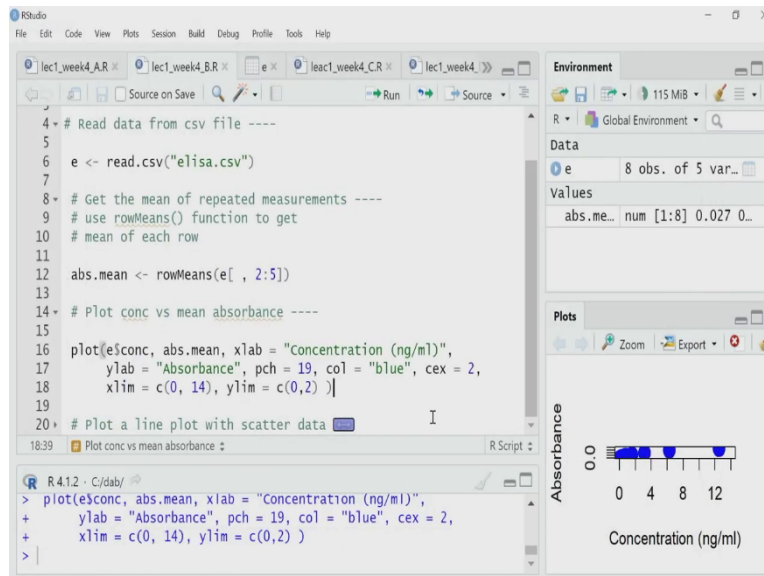


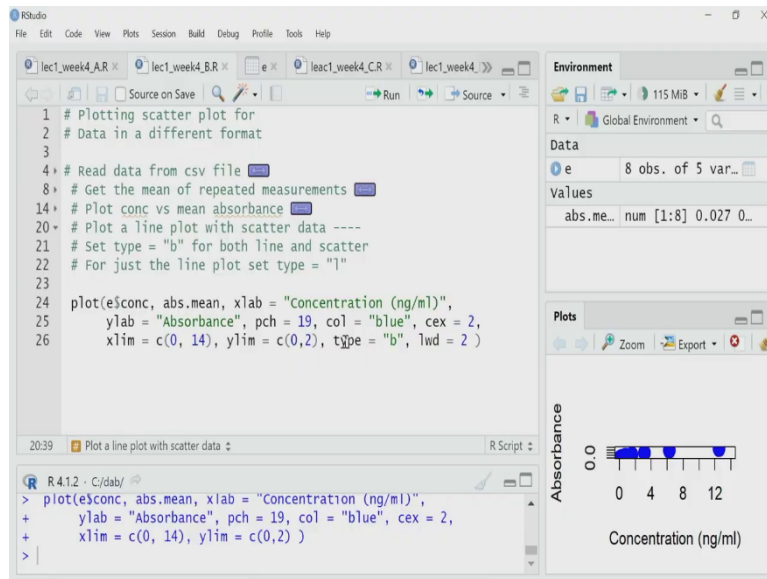
`plot(e$conc, abs.mean, xlab = "Concentration (ng/ml)",
ylab = "Absorbance", pch = 19, col = "blue", cex = 2,
xlim = c(0, 14), ylim = c(0, 2))`

So, if I run it, here is my plot, this is the plot same similar plot I have got earlier also. So, now, what I have done, we have discussed till now about scatterplot. I have plotted a scatter plot for the same data, but arranged in a different format. And I have initially plotted all the data points at scatter points and then I have plotted the mean. Now, you may be wondering that many a time we prefer to join these points with lines or sometimes we simply do not put these points as a symbol rather we join them as simpler lines, and we only do the line plot.

So, I can have two types of line plot line plot where we do not have any symbol for data point only the line and other can be the scatter plot combined with line. So, what I will do now, I will create a line plot for the same data where there will be scattered points also the points will be there the symbol circle, field circle will be there for each data point as well as these data points will get connected by lines.

(Refer Slide Time: 18:44)

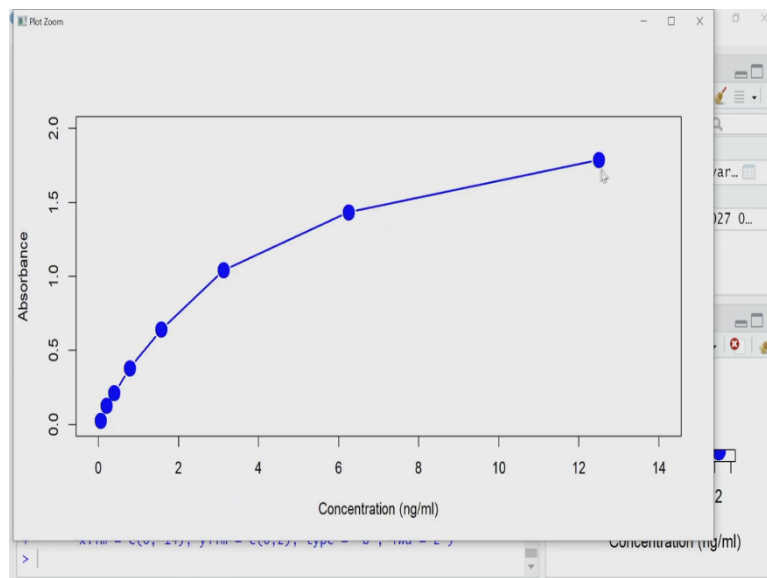
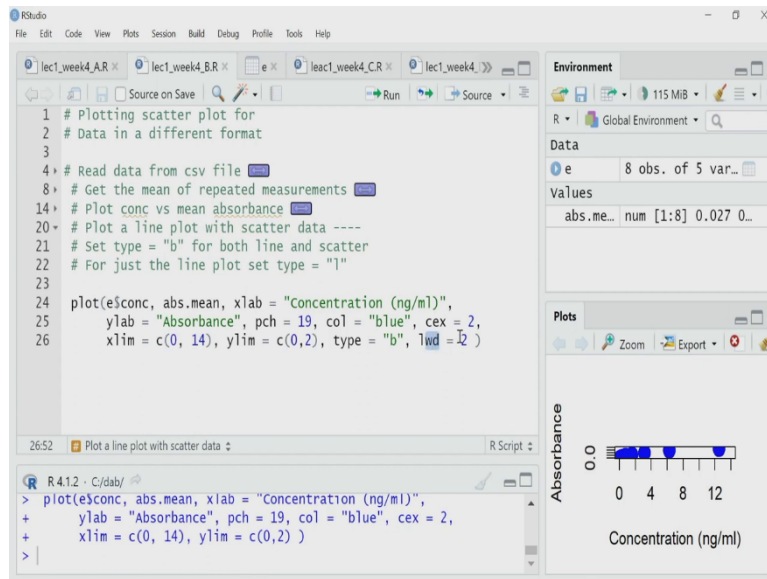




`plot(e$conc, abs.mean, xlab = "Concentration (ng/ml)",
ylab = "Absorbance", pch = 19, col = "blue", cex = 2,
xlim = c(0, 14), ylim = c(0, 2), type = "b", lwd = 2)`

Now, to do that, what I will do? I will use the same plot function, but as I said plot function is very powerful. So, what I will do? I will add few extra arguments, what the argument I have to add? In the plot function I will say, type is equal to b, b means both, by that plot function will understand I want both the symbols for each data point as well as I want to join them by a line. If you do not want those symbols, do not want to scatter points, then you put type equal to l lowercase l, that means you want on the line.

(Refer Slide Time: 19:26)



`plot(e$conc, abs.mean, xlab = "Concentration (ng/ml)",`

`ylab = "Absorbance", pch = 19, col = "blue", cex = 2,`

`xlim = c(0, 14), ylim = c(0, 2), type = "b", lwd = 2)`

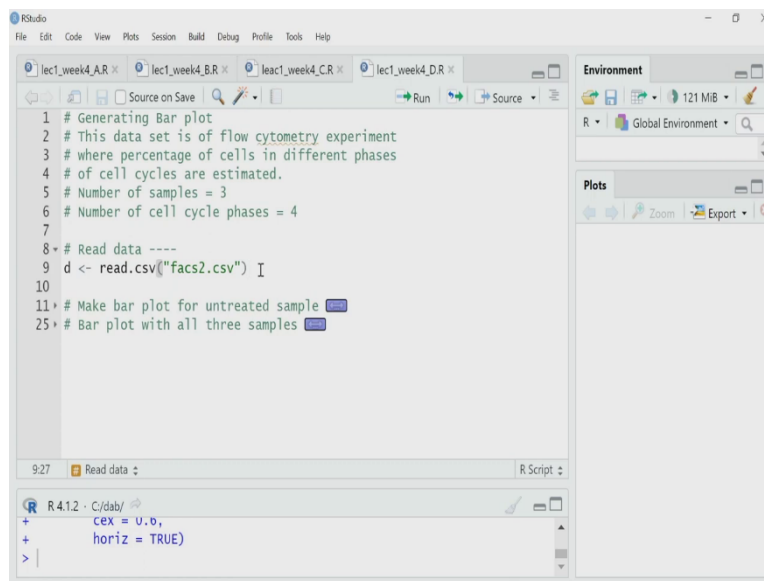
So, in this case I on both of them, and in a second you will be you will see how it looks and I want the width of the line `lwd` is equal to 2 here is the plot. So, you can see here now I have the scatter point and I have joined them by line segments. So, it is, it has both the scattered data points as well as the line. Now this one we should not confuse with data fitting, many times you may initially plot the scatter plot and then you may feed this data by a smooth line, which is a particular function.

Now, that is usually done when you are doing regression model, we will discuss those separately. Here what we are doing we are not adding any smooth line, we are just joining these data points by line segments. So, that is all for the scatter plot. Now, I will move to bar plot. bar plots are usually used when you have category based data and I will explain that with a particular example. To understand what is bar plot?

When to use a bar plot? And how to create a bar plot in R? I have a flow cytometry experiment data. Before I move into it, I want to highlight that for the scatter plot that I have shown earlier for my enzyme assay experiment, I can also use it the bar plot for that, but in that case, it does not add any extra value and it does not look aesthetically good also, that is why it is preferred that I will go for a scatterplot.

There are certain particular cases where scatter plot cannot do justice to my data, and in that case bar plot should be your first choice. So, for example, the flow cytometry data that I will discuss here is one such example where bar plot is a must, you cannot use a scatterplot. So, before I move into that data, let me load the data and explain what we have.

(Refer Slide Time: 21:21)



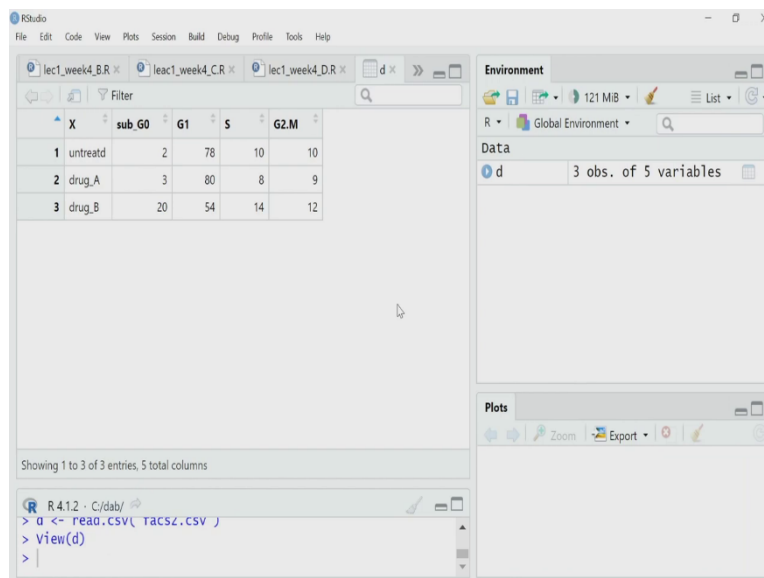
```
1 # Generating Bar plot
2 # This data set is of flow cytometry experiment
3 # where percentage of cells in different phases
4 # of cell cycles are estimated.
5 # Number of samples = 3
6 # Number of cell cycle phases = 4
7
8 # Read data ----
9 d <- read.csv("facs2.csv")
10
11 # Make bar plot for untreated sample
12 # Bar plot with all three samples
```

Environment: R, Global Environment, 121 MiB

Plots: Zoom, Export

R 4.1.2 · C:/dab/

```
+ cex = 0.6,
+ horiz = TRUE)
> |
```



Environment: R, Global Environment, 121 MiB

Data: d, 3 obs. of 5 variables

X	sub_G0	G1	S	G2.M	
1	untreatd	2	78	10	10
2	drug_A	3	80	8	9
3	drug_B	20	54	14	12

Showing 1 to 3 of 3 entries, 5 total columns

R 4.1.2 · C:/dab/

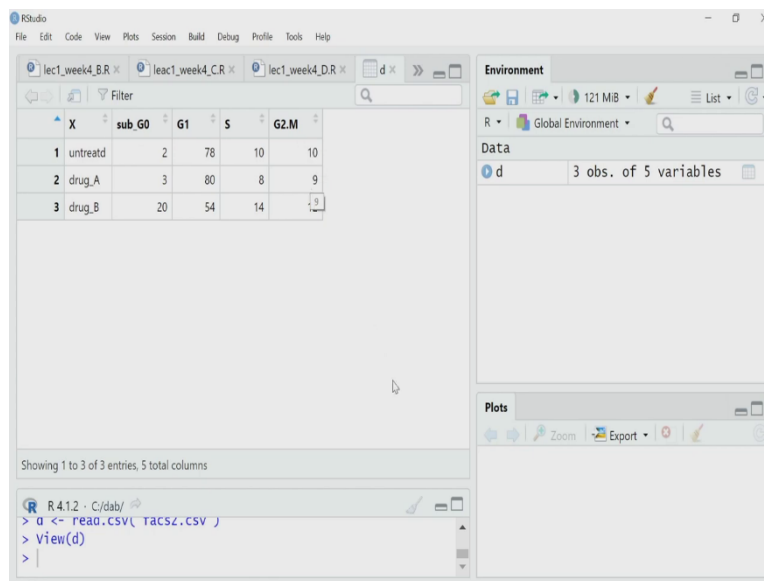
```
> d <- read.csv("facs2.csv")
> view(d)
> |
```

`d ← read.csv("facs2.csv")`

It is in this a facs2 dot csv file. So, let me read that first. Let us check the data. So, what I have in this experiment? What I am doing? I am checking whether a particular drug treatment can cause cell cycle arrest of some cells or not, I have two drug or you can imagine two doses of the drug, here I am labeling them as drug A and drug B.

And I have untreated cells. So, I have three sample untreated cell, cell treated with drug A and cell treated with drug B. And using flow cytometry, what I am doing? I am measuring percentage of cells in different phases of cell cycle, I hope you remember cell cycle

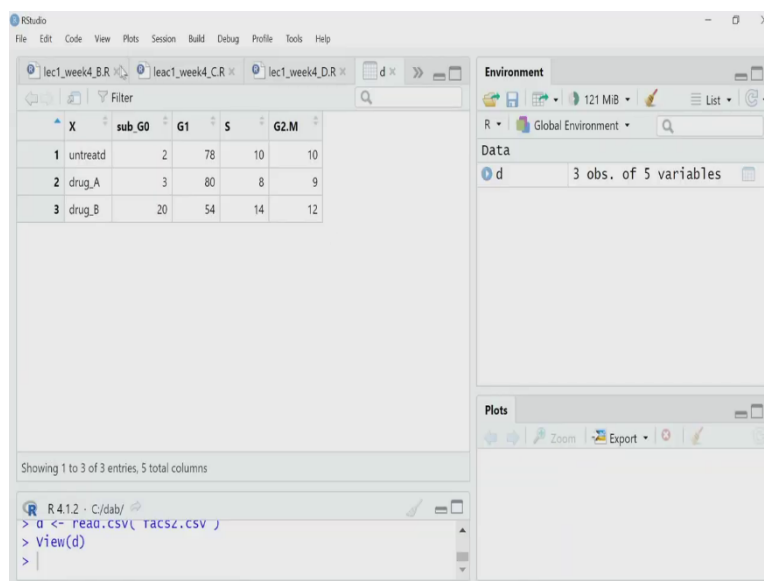
(Refer Slide Time: 22:12)



`d ← read.csv("facs2.csv")`

So, here, defined phases are sub G0, G1, S and G2 M. So, I have four cell cycle phases. And I have repeated this experiment, perform the flow cytometry, and the mean observation is shown here in this file. So, now I want to use a bar plot to represent data. So, first, what I will do? I will simply make a bar plot for this untreated cells, and each of the bar will correspond to each of these cell cycle phases.

(Refer Slide Time: 22:43)



```
8 # Read data ----
9 d <- read.csv("facs2.csv")
10
11 # Make bar plot for untreated sample ----
12
13 # Get the data and create a matrix
14
15 d.un <- d[1, 2:5]
16
17 d.un <- data.matrix(d.un)
18
19 # Plot the bar plot
20
21 barplot(d.un,
22         ylim = c(0,100),
23         ylab = "% of cells")
24
25 # Bar plot with all three samples
```

The screenshot shows the RStudio interface. The main editor window contains the R code above. The Environment pane on the right shows a variable 'd' with 3 observations and 5 variables. The Plots pane is empty. The console at the bottom shows the execution of the code, with the current line being 'Make bar plot for untreated sample'.

```
d.un ← d[1, 2:5]
```

```
d.un ← data.matrix(d.un)
```

```
barplot(d.un,
```

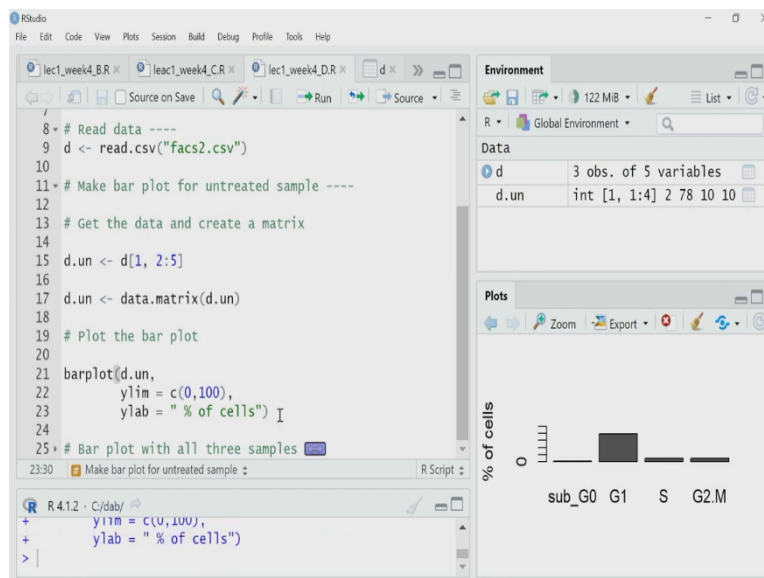
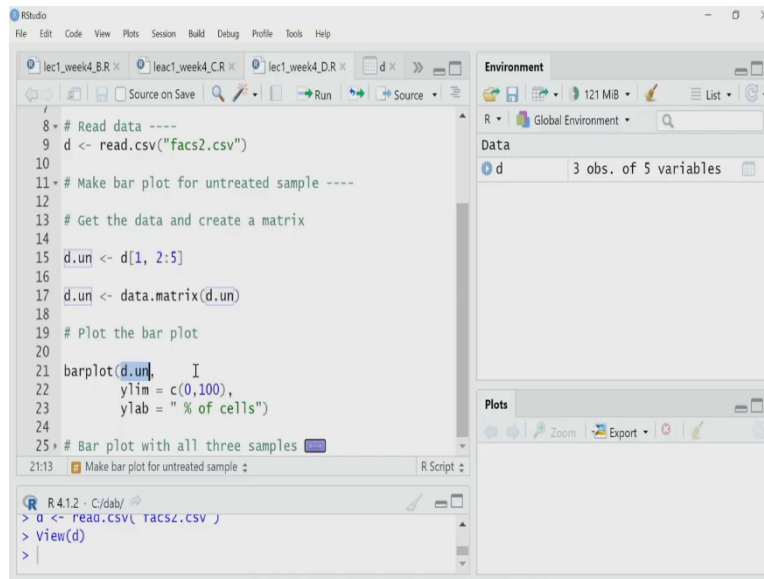
```
      ylim = c(0, 100),
```

```
      ylab = "% of cells")
```

To achieve that, what I will do? I will use the inbuilt bar plot function. But before I go into it, I have to do some transformation and processing of my data. So, what I want? I want the untreated samples. So, let me extract that. So d dot un is equal to d, first row, 2 to 5 column, so 2 colon 5. So, I am taking the first row of the data.

And I am taking from 2 to 5 because the first column has the Label, Label of the sample, I do not need that. And now to use bar plot, I have to convert this data frame into a matrix. To do that I am using data dot matrix function and using d dot un as a argument and I am saving that matrix again in dot d dot un.

(Refer Slide Time: 23:44)



```
d.un ← d[1, 2:5]
```

```
d.un ← data.matrix(d.un)
```

```
barplot(d.un,
```

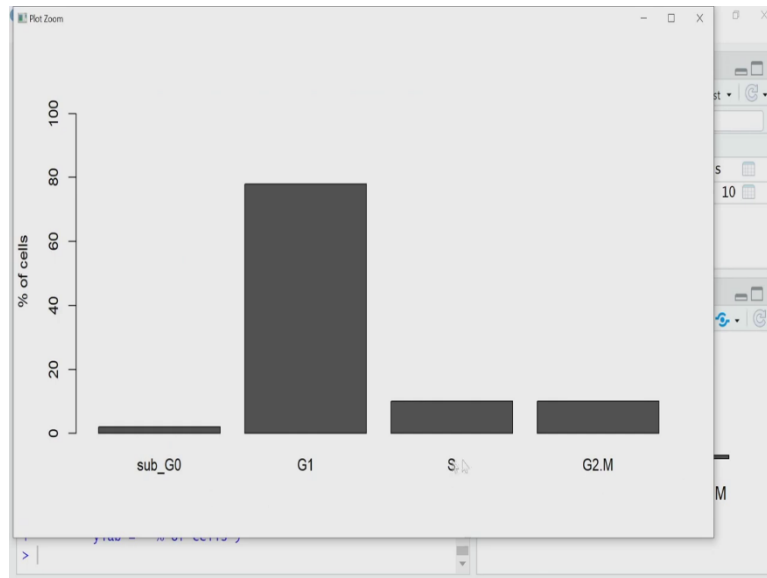
```
  ylim = c(0, 100),
```

```
  ylab = "% of cells")
```

So, now, I will use the d dot un as an argument for bar plot function to create bar plot. So, in bar plot function, the first argument is now, d dot un I am setting x y limit y will be the vertical axis this limit it is in percentage of cells so it can minimum value will be 0 maximum will be 100. So, I am setting y limit equal to c 0 comma 100. And the label of the vertical axis

is percentage of cells. So, if I execute this script, first I am creating the data set and then I am plotting the bar plot.

(Refer Slide Time: 24:17)



So, let me zoom in, this is a standard bar plot you expect. So, I have in the horizontal axis four categories, remember here, this is category data, So, I have four categories, sub G0, G1, S, G2 M, and for each category, these are untreated cells. So, they are a percentage of cell data. So for example, almost 80 percent of the cells in this untreated population are in G1 phase. Whereas very little, maybe 2 percent 3 percent are in sub G0 cell which is expected for a untreated sample.

Now in this case also you may be expecting that I should have error bars. Yes, actually, in bar plot, if you have multiple measurement, I should have the error bar because I am plotting the mean, again, bar plot does not have the default option to draw error bar here by default, there are advanced packages, which we will learn later on, where I have the option to put the error bar, and we will put a error bars then.

Now, usually, when you do multiple sample experiment, for example, I have three samples, right, you do not want three separate bar plot for them, you want the data for untreated, treated samples together in one single bar plot so that you can make a comparison between them. And that is what I will do, how I will do that?

(Refer Slide Time: 25:41)

RStudio

```

7
8 # Read data ----
9 d <- read.csv("facs2.csv")
10
11 # Make bar plot for untreated sample ----
12
13 # Get the data and create a matrix
14
15 d.un <- d[1, 2:5]
16
17 d.un <- data.matrix(d.un)
18
19 # Plot the bar plot
20
21 barplot(d.un,
22         ylim = c(0,100),
23         ylab = "% of cells")
24
25 # Bar plot with all three samples

```

23:30 Make bar plot for untreated sample

```

R 4.1.2 · C:/dab/
+ ylim = c(0,100),
+ ylab = "% of cells"
>

```

Environment

R · Global Environment

Data

d	3 obs. of 5 variables
d.un	int [1, 1:4] 2 78 10 10

Plots

Phase	% of cells
sub_G0	2
G1	78
S	10
G2.M	10

RStudio

```

1 # Generating Bar plot
2 # This data set is of flow cytometry experiment
3 # where percentage of cells in different phases
4 # of cell cycles are estimated.
5 # Number of samples = 3
6 # Number of cell cycle phases = 4
7
8 # Read data
11 # Make bar plot for untreated sample
25 # Bar plot with all three samples

```

23:30 Make bar plot for untreated sample

```

R 4.1.2 · C:/dab/
+ ylim = c(0,100),
+ ylab = "% of cells"
>

```

Environment

R · Global Environment

Data

d	3 obs. of 5 variables
d.un	int [1, 1:4] 2 78 10 10

Plots

Phase	% of cells
sub_G0	2
G1	78
S	10
G2.M	10

RStudio

```

4 # of cell cycles are estimated.
5 # Number of samples = 3
6 # Number of cell cycle phases = 4
7
8 # Read data
11 # Make bar plot for untreated sample ----
12
13 # Get the data and create a matrix
14
15 d.un <- d[1, 2:5]
16
17 d.un <- data.matrix(d.un)
18 barplot(height, ...)
19 # Plot the bar plot
20
21 barplot(d.un,
22         ylim = c(0,100),
23         ylab = "% of cells")
24

```

11:38 Make bar plot for untreated sample

```

R 4.1.2 · C:/dab/
+ ylim = c(0,100),
+ ylab = "% of cells"
>

```

Environment

R · Global Environment

Data

d	3 obs. of 5 variables
d.un	int [1, 1:4] 2 78 10 10

Plots

Phase	% of cells
sub_G0	2
G1	78
S	10
G2.M	10

RStudio

```

25 # Bar plot with all three samples ----
26
27 # Get the data and create a matrix
28
29 d.val <- data.matrix(d[,2:5])
30
31 # Create the bar plot
32
33 barplot(d.val,
34         beside = TRUE,
35         ylim = c(0, 100),
36         ylab = "% of cells",
37         legend.text = c("Untreated", "Drug A", "Drug B"),
38         args.legend = list(bty = "n", x = "topright"))
39
40
41 # Few formatting of the plot
42

```

Environment

R - Global Environment

Data

d 3 obs. of 5 variables

d.un int [1, 1:4] 2 78 10 10

Plots

```

R 4.1.2 · C:/dab/
+ y11m = c(0,100),
+ ylab = "% of cells"
>

```

RStudio

```

lec1_week4_BR x lec1_week4_CR x lec1_week4_DR x

```

X	sub_G0	G1	S	G2.M
1 untreated	2	78	10	10
2 drug_A	3	80	8	9
3 drug_B	20	54	14	12

Showing 1 to 3 of 3 entries, 5 total columns

```

R 4.1.2 · C:/dab/
+ y11m = c(0,100),
+ ylab = "% of cells"
>

```

Environment

R - Global Environment

Data

d 3 obs. of 5 variables

d.un int [1, 1:4] 2 78 10 10

Plots

RStudio

```

25 # Bar plot with all three samples ----
26
27 # Get the data and create a matrix
28
29 d.val <- data.matrix(d[,2:5])
30
31 # Create the bar plot
32
33 barplot(d.val,
34         beside = TRUE,
35         ylim = c(0, 100),
36         ylab = "% of cells",
37         legend.text = c("Untreated", "Drug A", "Drug B"),
38         args.legend = list(bty = "n", x = "topright"))
39
40
41 # Few formatting of the plot
42

```

Environment

R - Global Environment

Data

d 3 obs. of 5 variables

d.un int [1, 1:4] 2 78 10 10

Plots

```

R 4.1.2 · C:/dab/
+ y11m = c(0,100),
+ ylab = "% of cells"
>

```

```
d.val ← data.matrix(d[ , 2:5])  
barplot(d.val,  
        beside = TRUE,  
        ylim = c(0, 100),  
        ylab = "% of cells",  
        legend.text = c("Untreated", "Drug A", "Drug B"),  
        args.legend = list(bty = "n", x = "topright"))
```

Again, I will use the bar plot function, but I have to rearrange the data. And I have to add some arguments to that. So, to do that, first, what I have to do? Is that I have to get the data and convert that into matrix format. So, now in this case, I want all the rows from Column 2 to column 5. So, what I am doing, I am saying d, the row index, I am leaving empty, and then 2 colon 5, that means from column 2 to column 5, for all rows, you take it from d, and then convert that into matrix using this data dot matrix function.

(Refer Slide Time: 26:25)

The screenshot shows the RStudio interface with the following R code in the editor:

```
25 # Bar plot with all three samples ----
26
27 # Get the data and create a matrix
28
29 d.val <- data.matrix(d[, 2:5]) +
30
31 # Create the bar plot
32
33 barplot(d.val,
34         beside = TRUE,
35         ylim = c(0, 100),
36         ylab = "% of cells",
37         legend.text = c("Untreated", "Drug A", "Drug B"),
38         args.legend = list(bty = "n", x = "topright"))
39
40
41 # Few formatting of the plot
42
```

The Environment pane shows the following data:

Object	Class	Attributes
d	matrix	3 obs. of 5 variables
d.un	integer	[1, 1:4] 2 78 10 10

The Plots pane shows a bar plot with the y-axis labeled "% of cells" and the x-axis labeled "sub_G0 G1 S G2.M". The plot shows four bars of different heights, with G1 being the tallest.

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

```
> d.val <- data.matrix(d[, 2:5])
> View(d.val)
```

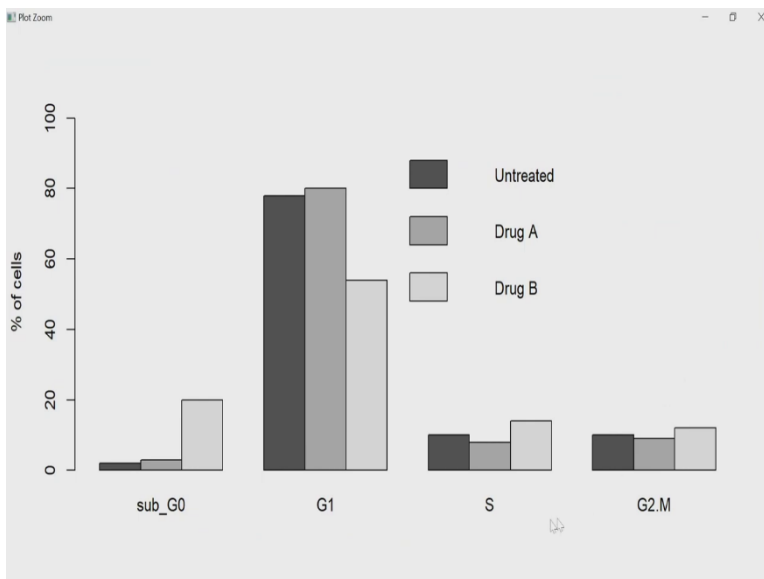
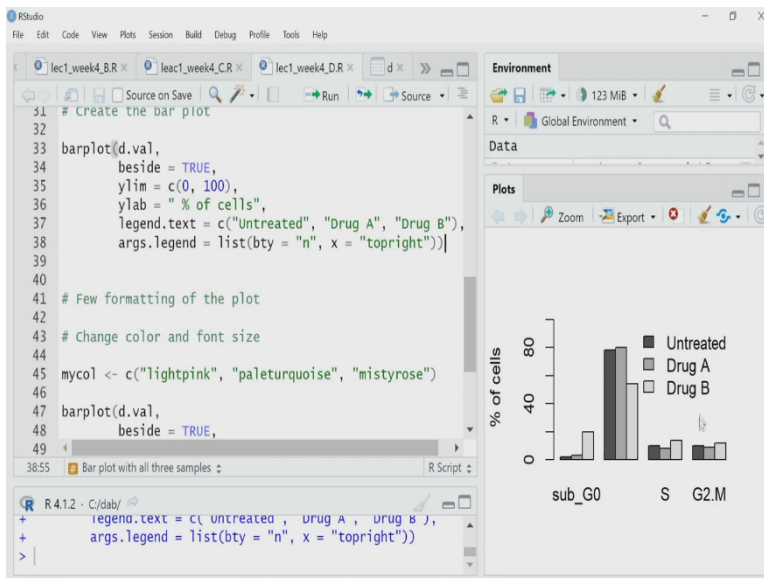
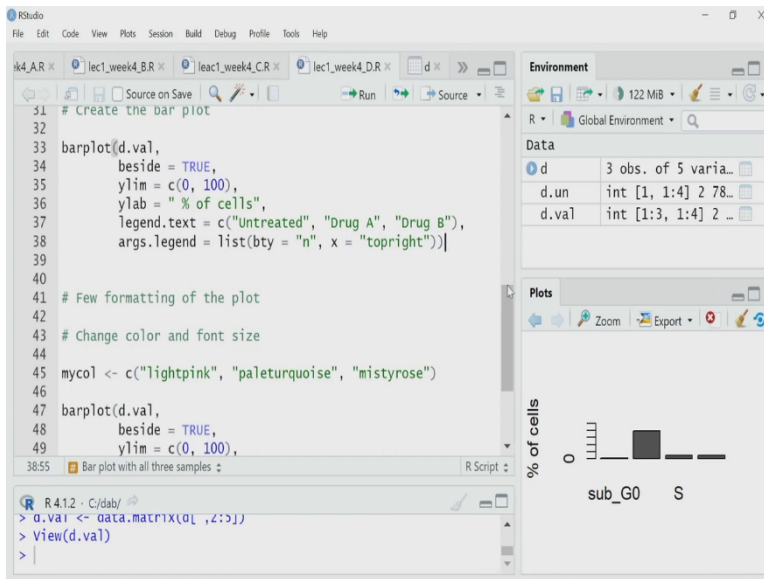
The Environment pane shows the following data:

Object	Class	Attributes
d	matrix	3 obs. of 5 variables
d.un	integer	[1, 1:4] 2 78 10 10
d.val	integer	[1:3, 1:4] 2 3 20 ...

The Plots pane shows the same bar plot as in the previous screenshot.

The console output shows the following data table:

	sub_G0	G1	S	G2.M
1	2	78	10	10
2	3	80	8	9
3	20	54	14	12



```
barplot(d.val,  
       beside = TRUE,  
       ylim = c(0, 100),  
       ylab = "% of cells",  
       legend.text = c("Untreated", "Drug A", "Drug B"),  
       args.legend = list(bty = "n", x = "topright"))
```

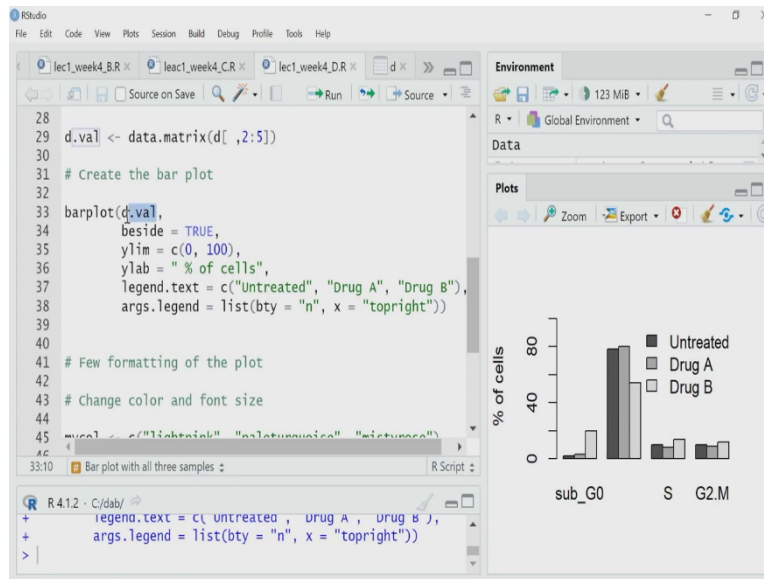
So, I have d dot val, which will store these values, I can look into that. Now I have a matrix with the values for each of the cell phases and cycle phases for each of these three samples. Now I will plot these data as bar plot. So, before I explain what type of argument I will put here, let me first plot it, so that it will be much easier to explain the argument.

So, I have plotted the plot and bar plot with all these three data set, let me zoom into so that you can easily see it clearly. So, what I have here? I have data for all these three samples treated, untreated like they are two treatment groups, drug A and drug B, and the data are got stacked side by side. So, how we have stacked?

So, look into the G1. The first one as per the legend drawn here is for untreated sample, the second bar is again G1 for drug A treated sample, the third one is for drug B treated samples. So, all the three samples data for G1 phase are stacked side by side.

Similarly, for other phases S, G2 M and sub G0, you have these data got stacked and in these you can easily understand that, in for drug B treated a large proportion of cells are now in sub G01 and that has caused in decrease in G1 phase cells, whereas some G2 M and S phase cells has also increased for drug B treated, whereas, drug A treated samples are almost similar to untreated one.

(Refer Slide Time: 28:16)



`d.val <- data.matrix(d[, 2:5])`

`barplot(d.val,`

`beside = TRUE,`

`ylim = c(0, 100),`

`ylab = "% of cells",`

`legend.text = c("Untreated", "Drug A", "Drug B"),`

`args.legend = list(bty = "n", x = "topright"))`

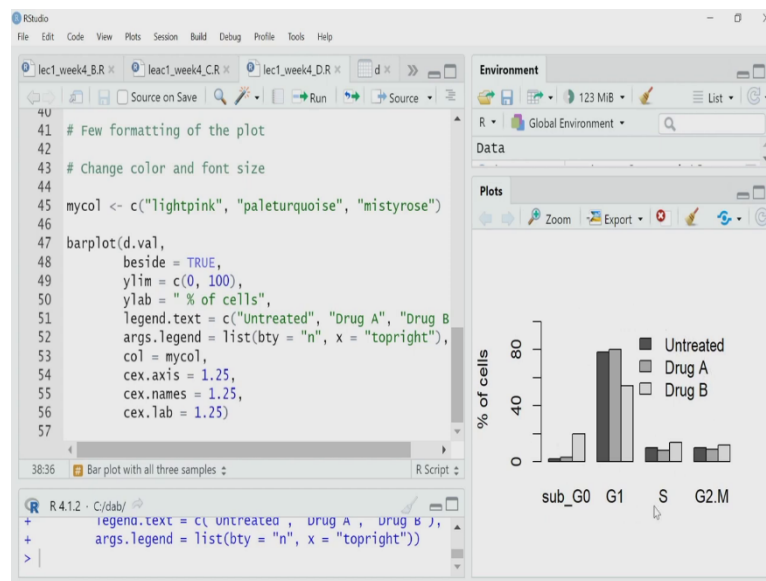
So, how I have achieved these using the bar plot function? Let me explain that now. So, the first thing is that obviously, the `d.val` is the argument. So, that data matrix with all these three sample having four phases of cell cycle is my input my argument, first input argument for bar plot.

Then, I have written `beside = true` that means, this bar will be plotted side by side, if I have not used that `beside = true` then data would have got stacked and then, I have set the limit of vertical axis to 0 to 100, I have labeled percentage of cell as the vertical axis label and by default bar plot is plotting legends. So, to do that, I have mentioned what should be the legend text.

So, I have said `untreated drug A and drug B` should be my legend text. And I have some added argument for legends. So, that is `args dot legend`, and that those args to set the position whether i want a box around the legend or not. So, I have set `x` equal to `top right`. That means

the legend should be arranged in the top right position. And I do not want any box. So, I have written bty equal to n.

(Refer Slide Time: 29:33)



`mycol ← c("lightpink", "paleturquoise", "mistyrose")`

`barplot(d.val,`

`beside = TRUE,`

`ylim = c(0, 100),`

`ylab = "% of cells",`

`legend.text = c("Untreated", "Drug A", "Drug B"),`

`args.legend = list(bty = "n", x = "topright").`

`col = mycol,`

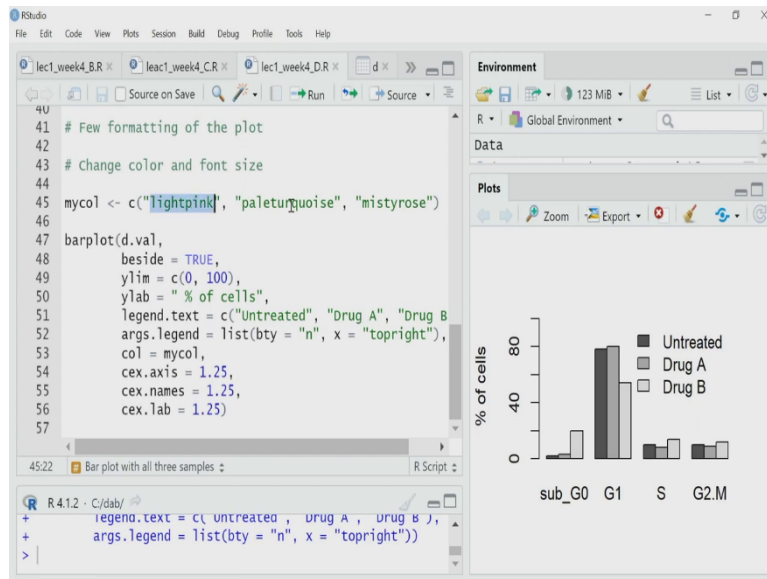
`cex.axis = 1.25,`

`cex.name = 1.25,`

`cex.lab = 1.25)`

Now, what I will do? I will add few more formats here formatting strings here. So, first thing what I will do? I will change the color of this bar. So, by default, R is using his default color scheme, which looks good but and is good for black and white printing and black and white a representation of data. But if you want to add some color, how can you do that?

(Refer Slide Time: 30:00)



`mycol ← c("lightpink", "paleturquoise", "mistyrose")`

`barplot(d.val,`

`beside = TRUE,`

`ylim = c(0, 100),`

`ylab = "% of cells",`

`legend.text = c("Untreated", "Drug A", "Drug B"),`

`args.legend = list(bty = "n", x = "topright").`

`col = mycol,`

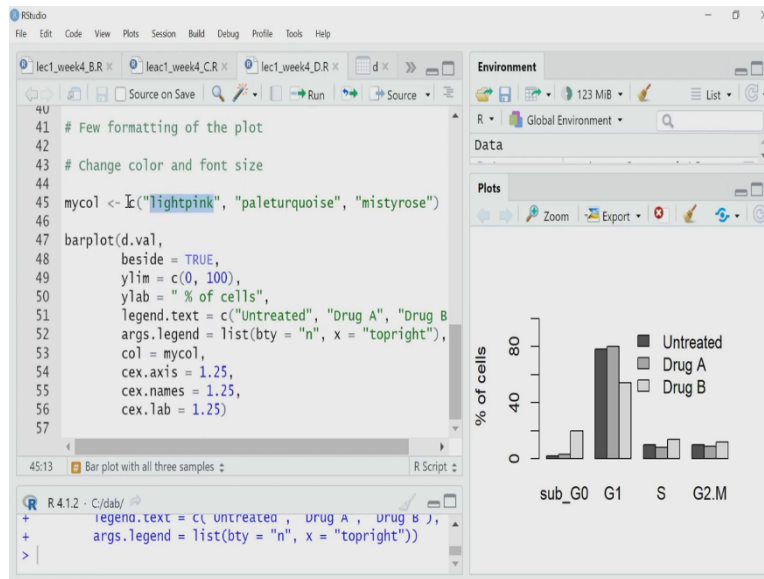
`cex.axis = 1.25,`

`cex.name = 1.25,`

`cex.lab = 1.25)`

So, to do that what I am doing I will be using three color because I have three sample, one will be light pink, and other will be a turquoise. And it is the pale turquoise actually, and another will be mistyrose, R has inbuilt lots of colors, hundreds of them, you have to go to the R documentation to find them, you can specify them by name, you can specify them by numbers, also, I am specifying them here by name.

(Refer Slide Time: 30:26)



`mycol ← c("lightpink", "paleturquoise", "mistyrose")`

`barplot(d.val,`

`beside = TRUE,`

`ylim = c(0, 100),`

`ylab = "% of cells",`

`legend.text = c("Untreated", "Drug A", "Drug B"),`

`args.legend = list(bty = "n", x = "topright").`

`col = mycol,`

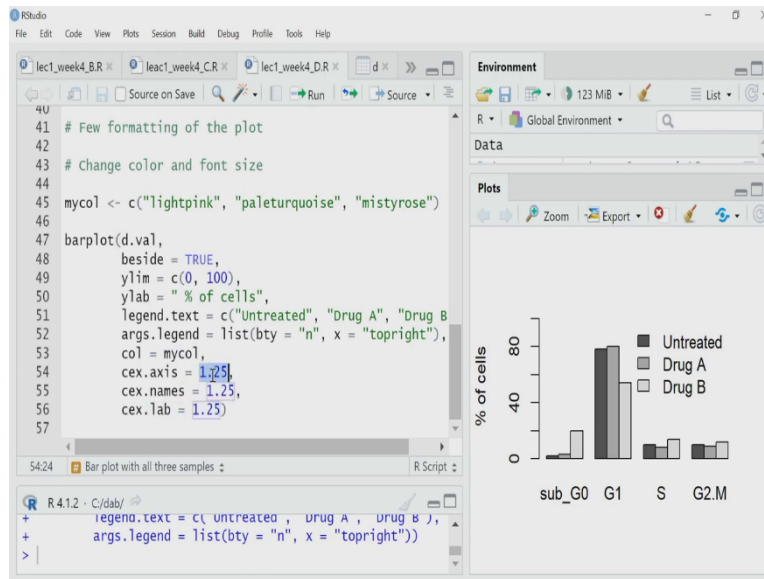
`cex.axis = 1.25,`

`cex.name = 1.25,`

`cex.lab = 1.25)`

So, I am creating a list of these three color and assigning that to mycol, so that I can you call it later on. So, now I will plot the bar plot, right what I am doing again, side by side, so beside is true the same data set, the only thing I am changing here is color equal to col equal to mycol, so the default color will not be used anymore. And also what I am doing, I am changing the font sizes.

(Refer Slide Time: 30:51)



`mycol ← c("lightpink", "paleturquoise", "mistyrose")`

`barplot(d.val,`

`beside = TRUE,`

`ylim = c(0, 100),`

`ylab = "% of cells",`

`legend.text = c("Untreated", "Drug A", "Drug B"),`

`args.legend = list(bty = "n", x = "topright").`

`col = mycol,`

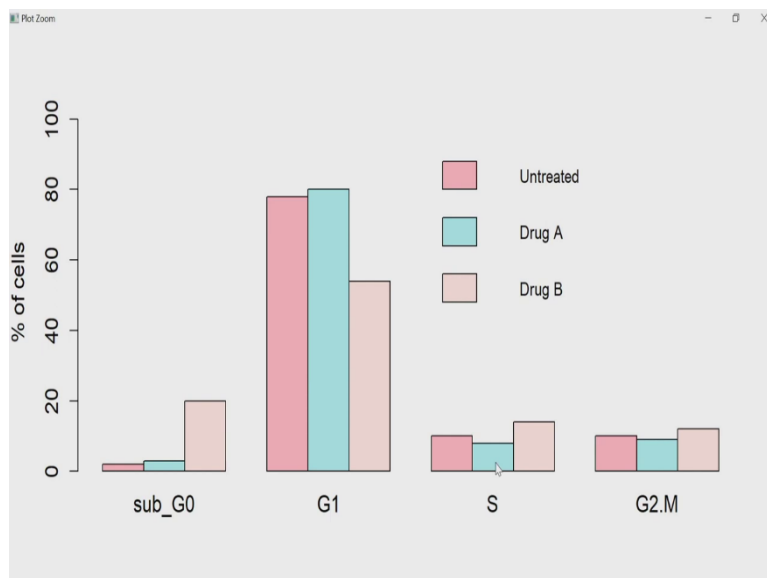
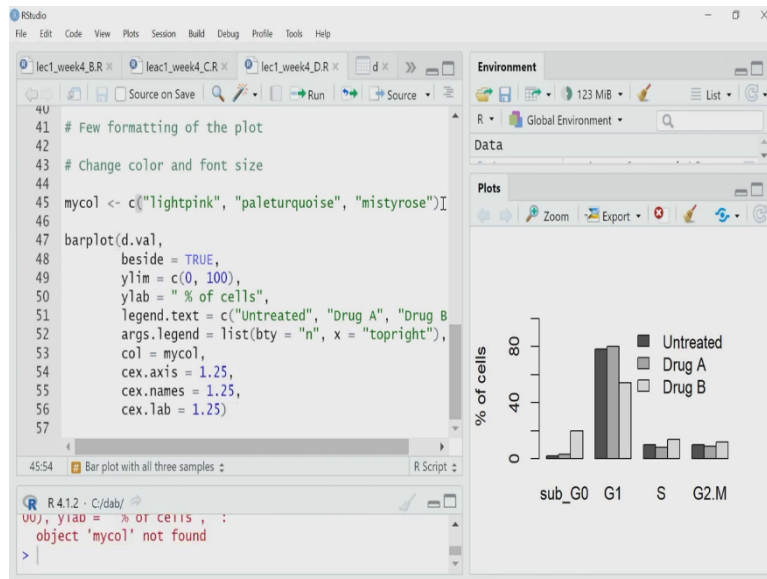
`cex.axis = 1.25,`

`cex.name = 1.25,`

`cex.lab = 1.25)`

So, I am saying, the axis labels should be 1.5 times of the default one. So, `cex dot axis` is 1.25. So, the numbers will increase their font size, the `cex name` 1.25 The names will be 1 point font size of the names will be 1.25 the default one and the label font will be also 1.25. So, the label is the percentage of cell in the vertical axis that I have written and the names are the names of these groups that he cells define cell cycle phases.

(Refer Slide Time: 31:23)



`mycol ← c("lightpink", "paleturquoise", "mistyrose")`

`barplot(d.val,`

`beside = TRUE,`

`ylim = c(0, 100),`

`ylab = "% of cells",`

`legend.text = c("Untreated", "Drug A", "Drug B"),`

`args.legend = list(bty = "n", x = "topright").`

`col = mycol,`

`cex.axis = 1.25,`

```
cex.name = 1.25,
```

```
cex.lab = 1.25)
```

So, if I execute it I get these types of diagrams. If I zoom in, you can see I have three different color. So, if you want a color bar plot, these may be the better way to do it, you can change the color as per your requirement. So, this brings us to the end of this lecture, what we have learned in this lecture, we have learned that when I have to go for scatterplot?

And when you have to go for a bar plot? and we have learned how to perform create better scatter plot? Scatter plot with lines and bar plot using R. As I said, we are using the default functions for R for plotting of R for plotting. Those default function has lots of limitation. At the same time, they have lots of options also, R itself is very flexible, you I have only scratched on the surface of these default functions for bar plot and scatter plot.

If you practice them, use them and play with the arguments, you may actually create very good quality plots, but there are packages in R dedicated to create high quality publication great graphs, and we will discuss them separately in separate lectures. So thank you for learning with me in this lecture. See you in those lectures too.