CONJ 620: Day 1

Intro to R & Rstudio

Meike Niederhausen, PhD

Welcome!

Goals for today

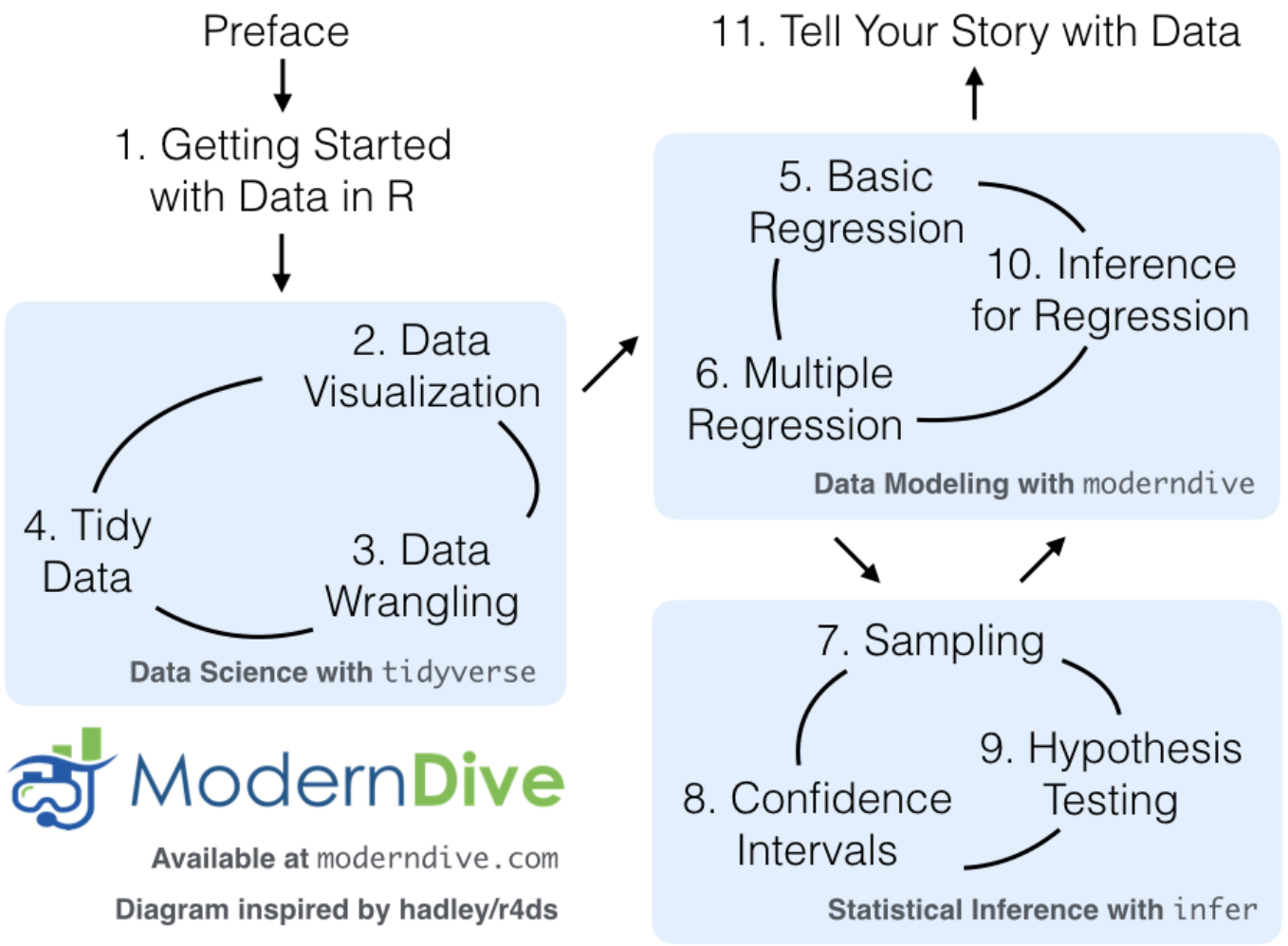
- CONJ 620 overview
- Introduction to R & RStudio
    - Basic coding in the console
    - Creating reproducible reports with R markdown
    - Overview of datasets in R
    - R packages: installing and loading packages

Learning outcomes

- Develop and clearly describe a statistical analysis plan, from start to finish.
- Work independently with data, including wrangling, tidying, visualizing, and basic analyses.
- Accurately interpret results from basic inferential statistical analyses.
- Understand assumptions and limitations to statistical procedures.
- Know about the types/scope of questions one can ask about data.
- Identify statistical questions/problems that require advanced methods out of skill set.
- Proficient use of tools and vocabulary to facilitate effective collaboration with advisors, biostatisticians, and other team members.

What does the class *not* cover?

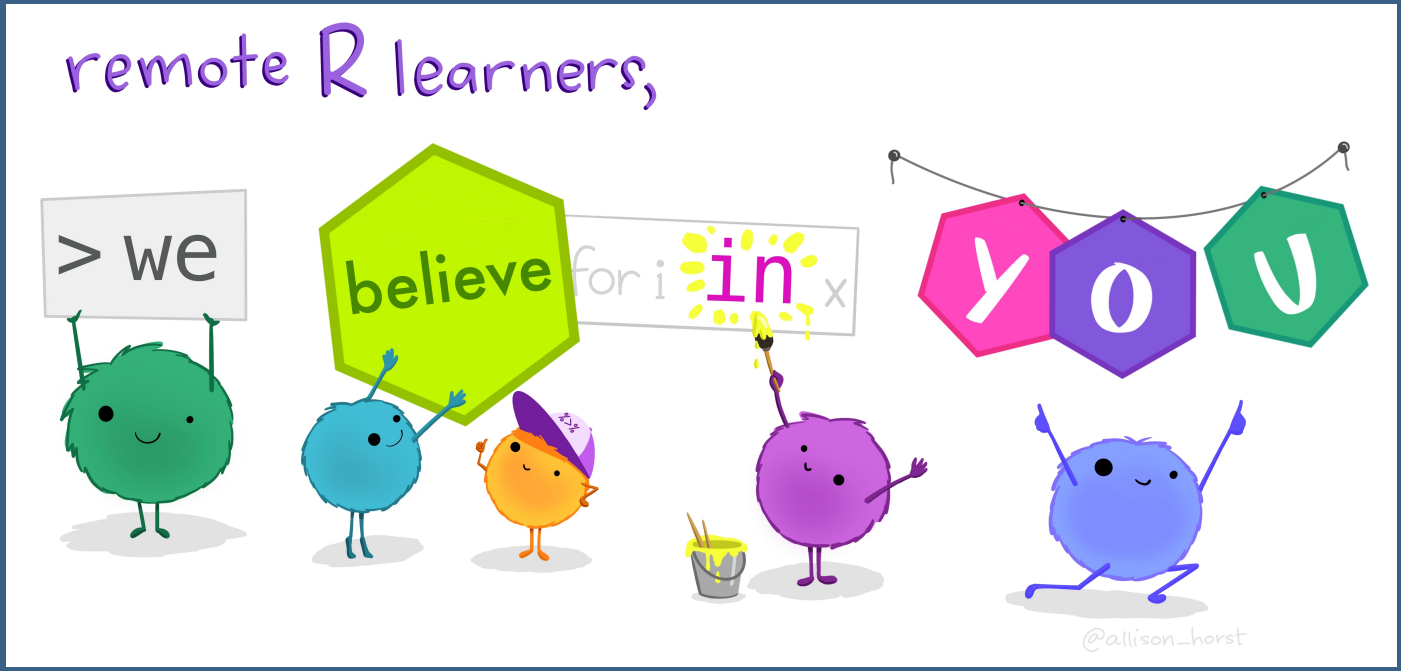
ModernDive textbook outline



Poll Questions

- Which program are you in?
- Which year are you?
- Have you taken a statistics class before?
- Have you used R?
- What other statistical software have you used?
- Has anyone used other programming languages (C, java, python, etc)?

Introduction to R



Allison Horst

What is R?

- A programming language
- Focus on statistical modeling and data analysis
  - import data, manipulate data, run statistics, make plots
- Useful for data science
- Great visualizations
- Also useful for most anything else you'd want to tell a computer to do
- Interfaces with other languages i.e. python, C++, bash

For the history and details: Wikipedia

- an interpreted language (run it through a command line)
- procedural programming with functions
- Why "R"?? Scheme inspired S (invented at Bell Labs in 1976) which inspired R (**free and open source!** in 1992)

What is RStudio?

R is a programming language

RStudio is an integrated development environment (IDE)
= an interface to use R (with perks!)

R: Engine	RStudio: Dashboard
	

Modern Dive

Open RStudio on your computer (not R!)

1.1.2 Using R via RStudio

Recall our car analogy from earlier. Much as we don't drive a car by interacting directly with the engine but rather by interacting with elements on the car's dashboard, we won't be using R directly but rather we will use RStudio's interface. After you install R and RStudio on your computer, you'll have two new *programs* (also called *applications*) you can open. We'll always work in RStudio and not in the R application. Figure 1.2 shows what icon you should be clicking on your computer.

R: Do not open this	RStudio: Open this
	

FIGURE 1.2: Icons of R versus RStudio on your computer.

BuzzR

RStudio anatomy

https://buzzrbeeline.blog/

Emma Rand

Script file

Write code here

To run code put your cursor on the line and click the run button

Edit to correct errors

⇒ record of commands that worked

Save scripts with the **.R** extension

⇒ syntax will be highlighted

⇒ good practice

<- is the assignment operator

⇒ puts what is on the right in to the object on the left

⇒ Assign results if you want to use them again

Console

When you click run, code is sent to the console and executed

> is the prompt

⇒ do not type it

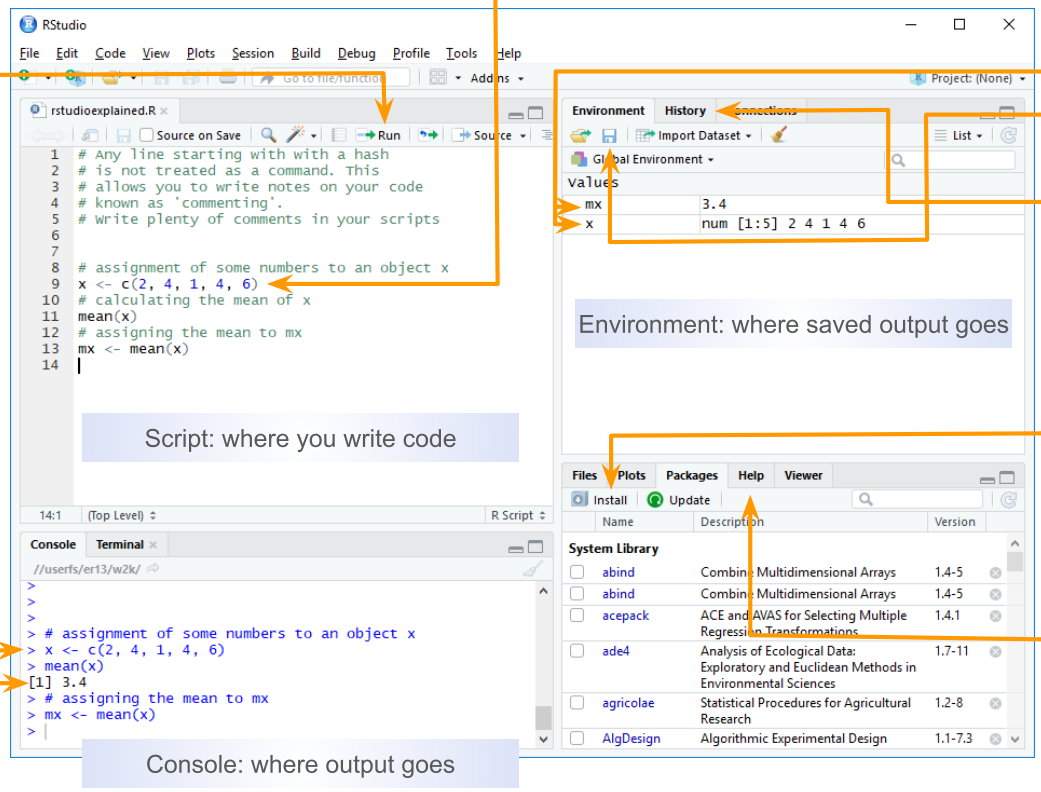
⇒ appears when R is ready for next command

Command output goes here by default

⇒ output is in a different colour

⇒ [1] indicates 3.4 is the first element of the output

⇒ many commands will not have output, the prompt just reappears



Environment

Name objects by assignment to use them again

All the objects you created in your session

Saving the environment saves all the objects, but not the code with a **.RData** extension

History

A history of every command you sent to the console, mistakes included.

File can be saved but usually you just need the script

Packages

Many functions come with R

A huge amount of extra functionality is available in packages

Packages can be installed by clicking the Install button

Help

Access to manual pages for all installed packages

Plots

Figure output appears here

Emma Rand

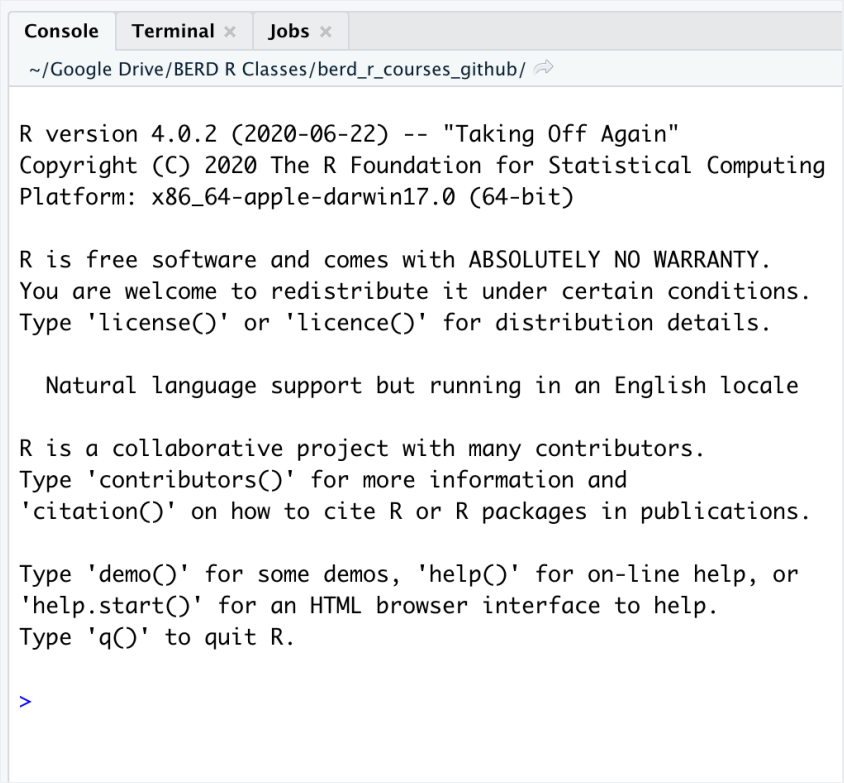
Read more about RStudio's layout in Section 3.4 of "Getting Used to R, RStudio, and R Markdown" (Ismay and Kennedy 2016)

Let's code! R Basics

Allison Horst

Coding in the console

When you first open R, the console should be empty.



Typing and executing code in the console

- Type code in the console (blue text)
- Press **return** to execute the code
- Output shown below in black

```
> 7
[1] 7
> 3 + 5
[1] 8
> "hello"
[1] "hello"
> # this is a comment, nothing happens
> # 5 - 8
> # separate multiple commands with ;
> 3 + 5; 4 + 8
[1] 8
[1] 12
>
```

Math calculations using R

```
> 10^2
```

```
[1] 100
```

```
> 3 ^ 7
```

```
[1] 2187
```

```
> 6/9
```

```
[1] 0.6666667
```

```
> 9-43
```

```
[1] -34
```

- Rules for order of operations are followed
- Spaces between numbers and characters are ignored

```
> 4^3-2* 7+9 /2
```

```
[1] 54.5
```

The equation above is computed as

$$4^3 - (2 \cdot 7) + \frac{9}{2}$$

Variables

Variables are used to store data, figures, model output, etc.

- Can assign a variable using either = or <-
  - **Using <- is preferable**
  - type name of variable to print

Assign just one value:

```
> x = 5
> x
```

```
[1] 5
```

```
> x <- 5
> x
```

```
[1] 5
```

Assign a **vector** of values:

- Consecutive integers using `:`

```
> a <- 3:10
> a
```

```
[1]  3  4  5  6  7  8  9 10
```

- **Concatenate** a string of numbers

```
> b <- c(5, 12, 2, 100, 8)
> b
```

```
[1]   5  12   2 100   8
```

We can do math with variables

Math using variables with just one value

```
> x <- 5
> x
```

```
[1] 5
```

```
> x + 3
```

```
[1] 8
```

```
> y <- x^2
> y
```

```
[1] 25
```

Math on vectors of values: **element-wise** computation

```
> a <- 3:6
> a
```

```
[1] 3 4 5 6
```

```
> a+2; a*3
```

```
[1] 5 6 7 8
```

```
[1]  9 12 15 18
```

```
> a*a
```

```
[1]  9 16 25 36
```

Variables can include text (characters)

```
> hi <- "hello"
> hi
```

```
[1] "hello"
```

```
> greetings <- c("Guten Tag", "Hola", hi)
> greetings
```

```
[1] "Guten Tag" "Hola"      "hello"
```

Using functions

- `mean()` is an example of a function
- functions have "arguments" that are specified within the `()`
- `?mean` in console will show help file for `mean()`

Function arguments specified by name:

```
> mean(x = 1:4)
```

```
[1] 2.5
```

```
> seq(from = 1, to = 12, by = 3)
```

```
[1]  1  4  7 10
```

```
> seq(by = 3, to = 12, from = 1)
```

```
[1]  1  4  7 10
```

Function arguments not specified, but listed in order:

```
> mean(1:4)
```

```
[1] 2.5
```

```
> seq(1,12,3)
```

```
[1]  1  4  7 10
```

Common console errors (1/2)

Incomplete commands

- When the console is waiting for a new command, the prompt line begins with >
  - If the console prompt is +, then a previous command is incomplete
  - You can finish typing the command in the console window

Example:

```
> 3 + (2*6
+ )
```

```
[1] 15
```

Common console errors (2/2)

Object is not found

- This happens when text is entered for a non-existent variable (object)

Example:

```
> hello
```

```
Error in eval(expr, envir, enclos): object 'hello' not found
```

- Can be due to missing quotes

```
> install.packages(dplyr) # need install.packages("dplyr")
```

```
Error in install.packages(dplyr): object 'dplyr' not found
```

Saving your code with R Markdown (Rmd)

or, creating reproducible reports

Allison Horst

R Markdown = .Rmd file = Code + text

`knitr` is a package that converts `.Rmd` files containing code + markdown syntax to a plain text `.md` markdown file, and then to other formats (html, pdf, Word, etc)

~/Google Drive/BERD R Classes/berd_rmarkdown_project - master - RStudio Source Editor

slides_ex.Rmd

```
1  ---
2  title: "Gapminder Report"
3  author: "Your Name"
4  date: "`r Sys.Date()`"
5  output:
6    html_document: default
7      keep_md:true
8  ---
9
10 ```{r setup, include=FALSE}
11 knitr::opts_chunk$set(echo = TRUE)
12 library(gapminder)
13 library(naniar)
14 library(tidyverse)
15 ```
16
17 # Background
18
19 This is an analysis of the gapminder data set with `r nrow(gapminder)` observations.
20
21 ```{r datasummary}
22 summary(gapminder)
23 ```
24
25 # Analysis
26
27 ## GDP vs Life Expectancy
28
29 ```{r}
30 ggplot(gapminder, aes(x = gdpPercap, y = lifeExp, color = continent)) +
31   geom_point()
32 ```
33
34
```

34:1 GDP vs Life Expectancy R Markdown

~/Google Drive/BERD R Classes/berd_rmarkdown_project/slides_example/slides_ex.html

slides_ex.html Open in Browser Find Publish

Gapminder Report

Your Name

2019-07-16

Background

This is an analysis of the gapminder data set with 1704 observations.

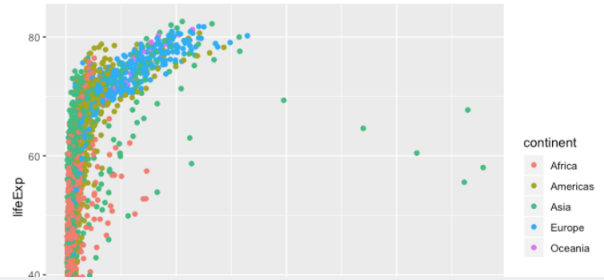
```
summary(gapminder)
```

```
##         country        continent        year         lifeExp
##  Afghanistan:  12   Africa  :624   Min.   :1952   Min.   :23.60
##  Albania    :  12   Americas:300   1st Qu.:1966   1st Qu.:48.20
##  Algeria    :  12   Asia    :396   Median :1980   Median :60.71
##  Angola     :  12   Europe  :360   Mean   :1980   Mean   :59.47
##  Argentina  :  12   Oceania : 24   3rd Qu.:1993   3rd Qu.:70.85
##  Australia  :  12                  Max.   :2007   Max.   :82.60
##  (Other)    :1632
##       pop              gdpPercap
##  Min.   :6.001e+04   Min.   :   241.2
##  1st Qu.:2.794e+06   1st Qu.:  1202.1
##  Median :7.024e+06   Median :  3531.8
##  Mean   :2.960e+07   Mean   :  7215.3
##  3rd Qu.:1.959e+07   3rd Qu.:  9325.5
##  Max.   :1.319e+09   Max.   :113523.1
##
```

Analysis

GDP vs Life Expectancy

```
ggplot(gapminder, aes(x = gdpPercap, y = lifeExp, color = continent)) +
  geom_point()
```



Basic R Markdown example

images from https://www.rstudio.com/products/rpackages/

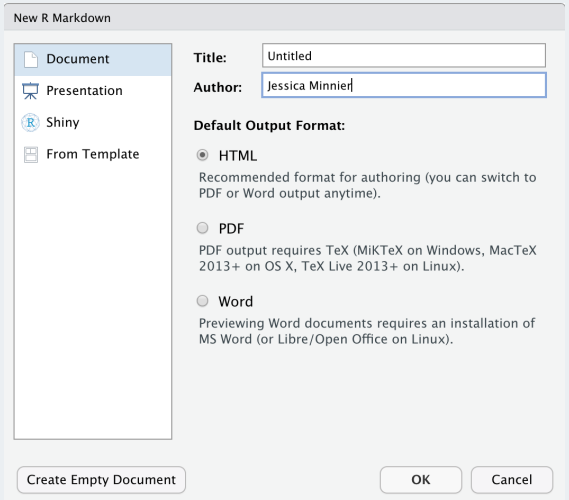
Create an R Markdown file (.Rmd)

Two options:

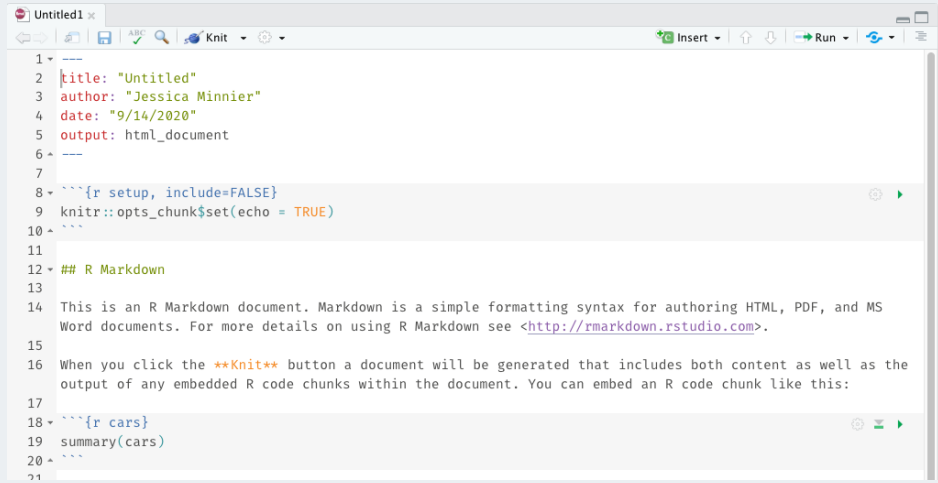
1. click on File → New File → R Markdown → OK , or
2. in upper left corner of RStudio click on → R Markdown...

Pop-up window:

- Enter a title and your name
- Keep default HTML output format
- Then click OK



- You should then see the following text in your editor window:



Save the Markdown file (`.Rmd`)

- **Save the file** by
  - selecting `File -> Save`,
  - or clicking on  (towards the left above the scripting window),
  - or keyboard shortcut
    - PC: *Ctrl + s*
    - Mac: *Command + s*

- You will need to specify
  - a **filename** to save the file as
    - ALWAYS use **.Rmd** as the filename extension for R markdown files
  - the **folder** to save the file in

Compare the .Rmd file with its html output

.Rmd file

default_html.Rmd

Knit | Insert | Run

```
---
title: "Untitled"
output: html_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```{r cars}
summary(cars)
```

Including Plots

You can also embed plots, for example:

```{r pressure, echo=FALSE}
plot(pressure)
```

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.
```

html output

~/Google Drive/BERD R Classes/berd_rmarkdown_project/default/default_html.html

default_html.html | Open in Browser | Find | Publish

Untitled

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

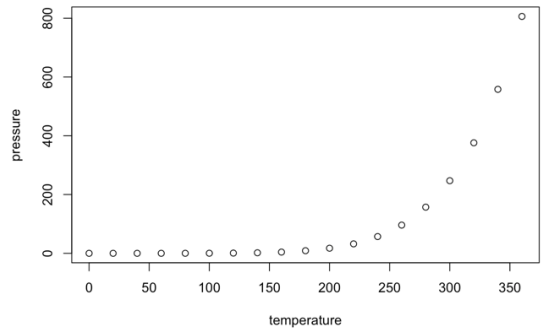
When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

```
##      speed           dist
##  Min.   : 4.0   Min.   :  2.00
##  1st Qu.:12.0   1st Qu.: 26.00
##  Median :15.0   Median : 36.00
##  Mean   :15.4   Mean   : 42.98
##  3rd Qu.:19.0   3rd Qu.: 56.00
##  Max.   :25.0   Max.   :120.00
```

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

Compare the .Rmd file with its html output

.Rmd file

default_html.Rmd

```
---
title: "Untitled"
output: html_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```{r cars}
summary(cars)
```

Including Plots

You can also embed plots, for example:

```{r pressure, echo=FALSE}
plot(pressure)
```

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.
```

YAML metadata

Text

Code chunk

html output

~/Google Drive/BERD R Classes/berd_rmarkdown_project/default/default_html.html

Untitled

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

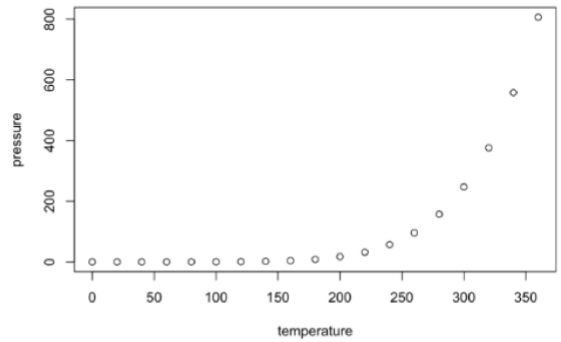
```
summary(cars)
```

```
##      speed           dist
##  Min.   : 4.0   Min.   :  2.00
##  1st Qu.:12.0   1st Qu.: 26.00
##  Median :15.0   Median : 36.00
##  Mean   :15.4   Mean   : 42.98
##  3rd Qu.:19.0   3rd Qu.: 56.00
##  Max.   :25.0   Max.   :120.00
```

Code output

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

How to create the html file? *Knit* the .Rmd file!

To **knit** the .Rmd file, either

1. click on the knit icon  at the top of the editor window
2. or use keyboard shortcuts
    - Mac: *Command+Shift+K*
    - PC: *Ctrl+Shift+K*

- A new window will open with the html output.
- You will now see both .Rmd and .html files in the folder where you saved the .Rmd file.

Note:

- The template .Rmd file that RStudio creates will knit to an html file by default

3 types of R Markdown content

1. *Text*
2. Code chunks
3. YAML metadata (will discuss more at a later date)

Formatting text

- Markdown is a markup language similar to html or LaTeX
- All text formatting is specified via code

Text in editor:

```
Time to learn how to format text using R Markdown!

If I put two spaces
at the end of a line it will force a line break and start a new line.

*This text is in italics*, but _so is this text_.

**Bold** also has __2 options__

~~Should this be deleted?~~

`Sometimes text needs to be verbatim`

>or even a block quote.

Need^superscripts^ or~subscripts~?
```

Output:

Time to learn how to format text using R Markdown!

If I put two spaces
at the end of a line it will force a line break and start a new line.

This text is in italics, but *so is this text*.

Bold also has **2 options**

~~Should this be deleted?~~

`Sometimes text needs to be verbatim`

> or even a block quote.

Need^{superscripts} or$_{subscripts}$?

Headers

- Organize your documents using headers to create sections and subsections

Text in editor:

```
# Header 1

## Header 2

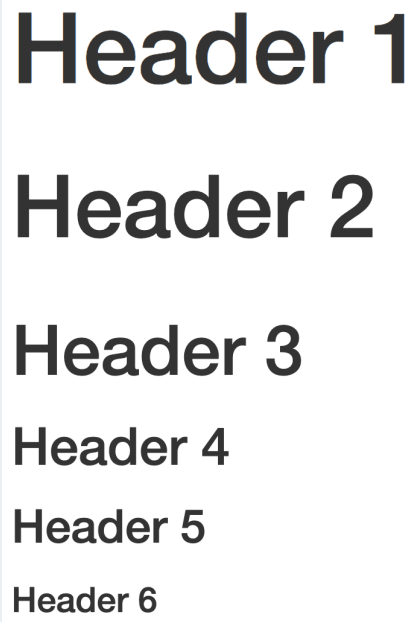
### Header 3

#### Header 4

##### Header 5

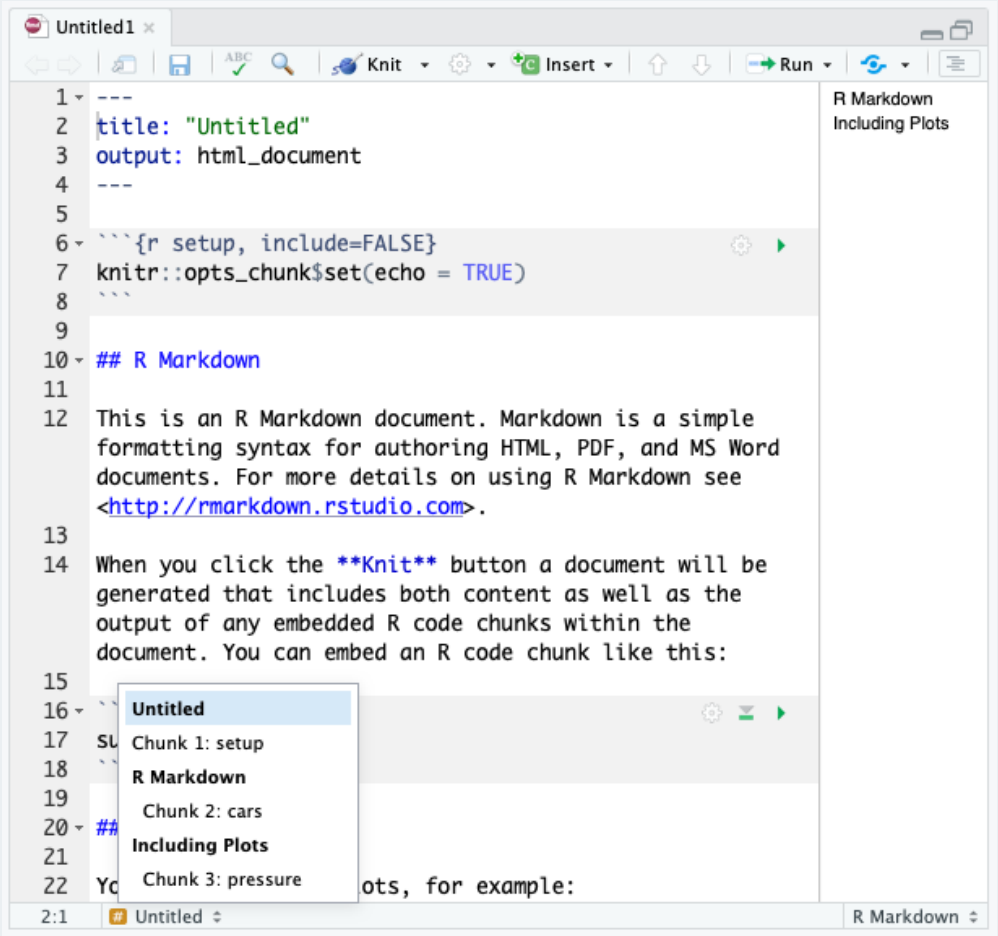
###### Header 6
```

Output:



RStudio tip

You can easily navigate through your .Rmd file if you use headers to outline your text



3 types of R Markdown content

1. Text
2. *Code chunks*
3. YAML metadata (will discuss more at a later date)

Recall the code chunks from before

.Rmd file

default_html.Rmd

```
---
title: "Untitled"
output: html_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```{r cars}
summary(cars)
```

Including Plots

You can also embed plots, for example:

```{r pressure, echo=FALSE}
plot(pressure)
```

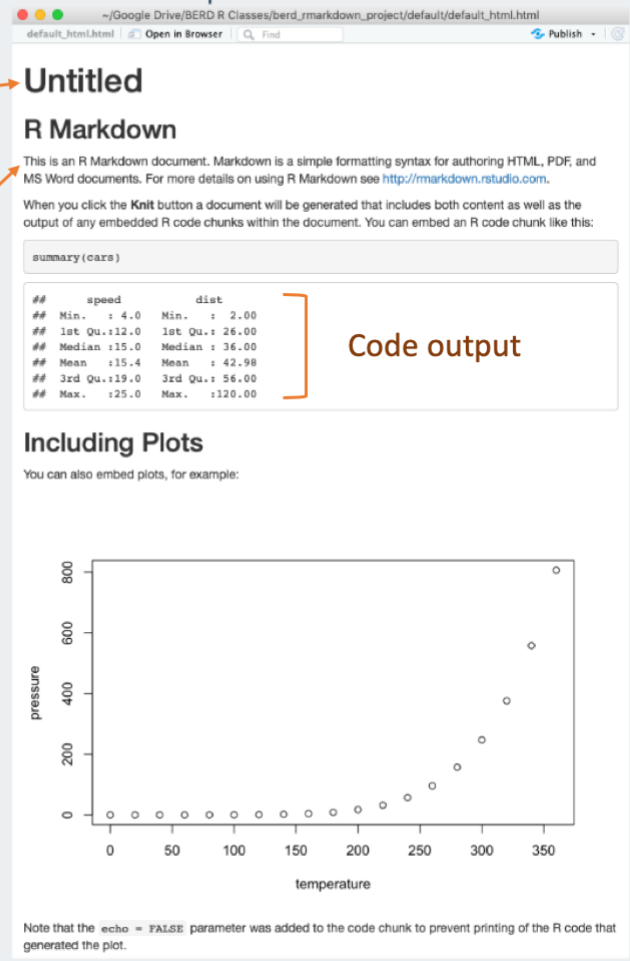
Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.
```

YAML metadata

Text

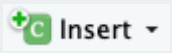
Code chunk

html output



Create a code chunk

Code chunks can be created by either

1. Clicking on  →  at top right of the editor window, or
2. **Keyboard shortcut**
    - Mac: *Command + Option + I*
    - PC: *Ctrl + Alt + I*

- An empty code chunk looks like this:

````
```{r}


```
````

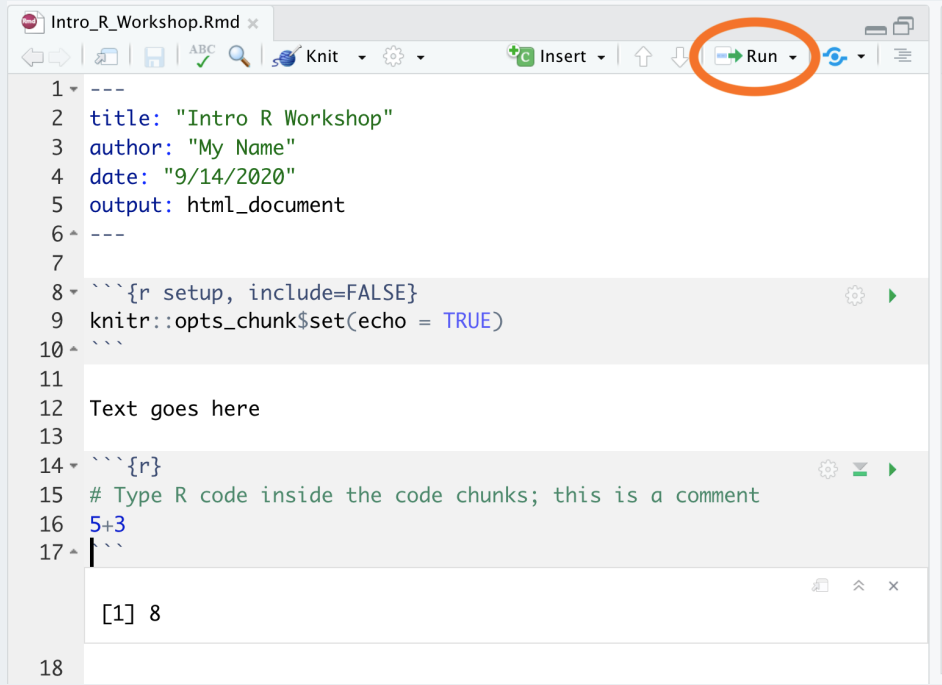
- Note that a code chunks start with ```` ```{r} ```` and ends with ```` ``` ````.

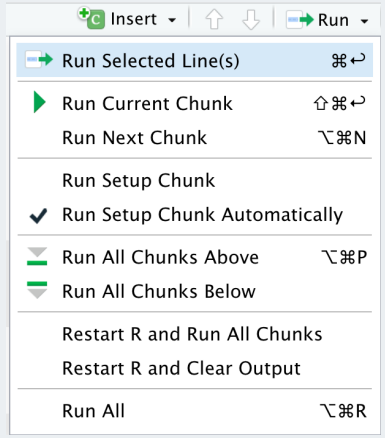
Enter and run code (1/n)

- **Type R code** inside code chunks
- **Select code** you want to run, by
  - placing the cursor in the line of code you want to run,
  - *or* highlighting the code you want to run

- **Run selected code** by
  - clicking on the Run button in the top right corner of the scripting window and choosing "Run Selected Line(s)",
  - or typing one of the following key combinations:
    - **Windows**: **ctrl + return**
    - **Mac**: **command + return**

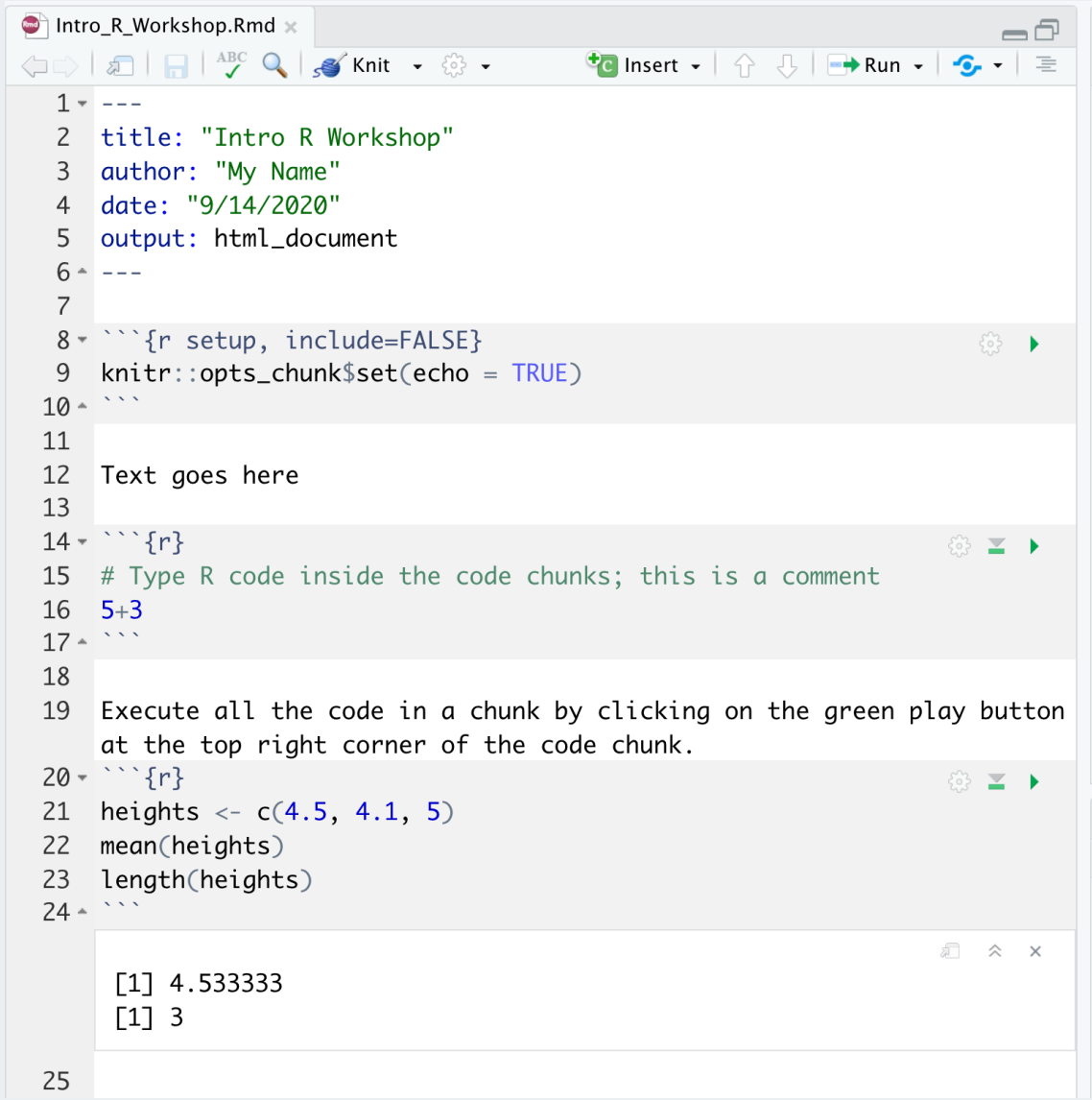
- *Where does the output appear?*





Enter and run code (2/n)

- **Run all code** in a chunk by
  - by clicking the play button in the top right corner of the chunk
- The code output appears below the code chunk



Useful keyboard shortcuts

action	mac	windows/linux
Run code in Rmd or script	cmd + enter	ctrl + enter
<–	option + -	alt + -

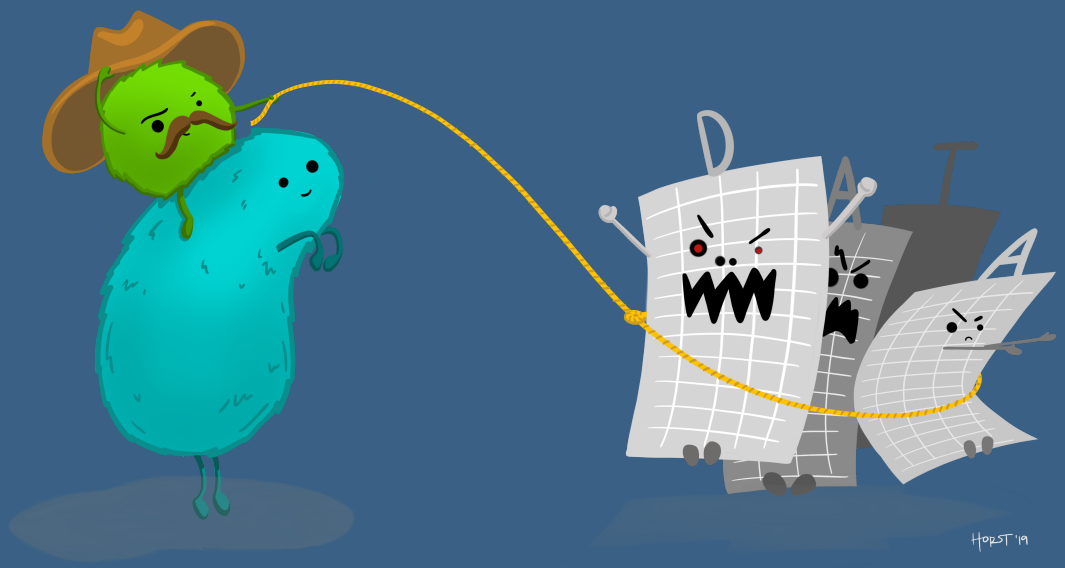
Try typing in Rmd (with shortcut) and running

```
y <- 5
y
```

Others: (see full list)

action	mac	windows/linux
interrupt currently executing command	esc	esc
in console, go to previously run code	up/down	up/down
keyboard shortcut help	option + shift + k	alt + shift + k

Intro to Data



Allison Horst

How are data stored, how do we use them?

- Often, data are in an Excel sheet, or a plain text file (.csv, .txt)
- .csv files open in Excel automatically, but actually are plain text
- Usually, columns are variables/measures and rows are observations (i.e. a person's measurements)

Data in R

- We can import data from many file types, including .csv, .txt., and .xlsx
  - We will cover this on a later date
- Once imported, R typically stores data as **data frames**, or **tibbles** if using the `tidyverse` package (more on this later).
  - For our purposes, these are essentially the same, and I will tend to use the terms interchangeably.
  - These are examples of what we call **object types** in R.

Data frame example

```
df <- data.frame(
  IDs=1:3,
  gender=c("male", "female", "Male"),
  age=c(28, 35.5, 31),
  trt = c("control", "1", "1"),
  Veteran = c(FALSE, TRUE, TRUE)
  )
df
```

```
##   IDs gender  age     trt Veteran
## 1   1   male 28.0 control   FALSE
## 2   2 female 35.5       1    TRUE
## 3   3   Male 31.0       1    TRUE
```

- **Vectors** vs. **data frames**
  - a data frame is a collection (or array or table) of vectors

- Different columns can be of different data types (i.e. numeric vs. text)
- Both numeric and text can be stored within a column (stored together as *text*).
- Vectors and data frames are examples of ***objects*** in R.
  - There are other types of R objects to store data, such as matrices, lists.

Variable (column) types

R type	stats type	description
integer	discrete	integer-valued numbers
double or **numeric**	continuous	**numbers that are decimals**
factor	categorical	categorical variables stored with levels (groups)
character	categorical	**text, "strings"**
logical	categorical	boolean (TRUE, FALSE)

- View the **structure** of our data frame to see what the variable types are:

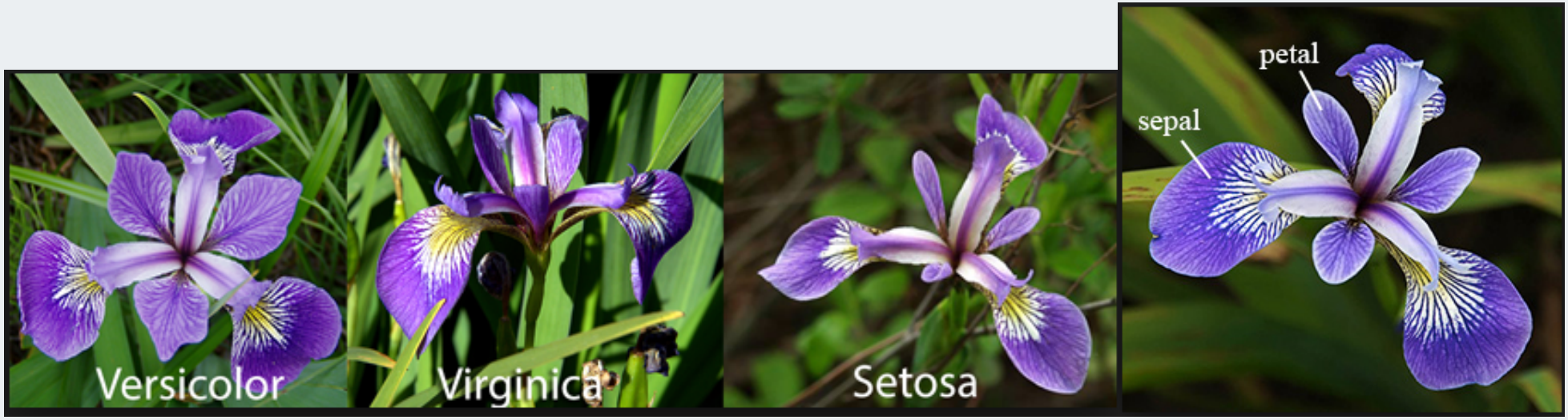
```
str(df)
```

```
## 'data.frame':    3 obs. of  5 variables:
##  $ IDs    : int  1 2 3
##  $ gender : chr  "male" "female" "Male"
##  $ age    : num  28 35.5 31
##  $ trt    : chr  "control" "1" "1"
##  $ Veteran: logi  FALSE TRUE TRUE
```

Data description: Fisher's (or Anderson's) Iris data set

- n = 150
- 3 species of Iris flowers (Setosa, Virginica, and Versicolour)
  - 50 measurements of each type of Iris
- variables:
  - sepal length, sepal width, petal length, petal width, and species

Can the flower species be determined by these variables?



Gareth Duffy

View the `iris` dataset

- The `iris` dataset is already pre-loaded in *base* R and ready to use.
- Type the following command in the console window
  - *Warning: this command cannot be knitted. It will give an error.*

```
View(iris)
```

A new tab in the scripting window should appear with the `iris` dataset.

Data structure

- What are the different **variable types** in this data set?

```
str(iris)   # structure of data
```

```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1
```

Data set summary

```
summary(iris)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
##        Species
##  setosa    :50
##  versicolor:50
##  virginica :50
##
##
##
```

Data set info

```
dim(iris)
```

```
## [1] 150   5
```

```
nrow(iris)
```

```
## [1] 150
```

```
ncol(iris)
```

```
## [1] 5
```

```
names(iris)
```

```
## [1] "Sepal.Length" "Sepal.Width"  "Petal.Length" "Petal.Width"  "Species"
```

View the beginning or end of a data set

```
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

```
tail(iris)
```

```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
## 145          6.7         3.3          5.7         2.5 virginica
## 146          6.7         3.0          5.2         2.3 virginica
## 147          6.3         2.5          5.0         1.9 virginica
## 148          6.5         3.0          5.2         2.0 virginica
## 149          6.2         3.4          5.4         2.3 virginica
## 150          5.9         3.0          5.1         1.8 virginica
```

Specify how many rows to view at beginning or end of a data set

```
head(iris, 3)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
```

```
tail(iris, 2)
```

```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
## 149          6.2         3.4          5.4         2.3 virginica
## 150          5.9         3.0          5.1         1.8 virginica
```

The $

- Suppose we want to single out the column of petal width values.
- One way to do this is to use the `$`
  - `DatSetName$VariableName`

```
iris$Petal.Width
```

```
##   [1] 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 0.2 0.2 0.1 0.1 0.2 0.4 0.4 0.3
##  [19] 0.3 0.3 0.2 0.4 0.2 0.5 0.2 0.2 0.4 0.2 0.2 0.2 0.2 0.4 0.1 0.2 0.2 0.2
##  [37] 0.2 0.1 0.2 0.2 0.3 0.3 0.2 0.6 0.4 0.3 0.2 0.2 0.2 0.2 1.4 1.5 1.5 1.3
##  [55] 1.5 1.3 1.6 1.0 1.3 1.4 1.0 1.5 1.0 1.4 1.3 1.4 1.5 1.0 1.5 1.1 1.8 1.3
##  [73] 1.5 1.2 1.3 1.4 1.4 1.7 1.5 1.0 1.1 1.0 1.2 1.6 1.5 1.6 1.5 1.3 1.3 1.3
##  [91] 1.2 1.4 1.2 1.0 1.3 1.2 1.3 1.3 1.1 1.3 2.5 1.9 2.1 1.8 2.2 2.1 1.7 1.8
## [109] 1.8 2.5 2.0 1.9 2.1 2.0 2.4 2.3 1.8 2.2 2.3 1.5 2.3 2.0 2.0 1.8 2.1 1.8
## [127] 1.8 1.8 2.1 1.6 1.9 2.0 2.2 1.5 1.4 2.3 2.4 1.8 1.8 2.1 2.4 2.3 1.9 2.3
## [145] 2.5 2.3 1.9 2.0 2.3 1.8
```

Example using the $

The `$` is helpful if you want to create a new dataset for just that one variable, or, more commonly, if you want to calculate summary statistics for that one variable.

```
mean(iris$Petal.Width)
```

```
## [1] 1.199333
```

```
sd(iris$Petal.Width)
```

```
## [1] 0.7622377
```

```
median(iris$Petal.Width)
```

```
## [1] 1.3
```

Inline code

- With markdown you can also report **R code output inline** with the text instead of using a chunk.

Text in editor:

```
The mean petal width for all 3 species combined
is `r round(mean(iris$Petal.Width),1)`
(SD = `r round(sd(iris$Petal.Width),1)`) cm.
```

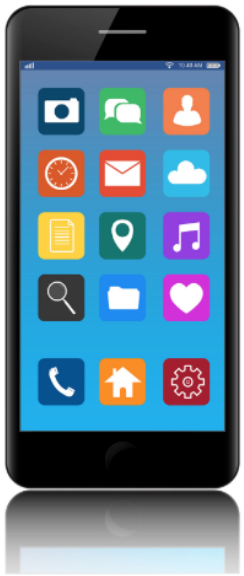
Output:

The mean petal width for all 3 species combined is 1.2 (SD = 0.8) cm.

- Reporting summary statistics this way in a report makes the numbers computationally reproducible.
- For example, if this were for an abstract and a year later you are wondering where the numbers came from, your R code will tell you exactly which dataset was used to calculate the values.

R Packages

A good analogy for R packages is that they
are like apps you can download onto a mobile phone:

R: A new phone	R Packages: Apps you can download
	

ModernDive Figure 1.4

Installing packages

- Packages contain additional functions and data

Two options to install packages:

1. `install.packages()` *or*
2. The "Packages" tab in Files/Plots/Packages/Help/Viewer window

```
install.packages("dplyr")   # only do this ONCE, use quotes
```

- **Only install pacakges once** *(unless you want to update them)*
- Installed from Comprehensive R Archive Network (CRAN) = package mothership

Load packages with `library()` command

- Tip: at the top of your Rmd file, create a chunk that loads all of the R packages you want to use in that file.
- Use the `library()` command to load each required package.
- **Packages need to be reloaded every time you open Rstudio.**

```
library(dplyr)    # run this every time you open Rstudio
```

- You can use a function without loading the package with `PackageName::CommandName`

```
dplyr::arrange(iris, Petal.Width)   # what does arrange do?
```

```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
## 1            4.9         3.1          1.5         0.1     setosa
## 2            4.8         3.0          1.4         0.1     setosa
## 3            4.3         3.0          1.1         0.1     setosa
## 4            5.2         4.1          1.5         0.1     setosa
## 5            4.9         3.6          1.4         0.1     setosa
## 6            5.1         3.5          1.4         0.2     setosa
## 7            4.9         3.0          1.4         0.2     setosa
```

Install and load the pacakges listed below

- `knitr`
  - this might actually already be installed
  - check your packages list
- `tidyverse`
  - this is actually a bundle of packages
  - it will take a while to install
  - see more info at https://tidyverse.tidyverse.org/
- `devtools`
  - used to create R packages
  - for our purposes, needed to install packages from github
- `oi_biostat_data`
  - see next slide for instructions

Directions for installing package `oibiostat`

- The package `oibiostat` is on github and contains datasets for the textbook *Introductory Statistics for the Life and Biomedical Sciences* by Vu & Harrington.
- Steps to install the package `oibiostat`:
  1. First install the `devtools` package if you have not already done so (see below).
  2. The code `devtools::install_github()` tells R to use the command `install_github()` from the `devtools` package without loading the entire package and all of its commands.

```
install.packages("devtools")
devtools::install_github("OI-Biostat/oi_biostat_data")
```

- After running the code above, put # in front of the commands so that RStudio doesn't evaluate them when knitting.
- Now load the `oibiostat` package
  - the code below needs to be run *every time* you restart R or knit an Rmd file

```
library(oibiostat)
```

You WILL get frustrated while learning R!

From Garrett Grolemund's Prologue of his book “Hands-On Programming with R”:

> As you learn to program, you are going to get frustrated. You are learning a new language, and it will take time to become fluent. But frustration is not just natural, it’s actually a positive sign that you should watch for. Frustration is your brain’s way of being lazy; it’s trying to get you to quit and go do something easy or fun. If you want to get physically fitter, you need to push your body even though it complains. If you want to get better at programming, you’ll need to push your brain. Recognize when you get frustrated and see it as a good thing: you’re now stretching yourself. Push yourself a little further every day, and you’ll soon be a confident programmer.

Grolemund, Garrett. 2014. Hands-on Programming with R. O’Reilly.

Allison Horst

More intro R and markdown slides

- These slides are heavily adapted from R workshops I presented together with Jessica Minnier, PhD and sponsored by *OCTRI Biostatistics, Epidemiology, Research & Design (BERD)*:
  - **Introduction to R and RStudio for Exploratory Data Analysis: Part 1** 2020/09/16
    - slides: bit.ly/berd_intro_part1
    - pdf: bit.ly/berd_intro_part1_pdf
    - recording of the presentation
  - **Reproducible Reports with R Markdown** 2019/09/26
    - slides: bit.ly/berd_rmd
    - pdf: bit.ly/berd_rmd_pdf
    - recording of the presentation