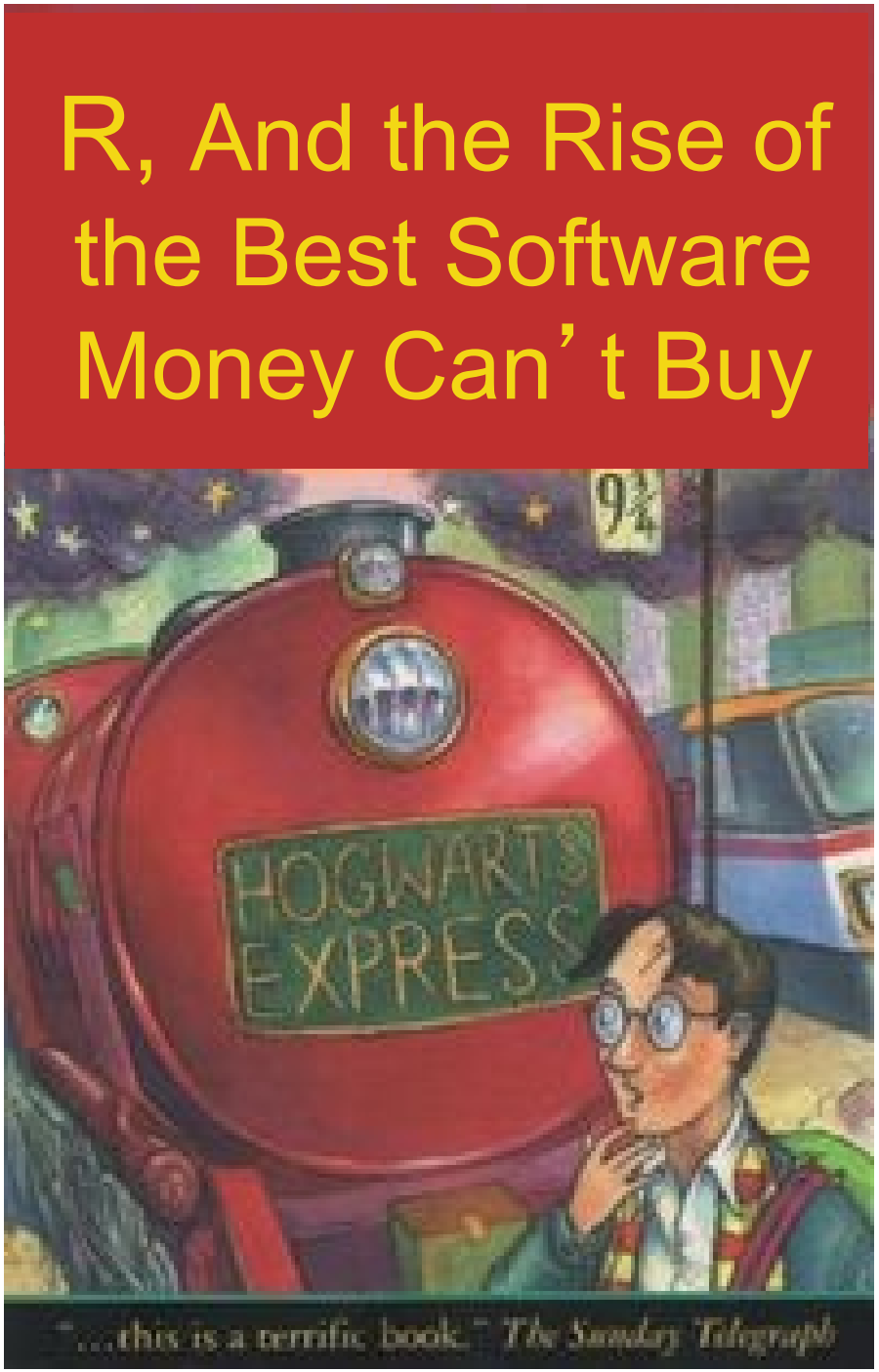
A very brief introduction to R

- Matthew Keller

Some material cribbed from: UCLA Academic Technology Services Technical Report Series (by Patrick Burns) and presentations (found online) by Bioconductor, Wolfgang Huber and Hung Chen, & various Harry Potter websites

R programming language is a lot like magic... except instead of spells you have functions.

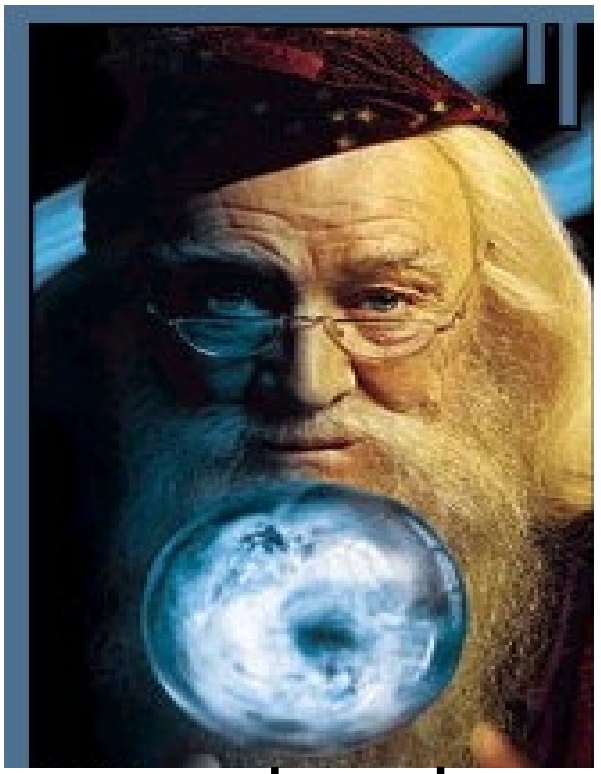


SPSS =

muggle

SPSS and SAS users are like muggles. They are limited in their ability to change their environment. They have to rely on algorithms that have been developed for them. The way they approach a problem is constrained by how SAS/SPSS employed programmers thought to approach them. And they have to pay money to use these constraining algorithms.

=



wizard

R users are like wizards. They can rely on functions (spells) that have been developed for them by statistical researchers, but they can also create their own. They don't have to pay for the use of them, and once experienced enough (like Dumbledore), they are almost unlimited in their ability to change their environment.

History of R

- S: language for data analysis developed at Bell Labs circa 1976
- Licensed by *AT&T/Lucent* to *Insightful Corp*. Product name: *S-plus*.
- R: initially written & released as an open source software by Ross Ihaka and Robert Gentleman at U Auckland during 90s (R plays on name "S")
- Since 1997: international R-core team ~15 people & 1000s of code writers and statisticians happy to share their libraries! AWESOME!

"Open source"... that just means I don't have to pay for it, right?

- No. Much more:
  - –Provides full access to algorithms and their implementation. Most of R is written in… R, making it easy to see what functions are actually doing.
  - –Gives the community ability to fix bugs/extend software
  - –Provides a forum allowing researchers to explore and expand the methods used to analyze data
  - –Ensures that scientists around the world - and not just ones in rich countries - are the co-owners to the software tools needed to carry out research
  - –Promotes reproducible research by providing open and accessible tools
  - **–Product of 1000s of leading experts in the fields they know best. It is CUTTING EDGE.**

What is it?

- R is an interpreted computer language.
  - Most user-visible functions are written in R itself, calling upon a smaller set of internal primitives.
  - It is possible to interface procedures written in C, C+, or FORTRAN languages for efficiency, and to write additional primitives.
  - System commands can be called from within R

- R is used for data manipulation, statistics, and graphics. It is made up of:
  - operators (+ - <- * %*% …) for calculations on arrays & matrices
  - large, coherent, integrated collection of functions
  - facilities for making unlimited types of publication quality graphics
  - user written functions & sets of functions (packages); 16000+ contributed packages so far & growing

R

Advantages

- Fast and free.
- State of the art: Statistical researchers provide their methods as R packages. SPSS and SAS are years behind R!
- 2nd only to MATLAB for graphics.
- Mx, WinBugs, and other programs use R.
- Active user community
- Excellent for simulation, programming, computer intensive analyses, etc.
- Forces you to *think* about your analysis.
- Interfaces with database storage software (SQL)

Disadvantages

R

Advantages

- Fast and free.
- State of the art: Statistical researchers provide their methods as R packages. SPSS and SAS are years behind R!
- 2nd only to MATLAB for graphics.
- Mx, WinBugs, and other programs use R.
- Active user community
- Excellent for simulation, programming, computer intensive analyses, etc.
- Forces you to *think* about your analysis.
- Interfaces with database storage software (SQL)
- Large vectors in 64 bit: 2^52 length

Disadvantages

- Not user friendly @ start - steep learning curve, minimal GUI.
- No commercial support; figuring out correct methods or how to use a function on your own can be frustrating.
- Working with large datasets is limited by RAM and some operations don't work on vectors > 2^31 length
- Not natively multi-threaded (easy work-arounds though)
- In the beginning, data prep & cleaning can be messier & more mistake prone in R vs. SPSS or SAS
- Some users complain about hostility on the R listserve

Learning R....

R-help listserve....

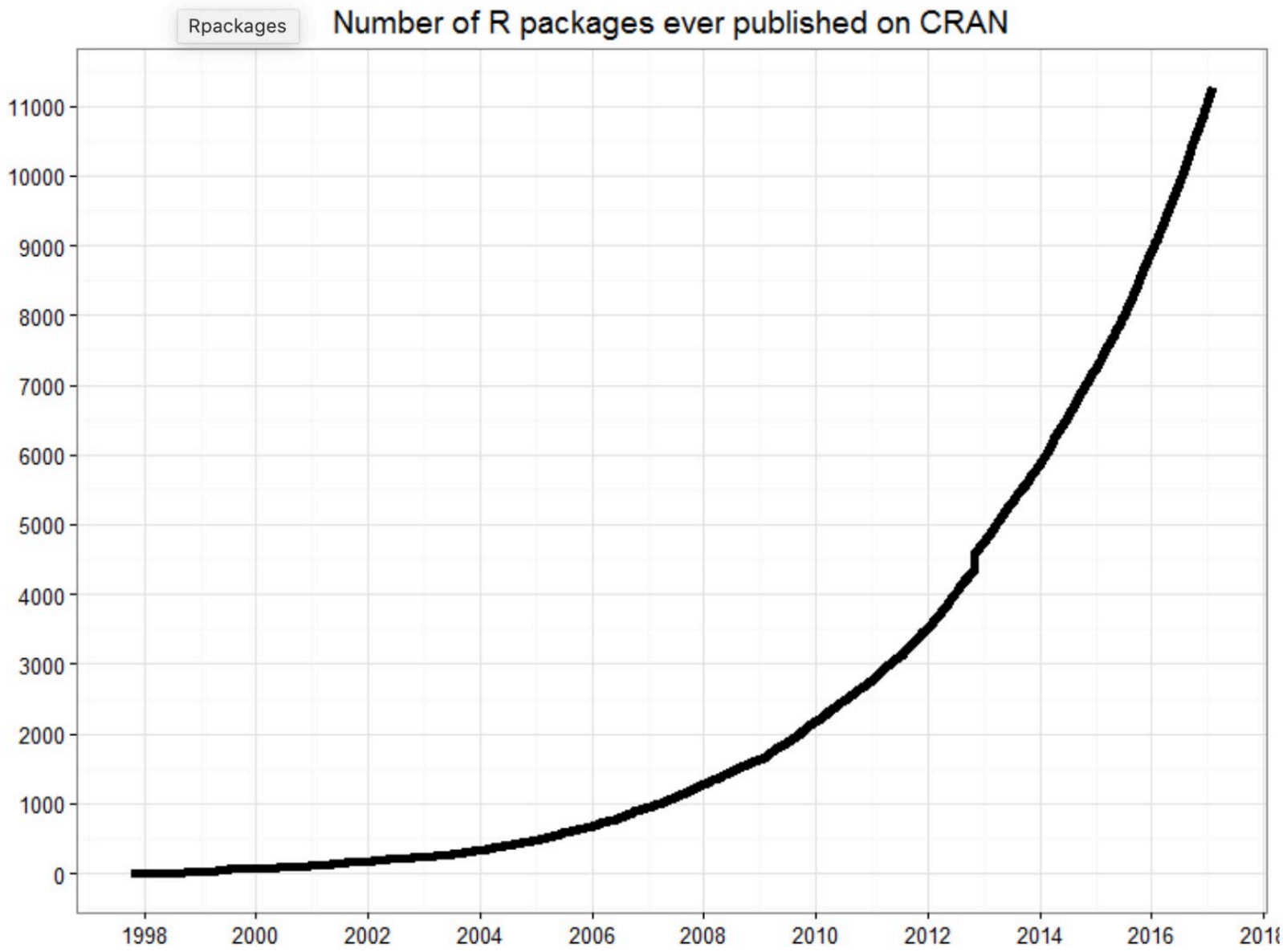
There are over 16K add-on packages

(http://cran.r-project.org/src/contrib/PACKAGES.html

http://www.bioconductor.org https://github.com/trending?l=r)

- This is an enormous advantage - new techniques available without delay, and they can be performed using the R language you already know.
- Allows you to build a customized statistical program suited to your own needs.
- Downside = as the number of packages grows, it is becoming difficult to choose the best package for your needs, & QC is an issue.

Growth of R packages through 2012

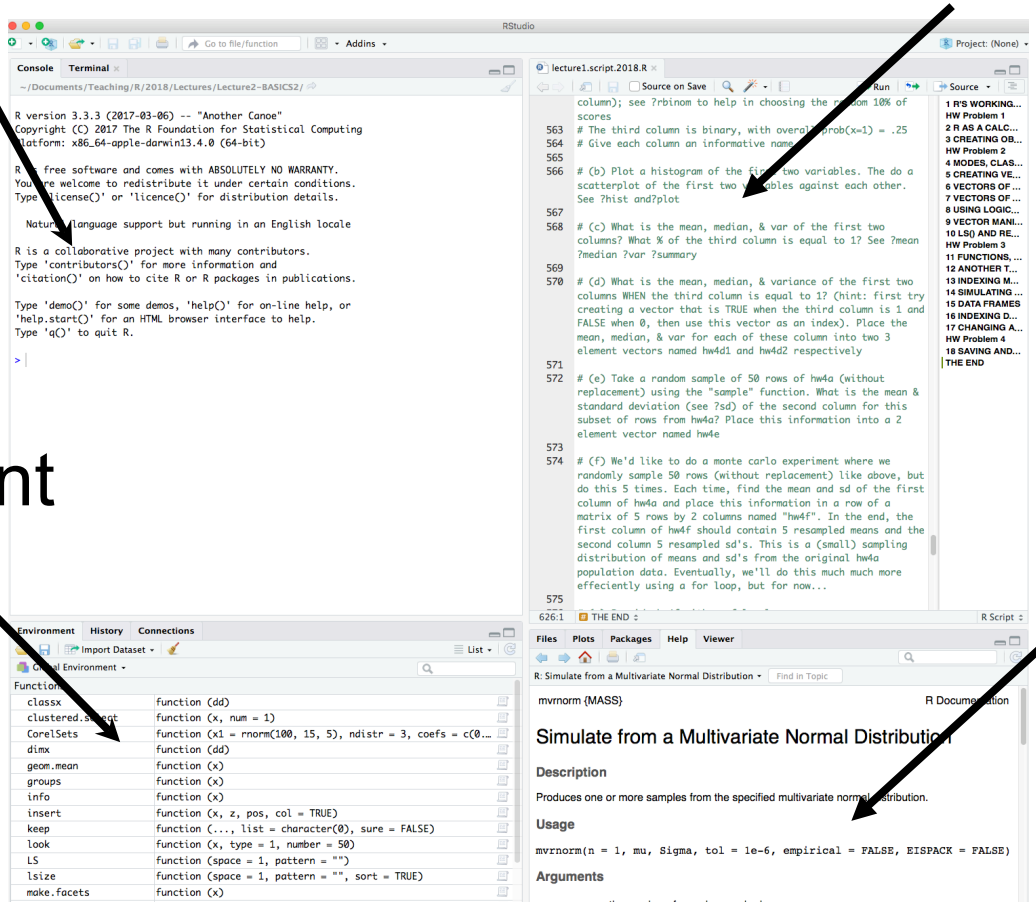


Will anything replace R in the future?

- Probably, but it's hard to know when, and I'd be my bottom dollar that it will be an object oriented, open-sourced language like R. (Thus translating your R knowledge will not be tough).
- One possible guess at this next language: JULIA (http://julialang.org), which is faster than R, able to work with very large datasets, and has sensible syntax (something R sometimes lacks). It already has 473 packages.

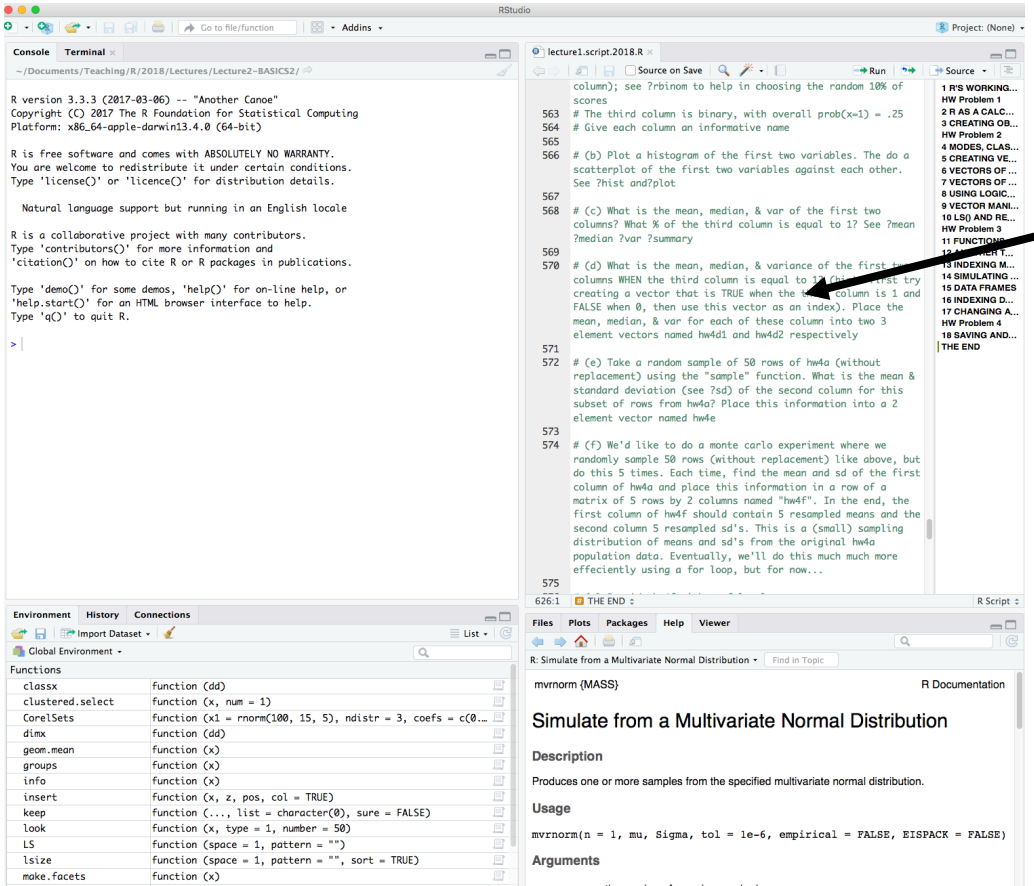
Typical Rstudio session

- Console – output & temporary input - usually unsaved
- Script – tells R what to do. Save this
- Environment
- Misc. windows, including help, files, etc.



Typical R session

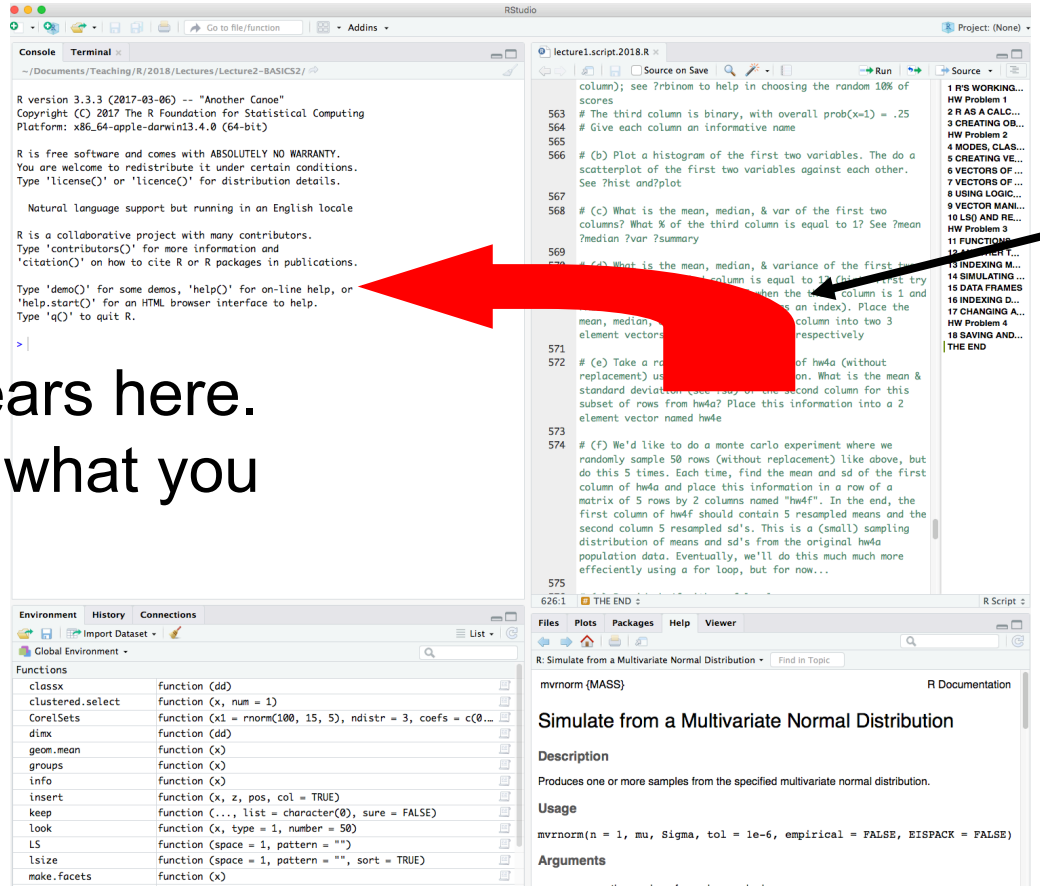
- R sessions are *interactive*



Write small bits of code here and run it

Typical R session

- R sessions are *interactive*

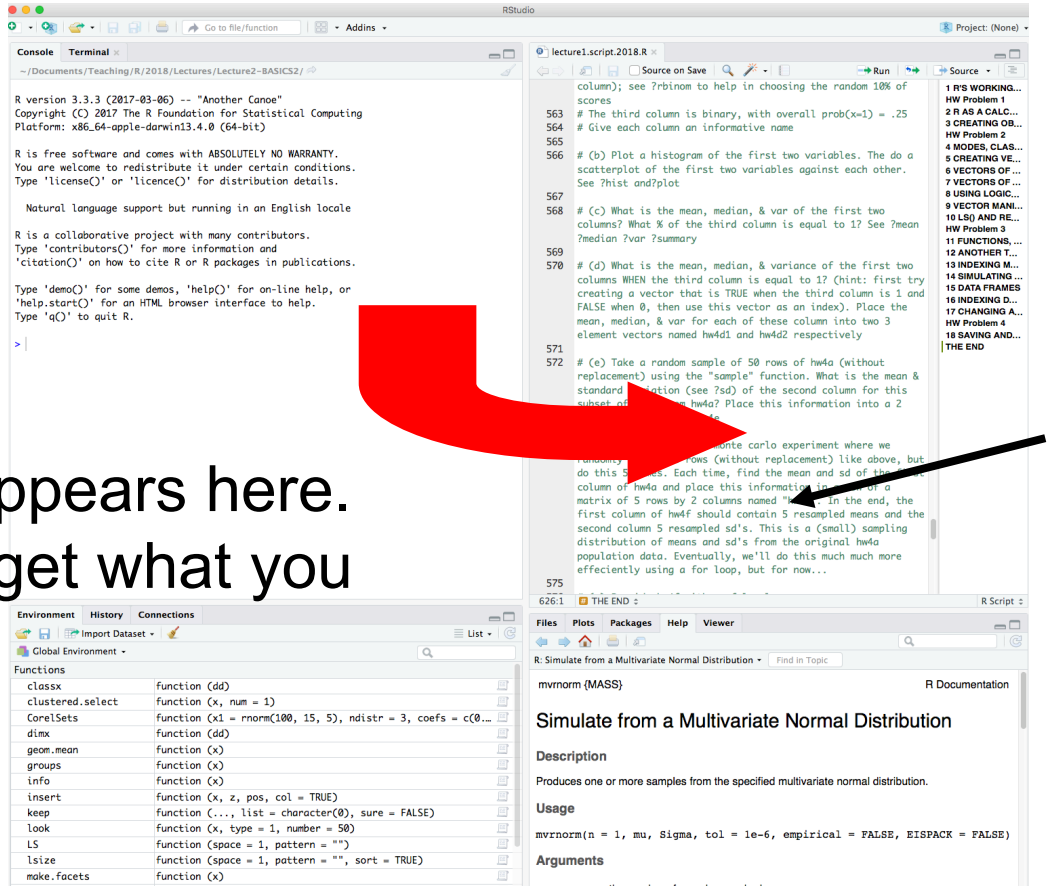


Output appears here. Did you get what you wanted?

Write small bits of code here and run it

Typical R session

- R sessions are *interactive*

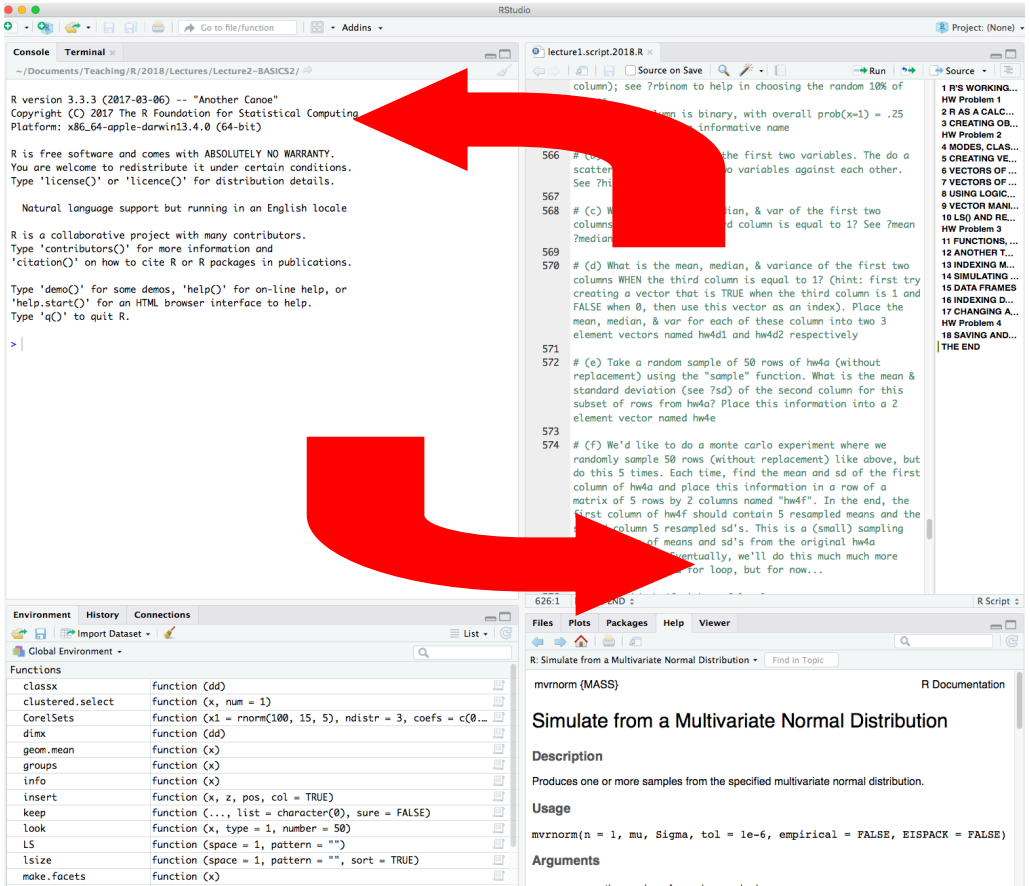


Output appears here. Did you get what you wanted?

Adjust your syntax here depending on this answer.

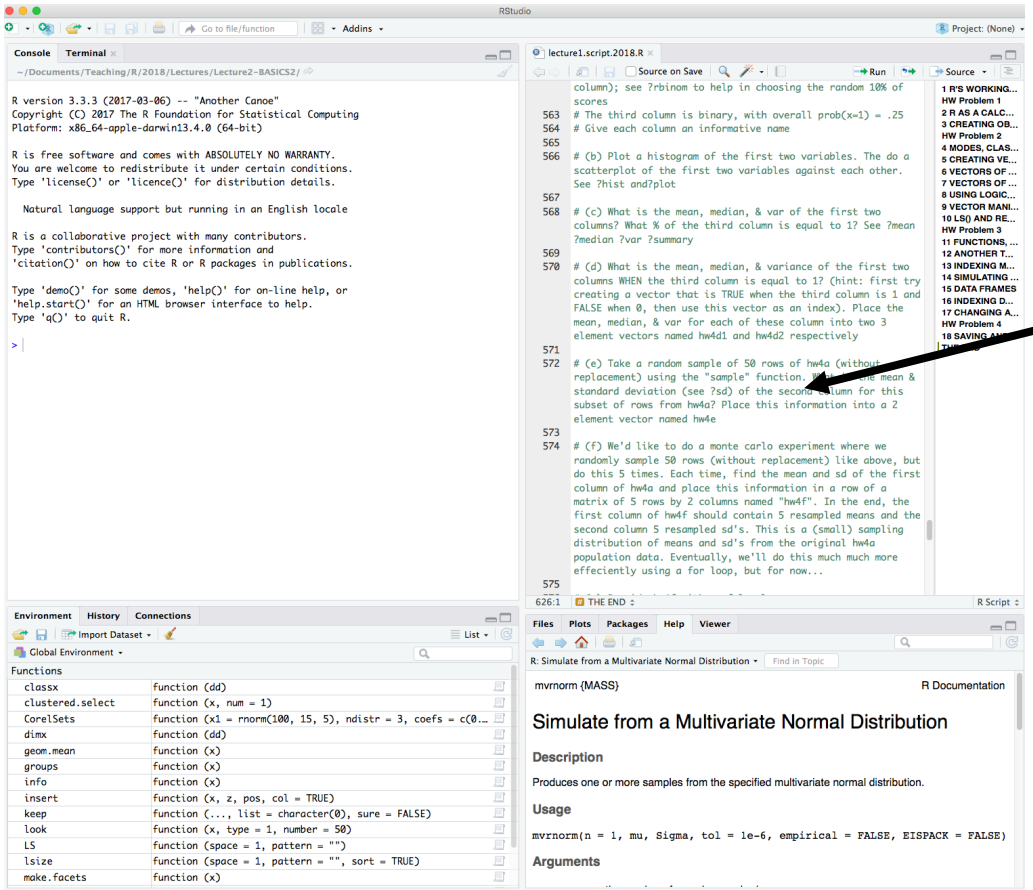
Typical R session

- R sessions are *interactive*



Typical R session

- R sessions are *interactive*



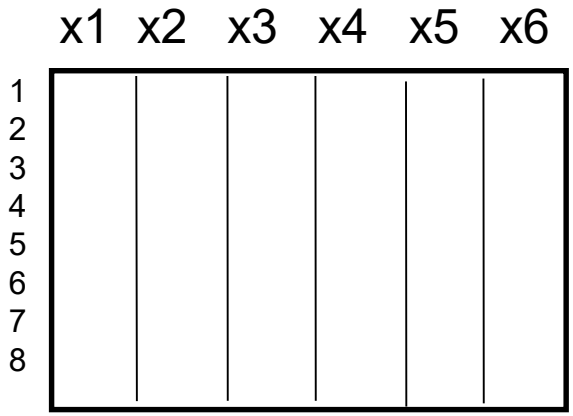
At end, all you need to do is save your script file(s) - which can easily be rerun later.

R Objects

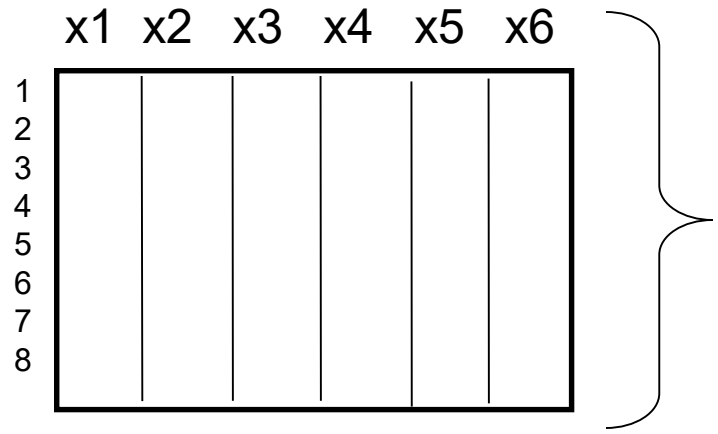
- Almost all things in R – functions, datasets, results, etc. – are OBJECTS.
  - (graphics are written out and are not stored as objects)
- Script can be thought of as a way to make objects. Your goal is usually to write a script that, by its end, has created the objects (e.g., statistical results) and graphics you need.
- Objects are classified by two criteria:
  - MODE: how objects are stored in R - character, numeric, logical, list, & function
  - CLASS: how objects are treated by functions (important to know!) - [vector], matrix, array, factor, data.frame, & 1000s of special classes created by specific functions

R Objects

Z <-



R Objects



The MODE of Z is determined automatically by the types of things stored in Z – numbers, characters, etc. Vectors & matrices must have their values all of the same mode. Lists can be a mix of modes.

R modes (to check, use mode() function):

numeric – numbers

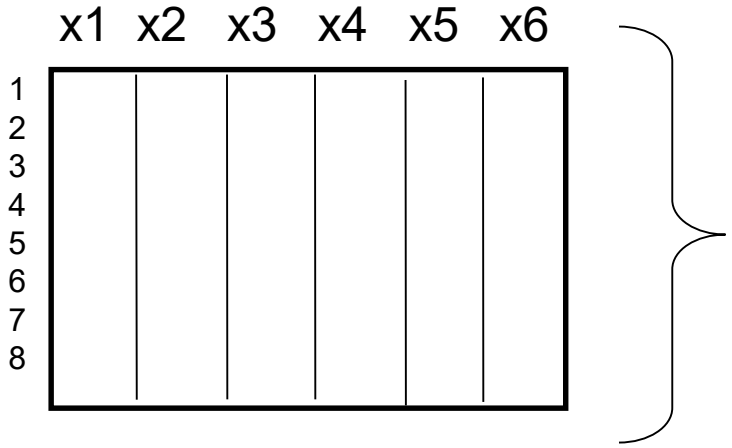
character

list – a concatenation of elements of different modes

logical – TRUE/FALSE

function

R Classes



The CLASS of Z is either set by default depending, on how it was created, or is explicitly set by user. You can check the objects' class and change it. It determines how functions deal with Z. If of class "lm", R searches for a function fun.lm

NOTE: If an object has two classes - c("first", "second") - R searches for a function called fun.first and, if it finds it, applies it to the object. If no such function is found, a function called fun.second is tried. If no class name produces a suitable function, the function fun.default is used.

R classes (to check, use class() function):

[for **vectors**, mode & class are same] - logical, numeric, character
[modes & class are same for these 2 as well] - function, list (when generic)
factor
matrix
array
data.frame

Learning R

- Read through the CRAN website & intro manual
- Know your objects' modes & classes: mode(x); class(x)
- Because R is interactive, errors are your friends!
- ?lm gives you help on lm function. Reading help files can be very… helpful
- MOST IMPORTANT - the more time you spend using R, the more comfortable you become with it. After doing your first real project in R, you won't look back. I promise.

Recommended Book

- An R and S-PLUS Companion to Applied Regression: An excellent overview of R, not just regression in R. Highly recommended. Many of the HWs we will do were inspired by Fox's book. If you are the type of person who likes to have a book, buy this one. $56 at Amazon.