R

About R:

1. This is more statistics and research.
2. Language for Statistical Computing. Developed by Ross Ihaka & Gentleman, Auckland, New Zealand.
3. This is Open Source Implementation of S language.
4. **Use of R:**
   1. For Exploarity work, R is easier for beginners.
   2. Statistical Models can be written with a few lines of code.
   3. Statistical Technique, Highly Extensible and Visualization Capabilities
5. Use rPython package to run the python code from R. Pass or get data from R.
6. **IDS:** R studio
7. **Why R for data analysis?**

R is not the only language that can be used for data analysis. Why R rather than another? Here is a list:

* interactive language
* data structures
* graphics
* missing values
* functions as first class objects
* packages
* community

Data analysis is inherently an interactive process — what you see at one stage determines what you want to do next. Interactivity is important. Language is important. The two together — an interactive language — is even more than their sum. But there is a down-side: compromises between interactive use and programming use are the cause of some user trauma.

R has a fantastic mechanism for creating data structures. Obviously if you are doing data analysis, you want to be able to put your data into a natural form. You don’t have to warp your data into a particular structure because that is all that is available.

Graphics should be central to data analysis. Humans are predominantly visual, we don’t intuitively grasp numbers like we do pictures. It is easy to produce graphs for exploring data. The default graphs can be tweaked to get publication-quality graphs.

Real data have missing values. Missing values are an integral part of the R language. Many functions have arguments that control how missing values are to be handled.

Functions, like mean and median, are objects that you can use like data. You can easily change your analysis to use the median (or some strange estimate you make up on the spot) rather than the mean.

R has a package system that makes it extremely easy for people to add their own functionality so it is indistinguishable from the central part of R. And people have. There are thousands of packages that do all sorts of extraordinary things.

The R community is very strong, and quite committed to improving data analysis.

1. **The Way We Think:**

One of the goals of S (and hence R), and one that I think has largely been successful, is that the language should mirror the way that people think. A simple example: suppose we think that weight is a function of (dependent on) height and girth. The R formula to express this is:

**Weight ~ height + girth**

The + is not + as in addition, but + as in “and”.

1. If the error message isn’t in English, run **Sys.setenv(LANGUAGE = "en")** and re-run the code; you’re more likely to find help for English error messages

dput() - The easiest way to include data in a question is to use dput() to generate the R code to recreate it.

For example, to recreate the mtcars dataset in R, I’d perform the following steps:

* Run dput (mtcars) in R
* Copy the output
* In my reproducible script, type mtcars <- then paste.

1. Data Editor Command: **fix()**
2. **What is the difference between package and library?**

A *package* is a standardized collection of material extending R, e.g. providing code, data, or documentation. A *library* is a place (directory) where R knows to find packages it can use (i.e., which were *installed*). R is told to use a package (to “load” it and add it to the search path) via calls to the function library. I.e., library () is employed to load a package from libraries containing packages.

1. **What object types can hold several different types at the same time?**

Data Frame, List

1. **R Data Types:**
   1. Vector => c()
      1. Sequence of Data Elements
      2. Same Basic Data Type
      3. 1D array of data elements
   2. List => list()
      1. Can hold any type R Objects/ Data Types
   3. Matrix => matrix()
      1. 2D array of data elements
      2. One atomic vector type (same basic data type). But using cbind() or rbind() we can create matrix with different data types

Ex., numerc <- matrix(1:6, nrow =2)

letters <- c('a','b','c','d','e','f')

char\_f <- matrix(letters, nrow=2)

rbind (numerc, char\_f)

**Result:**

[,1] [,2] [,3]

[1,] "1" "3" "5"

[2,] "2" "4" "6"

[3,] "a" "c" "e"

[4,] "b" "d" "f"

* 1. Array => array()
  2. Factors => factor()
     1. Two kind of variables:
        1. Categorical Variable => If categorical variable is there, then factor function will list out the categories. Also it will sort the alphabets. Also user can set the level in any order by using Levels and Labels in same factor (check with second Ex.,)

Ex., x <- c(“A”,”B”,”C”,”D”,”A”,”C”)

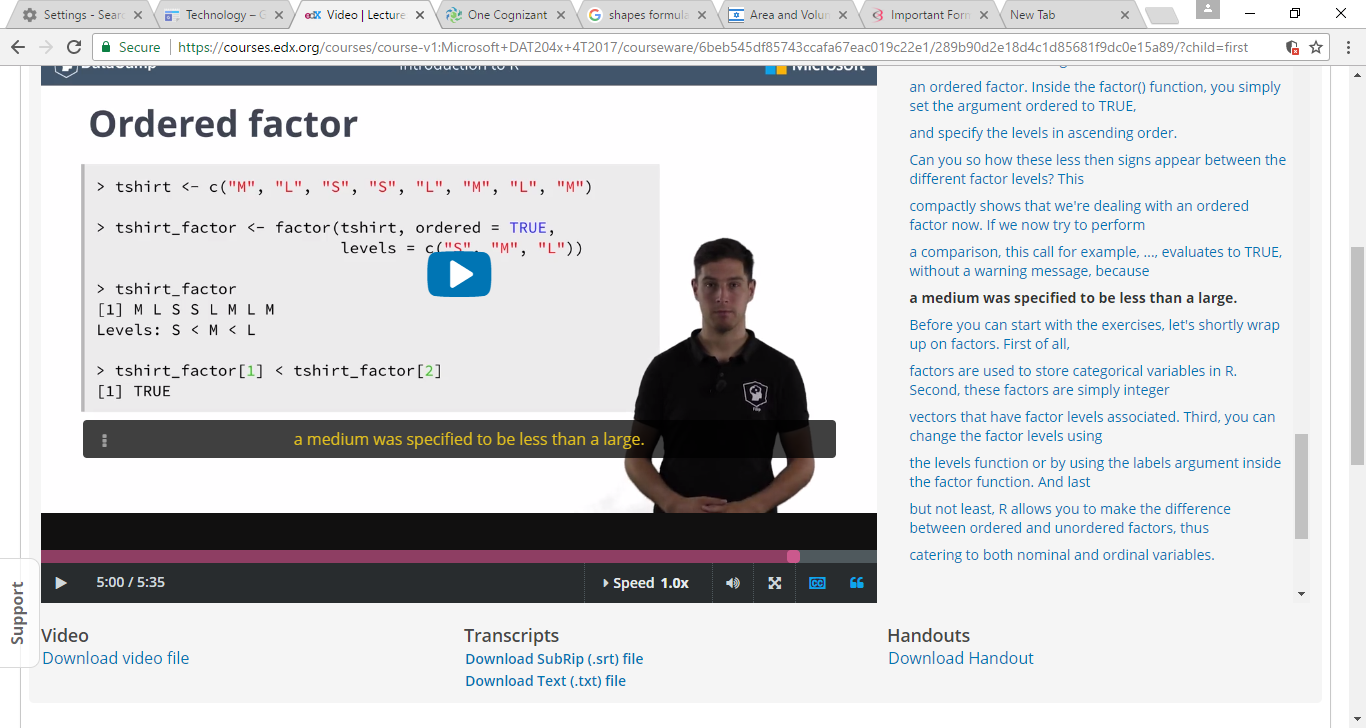
factor(x) =>

A B C D A C levels**: A B C D**

Ex., factor (x, levels = c(“D”,”C”,”B”,”A”),

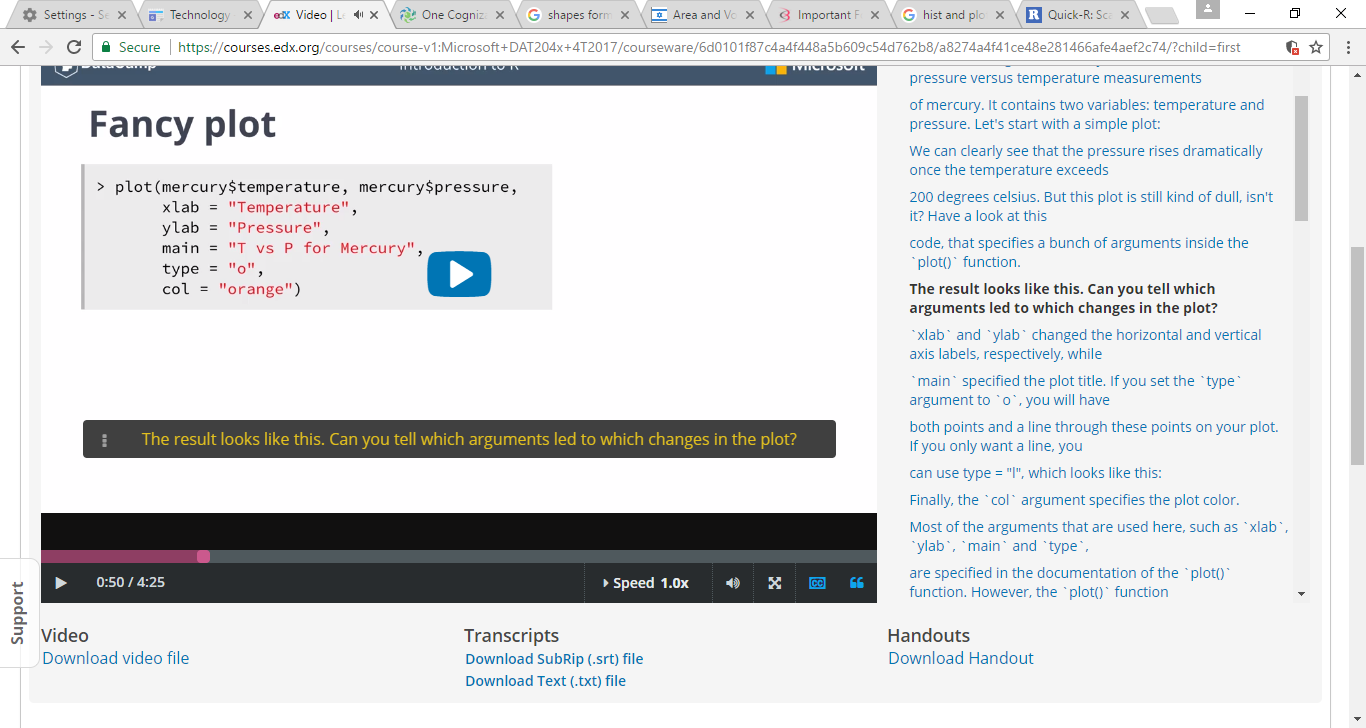
labels = c(“BT\_D”,”BT\_C”,”BT\_B”,”BT\_A”))

* + - 1. Nominal Variables

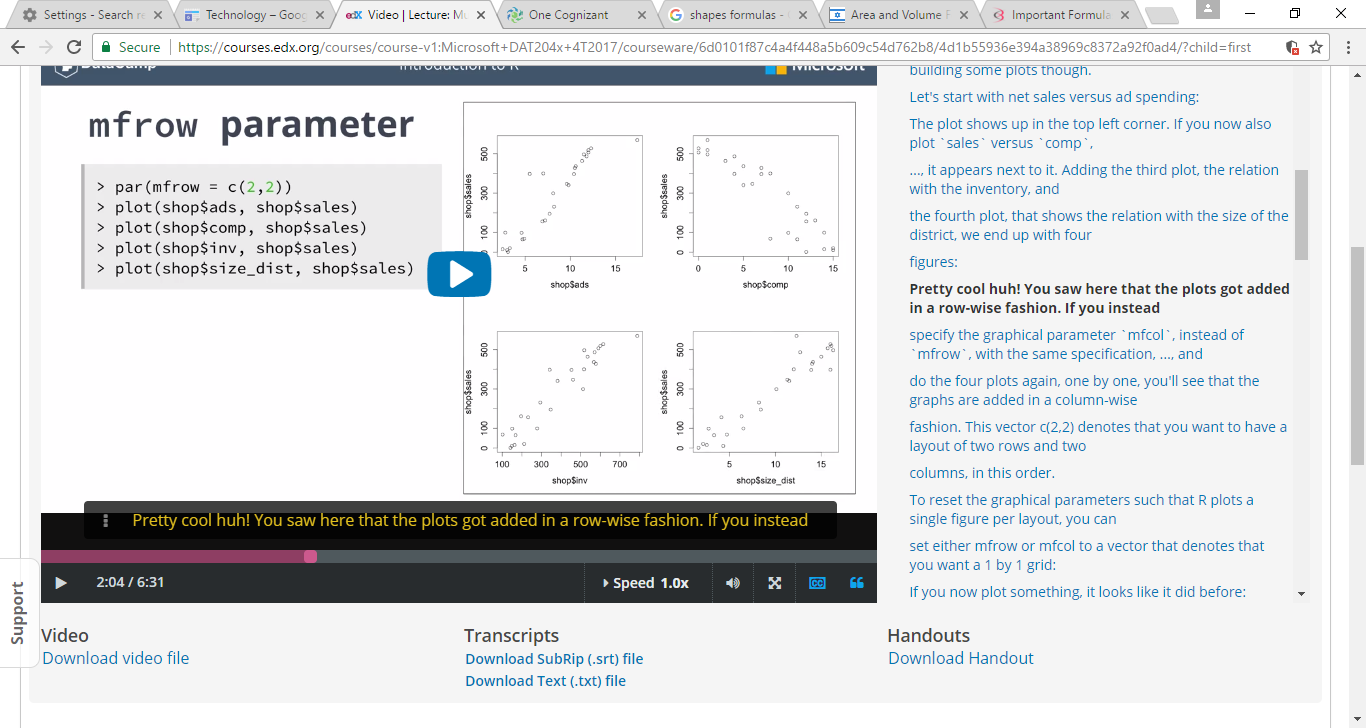


* 1. Data Frame => data.frame()
     1. Specifically for Datasets
     2. Rows called “Observation”
     3. Columns called “Variables”
     4. Contain different data types

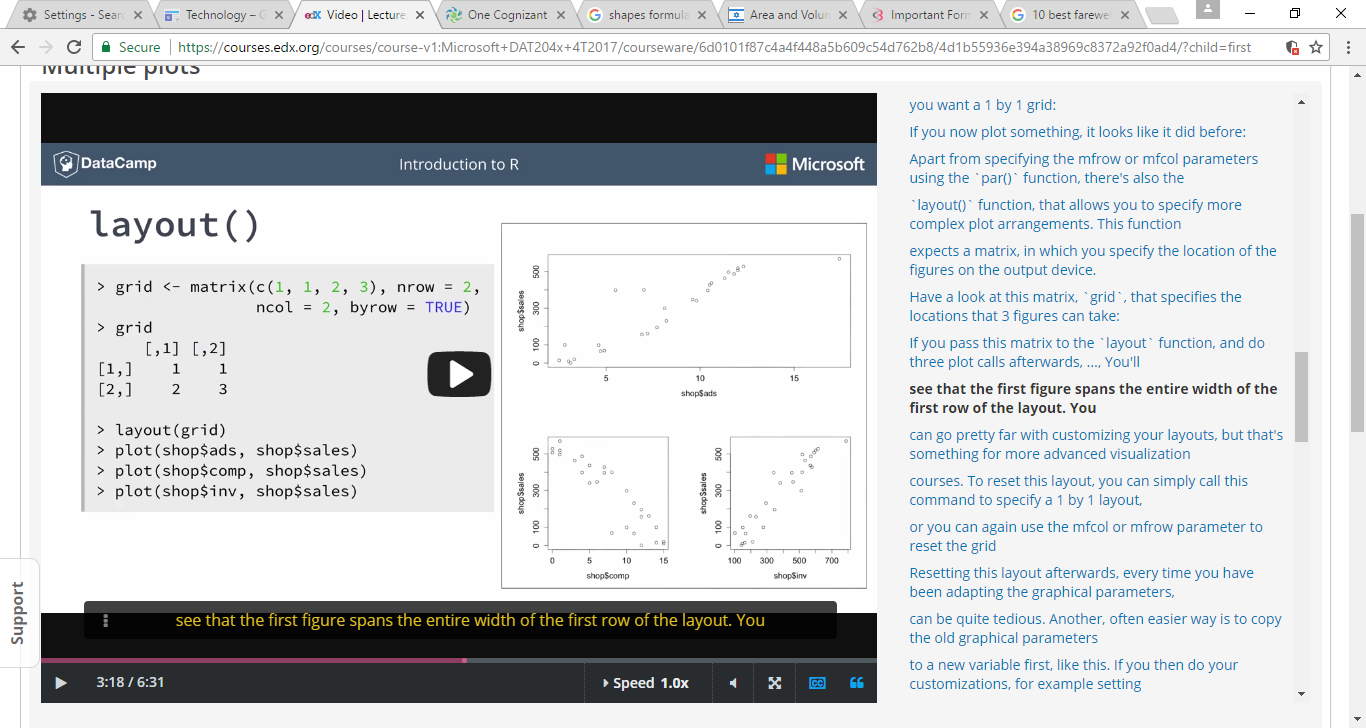
1. **Graphics**:
   1. Reproducibility is big advantage
   2. Package name: Graphics
      1. Plot()
      2. Hist()
   3. Important packages are, ggplot2, ggvis, lattice
   4. When plotting a categorical variable, R will automatically produce a **bar chart**. When plotting a continuous variable, R will produce an **index plot**.
   5. breaks argument specifies how many bins, or histogram cells, your plot should contain. By default, the "**Sturges**" algorithm is used to automatically determine the number of bins
   6. Naming the plots: **Customization**



* 1. Multiple plots in a grid:



Layout ():



1. Linear Regression:

Linear regression is used to predict the value of an outcome variable Y based on one or more input predictor variables X. The aim is to establish a linear relationship (a mathematical formula) between the predictor variable(s) and the response variable, so that, we can use this formula to estimate the value of the response Y, when only the predictors (Xs) values are known.

– Run R Online:

[http://www.roncloud.com](http://www.roncloud.com/)  
<http://www.compileonline.com/execute_r_online.php>  
[http://www.r-fiddle.org](http://www.r-fiddle.org/)  
<http://pbil.univ-lyon1.fr/Rweb/Rweb.general.html>