

Reproducible (and collaborative) science through RStudio

A whirlwind tour with R, RMarkdown, Python, LaTeX, and
more

Jenny Rieck & Derek Beaton

May 22 2019

The big outline

- ▶ Part 0: Background
- ▶ Part 1: A bit about R
- ▶ Part 2: RStudio & Project setup
- ▶ Part 3: R
- ▶ Part 4: RMarkdown & more
- ▶ Part 5: “Advanced” topics

Part 0: Background

To dive right in

If you want to skip over the background & RStudio, go straight to
Part 2: RStudio & Project setup

Background

- ▶ This is a taste and to bring you into a bigger world
 - ▶ R, Python, SQL, and JavaScript are critical data science tools/languages
- ▶ R (language and community) strongly emphasizes
 - ▶ Centralization & standards
 - ▶ Rigor & reproducibility (packages, RMarkdown)
 - ▶ An interesting language
- ▶ Functional
 - ▶ With a sublanguage (or dialect?): the tidyverse

R is a community (actually many communities!)

- ▶ Help and resources
- ▶ Package development and distribution
- ▶ An ideal example
 - ▶ Not quite always that way
 - ▶ Strong communal presence

R: Help!

- ▶ So many websites e.g., <https://www.statmethods.net/>
- ▶ Online forums (Stack Exchange, r-lists)
- ▶ SpringerLink
 - ▶ All R books for free (pdf format) or for minimal cost (printed)
- ▶ Vignettes
 - ▶ step-by-step instruction guides for packages
- ▶ Git
 - ▶ With open books (via bookdown)
- ▶ Twitter #rstats
- ▶ RStudio (website)
 - ▶ Videos, cheat sheets

R Packages

- ▶ Packages are bundles of code made by someone (or many people) for everyone to use
- ▶ There are packages for everything
- ▶ We'll cover some of the diversity throughout
- ▶ Comprehensive & Reproducible
 - ▶ Available primarily on CRAN
 - ▶ But also github (less so: r-forge)

Part 1: A bit about R

R Background

- ▶ Created in 1992 by Gentleman & Ihaka

[we] considered the problem of obtaining decent statistical software for our undergraduate Macintosh lab. After considering the options, we decided that the most satisfactory alternative was to write our own. [...] Finally we added some syntactic sugar to make it look somewhat like S. We call the result “R”.

What is R?

- ▶ R is general purpose programming
 - ▶ Design around & for statistics
 - ▶ “for and by statisticians”
- ▶ R is a collection of tools
 - ▶ Pre-packaged software at your disposal
- ▶ R is free (as in beer and speech)
 - ▶ No cost, no restrictions
 - ▶ E.g., Microsoft (nee Revolution) R
- ▶ R is a functional language
 - ▶ Turing complete
 - ▶ Mathematical functions
 - ▶ Pass expressions and functions to and from functions

Some syntax notes

Assignment

```
# allowed but not preferred
a_variable = 10 + 1
# preferred
a_variable <- 10 + 1
# a bonus
10 + 1 -> a_variable
```

Dots

```
# allowed but not preferred
a.variable = 10 + 1
## dots have 2 meanings in R,
## with a 3rd in the tidyverse

# preferred
a_variable <- 10 + 1
```

“Reserved” characters

- ▶ c, q, t, C, D, I, F, and T (via https://www.johndcook.com/blog/r_language_for_programmers/)
- ▶ Except that these can be redefined
 - ▶ With great power comes great responsibility (and danger)

R: Data types

- ▶ Stored as *vectors*
 - ▶ see `class()`
- ▶ numeric
 - ▶ real or decimal
 - ▶ Includes `NaN`, `Inf`, `-Inf`
- ▶ integer
- ▶ complex
- ▶ character
- ▶ logical
 - ▶ includes `NA`, `TRUE`, `FALSE`
- ▶ factor
 - ▶ factors are usually not your friends
 - ▶ with `read.csv()`: `stringsAsFactors = F` or convert these
 - ▶ or use tibbles in the tidyverse

R: factor disasters

```
a_numeric_vector <- c(3, 0, 1, -2, 2, 5, 5, 2, 1)
(a_numeric_vector + 1)

## [1] 4 1 2 -1 3 6 6 3 2

(a_numeric2factor_vector <- as.factor(a_numeric_vector))

## [1] 3 0 1 -2 2 5 5 2 1
## Levels: -2 0 1 2 3 5

(as.numeric(a_numeric2factor_vector))

## [1] 5 2 3 1 4 6 6 4 3

(as.numeric(as.character(a_numeric2factor_vector)))

## [1] 3 0 1 -2 2 5 5 2 1
```

R: Data structures

- ▶ Starts counting from 1
 - ▶ Not 0
- ▶ vector[1]
- ▶ matrix[1,1]
- ▶ array[1,1,1]
- ▶ list[[1]]
 - ▶ Can contain mixtures of types
 - ▶ or list\$name
- ▶ data.frame:
 - ▶ Is technically a list but access in three ways
 - ▶ data.frame[[1]][1]
 - ▶ data.frame[1,1]
 - ▶ data.frame\$name
 - ▶ tibbles: tidyverse data.frames

Cheatsheet for going from MatLab to R

R/S-Plus			MATLAB/Octave	Description
<code>a <- c(2,3,4,5)</code>			<code>a=[2 3 4 5];</code>	Row vector, \$1 \times n\$-matrix
<code>adash <- t(c(2,3,4,5))</code>			<code>adash=[2 3 4 5]';</code>	Column vector, \$m \times 1\$-matrix
Vectors				
R/S-Plus			MATLAB/Octave	Description
<code>seq(10)</code> or <code>1:10</code>			<code>1:10</code>	1,2,3, ...,10
<code>seq(0,length=10)</code>			<code>0:9</code>	0,0,1,0,2,0, ...,9,0
<code>seq(1,10,by=3)</code>			<code>1:3:10</code>	1,4,7,10
<code>seq(10,1)</code> or <code>10:-1:1</code>			<code>10:-1:1</code>	10,9,8, ...,1
<code>seq(from=10,to=1,by=-3)</code>			<code>10:-3:1</code>	10,7,4,1
<code>seq(1,10,length=7)</code>			<code>linspace(1,10,7)</code>	Linearly spaced vector of n=7 points
<code>rev(a)</code>			<code>reverse(a)</code>	Reverse
<code>a(:) = 3</code>				Set all values to same scalar value
Concatenation (vectors)				
R/S-Plus			MATLAB/Octave	Description
<code>c(a,a)</code>			<code>[a a]</code>	Concatenate two vectors
<code>c(1:4,a)</code>			<code>[1:4 a]</code>	

<http://mathesaurus.sourceforge.net/octave-r.html>

Cheatsheet for base R

Base R Cheat Sheet

Getting Help

Accessing the help files

```
?mean  
Get help of a particular function.  
help.search('weighted mean')  
Search the help files for a word or phrase.  
help(package = 'dplyr')  
Find help for a package.
```

More about an object

```
str(iris)  
Get a summary of an object's structure.  
class(iris)  
Find the class an object belongs to.
```

Using Libraries

```
install.packages('dplyr')  
Download and install a package from CRAN.
```

```
library(dplyr)  
Load the package into the session, making all its functions available to use.
```

```
dplyr::select  
Use a particular function from a package.
```

```
data(iris)  
Load a built-in dataset into the environment.
```

Working Directory

```
getwd()  
Find the current working directory (where inputs are found and outputs are sent).
```

```
setwd('C://file//path')  
Change the current working directory.
```

Use projects in RStudio to set the working directory to the folder you are working in.

Base R Cheat Sheet

Vectors

Creating Vectors

c(2, 4, 6)	2 4 6	Join elements into a vector
2:6	2 3 4 5 6	An integer sequence
seq(2, 3, by=0.5)	2.0 2.5 3.0	A complex sequence
rep(1:2, times=3)	1 2 1 2 1 2	Repeat a vector
rep(1:2, each=3)	1 1 1 2 2 2	Repeat elements of a vector

Vector Functions

sort(x)	rev(x)
Return x sorted.	Return x reversed.
table(x)	unique(x)
See counts of values.	See unique values.

Selecting Vector Elements

By Position

x[4]	The fourth element.
x[-4]	All but the fourth.
x[2:4]	Elements two to four.

x[-(2:4)]	All elements except two to four.
-----------	----------------------------------

x[c(1, 5)]	Elements one and five.
------------	------------------------

By Value

x[x == 10]	Elements which are equal to 10.
x[x < 0]	All elements less than zero.
x[x %in% c(1, 2, 5)]	Elements in the set 1, 2, 5.

Named Vectors

x[['apple']]	Element with name 'apple'.
--------------	----------------------------

Programming

For Loop

```
for (variable in sequence){  
  Do something  
}
```

Example

```
for (i in 1:5){  
  j <- i + 10  
  print(j)  
}
```

While Loop

```
while (condition){  
  Do something  
}
```

Example

```
while (i < 5){  
  print(i)  
  i <- i + 1  
}
```

Functions

```
function_name <- function(var){  
  Do something  
}  
else {  
  Do something different  
}
```

Example

```
square <- function(x){  
  squared <- x*x  
  return(squared)  
}
```

Reading and Writing Data

Input

```
df <- read.table('file.txt')
```

Output

```
write.table(df, 'file.txt')
```

Description

Read and write a delimited text file.

```
df <- read.csv('file.csv')
```

```
write.csv(df, 'file.csv')
```

Read and write a comma separated value file. This is a special case of readable/writable.

```
load('file.RData')
```

```
save(df, file = 'file.Rdata')
```

Read and write an R data file, a file type special for R.

Conditions

a == b	Are equal	a > b	Greater than	a >= b	Greater than or equal to	is.na(a)	Is missing
a != b	Not equal	a < b	Less than	a <= b	Less than or equal to	is.null(a)	Is null

Tidyverse

- ▶ R is “base R”
- ▶ tidyverse: “an opinionated collection of R packages designed for data science [that] share an underlying design philosophy, **grammar, and data structures.**”
 - ▶ Started(?) with/because of ggplot2
 - ▶ A sublanguage or dialect
 - ▶ Can do so because R is functional language
 - ▶ For full tidy immersion see: <https://style.tidyverse.org/>
- ▶ Strongly built around a style:
 - ▶ objects are nouns
 - ▶ functions are verbs
- ▶ Core packages:
 - ▶ ggplot2, dplyr, tidyr, readr, tibble, stringr
- ▶ Get them all with `install.packages("tidyverse")`
- ▶ Learn it!
 - ▶ But don't learn *only* the tidyverse, you'll be lost in base R

tidyverse cheatsheet

R For Data Science Cheat Sheet

Tidyverse for Beginners

Learn More R for Data Science [Interactively](#) at [www.datcamp.com](#).



Tidyverse

The tidyverse is a powerful collection of R packages that are actually data tools for transforming and visualizing data. All packages of the tidyverse share an underlying philosophy and common APIs.

The core packages are:

- **grid**, which implements the grammar of graphics. You can use it to visualize your data.
- **dplyr** is a grammar of data manipulation. You can use it to solve the most common data manipulation challenges.
- **tidyverse** helps you to create tidy data or `data` where each variable is in a column, each observation is a row and each value is a cell.
- **readr** is a fast and friendly way to read rectangular data.
- **purrr** enhances R's functional programming (FP) toolkit by providing a complete and consistent set of tools for working with functions and vectors.
- **tibble** is a modern re-imaging of the data frame.
- **stringr** provides a cohesive set of functions designed to make working with strings as easy as possible
- **forcats** provide a suite of useful tools that solve common problems with factors.

You can install the complete tidyverse with:

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

Then, load the core tidyverse and make it available in your current R session by running:

```
> library(tidyverse)
```

Note there are many other tidyverse packages with more specialized usage. They are not loaded automatically with `library(tidyverse)`, so you'll need to load each one with its own call to `library()`.

Useful Functions

- `tidyverse_conflicts()` Conflicts between tidyverse and other packages
- `tidyverse_deps()` List all tidyverse dependencies
- `tidyverse_logo()` Get tidyverse logo, using ASCII or unicode characters
- `tidyverse_packages()` List all tidyverse packages
- `tidyverse_update()` Update tidyverse packages

Loading in the data

- `library(datasets)` Load the datasets package
- `library(gapminder)` Load the gapminder package
- `attach(iris)` Attach Iris data to the R search path

dplyr

Filter

`filter()` allows you to select a subset of rows in a data frame.

```
> iris %>%  
  filter(Species=="virginica")  
> iris %>%  
  filter(Species=="virginica",  
         Sepal.Length > 6)
```

Select Iris data of species "virginica" and sepal length greater than 6.

Arrange

`arrange()` sorts the observations in a dataset in ascending or descending order based on one of its variables.

```
> iris %>%  
  arrange(Sepal.Length)  
> iris %>%  
  arrange(desc(Sepal.Length))
```

Sort in ascending order of sepal length
Sort in descending order of sepal length

Combine multiple dplyr verbs in a row with the pipe operator `%>%`:

```
> iris %>%  
  filter(Species=="virginica") %>%  
  arrange(desc(Sepal.Length))
```

Filter for species "virginica"
then arrange in descending order of sepal length

Mutate

`mutate()` allows you to update or create new columns of a data frame.

```
> iris %>%  
  mutate(Sepal.Length=Sepal.Length*10)
```

```
> iris %>%  
  mutate(SIMe=Sepal.length*10)
```

Change Sepal.Length to be in millimeters
Create a new column called SIMe

Combine the verbs `filter()`, `arrange()`, and `mutate()`:

```
> iris %>%  
  filter(Species=="virginica") %>%  
  mutate(SIMe=Sepal.Length*10) %>%  
  arrange(desc(SIMe))
```

Summarize

`summarise()` allows you to turn many observations into a single data point.

```
> iris %>%  
  summarise(medianSL=median(Sepal.Length))
```

```
> iris %>%  
  filter(Species=="virginica") %>%  
  summarise(medianSL=median(Sepal.Length))
```

Summarize to find the median sepal length
Filter for virginica then summarize the median sepal length

You can also summarize multiple variables at once:

```
> iris %>%  
  filter(Species=="virginica") %>%  
  summarise(medianSL=median(Sepal.Length),  
            maxSL=max(Sepal.Length))
```

```
group_by() allows you to summarize within groups instead of summarizing the entire dataset:
```

```
> iris %>%  
  group_by(Species) %>%  
  summarise(medianPL=median(Sepal.Length),  
            maxPL=max(Sepal.Length))
```

```
> iris %>%  
  filter(Sepal.Length>6) %>%  
  group_by(Species) %>%  
  summarise(medianPL=median(Petal.Length),  
            maxPL=max(Petal.Length))
```

Find median and max sepal length of each species

```
> iris %>%  
  filter(Sepal.Length>6) %>%  
  group_by(Species) %>%  
  summarise(maxPL=max(Petal.Length),  
            maxSL=max(Sepal.Length))
```

Find median and max petal length of each species with sepal length > 6

ggplot2

Scatter plot

Scatter plots allow you to compare two variables within your data. To do this with ggplot2, you use `geom_point()`:

```
> iris %>%  
  filter(Sepal.Length > 5)  
> ggplot(iris_small, aes(x=Petal.Length,  
                           y=Petal.Width)) +  
  geom_point()
```

Compare petal width and length

Additional Aesthetics

• Color

```
> ggplot(iris_small, aes(x=Petal.Length,  
                           y=Petal.Width,  
                           color=Species)) +  
  geom_point()
```

• Size

```
> ggplot(iris_small, aes(x=Petal.Length,  
                           y=Petal.Width,  
                           color=Species,  
                           size=Sepal.Length)) +  
  geom_point()
```

Faceting

```
> ggplot(iris_small, aes(x=Petal.Length,  
                           y=Petal.Width)) +  
  geom_point() +  
  facet_wrap(~Species)
```

Line Plots

```
> year %>%  
  group_by(year) %>%  
  summarise(medianGdpCap=median(gdpPerCap))  
> ggplot(by_year, aes(x=year,  
                           y=medianGdpPerCap)) +  
  geom_line() +  
  expand_limits(y=0)
```

Bar Plots

```
> by_species <- iris %>%  
  filter(Sepal.Length>6) %>%  
  group_by(Species) %>%  
  summarise(medianPL=median(Petal.Length))  
> ggplot(by_species, aes(x=Species,  
                           y=medianPL)) +  
  geom_col()
```

Histograms

```
> ggplot(iris_small, aes(x=Petal.Length)) +  
  geom_histogram()
```

Box Plots

```
> ggplot(iris_small, aes(x=Species,  
                           y=Sepal.Width)) +  
  geom_boxplot()
```

DataCamp

Learn R for Data Science [Interactively](#)



Part 2: RStudio & Project setup

RStudio

- ▶ IDE: Integrated development environment
- ▶ RStudio: Does so much
 - ▶ We scratch the surface here
- ▶ Quick walk through
 - ▶ Followed by specific set up
 - ▶ Generally, but
 - ▶ Also for this workshop

RStudio Setup

- ▶ Download R and Rstudio
 - ▶ Strongly recommend Microsoft R (<https://mran.microsoft.com/open>)
 - ▶ Comes with Intel MKL
- ▶ Plain R is fine (<https://cran.r-project.org/>)
 - ▶ Can relink to faster libraries
- ▶ Download RStudio (<https://www.rstudio.com/>)

RStudio Environment

The screenshot shows the RStudio interface with the following components:

- Editor Pane:** Displays R code for creating an ADNI data subset. The code includes library imports, data loading, cleaning, and merging steps. It also handles missing data and manually changes variable classes.
- Console Pane:** Shows statistical summaries for various variables like APOE4, FDG, AV45, CDRSB, ADAS13, and MOCA across different groups (e.g., Mean, Min, Max, Quartiles).
- File Browser:** Shows the project structure with files like .Renviron, README.md, and Rmd documents.
- Environment Tab:** Shows the global environment with objects like `merge_subset` (665 obs. of 17 variables) and `ids` (chr vector).

```
library(ADNImerger)
#####
## Load and clean data
#####
## 0.1 Specify the column names and participants you want (ie, baseline visit for all participants with MOCA>=1
admin.cols <- c("RID", "VISCODE", "DX", "AGE", "PTGENDER", "PTEDUCAT", "PTRECAT", "APOE4", "FDG", "AV45", "CDRSB", "ADAS13", "MOCA")
admin.rows <- c(adminmerge$VISCODE=="b1" & adminmerge$MOCA>=16)
merge_subset <- adminmerge[admin.rows,admin.cols]
#### remove participants with missing data
merge_subset <- merge_subset[complete.cases(merge_subset),]
## 0.2 Bring in modified hachkins
merge_subset$MSMSCORE <- modhach$MSMSCORE[match(merge_subset$RID, modhach$RID)]
## 0.3 Manually change variable classes (remove class 'labelled')
+
Mean :71.92      Mean :16.36
3rd Qu.:76.60     3rd Qu.:18.00
Min. :89.60       Max. :20.00
APOE4   FDG    AV45   CDRSB  ADAS13   MOCA
Min. :0.0000  Min. :0.6983  Min. :0.8385  Min. :0.0000  Min. :0.0  Min. :16.00
1st Qu.:0.0000  1st Qu.:1.1111  1st Qu.:1.1111  1st Qu.:0.0000  1st Qu.:8.4  1st Qu.:22.00
Median :0.0000  Median :2.8802  Median :2.8802  Median :0.0000  Median :1.0  Median :12.00
Mean   :0.5248  Mean   :11.2682  Mean   :11.1989  Mean   :1.2020  Mean   :11.8  Mean   :123.89
3rd Qu.:1.0000  3rd Qu.:11.3620  3rd Qu.:11.3714  3rd Qu.:12.0000  3rd Qu.:18.0  3rd Qu.:126.00
Max. :2.0000  Max. :11.7013  Max. :12.0256  Max. :15.5000  Max. :46.0  Max. :30.00
WholeBrain  Hippocampus  Midtemp  nPACtrailsB  MSMSCORE
Min. :14.421  Min. :1.1111  Min. :1.2213  Min. :-18.6883  Min. :0.0000
1st Qu.:98.8410  1st Qu.:6.510  1st Qu.:6.535  1st Qu.:1.051  1st Qu.:0.0000
Median :105.1621  Median :7.223  Median :7.0186  Median :-2.5250  Median :1.0000
Mean   :105.7026  Mean   :7.150  Mean   :20.302  Mean   :-3.6882  Mean   :0.588
3rd Qu.:112.0570  3rd Qu.:7.834  3rd Qu.:22.088  3rd Qu.:-0.3482  3rd Qu.:1.000
Max. :148.6036  Max. :10.602  Max. :32.189  Max. :5.3540  Max. :3.000
> view(merge_subset)
> |
```

RStudio Environment

The screenshot displays the RStudio interface with several windows open:

- Code Editor:** Shows a script named `create_ADNI_data.R` containing R code for data manipulation. The code includes library imports, data loading, merging datasets, and manual variable class changes.
- Console:** Shows the output of the R code execution. It includes descriptive statistics for various variables like APOE4, FDG, AV45, CDRSB, ADAS13, and MOCA across different brain regions (Wholebrain, Hippocampus, Midtemp, nPACCtailB, and MSMScore).
- File Browser:** Shows the project structure under `workshops/2019_Rstudio_Magic`, including files like `README.md`, `script.R`, and `output`.
- Environment View:** Shows the global environment with objects like `anmerge_subset` (665 obs. of 17 variables), `ids` (chr vector), and `MOCA` (num vector).

RStudio Environment

The screenshot shows the RStudio interface with several windows open:

- Script Editor:** Displays the R script `create_ADNI_data.R`. The code performs the following steps:
 - Imports required packages: `tidyverse`, `lapply`, `data.table`, and `stringr`.
 - Creates a function `PTRACCAT` that takes a list of data frames and merges them into a single data frame.
 - Specifies column names for the merged data frame.
 - Loads and cleans data from `ADNI_merge` and `MOCA` datasets.
 - Specifies column names and participants for the baseline visit.
 - Removes participants with missing data.
 - Brings in modified hachimaki functions.
 - Manually changes variable classes (removing `labelled` class).
- Console:** Shows statistical summaries for various variables across different datasets (e.g., APOE4, FDG, AV45, CDRSB, ADAS13, MOCA, Hippocampus, Midtemp, nPACCtrailsB, HMSCORE). For example, the APOE4 dataset has the following summary statistics:

	Mean	3rd Qu.	Min.	Max.
APOE4	:71.92	:76.60	:69.60	:83.85
FDG	:71.92	:76.60	:69.60	:83.85
AV45	:71.92	:76.60	:69.60	:83.85
CDRSB	:71.92	:76.60	:69.60	:83.85
ADAS13	:71.92	:76.60	:69.60	:83.85
MOCA	:71.92	:76.60	:69.60	:83.85

- Environment:** Shows the global environment with objects like `merge_subset` (665 obs. of 17 variables), `variable_type_map` (num [1:17, 1:3] 0 1 0 0 0 0 0 0 1 0 ...), and `ids` (chr [1:665] "2002" "2003" "2007" "2010" "2011" "2012").
- Files:** A red box highlights the file browser window, which shows the project structure:

 - Home
 - workshops > 2019_Rstudio_Magic
 - Renviron
 - 2019_Rstudio_Magic.Rproj
 - external
 - mac
 - output
 - R
 - README.md
 - Rmd

FILES, PLOTS, HELP

RStudio Environment

The screenshot shows the RStudio interface with several windows open:

- Code Editor:** Shows R code for creating an ADNI data subset. The code includes library imports, data loading, merging, and subset selection. A red box highlights the final command: `> view(merge_subset)`.
- Environment:** Shows the `merge_subset` object, which is a data frame with 665 observations and 17 variables. It lists variables like `ADAS13`, `CDRSB`, `MOCA`, and `PTEDUCAT`.
- File Browser:** Shows the project structure under `workshops > 2019_Rstudio_Magic`. It includes files like `README.md`, `environment.Rproj`, and `output`.
- Text Overlay:** A large red text overlay in the center-right area reads "VARIABLES, HISTORY, VERSION CONTROL".

RStudio Environment

The screenshot displays the RStudio interface with several panes:

- Code pane:** Shows R code for creating an ADNI data subset. The code includes library imports, data loading, cleaning, and subset selection. It also handles missing data and manually changes variable classes.
- Console pane:** Displays statistical summaries for variables like APOE4, FDG, AV45, CDRSB, ADAS13, and MOCA. For example, APOE4 has a mean of 71.92 and a median of 70.00. The FDG variable has a range from 89.60 to 208.00.
- Environment pane:** Shows the global environment with objects like `anmerge_subset`, `variable_type_map`, `Values`, and `Functions`.
- File browser pane:** Shows the project structure with files like `README.md`, `Renviron`, and `2019_Rstudio_Magic.Rproj`.

```
library(ADNImerGE)
#####
## Load and clean data
#####
## 0.1 Specify the column names and participants you want (ie, baseline visit for all participants with MOCA>=1
admin.cols <- c("RID", "VISCODE", "DX", "AGE", "PTGENDER", "PTEDUCAT", "PTETHCAT", "PTRACCAT", "APOE4", "FDG", "ADAS13", "CDRSB", "MOCA")
admin.rows <- c(adminmerge$VISCODE=="b1" & adminmerge$MOCA>=16)
anmerge_subset <- adminmerge[admin.rows,admin.cols]
#####
## remove participants with missing data
anmerge_subset <- anmerge_subset[complete.cases(anmerge_subset),]
#####
## 0.2 Bring in modified hachkins
anmerge_subset$MSMSCORE <- modhach$MSMSCORE[match(anmerge_subset$RID, modhach$RID)]
#####
## 0.3 Manually change variable classes (remove class 'labelled')
anmerge_subset$FDG <- as.numeric(as.character(anmerge_subset$FDG))
anmerge_subset$AV45 <- as.numeric(as.character(anmerge_subset$AV45))
anmerge_subset$ADAS13 <- as.numeric(as.character(anmerge_subset$ADAS13))
anmerge_subset$CDRSB <- as.numeric(as.character(anmerge_subset$CDRSB))
anmerge_subset$MOCA <- as.numeric(as.character(anmerge_subset$MOCA))
```

	APOE4	FDG	AV45	CDRSB	ADAS13	MOCA
Min.	:0.0000	Min. :0.6983	Min. :0.8385	Min. :0.0000	Min. :0.0	Min. :16.00
1st Qu.	:0.0000	1st Qu.:17.60	1st Qu.:17.60	1st Qu.:0.0000	1st Qu.:8.0	1st Qu.:22.00
Median	:0.0000	Median :28.02	Median :28.02	Median :0.0000	Median :12.00	Median :25.00
Mean	:0.5248	Mean :31.2682	Mean :31.2682	Mean :1.202	Mean :13.8	Mean :23.89
3rd Qu.	:1.0000	3rd Qu.:31.3620	3rd Qu.:31.3714	3rd Qu.:2.0000	3rd Qu.:18.0	3rd Qu.:26.00
Max.	:2.0000	Max. :31.7013	Max. :32.0256	Max. :5.500	Max. :46.0	Max. :30.00

> view(anmerge_subset)
> |

RStudio Environment

The screenshot displays the RStudio interface with several windows open:

- Data Viewer:** A central window titled "DATA VIEWER" showing a table of 665 observations across 17 variables. The variables include DX, AGE, PTGENDER, PTEDUCAT, PTRECAT, PTRACCAT, APOE4, FDG, AV45, CDRSB, ADAS13, MOCA, WholeBrain, and Hippocampus.
- Global Environment:** A window showing the global environment with objects like anerage_subset, variable_type_map, values, and functions.
- File Browser:** A window showing the file structure under "workshops > 2019_Rstudio_Magic".
- Console:** A window showing R code and its output, including descriptive statistics for variables like APOE4, FDG, AV45, CDRSB, ADAS13, MOCA, WholeBrain, Hippocampus, Midtemp, nPACCtrailsB, and HMSCore.
- Terminal:** A window showing the command line interface.
- Jobs:** A window showing the current jobs.

Some benefits of RStudio

- ▶ Built-in integration with version control (git or SVN)
- ▶ R Markdown
 - ▶ Save and execute code
 - ▶ Generate high quality reports that can be shared
 - ▶ Create presentations (like this one!)
 - ▶ Even write papers
 - ▶ This workshop
 - ▶ See https://github.com/jennyrieck/workshops/tree/master/2019_Rstudio_Magic
- ▶ Python, D3 (JavaScript), SQL, Shiny, LaTeX, Git/SVN, HTML/CSS, and so much more.

RStudio is more

- ▶ Not just an IDE
- ▶ A company
- ▶ A community
- ▶ A conference
- ▶ A centralized resource

RStudio Resources

The screenshot shows the RStudio website homepage. At the top, there's a navigation bar with links for Products, Resources, Pricing, About Us, Blogs, and a search icon. Below the navigation is a decorative banner featuring a colorful, abstract graphic of overlapping colored bands.

RStudio: A screenshot of the RStudio IDE interface, showing the code editor, workspace, and plots.

Shiny: An image of a map of the United States with a "ZIP explorer" interface overlaid.

R Packages: Icons for several popular R packages: `markdown`, `Shiny`, `tidyverse`, `knitr`, and `ggplot2`.

RStudio description: RStudio makes R easier to use. It includes a code editor, debugging & visualization tools.

Shiny description: Shiny helps you make interactive web applications for visualizing data. Bring R data analysis to life.

R Packages description: Our developers create popular packages to expand the features of R. Includes `ggplot2`, `dplyr`, `R Markdown` & more.

At the bottom, there are download and learn more buttons for each section, and a horizontal orange progress bar.

RStudio Resources

Online Learning - RStudio

https://www.rstudio.com/online-learning/

R Studio

Products Resources Pricing About Us Blogs

Online learning

A wealth of tutorials, articles, and examples exist to help you learn R and its extensions. Scroll down or click a link below for a curated guide to learning R and its extensions.

- R Programming
- Shiny
- R Markdown
- Data Science
- Books

R Programming
Read More >

Shiny
Read More >

R Markdown
Read More >

Data Science
Read More >

RStudio Resources

Cheatsheets - RStudio x + - □ x

https://www.rstudio.com/resources/cheatsheets/

R Studio Products Resources Pricing About Us Blogs SEARCH

RStudio Cheat Sheets

The cheat sheets below make it easy to learn about and use some of our favorite packages. From time to time, we will add new cheat sheets to the gallery. If you'd like us to drop you an email when we do, let us know by clicking the button to the right.

SUBSCRIBE TO CHEAT SHEET UPDATES HERE

- RStudio IDE
- R Markdown
- Shiny
- Package Development
- Data Import
- Data Transformation with dplyr
- Data Visualization with ggplot2
- Apply functions with purrr
- Deep Learning with Keras
- Data Science in Spark with Sparklyr
- String manipulation with stringr
- Dates and times with lubridate

Python with R and Reticulate Cheat Sheet

The reticulate package provides a comprehensive set of tools for interoperability between Python and R. With reticulate, you can call Python from R in a variety of ways including importing Python modules into R scripts, writing R Markdown Python chunks, sourcing Python scripts, and using Python interactively within the RStudio IDE. This cheatsheet will remind you how.
Updated 4/19.

Use Python with R with reticulate :: CHEAT SHEET

The reticulate package makes it easy to have and use Python in R. It's a Python interface, just like R itself.

Python in R Markdown

Object Conversion

Helpers



Project and Environment Setup

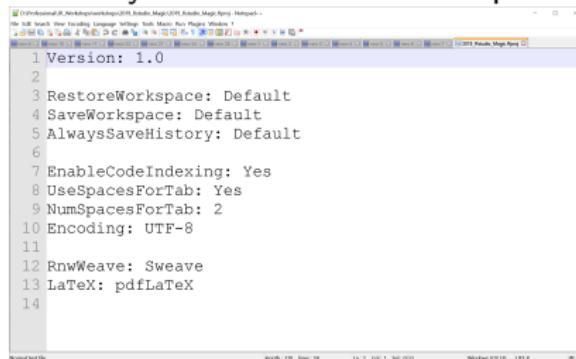
- ▶ Special & hidden files
- ▶ Having a structure

RStudio Setup

- ▶ See <https://jennybc.github.io/2014-05-12-ubc/r-setup.html> for a detailed guide

For safety & collaboration

- ▶ RStudio projects
 - ▶ “RStudio projects make it straightforward to divide your work into multiple contexts, each with their own working directory, workspace, history, and source documents.”
 - ▶ Allows for return to key states
- ▶ .Rproj files
 - ▶ Basically a text file with some parameters for start up



The screenshot shows the RStudio interface with a project titled "R Professional R_Neuroinformatics2019_RStudio_Magic2019_RStudio_Magic_Spring_Histogram". The code editor displays an R script with the following content:

```
1 Version: 1.0
2
3 RestoreWorkspace: Default
4 SaveWorkspace: Default
5 AlwaysSaveHistory: Default
6
7 EnableCodeIndexing: Yes
8 UseSpacesForTab: Yes
9 NumSpacesForTab: 2
10 Encoding: UTF-8
11
12 RnwWeave: Sweave
13 LaTeX: pdfLaTeX
14
```

The status bar at the bottom indicates the file is a "Normal text file" with a length of 720, 14 lines, and 1000 characters.

Projects

Create a new one for:

- ▶ a folder
- ▶ packages
- ▶ (and from) git repos:

New Project

Create Project

 **New Directory**
Start a project in a brand new working directory >

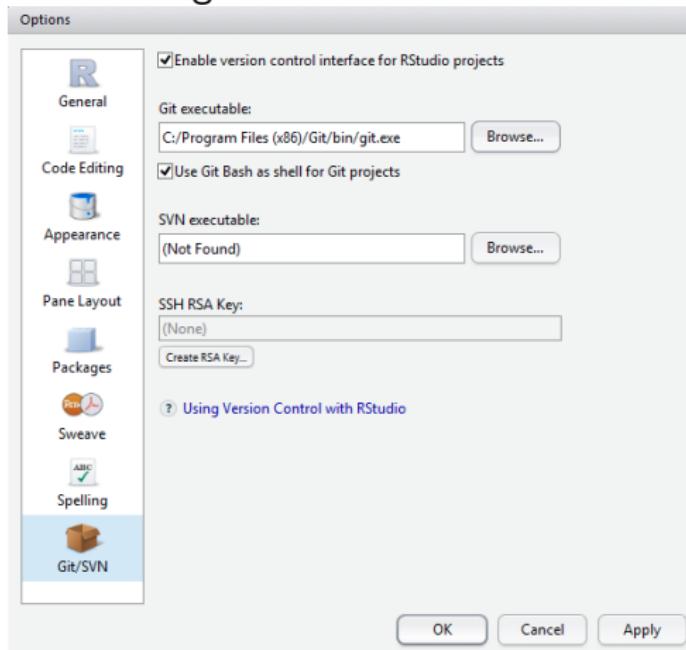
 **Existing Directory**
Associate a project with an existing working directory >

 **Version Control**
Checkout a project from a version control repository >

Cancel

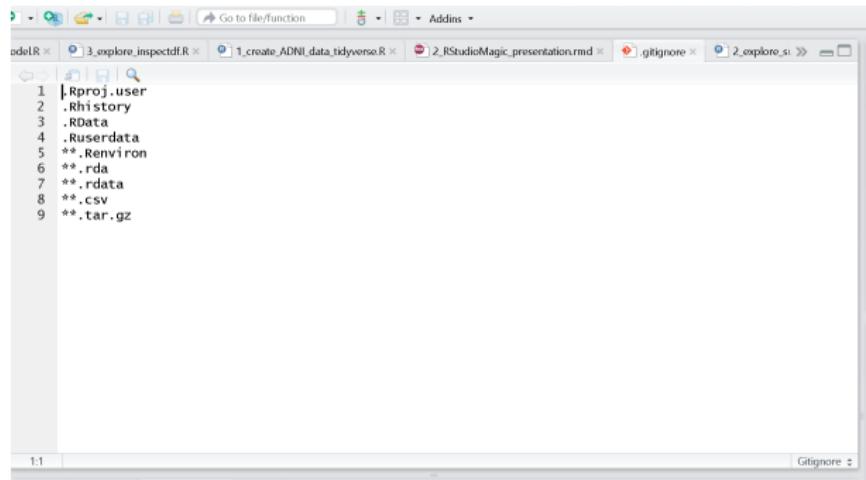
Git & Projects

- ▶ Git
- ▶ Download git and link executable within RStudio



Format .gitignore

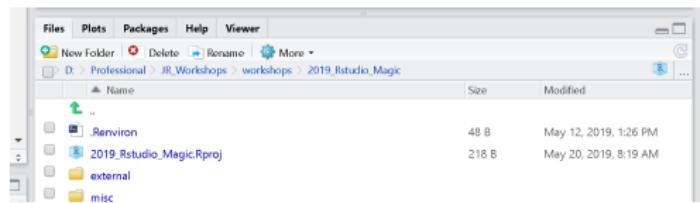
- ▶ File types to ignore via version control
- ▶ ****** before each extensions will match directories anywhere in the repo



The screenshot shows the RStudio interface with the .gitignore tab selected in the top navigation bar. The code editor displays the following content:

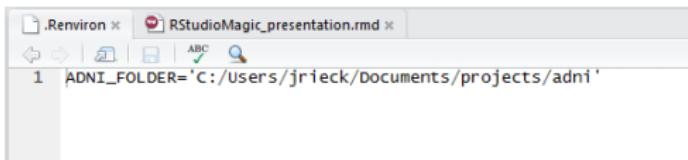
```
1 .Rproj.user
2 .Rhistory
3 .RData
4 .Ruserdata
5 **.Renviron
6 **.rda
7 **.rdata
8 **.csv
9 **.tar.gz
```

Environmental variables



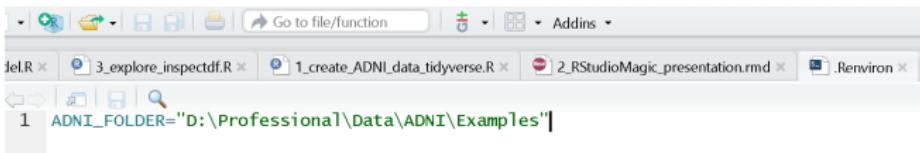
Format environmental variables

- ▶ Set environmental variables (ie, directory location of data) to make code generalizable across computers
 - ▶ Don't commit or share these
- ▶ In **your** project folder create a `.Renvironment` file and define variables
 - ▶ Jenny's:



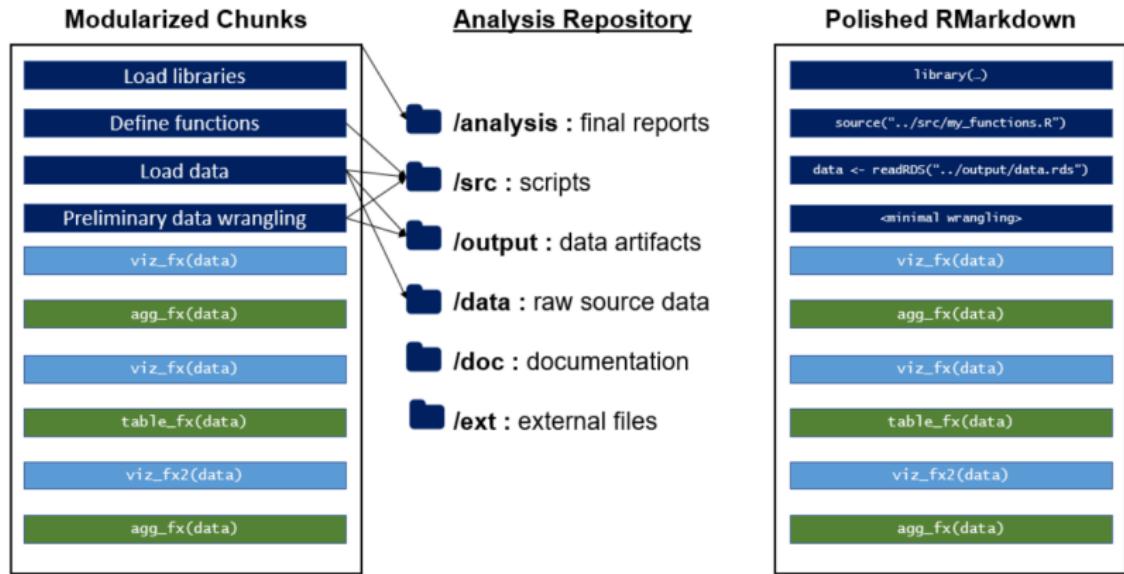
```
1 ADNI_FOLDER='C:/Users/jrieck/Documents/projects/adni'
```

- ▶ Derek's:



```
1 ADNI_FOLDER="D:\Professional\Data\ADNI\Examples"
```

Organize your project folders and markdown



<https://emilyriederer.netlify.com/post/rmarkdown-driven-development/>

Organize your project folders and markdown

- ▶ What works for you?
- ▶ What works for your organization or team?
- ▶ Maximize utility, minimize complexity

This works for us

 [jennyrieck / workshops](#)

 Watch ▾ 1  Star 0  Fork 0

 Code  Issues 0  Pull requests 0  Projects 0  Wiki  Insights  Settings

Branch: master ▾ [workshops / 2019_Rstudio_Magic /](#)

 Create new file  Upload files  Find file  History

 jennyrieck added our favoRite things ... Latest commit d818f26 6 hours ago

..

	R	more updates to manuscript example!	23 hours ago
	Rmd	added our favoRite things	6 hours ago
	external/images	reorganizing pngs	6 hours ago
	misc	reorganizing pngs	6 hours ago
	2019_Rstudio_Magic.Rproj	initial folder structure	5 days ago
	README.md	create readme	5 days ago
 README.md			
Rstudio magic for BrainHack Toronto 2019			

This works for us

Screenshot of a GitHub repository interface showing a list of commits.

Branch: master workshops / 2019_Rstudio_Magic / R /

Create new file Upload files Find file History

derekbeaton almost done now we hope Latest commit e87b65d 16 hours ago

..

File	Message	Time Ago
0_create_ADNI_data_base.R	fixed rownames/race coding	8 days ago
1_create_ADNI_data_tidyverse.R	fixed rownames/race coding	8 days ago
2_explore_summarytools.R	almost done now we hope	16 hours ago
3_explore_inspectdf.R	almost done now we hope	16 hours ago
4_explore_DataExplorer_one_liner.R	almost done now we hope	16 hours ago
5_linear_model.R	almost done now we hope	16 hours ago
6_covstatis_example.R	please don't collide.	2 days ago

This works for us

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights

Branch: master workshops / 2019_Rstudio_Magic / Rmd / Create new file Upload files Find file History

derekbeaton small update Latest commit 74c5384 14 hours ago

1_a_Simple_RMarkdown_PDF_files/figure-latex more updates to manuscript example! 3 days ago

3_RMarkdown APA Manuscript_files updated numbers & structures 16 hours ago

1_a_Simple_RMarkdown_PDF.Rmd almost done now we hope 16 hours ago

1_a_Simple_RMarkdown_PDF.log whatever 2 days ago

1_a_Simple_RMarkdown_PDF.pdf more updates to manuscript example! 3 days ago

1_a_Simple_RMarkdown_PDF.tex tons of bells-and-whistles via the manuscript. 4 days ago

2_RStudioMagic_presentation.pdf small update 14 hours ago

2_RStudioMagic_presentation.rmd small update 14 hours ago

2_RStudioMagic_presentation.tex small update 14 hours ago

3_RMarkdown APA Manuscript.Rmd updated numbers & structures 16 hours ago

3_RMarkdown APA Manuscript.docx updated numbers & structures 16 hours ago

3_RMarkdown APA Manuscript.pdf updated numbers & structures 16 hours ago

3_RMarkdown APA Manuscript.tex updated numbers & structures 16 hours ago

r-references.bib updated numbers & structures 16 hours ago

Get the packages you need

e.g.,

```
#to install from CRAN
install.packages('devtools', dependencies = TRUE)

#to install from a git  (requires the devtools package)
devtools::install_github(Gibbsdavid/CatterPlots)

#to install from a file
install.packages('/mypath/to/package/ADNIMERGE.tar.gz',
                 type='source', repos=NULL)
```

(hint: you'll need more than this!)

Part 3: R

Let's take a closer look at R

- ▶ The Alzheimer's Disease Neuroimaging Initiative (ADNI).
- ▶ We use the ADNIMERGE R package
 - ▶ Lots and lots of data prepared *for you*
- ▶ We use this for all of our examples

The R examples

- ▶ Making & prepping data
 - ▶ 0_create_ADNI_data_base.R
 - ▶ 1_create_ADNI_data_tidyverse.R
- ▶ Exploring data
 - ▶ 2_explore_summarytools.R
 - ▶ 3_explore_inspectdf.R
 - ▶ 4_explore_DataExplorer_one_liner.R
 - ▶ With a small rant from Derek
- ▶ Some stats: Linear models
 - ▶ 5_linear_model.R
- ▶ Experimental packages (covSTATIS via Github)
 - ▶ 6_covstatis_example.R

Derek's rant break

- ▶ DataExplorer committed a crime.
 - ▶ “coding categorical variables with the indicator matrix of dummy variables and considering them as Gaussian, for instance, is almost a crime.”
- ▶ See [Derek's PCA & multiple correspondence analysis workshop](#)
 - ▶ Same data as this workshop
 - ▶ All in R & RMarkdown

Part 4: RMarkdown & more

RMarkdown

- ▶ Yuhui Xie:
 - ▶ <https://bookdown.org/yihui/rmarkdown/>
- ▶ What it is & why to use it
- ▶ Fancy helpers:
 - ▶ `kable` & `kableExtra`
 - ▶ `grid` & `gridExtra`
- ▶ Deviations for:
 - ▶ LaTeX
 - ▶ Python
- ▶ Tying it all together through here

RMarkdown Don'ts

- ▶ Don't hardcode values or absolute file paths
 - ▶ see `here::here()`
 - ▶ Use projects (`.Rproj`)
- ▶ Don't do complicated or expensive stuff
 - ▶ Database queries
 - ▶ Resampling
- ▶ avoid `eval=FALSE`
 - ▶ except to help illustrate code...
- ▶ Reduce repeated code
 - ▶ One time: Script
 - ▶ Two times: Function
 - ▶ Three times: Package

RMarkdown dos & don'ts

- ▶ Suggestions & More:
 - ▶ <https://emilyriederer.netlify.com/post/rmarkdown-driven-development/>
- ▶ We're using this here

Let's take a look

- ▶ 1_a_Simple_RMarkdown_PDF.Rmd
- ▶ 2_RStudioMagic_presentation.rmd
 - ▶ This presentation!
- ▶ 3_RMarkdown APA Manuscript.Rmd

Part 5: “Advanced” topics

Some advanced/other things we're not covering

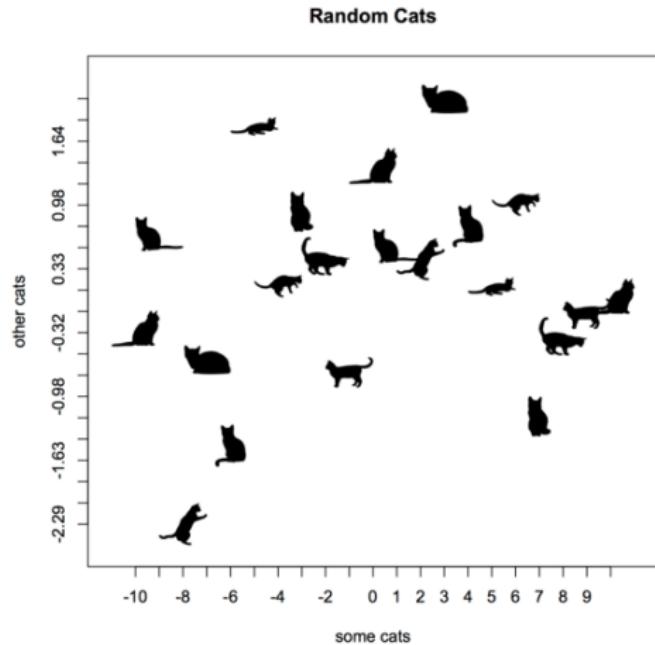
- ▶ package development
- ▶ Shiny
- ▶ SQL
- ▶ C/C++
- ▶ R2D3 (JavaScript)

A few of our favorite things

- ▶ Fun R do-dads

CatterPlot for feline based graphics:

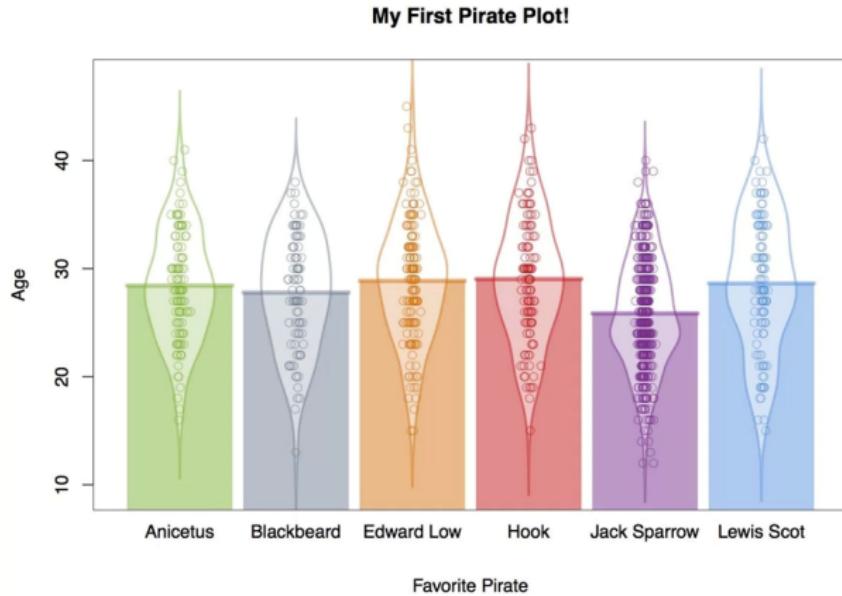
```
devtools::install_github(Gibbsdavidl/CatterPlots)
```



<https://github.com/Gibbsdavidl/CatterPlots>

What's a pirate's favorite programming language?

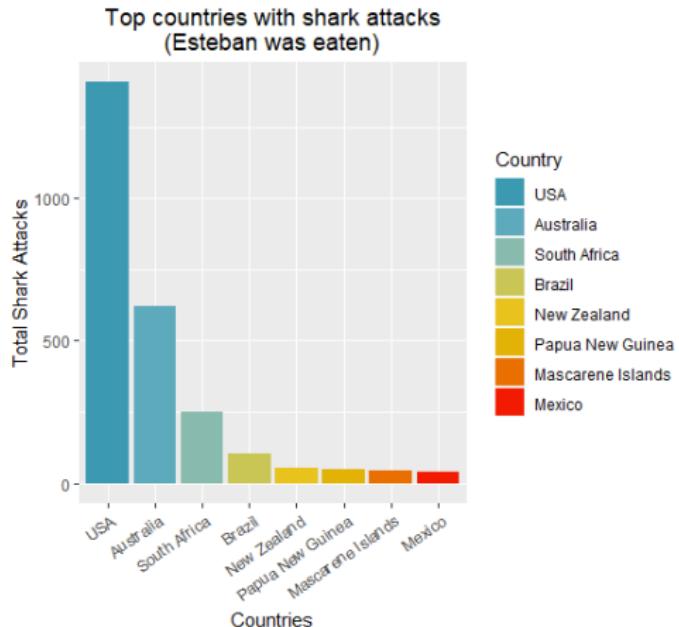
```
install.packages('yarrr')
```



<https://cran.r-project.org/web/packages/yarrr/vignettes/pirateplot.html>

Color palettes to fit your mood

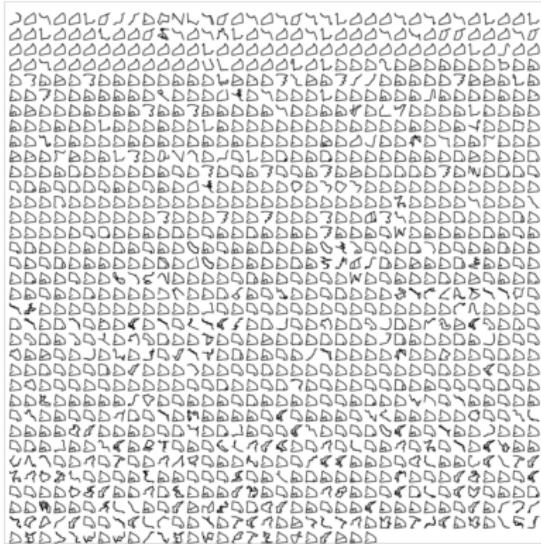
```
devtools::install_github(karthik/wesanderson)
```



<https://github.com/karthik/wesanderson>

Mapping your Strava routes

```
devtools::install_github(marcusvolz/strava)
```

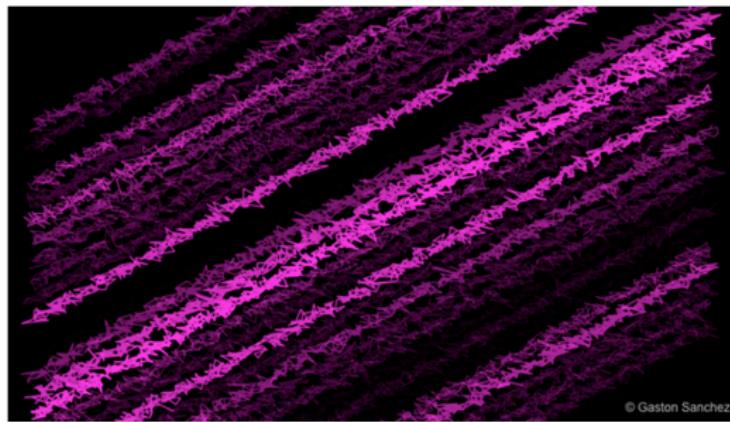


<https://www.r-bloggers.com/strava-rides-map-in-r/>

ALSO <https://marcusvolz.com/?p=4068>

Make aRt!

- ▶ R Graph Gallery
 - ▶ <http://www.r-graph-gallery.com/>
- ▶ Rtist: Gaston Sanchez
 - ▶ <http://gastonsanchez.com/Rtist/>



© Gaston Sanchez

```
# Pink Barbs
# -----
# generate pairs of x-y values
x <- seq(1, 100, length = 1000)
y <- x + rnorm(1000)

# pink_barbs.png
# set graphical parameters
op <- par(bg = "black", mar = rep(0, 4))
# plot
plot(x, y, type = "n")
for (i in seq(-80, 70, by = 5)) {
  lines(x + rnorm(1000), x + i + rnorm(1000, 2), pch = 19,
        col = hsv(0.85, 1, 1, runif(1000)),
        lwd = sample(seq(0.3, 2, length = 20), 1))
}
# signature
legend("bottomright", legend = "@ Gaston Sanchez", bty = "n",
       text.col = "gray70")
# reset par
par(op)
dev.off()
```

Wouldn't be a brain hack without brains

R & Brain Imaging

- ▶ Slow to take hold
 - ▶ Getting there
- ▶ Neuroimaging: one of the last fields in R
 - ▶ Virtually every other field has an R community
 - ▶ Bioinformatics especially (see [Bioconductor](#))

Reading & manipulating

- ▶ The neuroimaging ecosystem is growing in R
 - ▶ neuroim
 - ▶ ANTsR
 - ▶ oro.nifti - been around a while
 - ▶ RMINC

More brains

- ▶ MRI in Shiny
- ▶ brainGraph
- ▶ ggSeg



Neuroconductor

- ▶ Conceptually based on Bioconductor
- ▶ <https://neuroconductor.org/>

