MH2010
Practical session 29th Jan 2020

- Introduction to RStudio

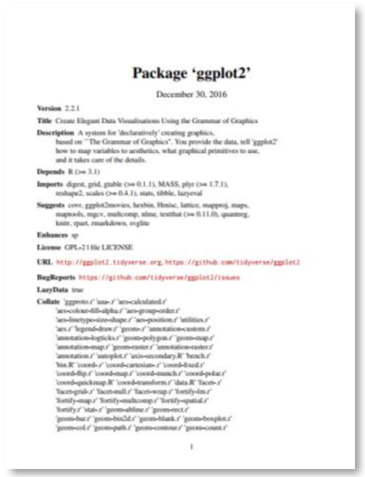
Brendan Palmer,
Clinical Research Facility - Cork &
School of Public Health
@B_A_Palmer

CRF-C
HRB Clinical Research Facility Cork

UCC
University College Cork, Ireland
Coláiste na hOllscoile Corcaigh
School of Public Health

R is for Resources

Vignettes



Package 'ggplot2'

December 30, 2016

Version 2.2.1

Title Create Elegant Data Visualisations Using the Grammar of Graphics

Description A system for 'declaratively' creating graphics, based on "The Grammar of Graphics". You provide the data, tell 'ggplot2' how to map variables to aesthetics, what graphical primitives to use, and it takes care of the details.

Depends R (>= 3.1)

Imports digest, grid, gtable (>= 0.1.1), MASS, plyr (>= 1.7.1), reshape2, scales (>= 0.4.1), stats, tibble, lazyeval

Suggests covr, ggplot2movies, hexbin, Hmisc, lattice, mapproj, maps, maptools, mgcv, multcomp, nlme, testthat (>= 0.11.0), quantreg, knitr, rpart, rmarkdown, svglite

Enhances sp

License GPL-2 | file LICENSE

URL http://ggplot2.tidyverse.org, https://github.com/tidyverse/ggplot2

BugReports https://github.com/tidyverse/ggplot2/issues

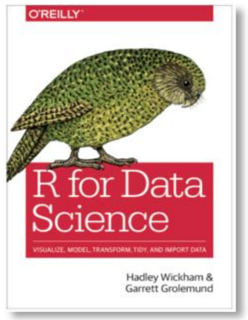
LazyData true

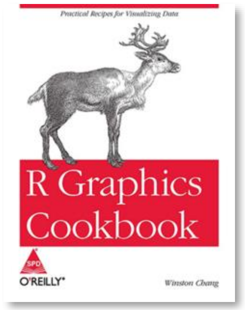
Collate 'ggproto.r' 'aaa-.r' 'aes-calculated.r' 'aes-colour-fill-alpha.r' 'aes-group-order.r' 'aes-linetype-size-shape.r' 'aes-position.r' 'utilities.r' 'aes.r' 'legend-draw.r' 'geom-.r' 'annotation-custom.r' 'annotation-logticks.r' 'geom-polygon.r' 'geom-map.r' 'annotation-map.r' 'geom-raster.r' 'annotation-raster.r' 'annotation.r' 'autoplot.r' 'axis-secondary.R' 'bench.r' 'bin.R' 'coord-.r' 'coord-cartesian-.r' 'coord-fixed.r' 'coord-flip.r' 'coord-map.r' 'coord-munch.r' 'coord-polar.r' 'coord-quickmap.R' 'coord-transform.r' 'data.R' 'facet-.r' 'facet-grid-.r' 'facet-null.r' 'facet-wrap.r' 'fortify-lm.r' 'fortify-map.r' 'fortify-multcomp.r' 'fortify-spatial.r' 'fortify.r' 'stat-.r' 'geom-abline.r' 'geom-rect.r' 'geom-bar.r' 'geom-bin2d.r' 'geom-blank.r' 'geom-boxplot.r' 'geom-col.r' 'geom-path.r' 'geom-contour.r' 'geom-count.r'

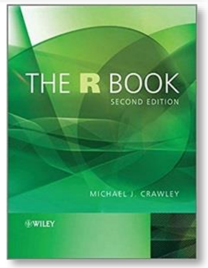
1

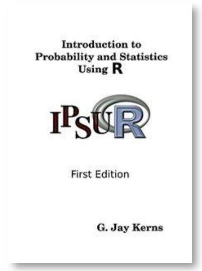
Webpages

eBooks









Cheatsheets

Base R
Data Transformation with dplyr :: CHEAT SHEET
R Markdown
R Markdown :: CHEAT SHEET
Data Science in Spark with Sparklyr :: CHEAT SHEET
Data Visualization with ggplot2 :: CHEAT SHEET
RStudio IDE :: CHEAT SHEET

Twitter

Mara Averick
@dataandme

tidyverse 🥑 @rstudio, 🏀 hoop head, gnashgab, blatherskite, lesser ½ of @batpigandme

Massachusetts

maraaverick.rbind.io

One R Tip a Day
@RLangTip

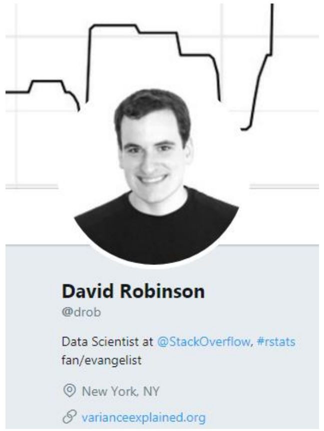
One tip per day M-F on the R programming language #rstats. Brought to you by the R community team at Microsoft.

Hadley Wickham
@hadleywickham

R, data, visualisation.

Houston, TX

hadley.nz



David Robinson
@drob

Data Scientist at @StackOverflow, #rstats fan/evangelist

New York, NY

varianceexplained.org

Jenny Bryan
@JennyBryan

Software engineer @rstudio, humane #rstats, adjunct prof @UBC where I created @STAT545, part of @ropensci

Vancouver, BC

jennybryan.org

Darren L Dahly
@statsepi Follows you

Principal Statistician, Epidemiologist, Sr Lecturer | @HRBIreland Clinical Research Facility @CRF_CORK | Cork #Rstats Users Group meetup.com/Cork-Ireland-R...

Cork, Ireland

darrendahly.github.io

Data Scientists IRL
@DataSci_Ireland Follows you

Promoting the Data Science professions in Ireland.

Ireland

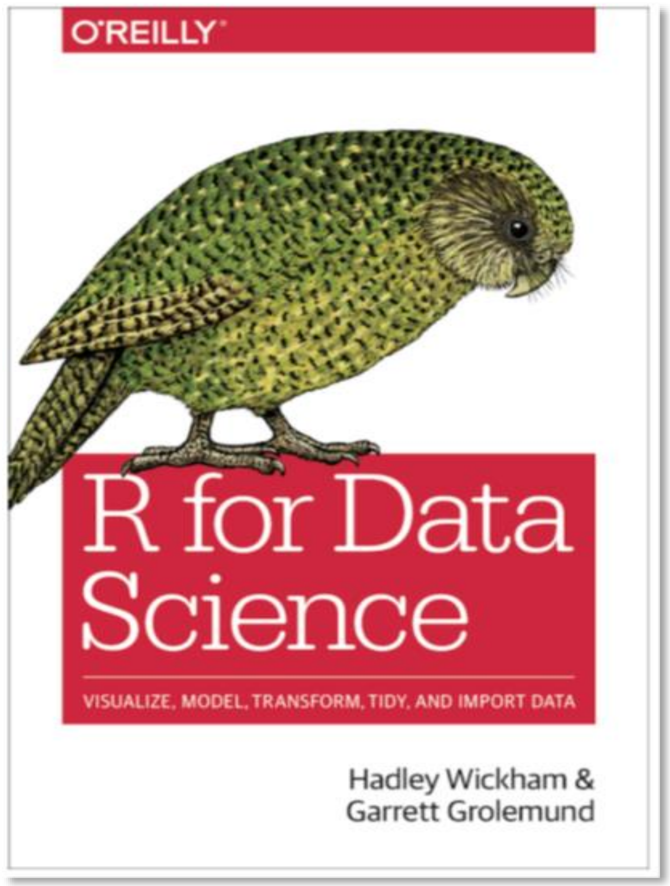
facebook.com/DataScientists...

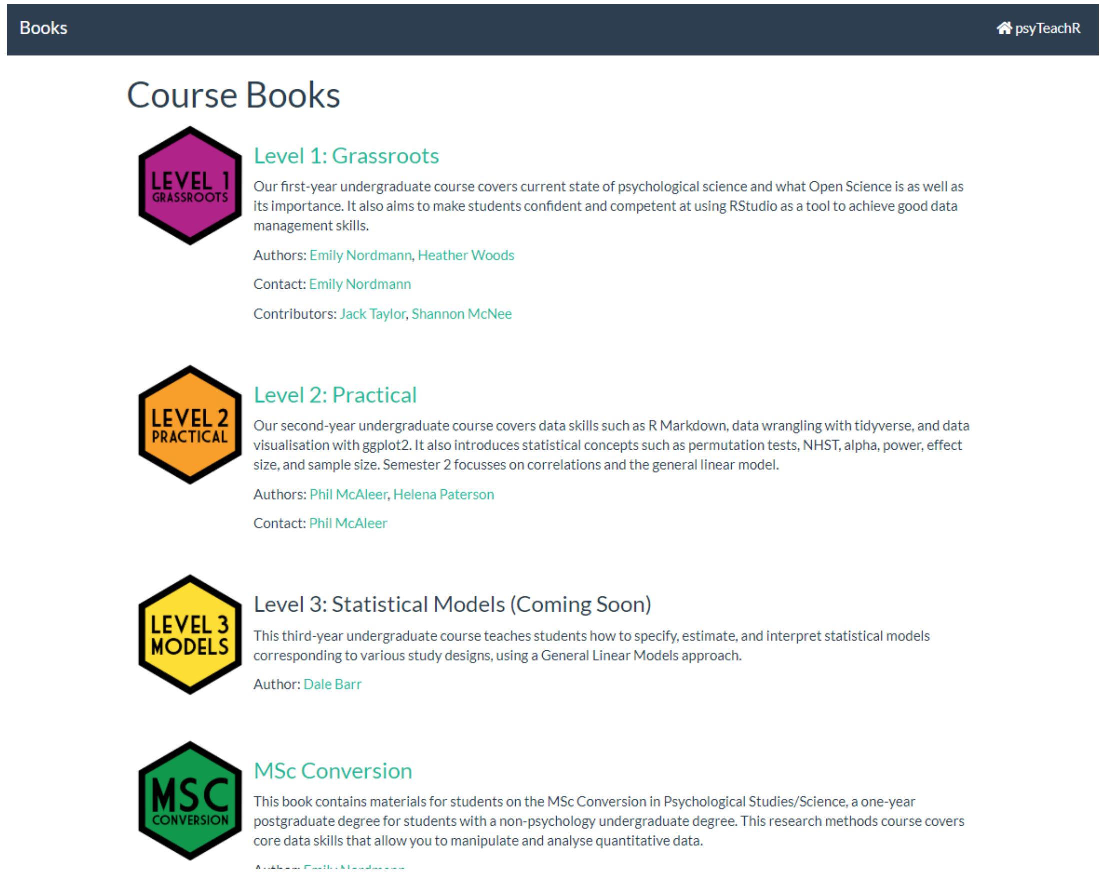
Kara Woo
@kara_woo

Research scientist at @sagebio. Data curation, visualization, #rstats, reproducibility, open science, ballet

karawoo.com

R is for Resources


O'REILLY
R for Data Science
VISUALIZE, MODEL, TRANSFORM, TIDY, AND IMPORT DATA
Hadley Wickham & Garrett Grolemund



Course Books

Level 1: Grassroots

Our first-year undergraduate course covers current state of psychological science and what Open Science is as well as its importance. It also aims to make students confident and competent at using RStudio as a tool to achieve good data management skills.

Authors: Emily Nordmann, Heather Woods

Contact: Emily Nordmann

Contributors: Jack Taylor, Shannon McNee

Level 2: Practical

Our second-year undergraduate course covers data skills such as R Markdown, data wrangling with tidyverse, and data visualisation with ggplot2. It also introduces statistical concepts such as permutation tests, NHST, alpha, power, effect size, and sample size. Semester 2 focusses on correlations and the general linear model.

Authors: Phil McAleer, Helena Paterson

Contact: Phil McAleer

Level 3: Statistical Models (Coming Soon)

This third-year undergraduate course teaches students how to specify, estimate, and interpret statistical models corresponding to various study designs, using a General Linear Models approach.

Author: Dale Barr

MSc Conversion

This book contains materials for students on the MSc Conversion in Psychological Studies/Science, a one-year postgraduate degree for students with a non-psychology undergraduate degree. This research methods course covers core data skills that allow you to manipulate and analyse quantitative data.

Hadley Wickham
@hadleywickham

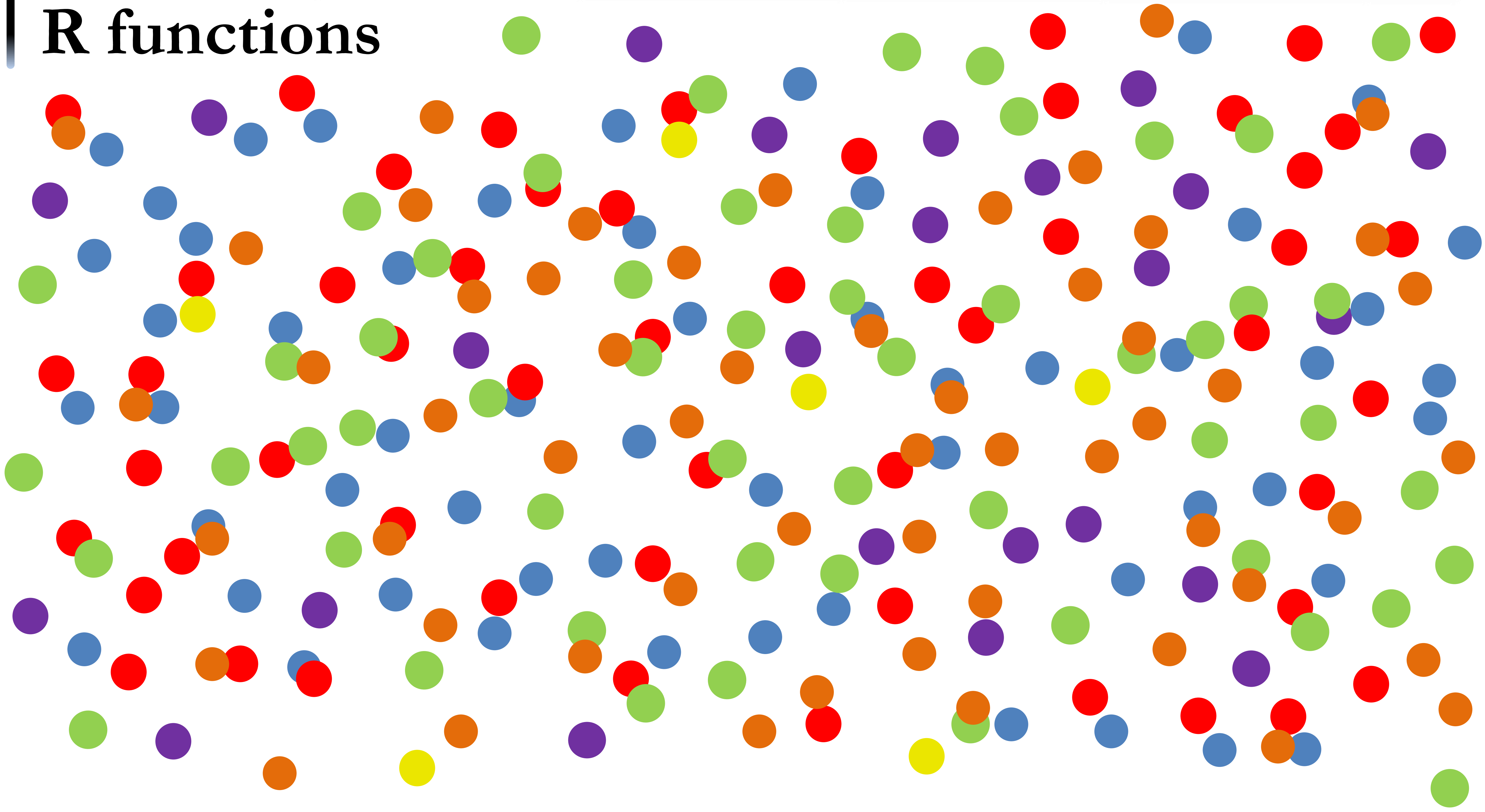
Garrett Grolemund
@StatGarrett

Emily Nordmann
@emilynordmann

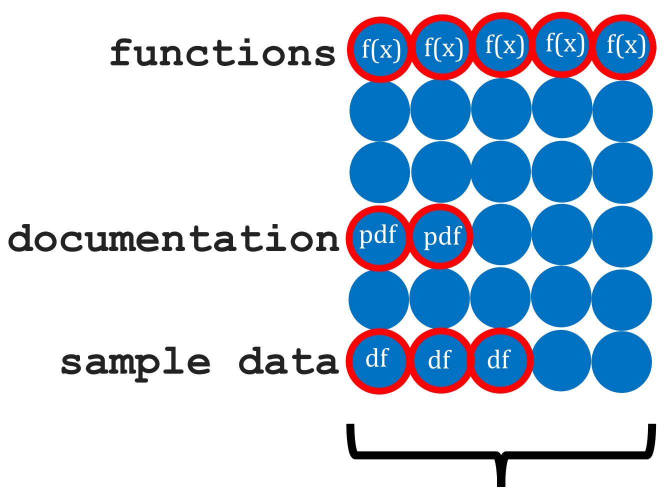
Lisa DeBruine 🏳️‍🌈
@LisaDeBruine

Phil McAleer
@McAleerP

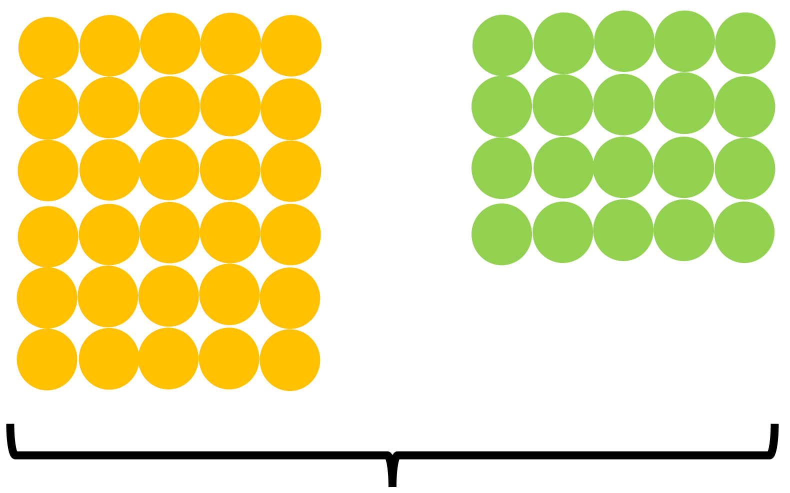
R functions



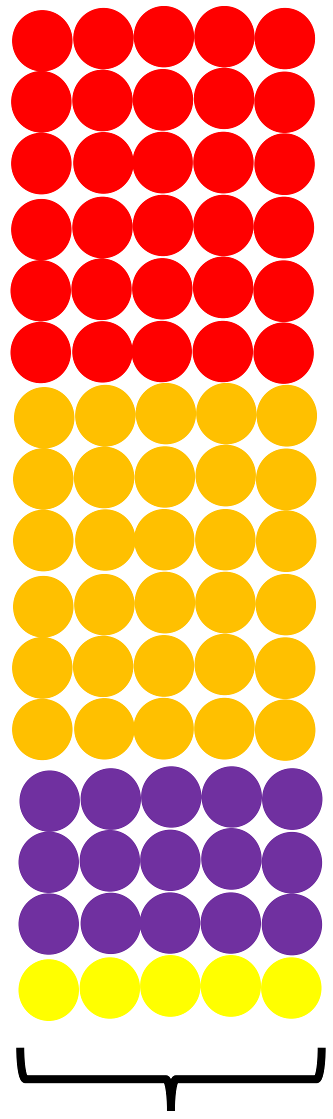
R packages



R comes pre-loaded with ~30 other packages (e.g. base, stats, graphics etc.)

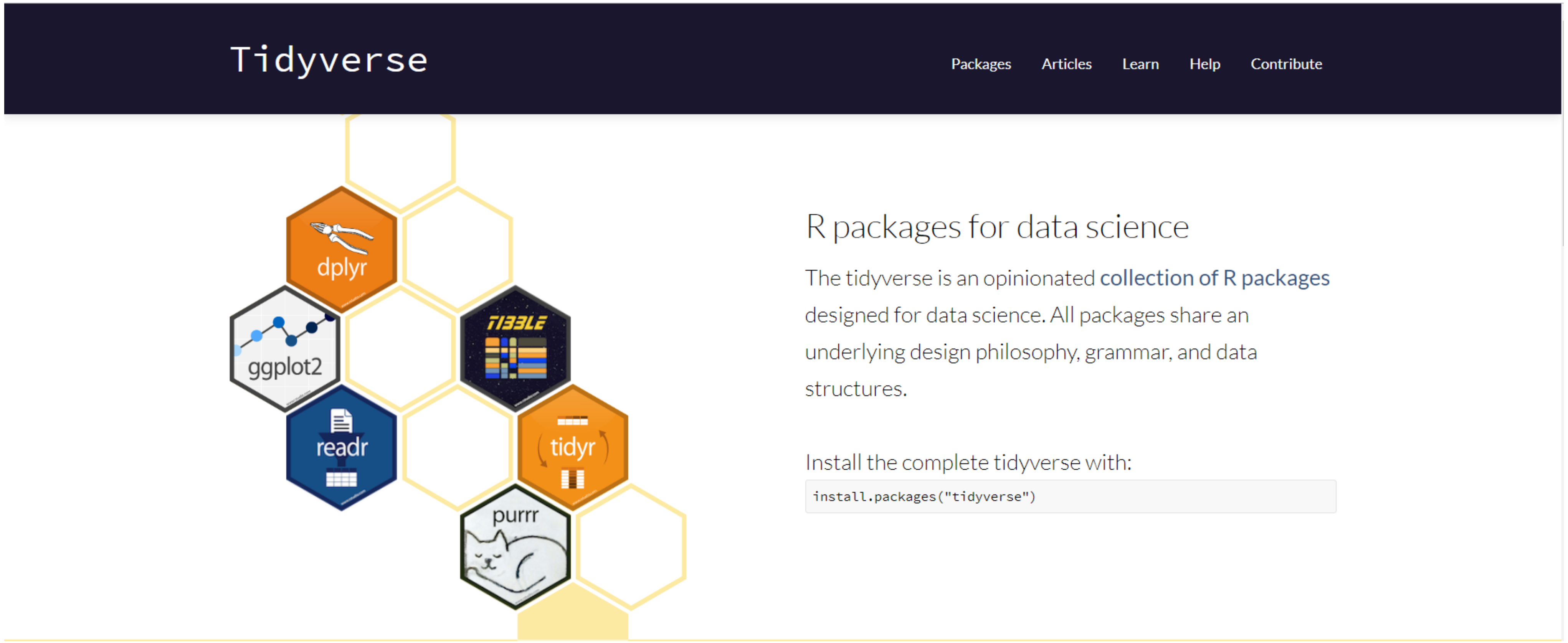


Other packages:
Install once
Update regularly
Load each session



tidyverse

What is the tidyverse?



R packages for data science

The tidyverse is an opinionated collection of R packages designed for data science. All packages share an underlying design philosophy, grammar, and data structures.

Install the complete tidyverse with:

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

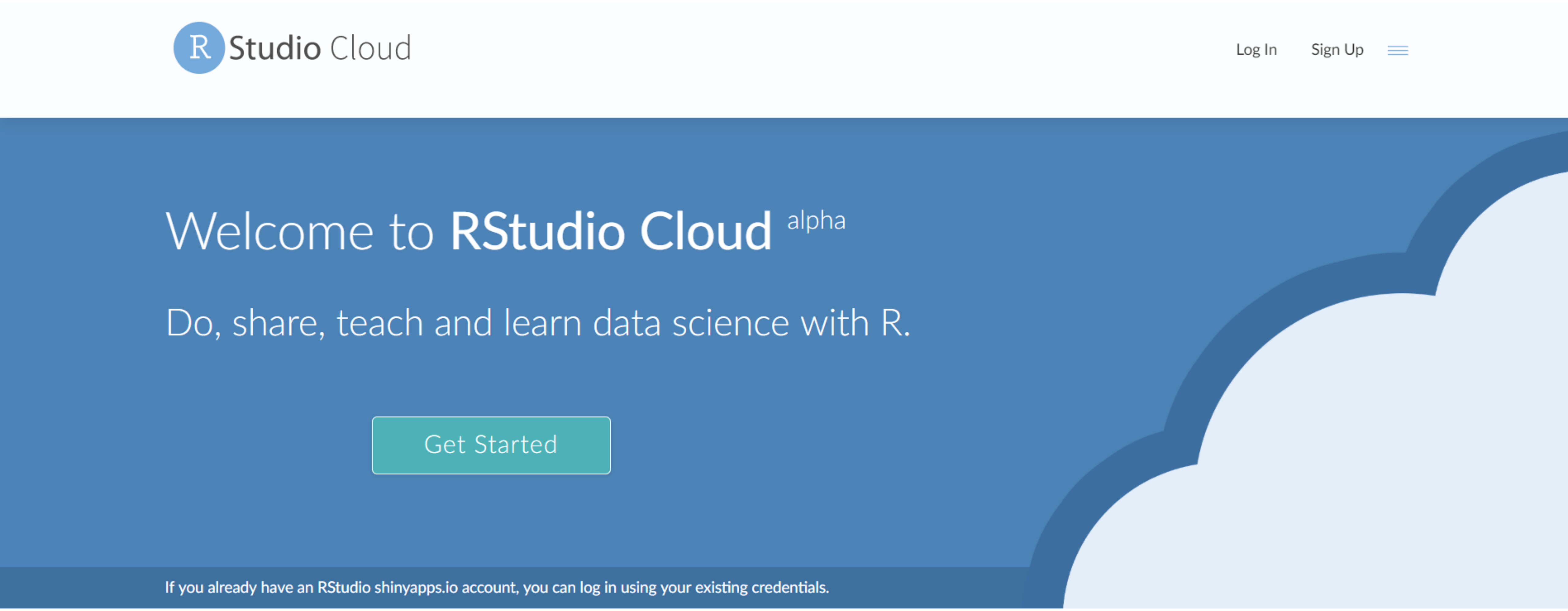
- Joined up collection of packages for data analysis
    - Consistent functions
    - Uses (tidy) data
    - Supports end-to-end workflows

What is the tidyverse?

```
> install.packages(c(“broom”, “cli2”, “crayon”,
“dbplyr”, “dplyr”, “forcats”, “ggplot2”, “haven”,
“hms”, “httr”, “jsonlite”, “lubridate”,
“magrittr”, “modelr”, “pillar”, “purrr”, “readr”,
“readxl”, “reprex”, “rlang”, “rstudioapi”,
“rvest”, “stringr”, “tibble”, “tidyr”, “xml2”)
```

```
> install.packages(“tidyverse”)
```

RStudio Cloud

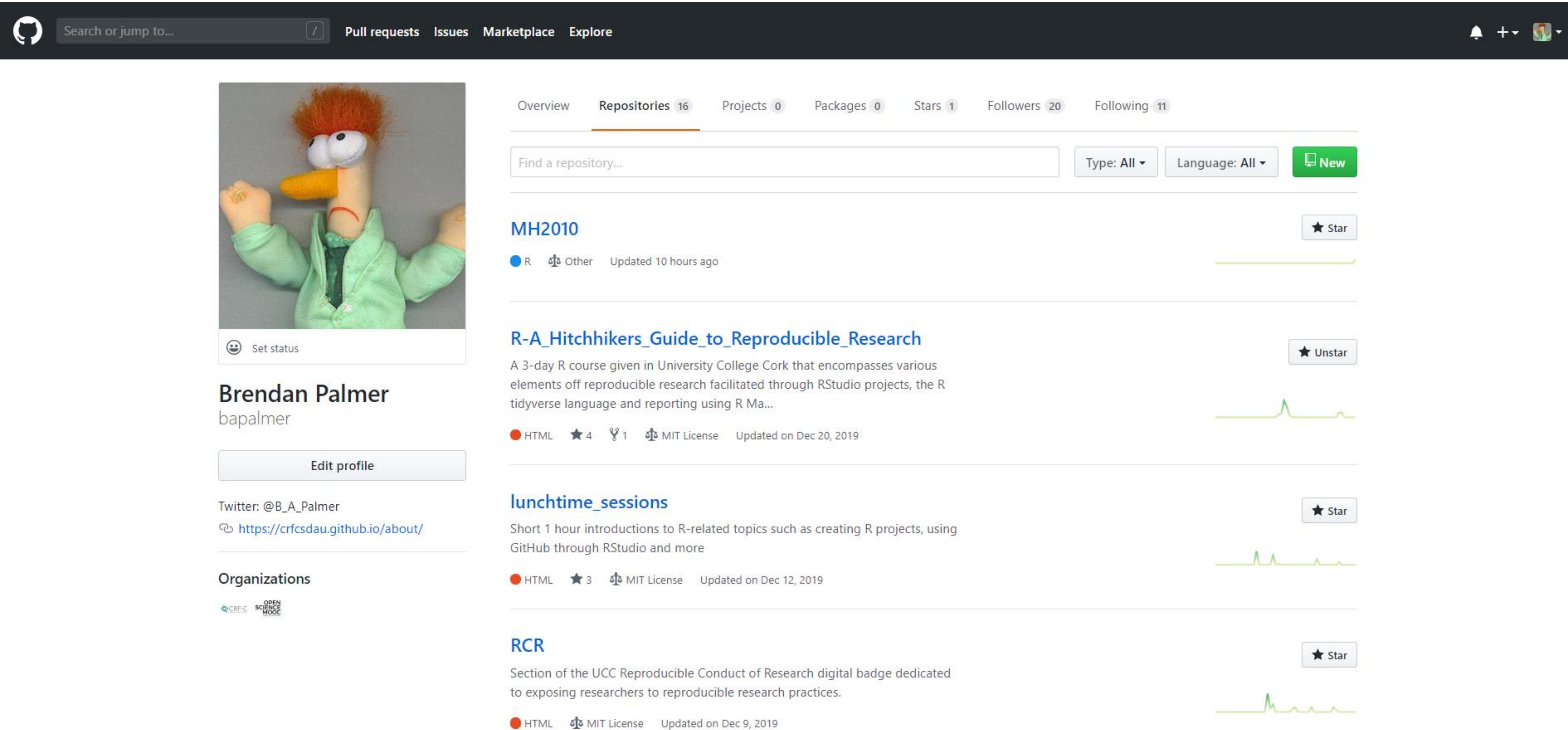


Welcome to **RStudio Cloud** alpha

Do, share, teach and learn data science with R.

If you already have an RStudio shinyapps.io account, you can log in using your existing credentials.

GitHub



Brendan Palmer
bapalmer

Twitter: @B_A_Palmer
https://crfcsdau.github.io/about/

Organizations

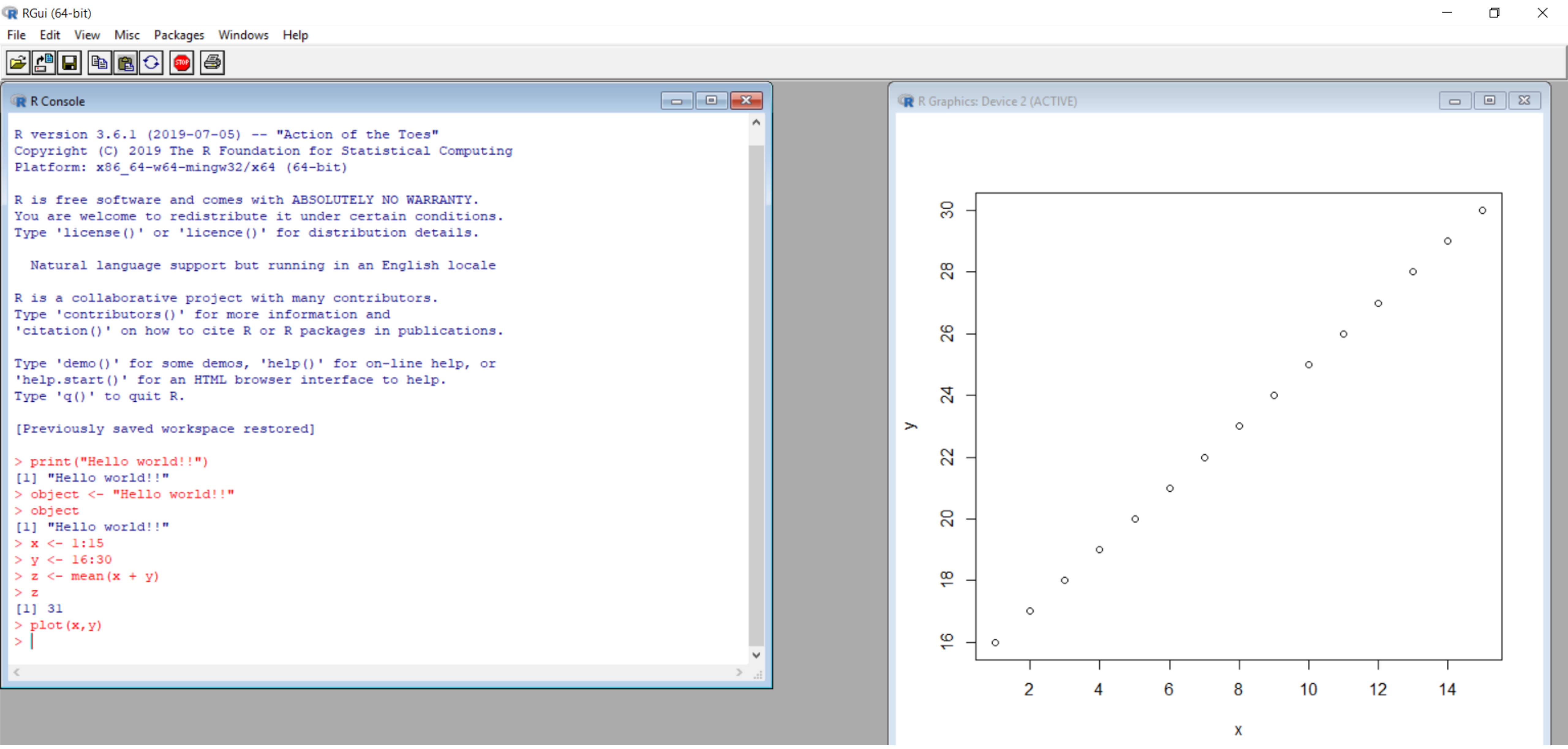
MH2010
R · Other · Updated 10 hours ago

R-A_Hitchhikers_Guide_to_Reproducible_Research
A 3-day R course given in University College Cork that encompasses various elements off reproducible research facilitated through RStudio projects, the R tidyverse language and reporting using R Ma...
HTML · ★ 4 · 1 · MIT License · Updated on Dec 20, 2019

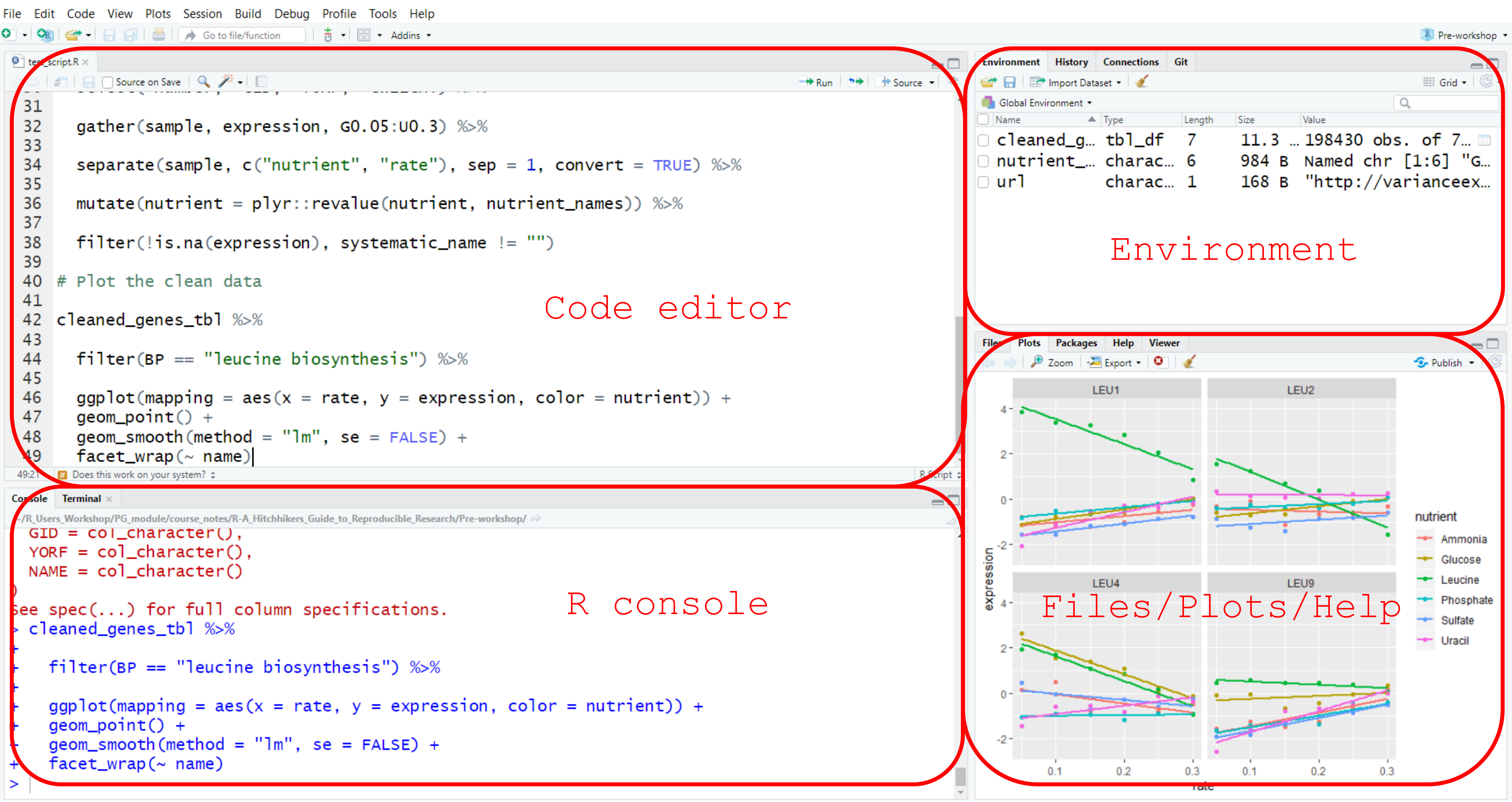
lunchtime_sessions
Short 1 hour introductions to R-related topics such as creating R projects, using GitHub through RStudio and more
HTML · ★ 3 · MIT License · Updated on Dec 12, 2019

RCR
Section of the UCC Reproducible Conduct of Research digital badge dedicated to exposing researchers to reproducible research practices.
HTML · MIT License · Updated on Dec 9, 2019

R user interface versus RStudio


RGui (64-bit)
File Edit View Misc Packages Windows Help
R Console
R Graphics: Device 2 (ACTIVE)
y
x
16
18
20
22
24
26
28
30
2
4
6
8
10
12
14

```
R version 3.6.1 (2019-07-05) -- "Action of the Toes"
Copyright (C) 2019 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Previously saved workspace restored]

> print("Hello world!!")
[1] "Hello world!!"
> object <- "Hello world!!"
> object
[1] "Hello world!!"
> x <- 1:15
> y <- 16:30
> z <- mean(x + y)
> z
[1] 31
> plot(x,y)
> |
```



```
  gather(sample, expression, G0.05:U0.3) %>%

  separate(sample, c("nutrient", "rate"), sep = 1, convert = TRUE) %>%

  mutate(nutrient = plyr::revalue(nutrient, nutrient_names)) %>%

  filter(!is.na(expression), systematic_name != "")

# Plot the clean data

cleaned_genes_tbl %>%

  filter(BP == "leucine biosynthesis") %>%

  ggplot(mapping = aes(x = rate, y = expression, color = nutrient)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  facet_wrap(~ name)
```

49:21 Does this work on your system? R Script

Name	Type	Length	Size	Value
cleaned_g…	tbl_df	7	11.3 …	198430 obs. of 7…
nutrient_…	charac…	6	984 B	Named chr [1:6] "G…
url	charac…	1	168 B	"http://varianceex…

~/R_Users_Workshop/PG_module/course_notes/R-A_Hitchhikers_Guide_to_Reproducible_Research/Pre-workshop/

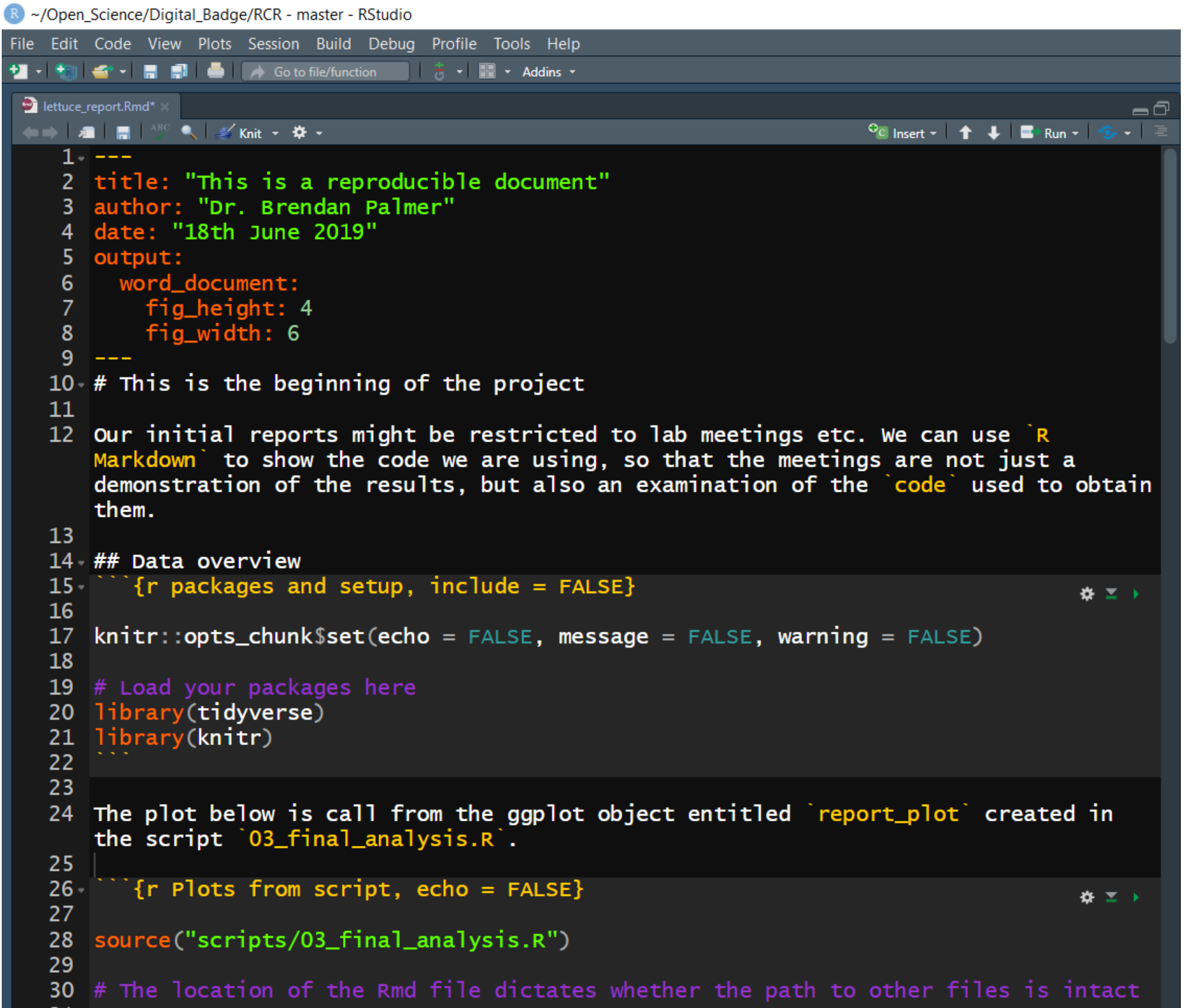
```
  GID = col_character(),
  YORF = col_character(),
  NAME = col_character()
)
See spec(...) for full column specifications.
> cleaned_genes_tbl %>%
+
+   filter(BP == "leucine biosynthesis") %>%
+
+   ggplot(mapping = aes(x = rate, y = expression, color = nutrient)) +
+   geom_point() +
+   geom_smooth(method = "lm", se = FALSE) +
+   facet_wrap(~ name)
>
```

Central theme of this session

You don't become an expert by doing expert things

Dr Felienne Hermans, RStudio Conference 2019

R Markdown


~/Open_Science/Digital_Badge/RCR - master - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
lettuce_report.Rmd*
Knit
Insert
Run

~~~
---
title: "This is a reproducible document"
author: "Dr. Brendan Palmer"
date: "18th June 2019"
output:
  word_document:
    fig_height: 4
    fig_width: 6
---
# This is the beginning of the project

Our initial reports might be restricted to lab meetings etc. We can use `R
Markdown` to show the code we are using, so that the meetings are not just a
demonstration of the results, but also an examination of the `code` used to obtain
them.

## Data overview
```{r packages and setup, include = FALSE}

knitr::opts_chunk$set(echo = FALSE, message = FALSE, warning = FALSE)

# Load your packages here
library(tidyverse)
library(knitr)
~~~

The plot below is call from the ggplot object entitled `report_plot` created in
the script `03_final_analysis.R`.

```{r Plots from script, echo = FALSE}

source("scripts/03_final_analysis.R")

# The location of the Rmd file dictates whether the path to other files is intact
```

This is a reproducible document

Dr. Brendan Palmer

18th June 2019

This is the beginning of the project

Our initial reports might be restricted to lab meetings etc. We can use `R Markdown` to show the code we are using, so that the meetings are not just a demonstration of the results, but also an examination of the code used to obtain them.

Data overview

The plot below is call from the ggplot object entitled `report_plot` created in the script `03_final_analysis.R`.

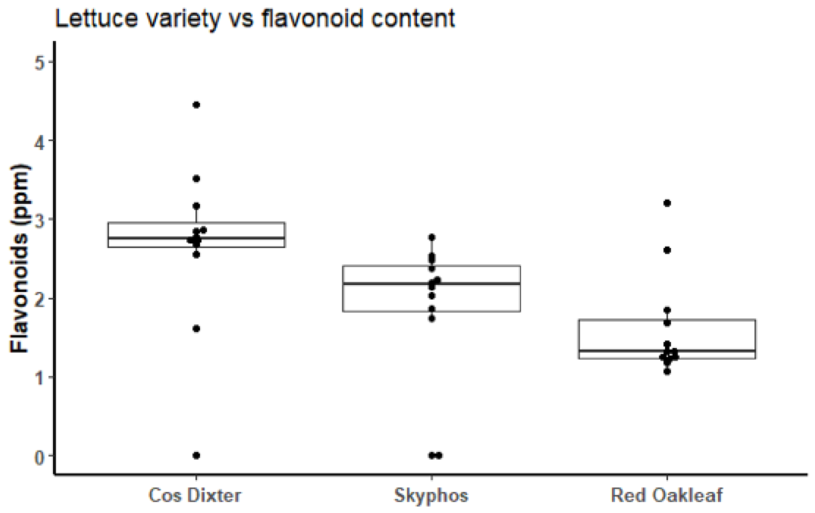
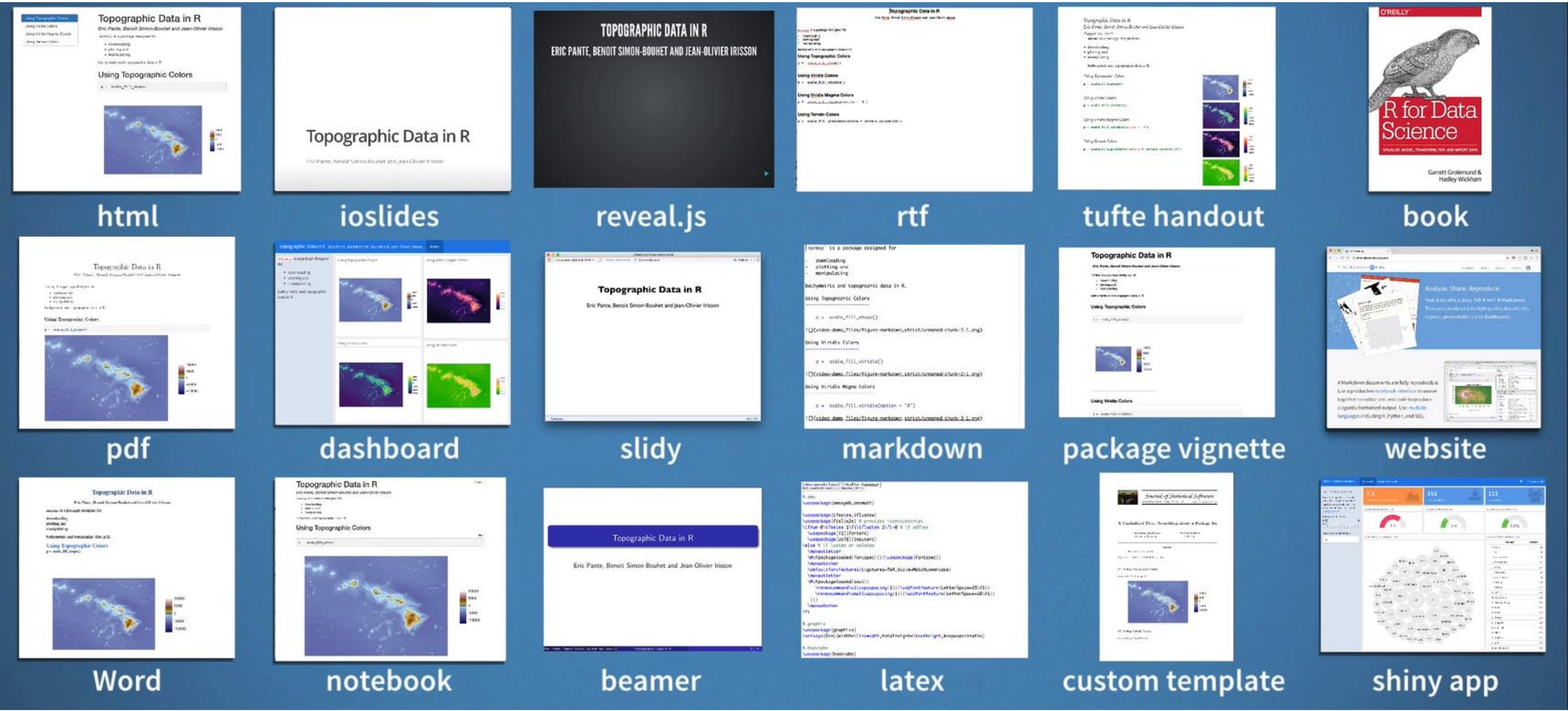


Fig. 1. Flavonoid content of three lettuce varieties under three experimental conditions.

Or we can also recreate the code within the R Markdown document as seen below.

What has R Markdown ever done for us?



R Markdown

YAML header

````
---
title: "Diamond sizes"
date: 2016-08-25
output: html_document
---
````

Chunks of code

~~~
```{r setup, include = FALSE}
library(ggplot2)
library(dplyr)
smaller <- diamonds %>%
filter(carat <= 2.5)
```
~~~

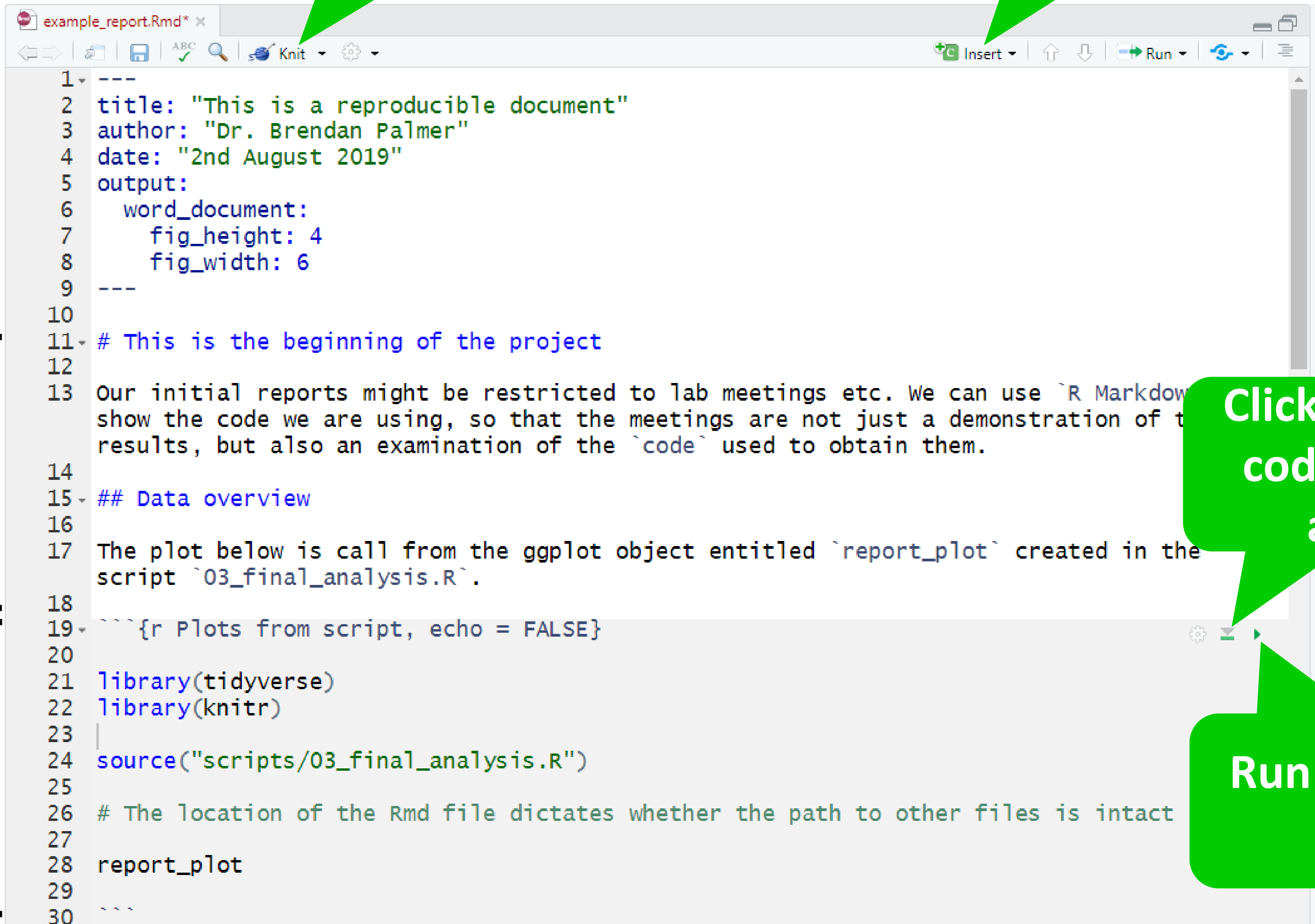
Plain text with data outputs from R code

````
We have data about `r nrow(diamonds)`
diamonds. Only
`r nrow(diamonds) - nrow(smaller)` are
larger than
2.5 carats. The distribution of the
remainder is shown below:
````

Chunks of code

~~~
```{r, echo = FALSE}
smaller %>%
ggplot(aes(carat)) +
geom_freqpoly(binwidth = 0.01)
```
~~~

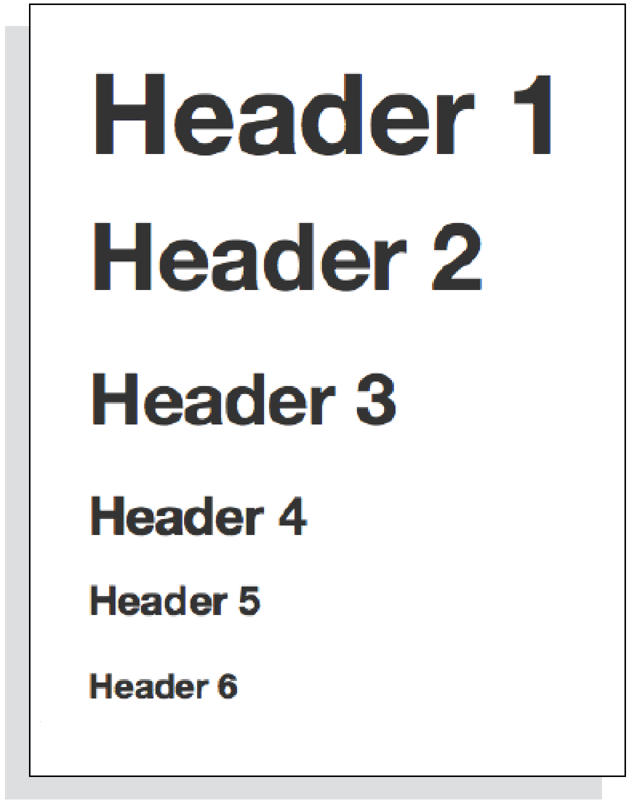
R Markdown



~~~
---
title: "This is a reproducible document"
author: "Dr. Brendan Palmer"
date: "2nd August 2019"
output:
  word_document:
    fig_height: 4
    fig_width: 6
---

# This is the beginning of the project

Our initial reports might be restricted to lab meetings etc. We can use `R Markdow
show the code we are using, so that the meetings are not just a demonstration of t
results, but also an examination of the `code` used to obtain them.

## Data overview

The plot below is call from the ggplot object entitled `report_plot` created in the
script `03_final_analysis.R`.

```{r Plots from script, echo = FALSE}

library(tidyverse)
library(knitr)

source("scripts/03_final_analysis.R")

# The location of the Rmd file dictates whether the path to other files is intact

report_plot

```
~~~

R Markdown - Headers

```
# Header 1
## Header 2
### Header 3
#### Header 4
##### Header 5
###### Header 6
```


Header 1

Header 2

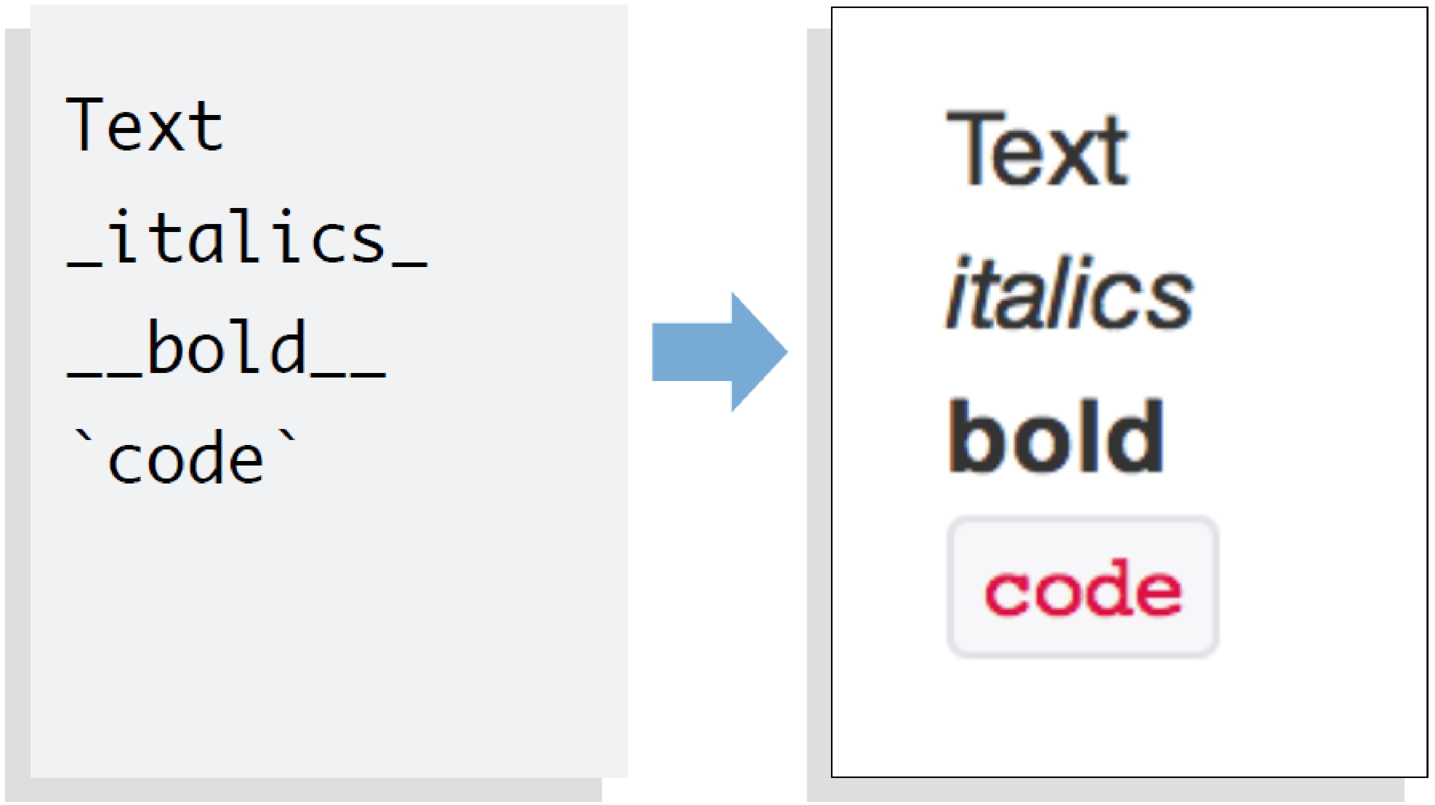
Header 3

Header 4

Header 5

Header 6

R Markdown - Formatting



```
Text
_italics_
__bold__
`code`
```

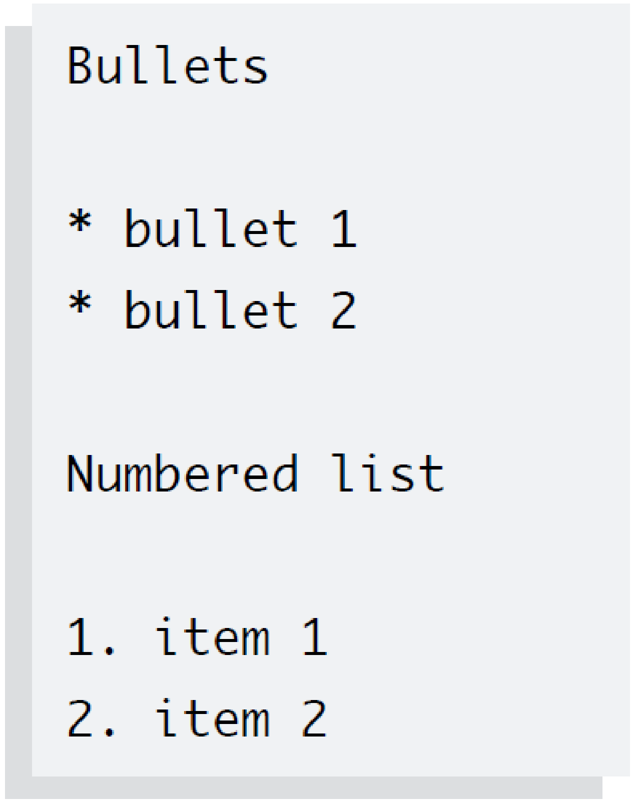
Text

italics

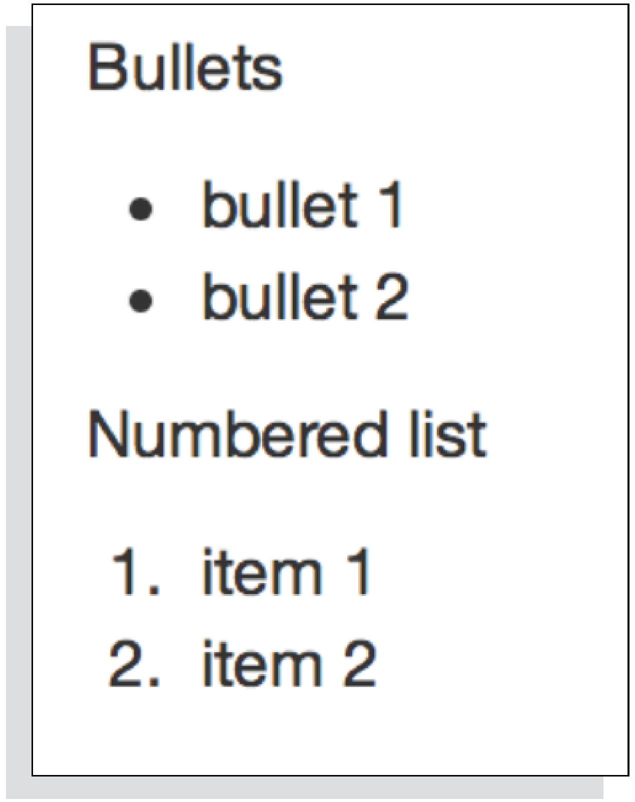
bold

`code`

R Markdown - Lists



```
Bullets

* bullet 1
* bullet 2

Numbered list

1. item 1
2. item 2
```



Bullets

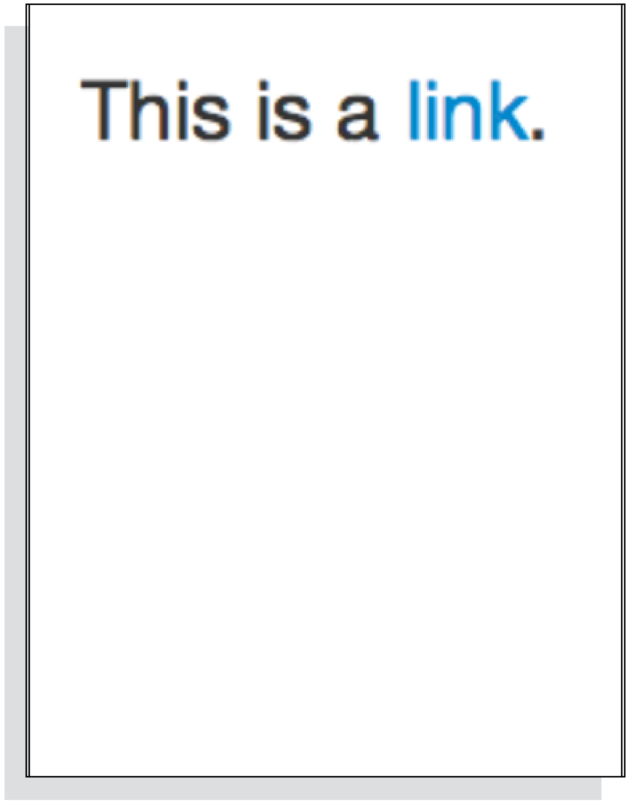
- bullet 1
- bullet 2

Numbered list

1. item 1
2. item 2

R Markdown - Hyperlinks

```
This is a
[link](www.git.com).
```



This is a link.

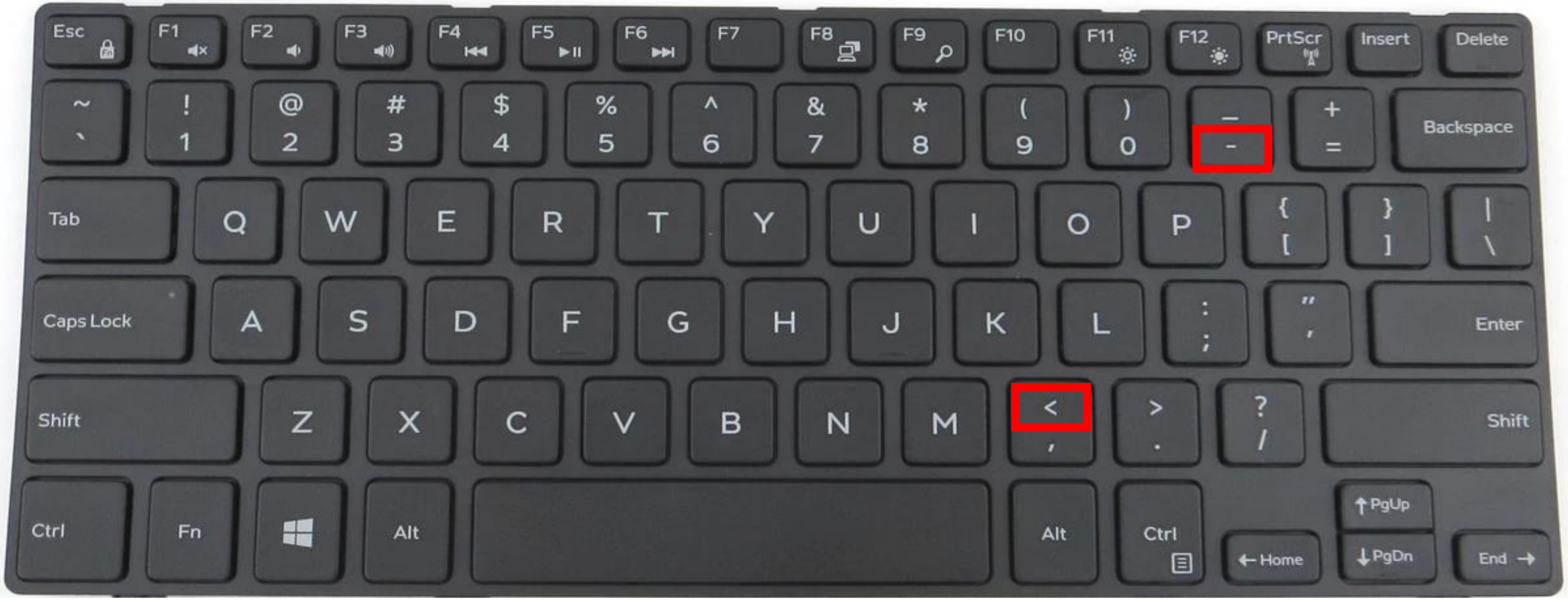
Basics of R code

Symbol	What it does	Example 1	Example 2
<-	Assign operator Creates new objects	> x <- 5 > x [1] 5	> y <- "This" > y [1] "This"
c()	Helps create objects with more than one element	> v <- c(5,6,7,8) > v [1] 5 6 7 8	> w <- c("This", "is", "easy! ") > w [1] "This" "is" "easy!"
#	Computer ignores what is written. Used for adding notes to code	> # print("hello") >	> print("hello") [1] "hello"
%>%	Literally translates as "then do this"	> data %>% do_something_to(data)	> data %>% do_something_to(data) %>% do_something_else_to(data)
%in%	returns a logical vector indicating if there is a match	> "x" %in% c("x", "y", "z") [1] TRUE	> c("x", "y", "z") %in% "x" [1] TRUE FALSE FALSE
?	Access help	> ?mean()	> ?geom_point()

FYI: R is case sensitive!! Name.of.data ≠ name.of.data

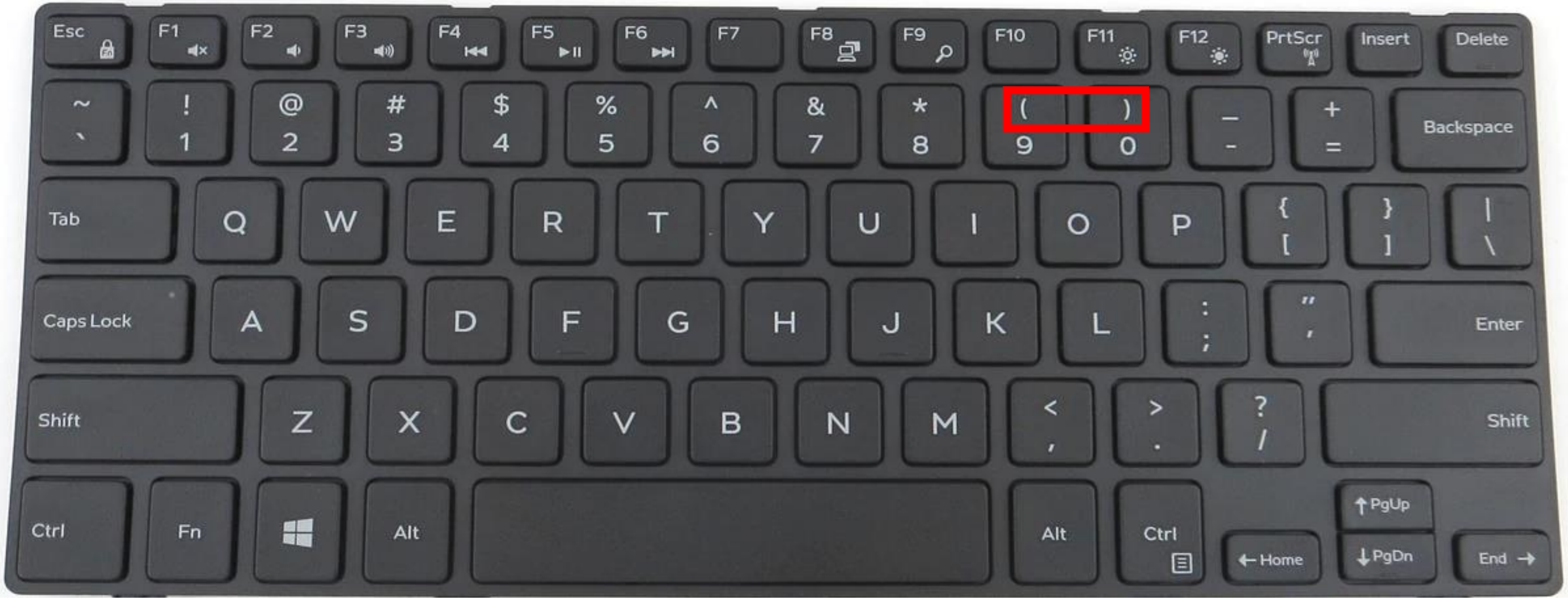
Say hello to the lesser known keyboard keys

`Assignment operator: <-`



Say hello to the lesser known keyboard keys

`Functions take arguments inside round brackets: function()`



Say hello to the lesser known keyboard keys

Indexing occurs inside square brackets: []

Say hello to the lesser known keyboard keys

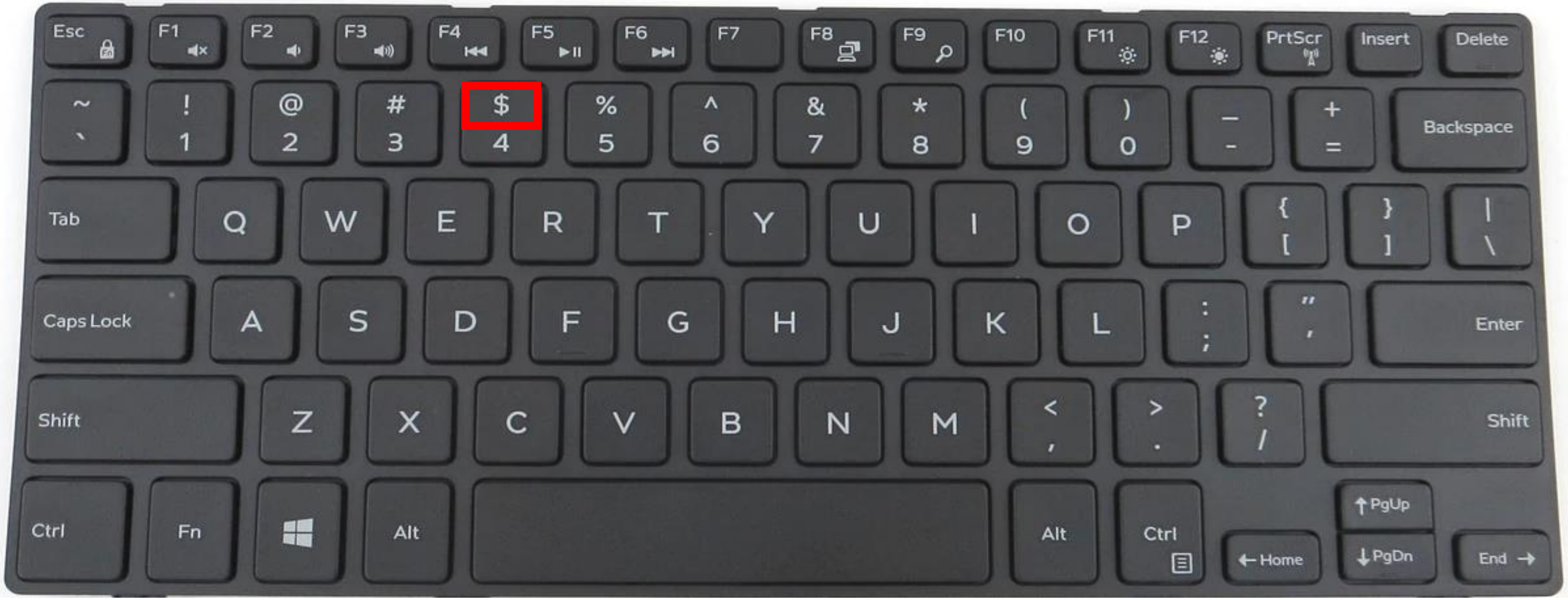
`Functions are defined inside curly brackets: {}`

Say hello to the lesser known keyboard keys

`You can comment out your code using the hash key: #`

Say hello to the lesser known keyboard keys

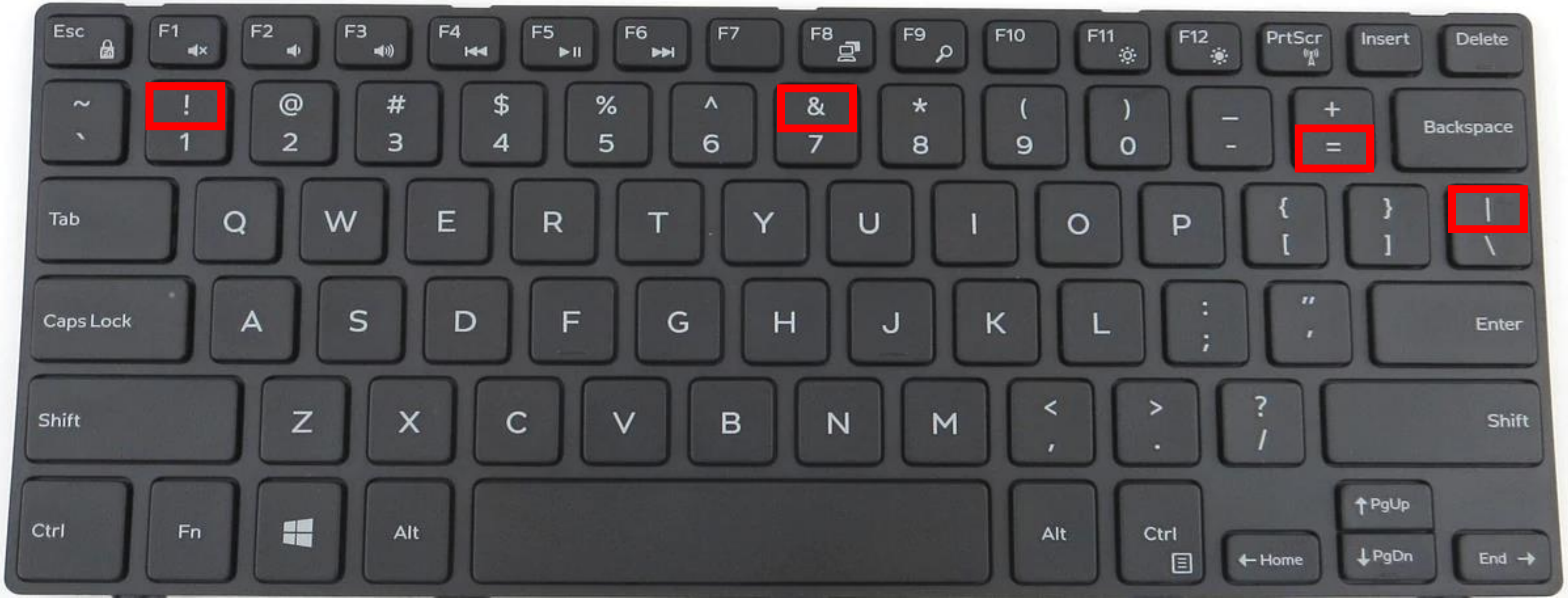
`Dollar sign allows you extract elements by name: $`



Say hello to the lesser known keyboard keys

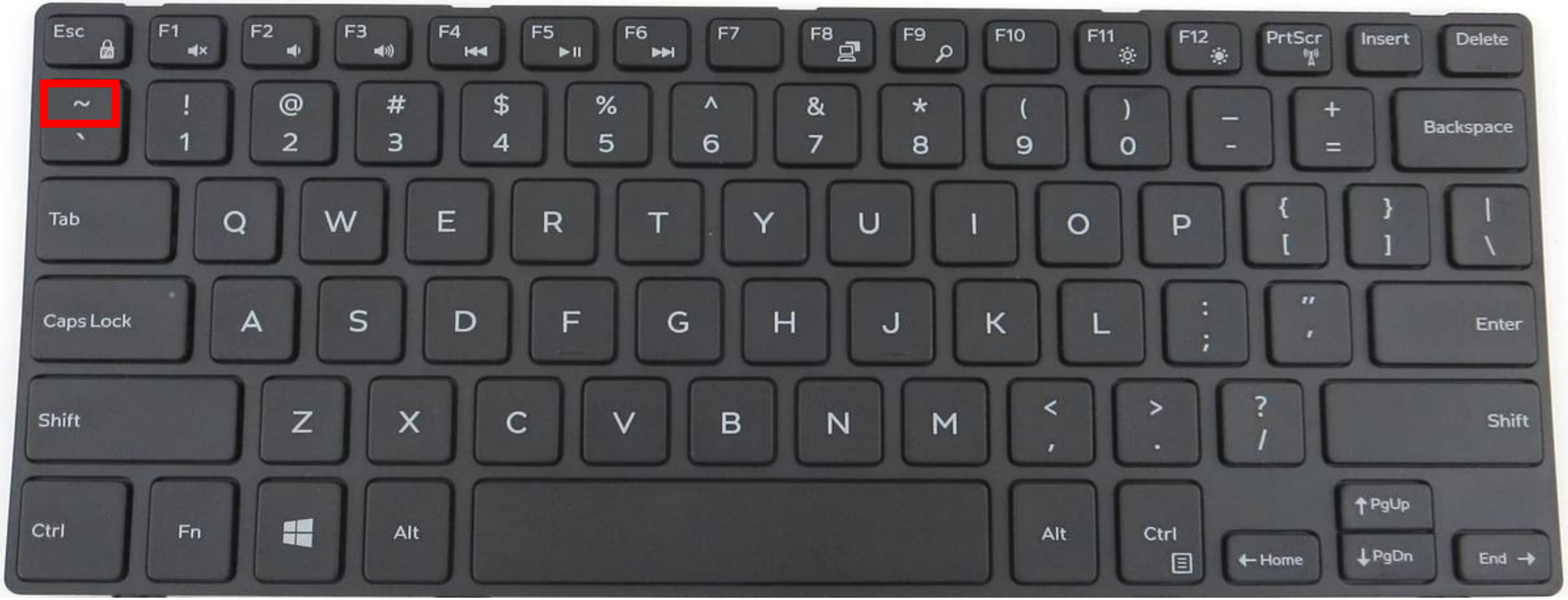
Logical TRUE/FALSE operators equals, not equals, and, or:

`==, !=, &, |`



Say hello to the lesser known keyboard keys

Tilde operator for use in formulas: ~



Say hello to the lesser known keyboard keys

Tab key for autocomplete

To understand R, remember the following

- Everything that exists is an object

- Everything that happens is a function

Creating objects

```
For most of us, R is simply the creation of and manipulation
of objects:
new_object <- c(1, 2, 3)

- the objects are then fed into functions to create amazing
  new objects

amazing_new_object <- function(new_object)

Broadly speaking the following is true in R:
- information
> data_frame  <- function(information)
> plot        <- function(data_frame)
> model       <- function(data_frame)
```

Naming objects

```
There are a few simple rules to follow initially:

- Object names must start with a letter and can only contain
  letters, numbers, '_' and '.'

- Certain characters should not be used, e.g:
     - c is the concatenate function 'c()'
     - T is used as shorthand for TRUE
     - F is used as shorthand for FALSE
```

Three main types of data structure

```
The main data types are;

# double (for double precision floating point numbers)
typeof(1.23)

# character
typeof("string")

# logical
typeof(FALSE)

# missing values are represented by NA
example <- c(1, 2, NA, 4)
```

Nested functions

It is possible to 'nest' functions within functions…

```
round(mean(sample(pop, n, replace = FALSE)), 2)
```

- `sample(pop, n, replace = FALSE)` will give us a numeric vector
- `mean(numeric vector)` will calculate the mean of this vector
- `round(mean, 2)` will round up the answer to two decimal places

Nested functions

It is possible to 'nest' functions within functions…

```
round(mean(sample(pop, n, replace = FALSE)), 2)
```

- sample(pop, n, replace = FALSE) will give us a numeric vector
- mean(numeric vector) will calculate the mean of this vector
- round(mean, 2) will round up the answer to two decimal places

Nested functions

It is possible to 'nest' functions within functions…

```
round(mean(sample(pop, n, replace = FALSE)), 2)
```

- `sample(pop, n, replace = FALSE)` will give us a numeric vector
- `mean(numeric vector)` will calculate the mean of this vector
- `round(mean, 2)` will round up the answer to two decimal places

Types of data structure

- Vectors come in two forms

A: Atomic vectors contain exactly one type of data

```
all_numbers          <- c(1, 2, 0.5, -0.5, 3.4)
all_characters       <- c("One", "too", "3")
all_logical          <- c(TRUE, FALSE) # NOTE: Type it out
```

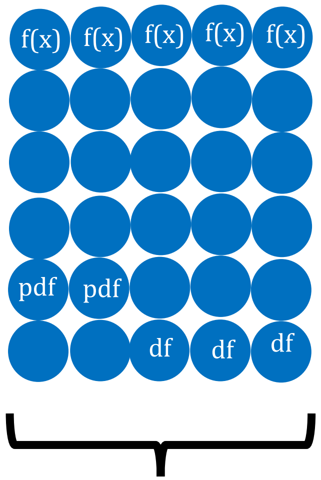
B: Lists allow combinations of different types of data

```
this_is_a_list       <- list(1, TRUE, "Three", "4")
typeof(this_is_a_list)
[1] "list"

this_is_also_a_list <- list(all_numbers, all_characters)
```

Walkthrough example script

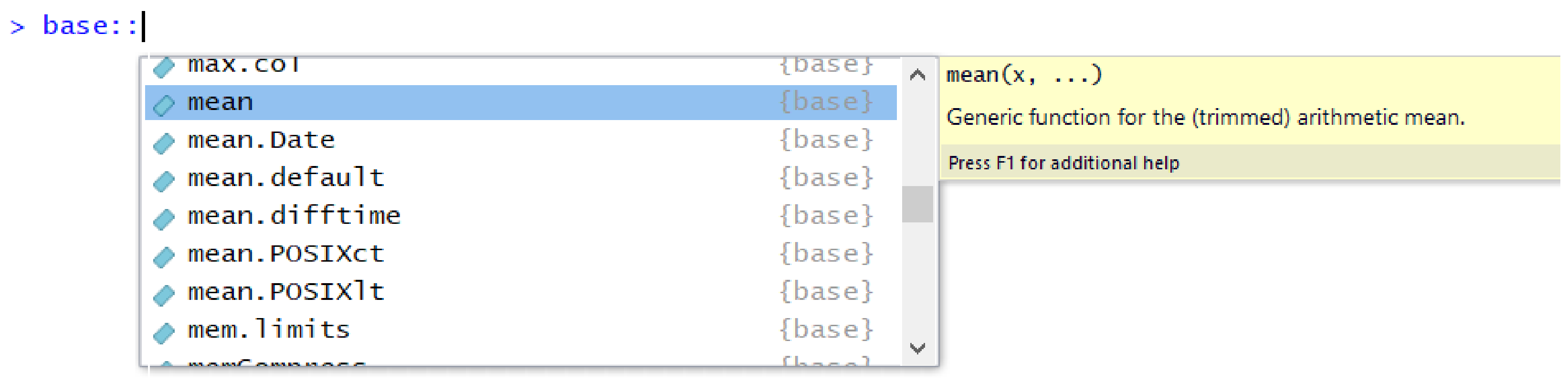
```
- scripts/01_baseR_introduction.R
```

Package contents



Base R: Comes pre-loaded

Functions



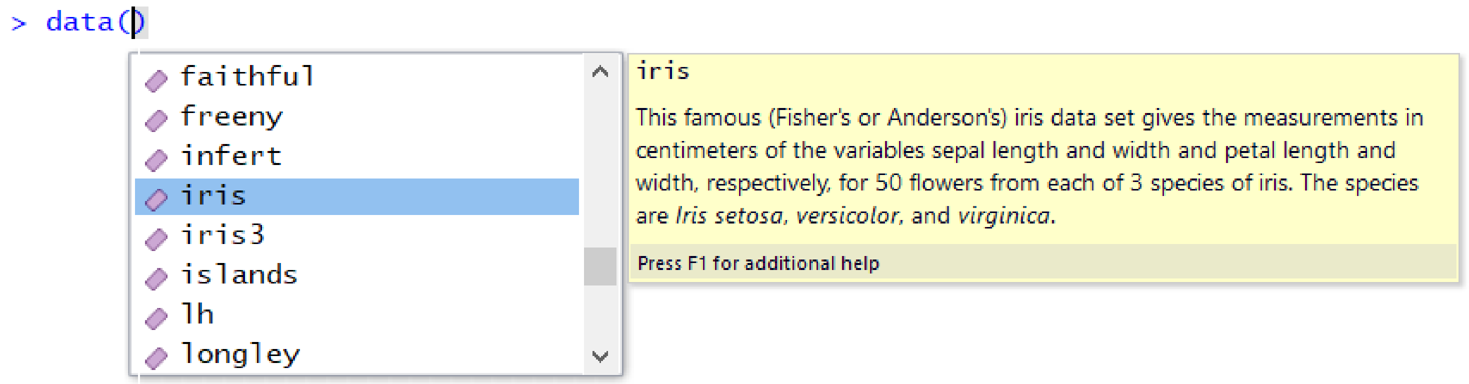
```
> base::|
max.col                {base}
mean                   {base}
mean.Date              {base}
mean.default           {base}
mean.difftime          {base}
mean.POSIXct           {base}
mean.POSIXlt           {base}
mem.limits             {base}
memCompress            {base}
```

mean(x, ...)

Generic function for the (trimmed) arithmetic mean.

Press F1 for additional help

Data sets



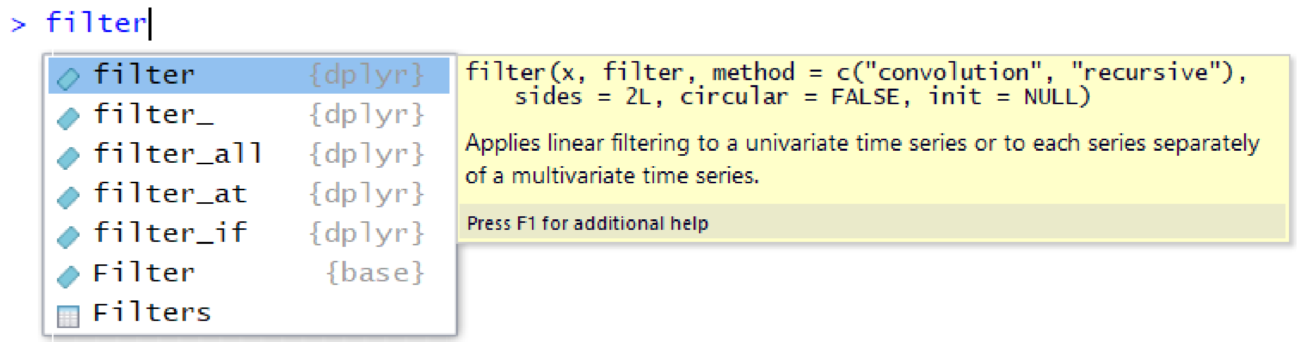
```
> data()
faithful
freeny
infert
iris
iris3
islands
lh
longley
```

iris

This famous (Fisher's or Anderson's) iris data set gives the measurements in centimeters of the variables sepal length and width and petal length and width, respectively, for 50 flowers from each of 3 species of iris. The species are *Iris setosa*, *versicolor*, and *virginica*.

Press F1 for additional help

Conflicts



```
> filter|
filter       {dplyr}
filter_      {dplyr}
filter_all   {dplyr}
filter_at    {dplyr}
filter_if    {dplyr}
Filter        {base}
Filters
```

```
filter(x, filter, method = c("convolution", "recursive"),
    sides = 2L, circular = FALSE, init = NULL)
```

Applies linear filtering to a univariate time series or to each series separately of a multivariate time series.

Press F1 for additional help

Walkthrough example script

- `scripts/02_navigating_R_packages.R`

Practical notes

- `docs/frequentist_inference.Rmd`

OPTIONAL EXTRAS

Types of data structure

```
# Matrices/Arrays:

- You can have a matrix of two or more dimensions
  a_matrix <- matrix(1:9, 3, 3)

- Vectors and matrices can only contain one type of data

- VERY VERY VERY NB: If you try to create a vector with more
  than one data type, then it will undergo coercion to the
  least common denominator

- The coercion rule goes:
      logical -> integer -> numeric -> complex -> character

- You can perform coercions yourself on vectors
```

Types of data structures

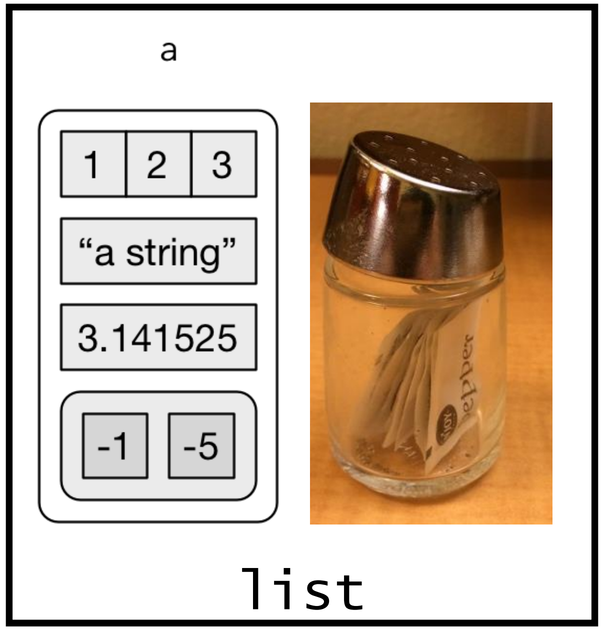
```
# Dataframes:
- These are a special type of list
- Observations are in rows
- Variables are in columns
- Labels or other metadata may also be present

> a_data_frame <- data.frame(number = 1:10,
                      char = sample(letters, 10),
                      this_really_a_col_name  = rep(c(TRUE, FALSE), 5))
```

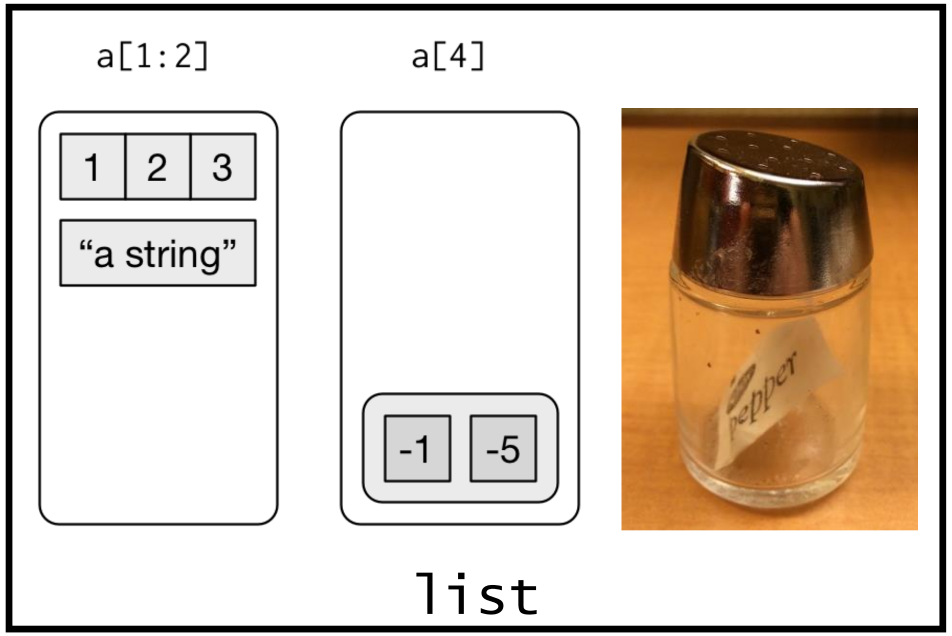
Indexing

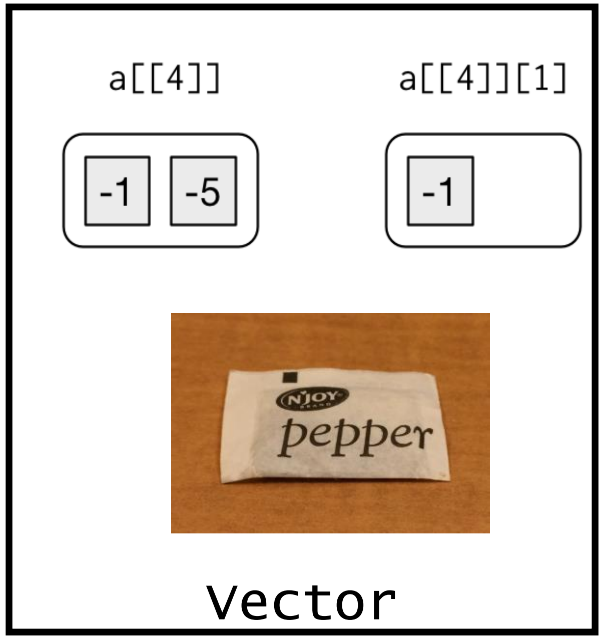
```
- Indexing can occur in one or two dimensions
- One dimension:
     new_object <- c(1, 2, 3)
     new_object[1]
     [1] 1

- Two dimensions
     a_data_frame[1, 1] # i.e [Row number 1, Column number 1]
     a_data_frame$number[1] # i.e. Column called number, row 1
```

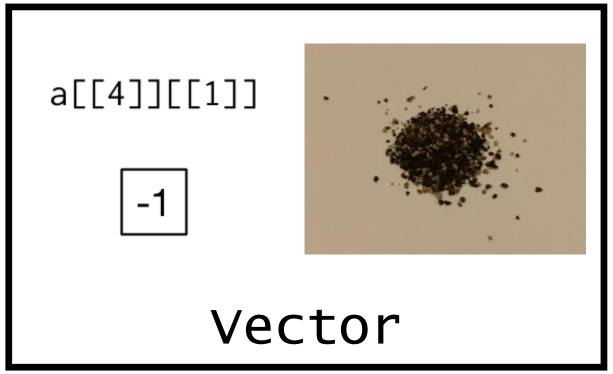
Indexing

```
# Recall
- this_is_also_a_list <- list(all_numbers, all_characters, all_logical)
```









```
# Important
-   [ extracts a sublist, results will be a list
-   [[ extracts a single component
```

Image: R for Data Science, Wickham and Grolemund

Try it out for yourself

```
- scripts/03_practice_worksheet.R
```