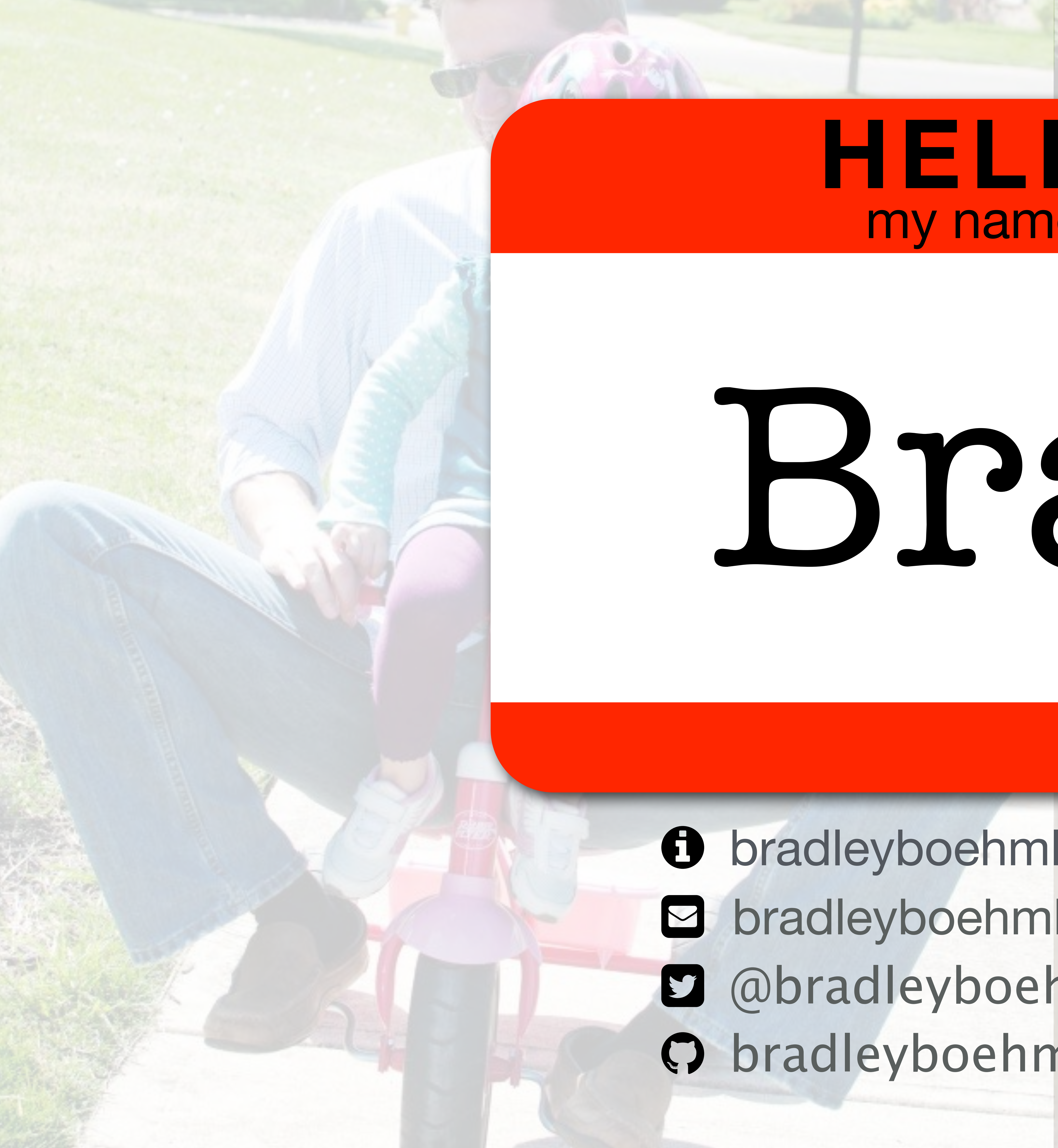
R PROGRAMMING

An Introduction

INTRODUCTIONS



HELLO
my name is

Brad

bradleyboehmke.github.io

bradleyboehmke@gmail.com

@bradleyboehmke

bradleyboehmke

COURSE OBJECTIVES

Provides an intensive, hands-on introduction to the R programming language.

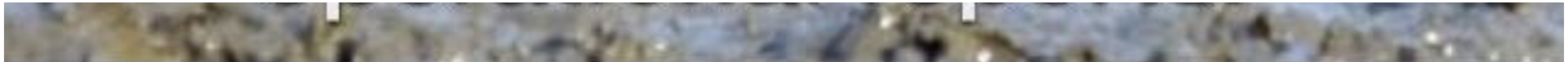
- Get up and running with R & RStudio
- Understand the basics of data prep
- Perform data transformation
- Visualize data
- Compute basic statistical inference

SCHEDULE

8:00 - 9:00:	Intro
9:00 - 10:00:	Data Prep
10:00 - 10:20:	Break
10:20 - 12:00:	Data Transformation
12:00 - 1:30:	Lunch
1:30 - 2:45:	Visualization
2:45 - 3:00:	Break
3:00 - 4:30:	Interactive Tutorial

Lots of hands-on coding exercises

You will be overwhelmed!

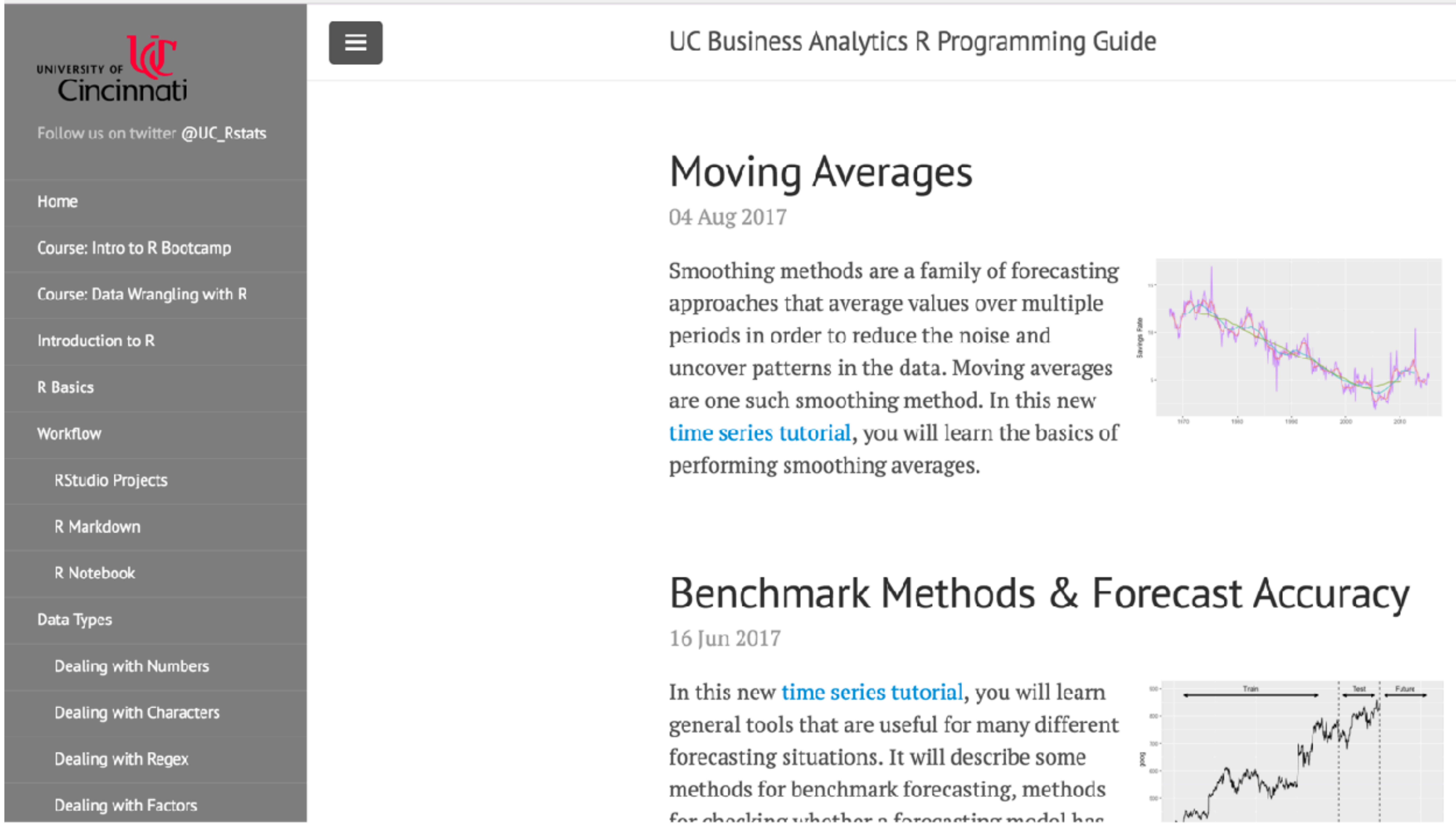


Strong proponent of collaborative work!

CLASS MATERIAL

https://uc-r.github.io

- Class material
- Tutorials
- Additional resources links
- Exercises



UC Business Analytics R Programming Guide

Moving Averages

04 Aug 2017

Smoothing methods are a family of forecasting approaches that average values over multiple periods in order to reduce the noise and uncover patterns in the data. Moving averages are one such smoothing method. In this new time series tutorial, you will learn the basics of performing smoothing averages.

Benchmark Methods & Forecast Accuracy

16 Jun 2017

In this new time series tutorial, you will learn general tools that are useful for many different forecasting situations. It will describe some methods for benchmark forecasting, methods for checking whether a forecasting model has

PROGRAMMING & ANALYSIS

Download and install R, a free software environment for statistical computing and graphics from CRAN, the Comprehensive R Archive Network. It is highly recommended to install a precompiled binary distribution for your operating system; follow these instructions:

1. Go to https://cran.r-project.org/
2. Click "Download R for Mac/Windows"
3. Download the appropriate file:
   - Windows users click Base, and download the installer for the latest R version
   - Mac users select the file R-3.X.X.pkg that aligns with your OS version
4. Follow the instructions of the installer.

PROGRAMMING & ANALYSIS

Install RStudio's IDE (stands for integrated development environment), a powerful user interface for R. RStudio includes a text editor, so you do not have to install another stand-alone editor. Follow these instructions:

1. Go to RStudio for desktop https://www.rstudio.com/products/rstudio/download/
2. Select the install file for your OS
3. Follow the instructions of the installer.

There are other R IDE's available: Emacs, Microsoft R Open, Notepad++, etc; however, I have found RStudio to be my preferred route. When you are done installing RStudio click on the icon.

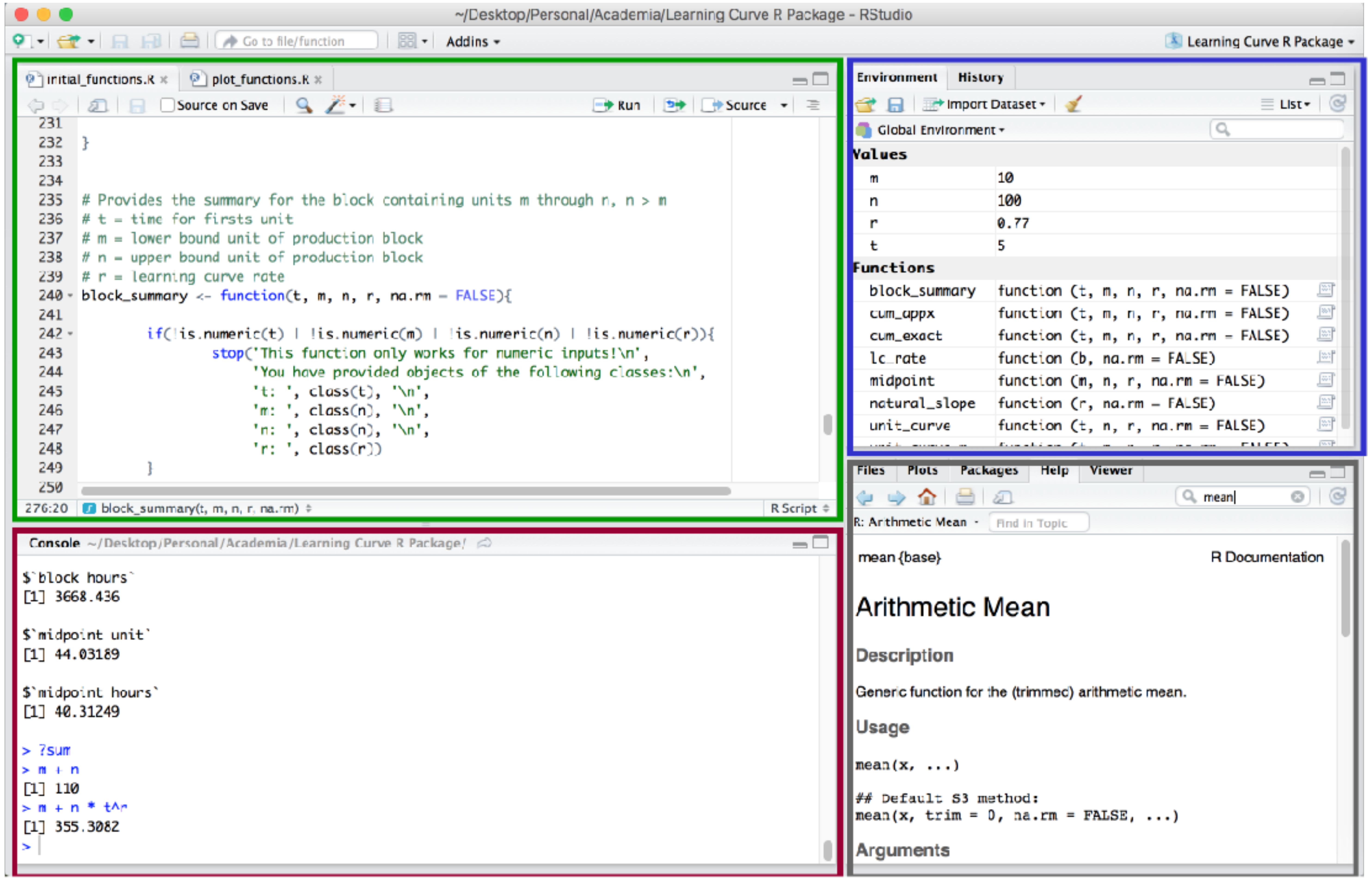
QUESTIONS ABOUT THE CLASS?

FUNDAMENTALS

UNDERSTANDING THE RSTUDIO IDE

Script files
- Saves your script
- Allows code & comments
- Can have multiple files open at a time

Console/Command line
- Can use as calculator
- Does not save code
- This is where your output is displayed



```r
}


# Provides the summary for the block containing units m through n, n > m
# t = time for firsts unit
# m = lower bound unit of production block
# n = upper bound unit of production block
# r = learning curve rate
block_summary <- function(t, m, n, r, na.rm = FALSE){

        if(!is.numeric(t) | !is.numeric(m) | !is.numeric(n) | !is.numeric(r)){
                stop('This function only works for numeric inputs!\n',
                     'You have provided objects of the following classes:\n',
                     't: ', class(t), '\n',
                     'm: ', class(n), '\n',
                     'n: ', class(n), '\n',
                     'r: ', class(r))
        }

```

Values

m	10
n	100
r	0.77
t	5

Functions

block_summary	function (t, m, n, r, na.rm = FALSE)
cum_appx	function (t, m, n, r, na.rm = FALSE)
cum_exact	function (t, m, n, r, na.rm = FALSE)
lc_rate	function (b, na.rm = FALSE)
midpoint	function (m, n, r, na.rm = FALSE)
natural_slope	function (r, na.rm = FALSE)
unit_curve	function (t, n, r, na.rm = FALSE)

Console ~/Desktop/Personal/Academia/Learning Curve R Package/

```
$`block hours`
[1] 3668.436

$`midpoint unit`
[1] 44.03189

$`midpoint hours`
[1] 40.31249

> ?sum
> m + n
[1] 110
> m + n * t^r
[1] 355.3082
>
```

mean {base} R Documentation

Arithmetic Mean

Description

Generic function for the (trimmed) arithmetic mean.

Usage

```
mean(x, ...)

## Default S3 method:
mean(x, trim = 0, na.rm = FALSE, ...)
```

Arguments

Workspace environment
- Holds your objects
- Can review history

Misc - Displays:
- files in working directory
- plots when produced
- help files/search

Thorough tutorial regarding the RStudio console: http://dss.princeton.edu/training/RStudio101.pdf

YOUR TURN!

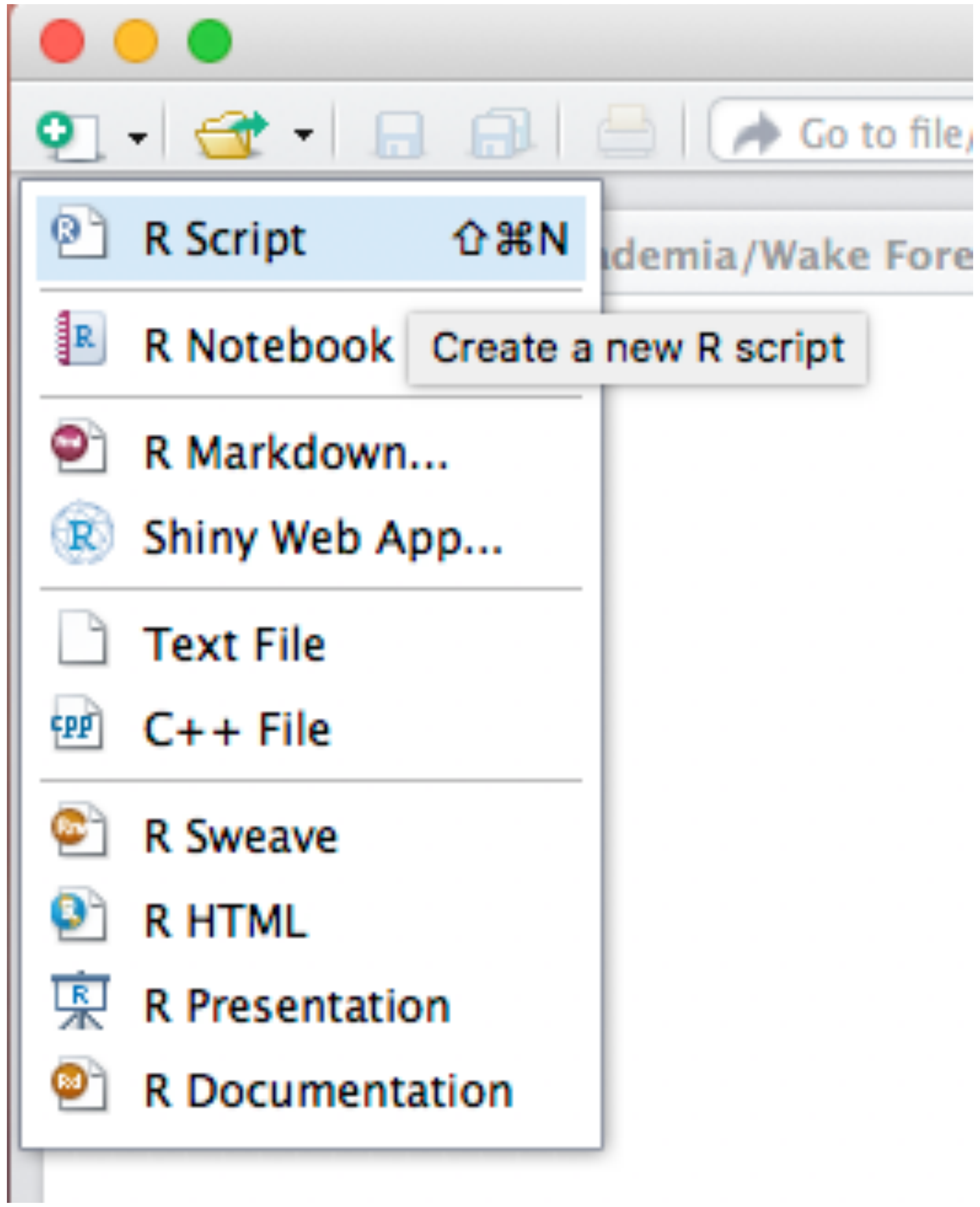
1. *Create a new .R script named my_first_script.R*

2. *Write and execute the following code in the .R script and identify where in the IDE the outputs can be found.*

```
mtcars
?sum
hist(mtcars$mpg)
avg_mpg <- mean(mtcars$mpg)
random_numbers <- runif(25)
history()
```

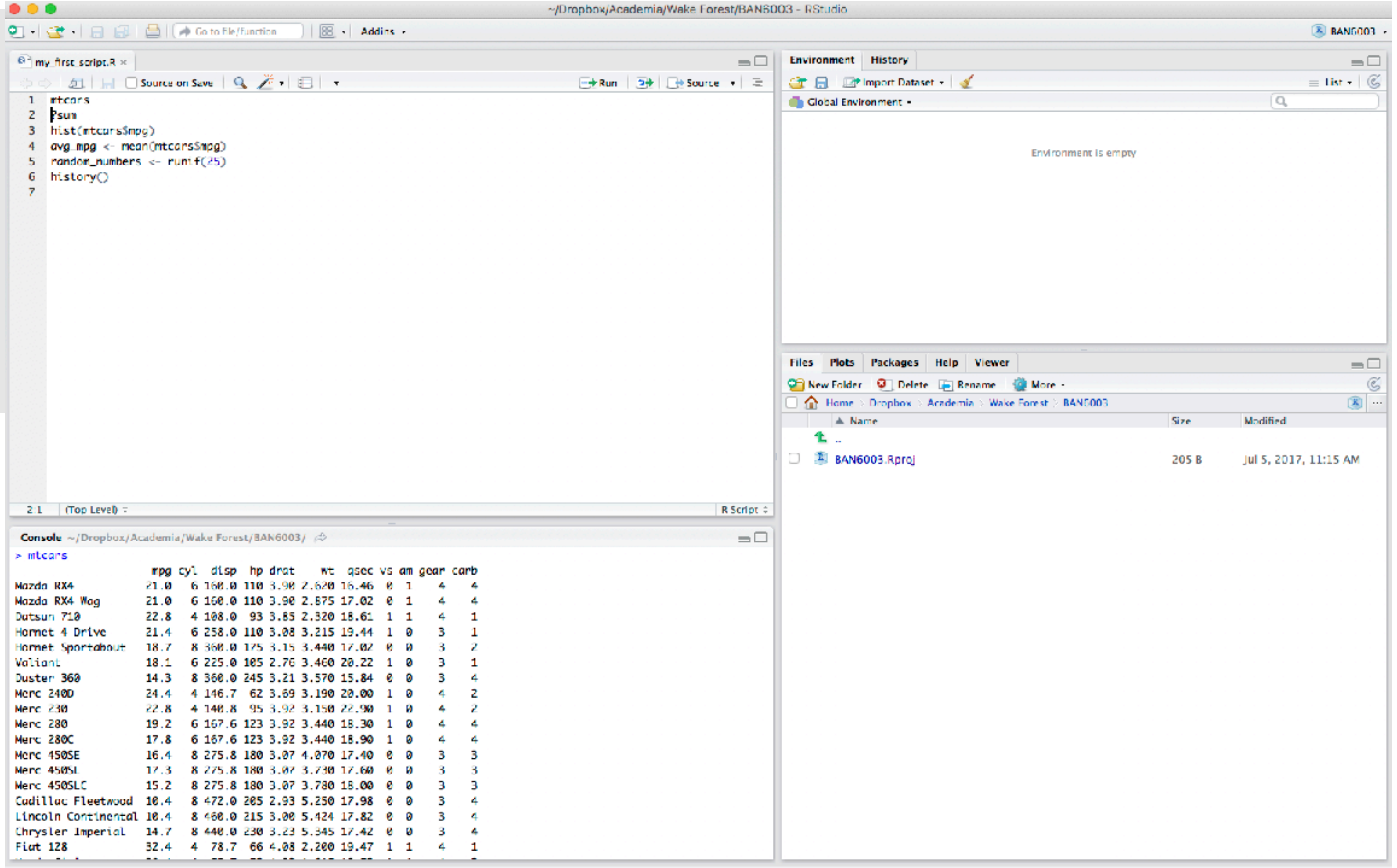
SOLUTION

1. *Create a new .R script*



SOLUTION

```
mtcars
?sum
hist(mtcars$mpg)
avg_mpg <- mean(mtcars$mpg)
random_numbers <- runif(25)
history()
```



```
mtcars
?sum
hist(mtcars$mpg)
avg_mpg <- mean(mtcars$mpg)
random_numbers <- runif(25)
history()
```

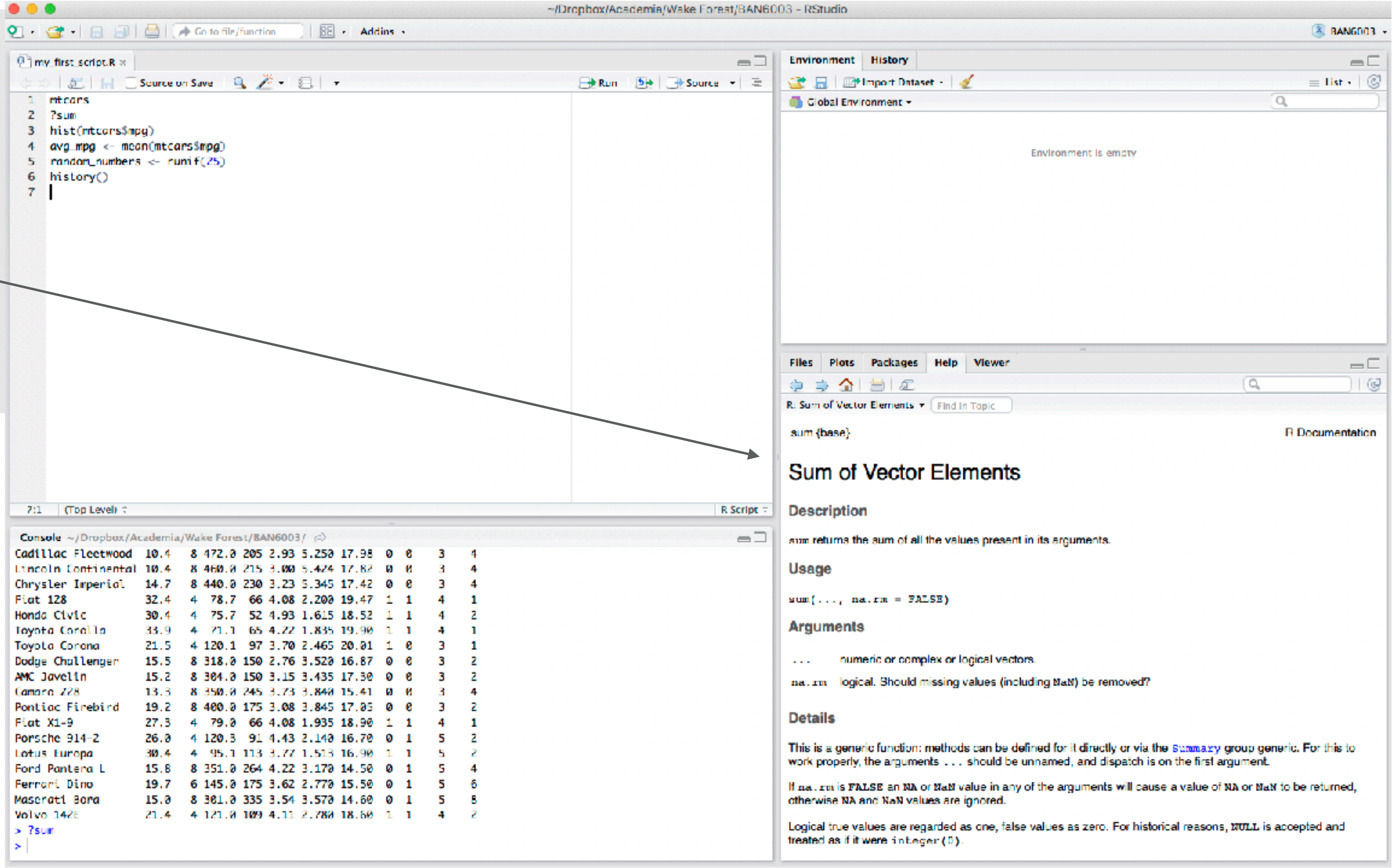
```
> mtcars
                     mpg cyl  disp  hp drat    wt  qsec vs am gear carb
Mazda RX4           21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag       21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
Datsun 710          22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive      21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout   18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
Valiant             18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
Duster 360          14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
Merc 240D           24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
Merc 230            22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
Merc 280            19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
Merc 280C           17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
Merc 450SE          16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
Merc 450SL          17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
Merc 450SLC         15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
Fiat 128            32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
```

Environment is empty

Name	Size	Modified
..		
BAN6003.Rproj	205 B	Jul 5, 2017, 11:15 AM

SOLUTION

```
mtcars
?sum
hist(mtcars$mpg)
avg_mpg <- mean(mtcars$mpg)
random_numbers <- runif(25)
history()
```



```
mtcars
?sum
hist(mtcars$mpg)
avg_mpg <- mean(mtcars$mpg)
random_numbers <- runif(25)
history()
```

Console ~/Dropbox/Academia/Wake Forest/BAN6003/

```
Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
Fiat 128            32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
Honda Civic         30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
Toyota Corolla      33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
Toyota Corona       21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
Dodge Challenger    15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
AMC Javelin         15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
Camaro Z28          13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
> ?sum
>
```

sum {base}   R Documentation

Sum of Vector Elements

Description

`sum` returns the sum of all the values present in its arguments.

Usage

```
sum(..., na.rm = FALSE)
```

Arguments

`...` numeric or complex or logical vectors.

`na.rm` logical. Should missing values (including `NaN`) be removed?

Details

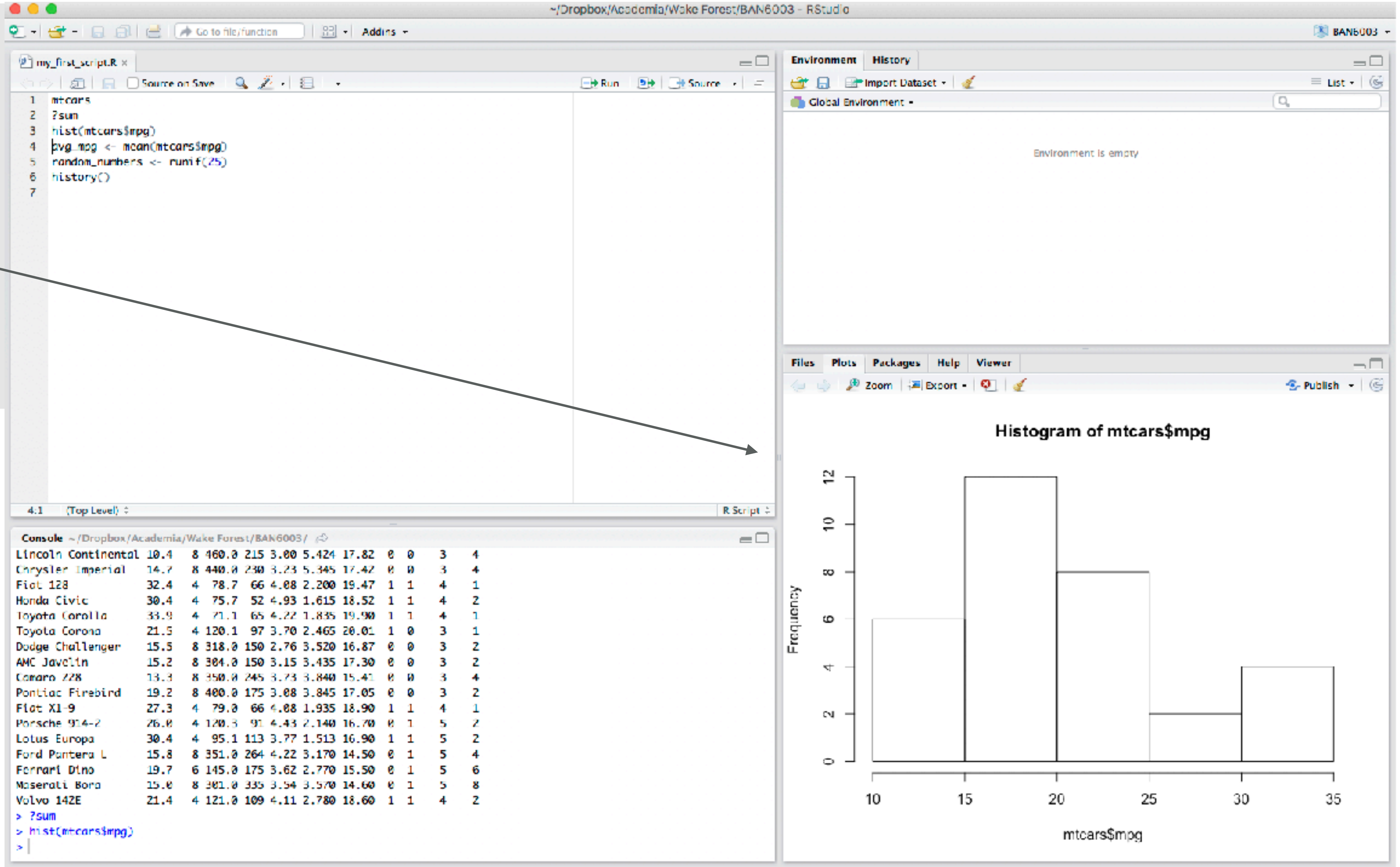
This is a generic function: methods can be defined for it directly or via the `Summary` group generic. For this to work properly, the arguments `...` should be unnamed, and dispatch is on the first argument.

If `na.rm` is `FALSE` an `NA` or `NaN` value in any of the arguments will cause a value of `NA` or `NaN` to be returned, otherwise `NA` and `NaN` values are ignored.

Logical true values are regarded as one, false values as zero. For historical reasons, `NULL` is accepted and treated as if it were `integer(0)`.

SOLUTION

```
mtcars
?sum
hist(mtcars$mpg)
avg_mpg <- mean(mtcars$mpg)
random_numbers <- runif(25)
history()
```



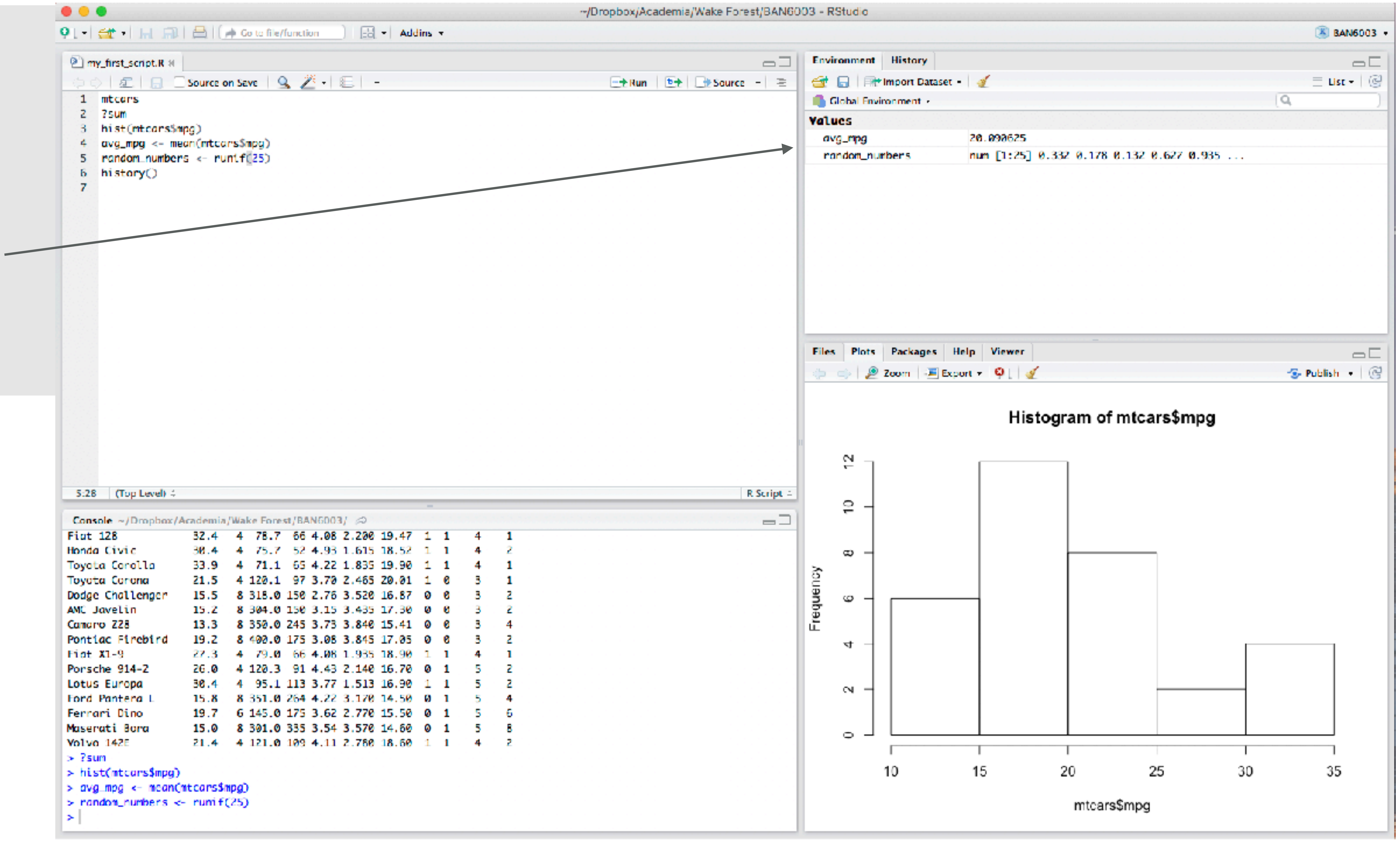
```
mtcars
?sum
hist(mtcars$mpg)
avg_mpg <- mean(mtcars$mpg)
random_numbers <- runif(25)
history()
```

Console ~/Dropbox/Academia/Wake Forest/BAN6003/

```
Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
Fiat 128            32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
Honda Civic         30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
Toyota Corolla      33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
Toyota Corona       21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
Dodge Challenger    15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
AMC Javelin         15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
Camaro Z28          13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
> ?sum
> hist(mtcars$mpg)
> 
```

SOLUTION

```
mtcars
?sum
hist(mtcars$mpg)
avg_mpg <- mean(mtcars$mpg)
random_numbers <- runif(25)
history()
```



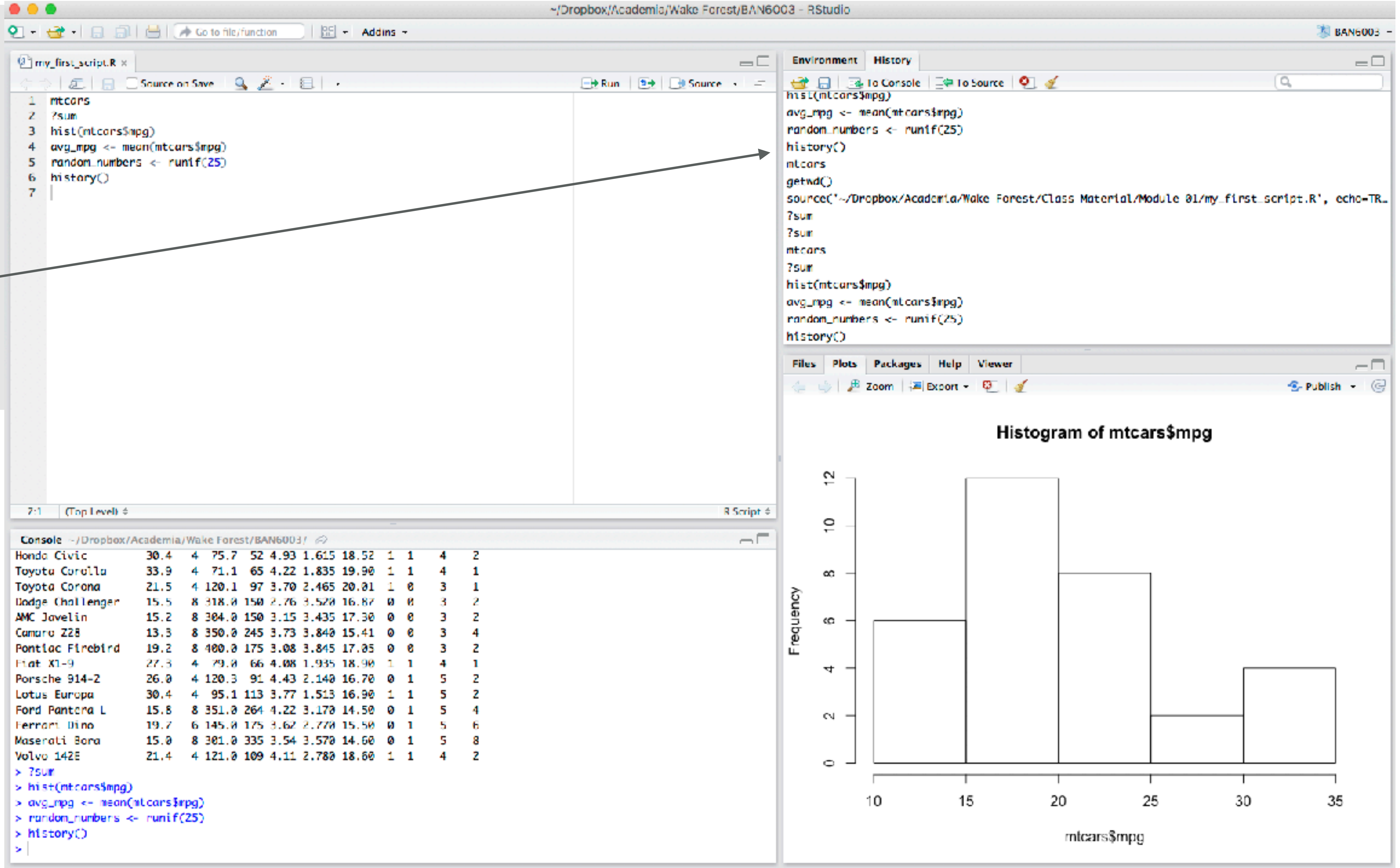
```
mtcars
?sum
hist(mtcars$mpg)
avg_mpg <- mean(mtcars$mpg)
random_numbers <- runif(25)
history()
```

```
Values
avg_mpg          20.090625
random_numbers   num [1:25] 0.332 0.178 0.132 0.627 0.935 ...
```

```
Console ~/Dropbox/Academia/Wake Forest/BAN6003/
Fiat 128            32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
Honda Civic         30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
Toyota Corolla      33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
Toyota Corona       21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
Dodge Challenger    15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
AMC Javelin         15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
Camaro Z28          13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
> ?sum
> hist(mtcars$mpg)
> avg_mpg <- mean(mtcars$mpg)
> random_numbers <- runif(25)
>
```

SOLUTION

```
mtcars
?sum
hist(mtcars$mpg)
avg_mpg <- mean(mtcars$mpg)
random_numbers <- runif(25)
history()
```



```
mtcars
?sum
hist(mtcars$mpg)
avg_mpg <- mean(mtcars$mpg)
random_numbers <- runif(25)
history()

```

```
hist(mtcars$mpg)
avg_mpg <- mean(mtcars$mpg)
random_numbers <- runif(25)
history()
mtcars
getwd()
source('~/Dropbox/Academia/Wake Forest/Class Material/Module 01/my_first_script.R', echo=TR…
?sum
?sum
mtcars
?sum
hist(mtcars$mpg)
avg_mpg <- mean(mtcars$mpg)
random_numbers <- runif(25)
history()
```

```
Honda Civic        30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
Toyota Corolla     33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
Toyota Corona      21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
Dodge Challenger   15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
AMC Javelin        15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
Camaro Z28         13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
Pontiac Firebird   19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
Fiat X1-9          27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
Porsche 914-2      26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
Lotus Europa       30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
Ford Pantera L     15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
Ferrari Dino       19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
Maserati Bora      15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
Volvo 142E         21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
> ?sum
> hist(mtcars$mpg)
> avg_mpg <- mean(mtcars$mpg)
> random_numbers <- runif(25)
> history()
> 
```

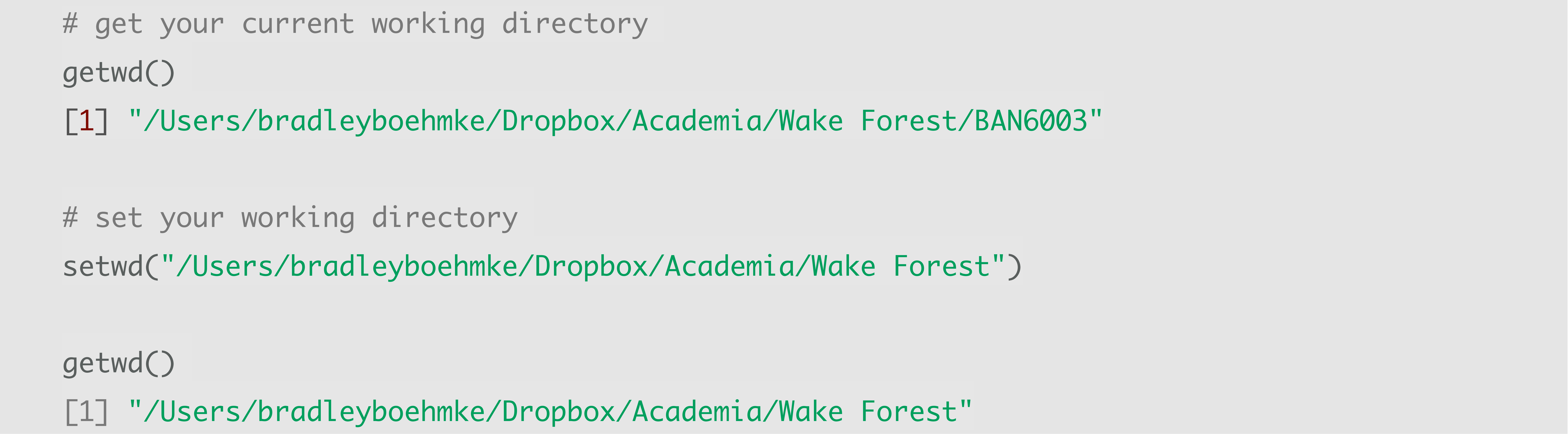
GETTING HELP

```
# provides details for specific function
help(sqrt)

# provides same information as help(functionname)
?sqrt

# provides examples for said function
example(sqrt)
```

External to R:

Google: just add "with R" at the end of any search.
Stack Overflow: a searchable Q&A site oriented toward programming issues. 75% of my answers come from SO
Cross Validated: a searchable Q&A site oriented toward statistical analysis.
R-bloggers: a central hub of content from over 500 bloggers who provide news and tutorials about R.

SET YOUR WORKING DIRECTORY



```
# get your current working directory
getwd()
[1] "/Users/bradleyboehmke/Dropbox/Academia/Wake Forest/BAN6003"

# set your working directory
setwd("/Users/bradleyboehmke/Dropbox/Academia/Wake Forest")

getwd()
[1] "/Users/bradleyboehmke/Dropbox/Academia/Wake Forest"
```

Keeping your files organized is critical

YOUR TURN!

Set your working directory to the "Intro to R Bootcamp" folder you downloaded for this class.

R AS A CALCULATOR

```
# Uses PEMDAS convention for order of operations
4 + 3 / 10 ^ 2
## [1] 4.03
4 + (3 / 10 ^ 2)
## [1] 4.03
(4 + 3) / 10 ^ 2
## [1] 0.07

# large/small numbers will be displayed in scientific notation
1 / 17 ^ 7
## [1] 2.437011e-09

# Undefined calculations result in Inf or NaN
1 / 0
## [1] Inf
Inf - Inf
## [1] NaN
```

THE ASSIGNMENT (<-) OPERATOR

```
x <- 3     # GOOD
x = 3      # BAD


# we can increment (build onto) existing objects
x <- x + 1
x
## [1] 4


# must be specific
X
Error: object 'x' not found
```

YOUR TURN!

Economic Order Quantity Model:

$$Q = \sqrt{\frac{2DK}{h}}$$

*Calculate **Q** where:*

D = 1000

K = 5

h = 0.25

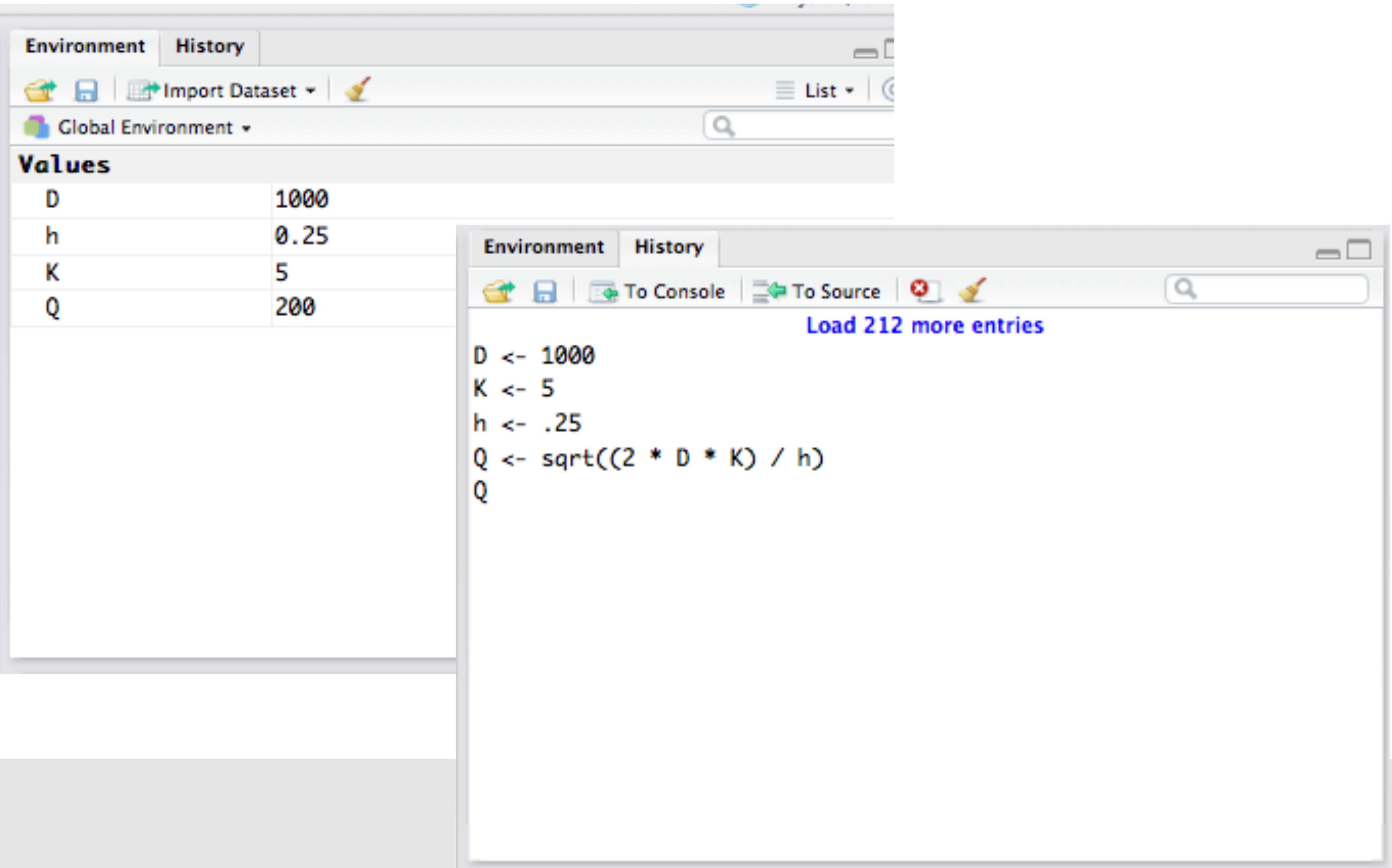
hint: `sqrt(x)` = $\sqrt{x}$

SOLUTION

```
D <- 1000
K <- 5
h <- .25

Q <- sqrt((2 * D * K) / h)
Q
## [1] 200
```

WORKSPACE ENVIRONMENT

- You should now have 4 objects in your global environment
- History tab will show your recent code



Values

```
D        1000
h        0.25
K        5
Q        200
```

```
D <- 1000
K <- 5
h <- .25
Q <- sqrt((2 * D * K) / h)
Q
```

To list and remove objects in your global environment:

```
# list all objects
ls()
## [1] "D" "h" "K" "Q"

# remove defined object from the environment
rm(D)

# removes everything in the working environment -- use with caution!
rm(list = ls())
```

PACKAGES

The fundamental unit of shareable code is the package.

CRAN:	10,000+
Bioconductor:	1,000+
GitHub:	Many more plus beta versions for updated packages not yet published

So how do we install these packages?

```
# install packages from CRAN
install.packages("packagename")

# install packages from Bioconductor
source("http://bioconductor.org/biocLite.R")          # only required the first time
biocLite()                                            # only required the first time
biocLite("packagename")

# install packages from GitHub
install.packages("devtools")                          # only required the first time
devtools::install_github("username/packagename")
```

YOUR TURN!

Download these packages from CRAN:

```
tidyverse
nycflights13
```

SOLUTION

```
install.packages("tidyverse")
install.packages("nycflights13")


# alternative
install.packages(c(“tidyverse”, “nycflights13”))
```

For a full list of useful packages see this guide: [*http://bit.ly/1x9vkzV*](http://bit.ly/1x9vkzV)

PACKAGES

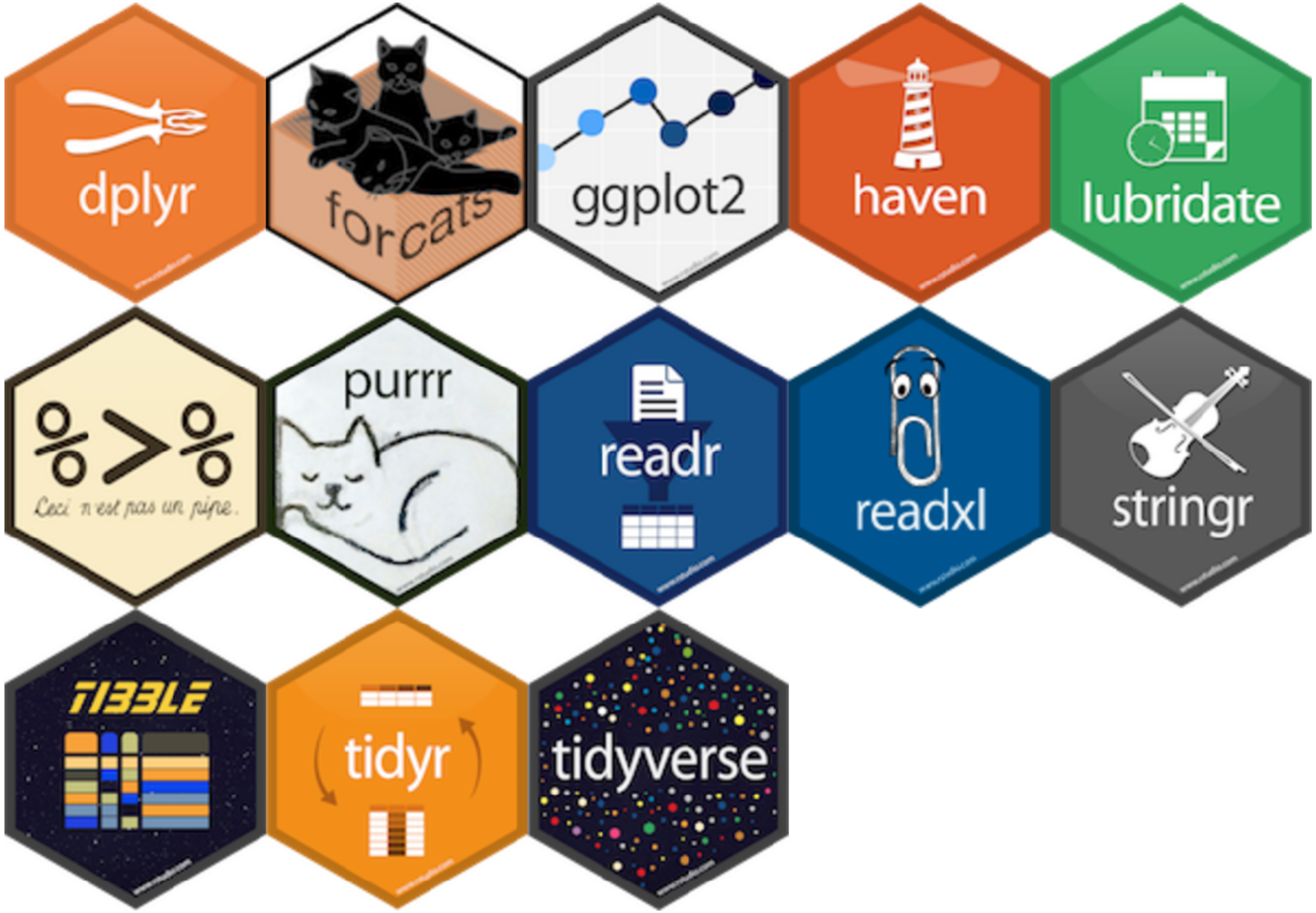
Loading packages:

```
# load the package to use in the current R session
library(tidyverse)

# use a particular function within a package without loading the package
stringr::str_replace()
```

Getting help on packages:

```
# provides details regarding contents of a package
help(package = "tidyr")

# list vignettes available for a specific package
vignette(package = "tidyr")

# view specific vignette
vignette(“tidy-data")
```

WHAT IS TIDYVERSE?



WHAT TO REMEMBER

FUNCTIONS TO REMEMBER

Operator/Function	Description
`help(), ?, example()`	Get help on functions and provide examples
`getwd(), setwd()`	Get and set your working directory
`+, -, *, /, ^`	Arithmetic
`<-`	Assignment operator
`ls(), rm()`	list and remove objects in your global environment
`install.packages(), library()`	Install and load packages
`vignette()`	View/list package vignette