Introduction to R

Santiago Caño Muñiz, Dr Aaron Weimann, Dr Chris Ruis

https://aweimann.github.io/floto-lab-learning-bioinformatics/docs/

The goal

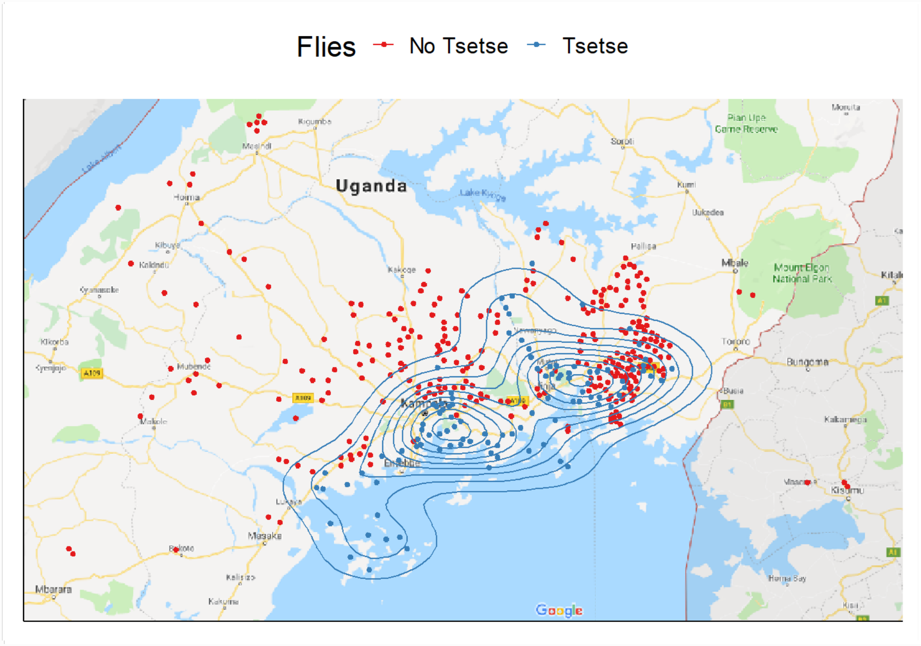
Learning to program is learning to think

Learning R in a workshop is an impossible task. Hence, our goal is to illustrate the capabilities of this tool. How programming can make your work much easier.

Why learn R?

A way to represent ideas

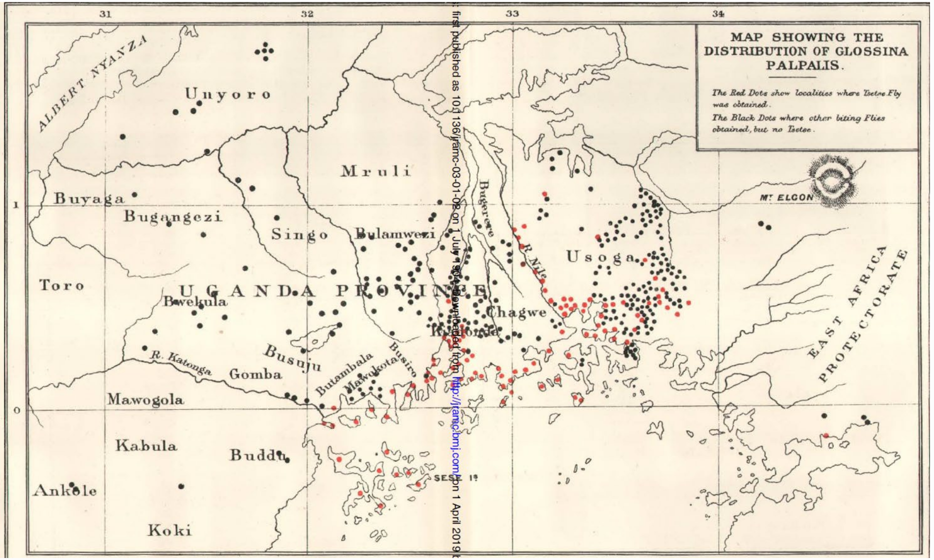
Age	Sex	District	Shamba	Trypanosoma	Filaria
25	M	Sese Island	Sewana	+	+
20	M	Sese Island	Kaganda I	-	+
25	M	Sese Island	Semagala I	+	+
30	M	Sese Island	Kaganda I	-	+
20	M	Sese Island	Semagala I	-	+
25	M	Sese Island	Buvovu I	-	+
25	M	Sese Island	Kaganda I	-	+
20	M	Sese Island	Semagala I	-	+
30	M	Sese Island	Buvu I	-	+
35	M	Sese Island	Semagala I	-	-
20	M	Sese Island	Semagala I	-	-
......	......	......	......	......	......
......	......	......	......	......	......
......	......	......	......	......	......
25	M	Sese Island	Semagala I	+	-
35	M	Sese Island	Bunami I	+	+



Why learn R?

A way to represent ideas

Age	Sex	District	Shamba	Trypanosoma	Filaria
25	M	Sese Island	Sewana	+	+
20	M	Sese Island	Kaganda I	-	+
25	M	Sese Island	Semagala I	+	+
30	M	Sese Island	Kaganda I	-	+
20	M	Sese Island	Semagala I	-	+
25	M	Sese Island	Buvovu I	-	+
25	M	Sese Island	Kaganda I	-	+
20	M	Sese Island	Semagala I	-	+
30	M	Sese Island	Buvu I	-	+
35	M	Sese Island	Semagala I	-	-
20	M	Sese Island	Semagala I	-	-
……	……	……	……	……	……
……	……	……	……	……	……
……	……	……	……	……	……
25	M	Sese Island	Semagala I	+	-
35	M	Sese Island	Bunami I	+	+

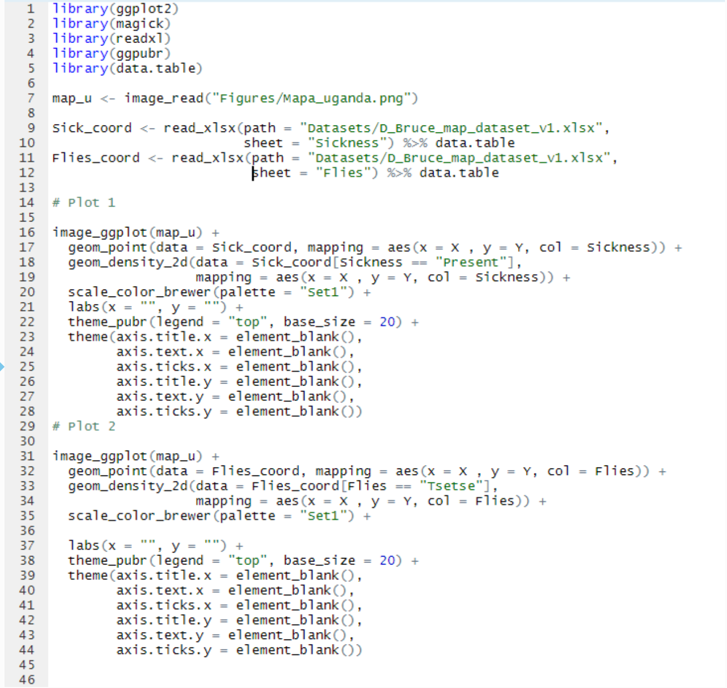

MAP SHOWING THE DISTRIBUTION OF GLOSSINA PALPALIS.
The Red Dots show localities where Tsetse Fly was obtained.
The Black Dots where other biting Flies obtained, but no Tsetse.
31
32
33
34
1
0
ALBERT NYANZA
Unyoro
Mruli
Buyaga
Bugangezi
Singo
Bulamwezi
Usoga
Toro
UGANDA PROVINCE
Bwekula
Chagwe
Kyagwe
R. Katonga
Gomba
Busuju
Butambala
Mawokota
Busiro
Bugerere
R. Nile
Mawogola
Kabula
Buddu
Ankole
Koki
SESE IS
MT ELGON
EAST AFRICA PROTECTORATE
first published as 10.1136/jramc-03-01-08 on 1 July 1904. Downloaded from http://jramc.bmj.com/ on 1 April 2019

Dr D. Bruce, 1903

Why learn R?

A way to represent ideas

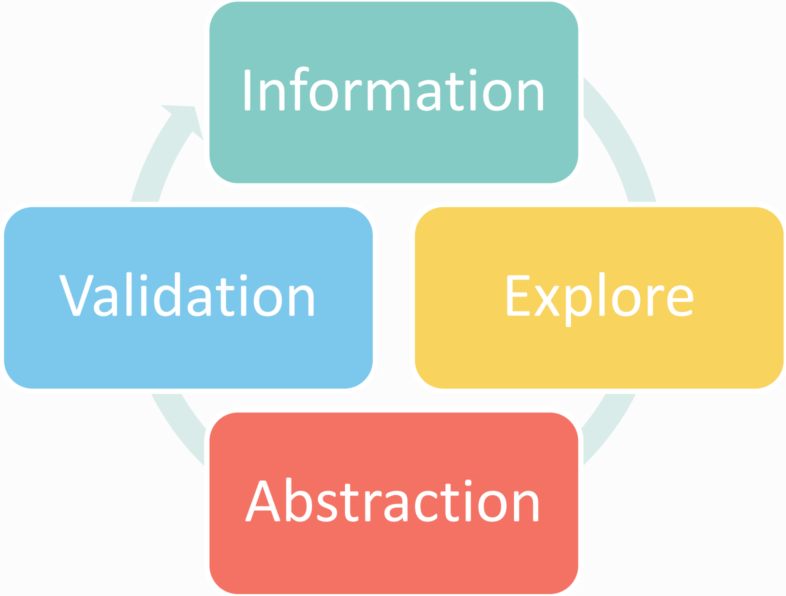
Age	Sex	District	Shamba	Trypanosoma	Filaria
25	M	Sese Island	Sewana	+	+
20	M	Sese Island	Kaganda I	-	+
25	M	Sese Island	Semagala I	+	+
30	M	Sese Island	Kaganda I	-	+
20	M	Sese Island	Semagala I	-	+
25	M	Sese Island	Buvovu I	-	+
25	M	Sese Island	Kaganda I	-	+
20	M	Sese Island	Semagala I	-	+
30	M	Sese Island	Buvu I	-	+
35	M	Sese Island	Semagala I	-	-
20	M	Sese Island	Semagala I	-	-
......	......	......	......	......	......
......	......	......	......	......	......
......	......	......	......	......	......
25	M	Sese Island	Semagala I	+	-
35	M	Sese Island	Bunami I	+	+



```r
library(ggplot2)
library(magick)
library(readxl)
library(ggpubr)
library(data.table)

map_u <- image_read("Figures/Mapa_uganda.png")

Sick_coord <- read_xlsx(path = "Datasets/D_Bruce_map_dataset_v1.xlsx",
                        sheet = "Sickness") %>% data.table
Flies_coord <- read_xlsx(path = "Datasets/D_Bruce_map_dataset_v1.xlsx",
                         sheet = "Flies") %>% data.table


# Plot 1

image_ggplot(map_u) +
  geom_point(data = Sick_coord, mapping = aes(x = X , y = Y, col = Sickness)) +
  geom_density_2d(data = Sick_coord[Sickness == "Present"],
                  mapping = aes(x = X , y = Y, col = Sickness)) +
  scale_color_brewer(palette = "Set1") +
  labs(x = "", y = "") +
  theme_pubr(legend = "top", base_size = 20) +
  theme(axis.title.x = element_blank(),
        axis.text.x = element_blank(),
        axis.ticks.x = element_blank(),
        axis.title.y = element_blank(),
        axis.text.y = element_blank(),
        axis.ticks.y = element_blank())
# Plot 2

image_ggplot(map_u) +
  geom_point(data = Flies_coord, mapping = aes(x = X , y = Y, col = Flies)) +
  geom_density_2d(data = Flies_coord[Flies == "Tsetse"],
                  mapping = aes(x = X , y = Y, col = Flies)) +
  scale_color_brewer(palette = "Set1") +

  labs(x = "", y = "") +
  theme_pubr(legend = "top", base_size = 20) +
  theme(axis.title.x = element_blank(),
        axis.text.x = element_blank(),
        axis.ticks.x = element_blank(),
        axis.title.y = element_blank(),
        axis.text.y = element_blank(),
        axis.ticks.y = element_blank())
```

The cycle of data analysis

One ring to model them all



Information

Explore

Abstraction

Validation

What’s R

A tool to make life easier

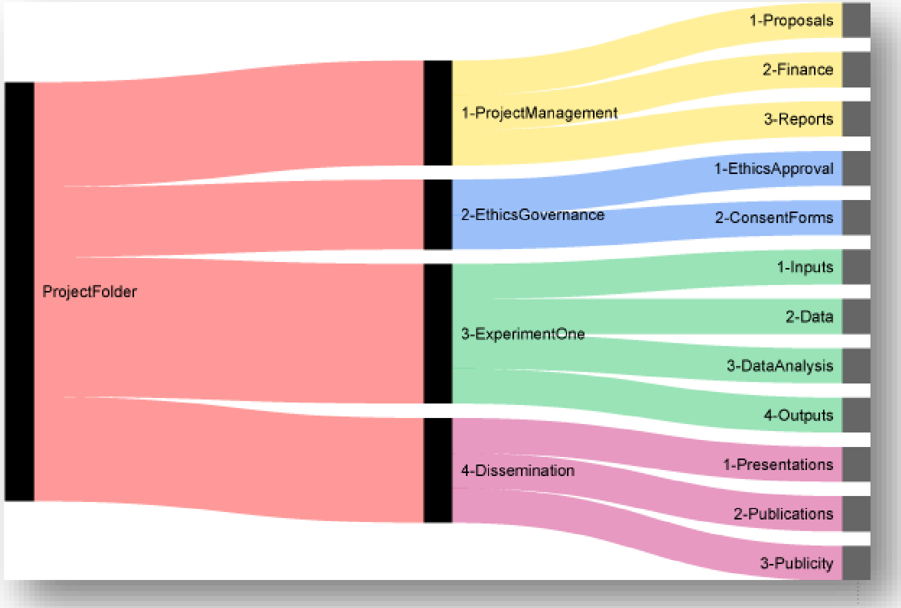
- **Dynamic programming**
- Open source
- Active community
- Endless packages

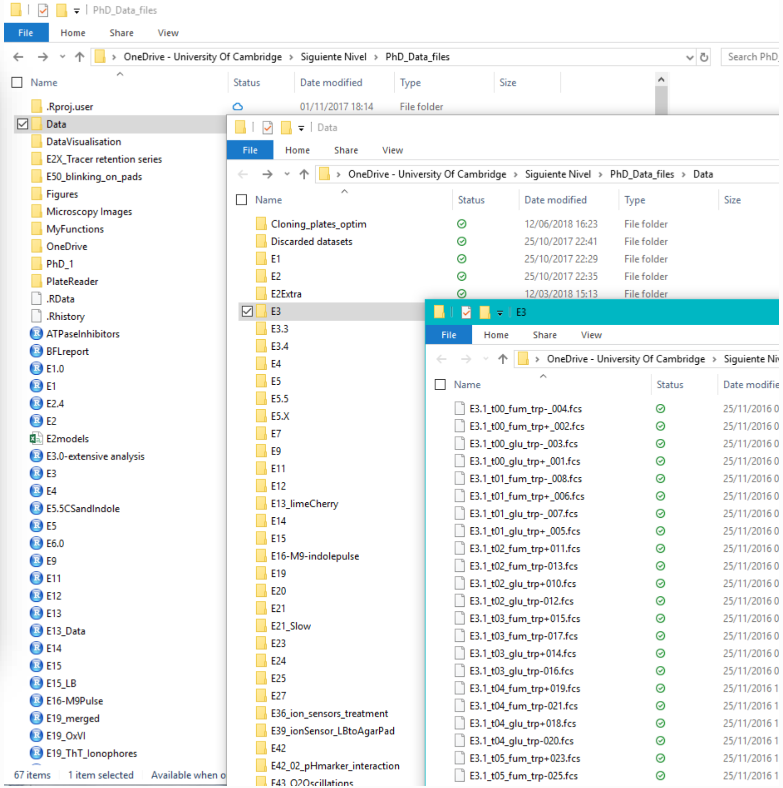
What’s R?

How was my first experience

Arrange files and folders

Thinking in hierarchical order





OneDrive - University Of Cambridge › Siguiente Nivel › PhD_Data_files

Name	Status	Date modified	Type	Size
.Rproj.user	☁	01/11/2017 18:14	File folder	
☑ Data				
DataVisualisation				
E2X_Tracer retention series				
E50_blinking_on_pads				
Figures				
Microscopy Images				
MyFunctions				
OneDrive				
PhD_1				
PlateReader				
.RData				
.Rhistory				
ATPaseInhibitors				
BFLreport				
E1.0				
E1				
E2.4				
E2				
E2models				
E3.0-extensive analysis				
E3				
E4				
E5.5CSandIndole				
E5				
E6.0				
E9				
E11				
E12				
E13				
E13_Data				
E14				
E15				
E15_LB				
E16-M9Pulse				
E19_merged				
E19_OxVI				
E19_ThT_Ionophores				

67 items 1 item selected Available when o

OneDrive - University Of Cambridge › Siguiente Nivel › PhD_Data_files › Data

Name	Status	Date modified	Type	Size
Cloning_plates_optim	✓	12/06/2018 16:23	File folder	
Discarded datasets	✓	25/10/2017 22:41	File folder	
E1	✓	25/10/2017 22:29	File folder	
E2	✓	25/10/2017 22:35	File folder	
E2Extra	✓	12/03/2018 15:13	File folder	
☑ E3				
E3.3				
E3.4				
E4				
E5				
E5.5				
E5.X				
E7				
E9				
E11				
E12				
E13_limeCherry				
E14				
E15				
E16-M9-indolepulse				
E19				
E20				
E21				
E21_Slow				
E23				
E24				
E25				
E27				
E36_ion_sensors_treatment				
E39_ionSensor_LBtoAgarPad				
E42				
E42_02_pHmarker_interaction				
E43_O2Oscillations				

OneDrive - University Of Cambridge › Siguiente Ni

Name	Status	Date modifie
E3.1_t00_fum_trp-_004.fcs	✓	25/11/2016 0
E3.1_t00_fum_trp+_002.fcs	✓	25/11/2016 0
E3.1_t00_glu_trp-_003.fcs	✓	25/11/2016 0
E3.1_t00_glu_trp+_001.fcs	✓	25/11/2016 0
E3.1_t01_fum_trp-_008.fcs	✓	25/11/2016 0
E3.1_t01_fum_trp+_006.fcs	✓	25/11/2016 0
E3.1_t01_glu_trp-_007.fcs	✓	25/11/2016 0
E3.1_t01_glu_trp+_005.fcs	✓	25/11/2016 0
E3.1_t02_fum_trp+011.fcs	✓	25/11/2016 0
E3.1_t02_fum_trp-013.fcs	✓	25/11/2016 0
E3.1_t02_glu_trp+010.fcs	✓	25/11/2016 0
E3.1_t02_glu_trp-012.fcs	✓	25/11/2016 0
E3.1_t03_fum_trp+015.fcs	✓	25/11/2016 0
E3.1_t03_fum_trp-017.fcs	✓	25/11/2016 0
E3.1_t03_glu_trp+014.fcs	✓	25/11/2016 0
E3.1_t03_glu_trp-016.fcs	✓	25/11/2016 0
E3.1_t04_fum_trp+019.fcs	✓	25/11/2016 1
E3.1_t04_fum_trp-021.fcs	✓	25/11/2016 1
E3.1_t04_glu_trp+018.fcs	✓	25/11/2016 1
E3.1_t04_glu_trp-020.fcs	✓	25/11/2016 1
E3.1_t05_fum_trp+023.fcs	✓	25/11/2016 1
E3.1_t05_fum_trp-025.fcs	✓	25/11/2016 1

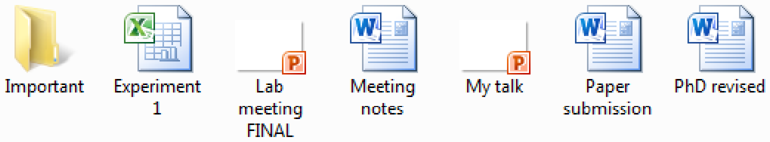
® http://nikola.me/folder_structure.html

Arrange files and folders

Naming… really matters?

Copyright: http://10pm.com/

PSD
Ps
new.psd
newfinal.psd
newfinalfinal.psd
newfinalestfinal.psd
newfinalestfinal forsure.psd
newfinalestfuckthis shitfinal.psd
©10PM.COM



Will we remember what does it means at the end of the thesis or Project?

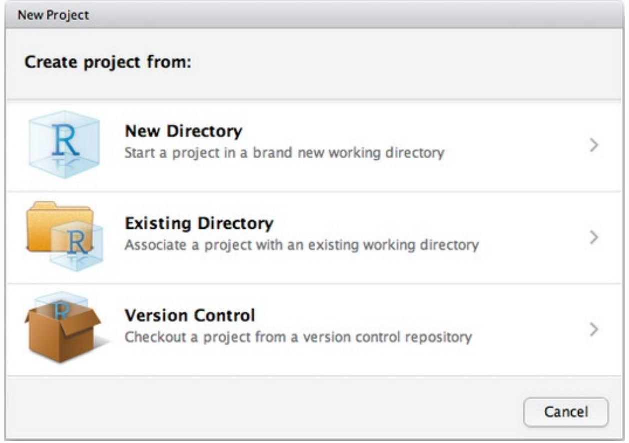
What I do

EXP_AAAAMMDD_NOMBRE_VX.X.csv

Avoid: ?, $, %, ^, &, *, (,),-,#, ?,,,<,>, /, |, \, [,], {, and };

RStudio

A helper in our quest



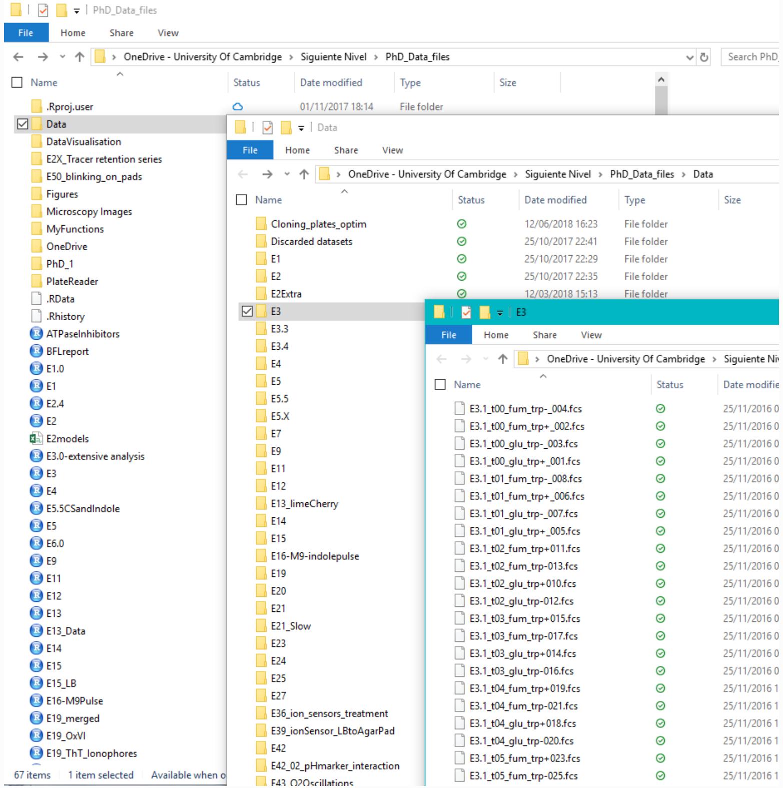
New Project

Create project from:

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



PhD_Data_files
File
Home
Share
View
OneDrive - University Of Cambridge › Siguiente Nivel › PhD_Data_files
Search PhD

Name	Status	Date modified	Type	Size
.Rproj.user	☁	01/11/2017 18:14	File folder	
☑ Data				
DataVisualisation				
E2X_Tracer retention series				
E50_blinking_on_pads				
Figures				
Microscopy Images				
MyFunctions				
OneDrive				
PhD_1				
PlateReader				
.RData				
.Rhistory				
ATPaseInhibitors				
BFLreport				
E1.0				
E1				
E2.4				
E2				
E2models				
E3.0-extensive analysis				
E3				
E4				
E5.5CSandIndole				
E5				
E6.0				
E9				
E11				
E12				
E13				
E13_Data				
E14				
E15				
E15_LB				
E16-M9Pulse				
E19_merged				
E19_OxVI				
E19_ThT_Ionophores				

67 items | 1 item selected | Available when o

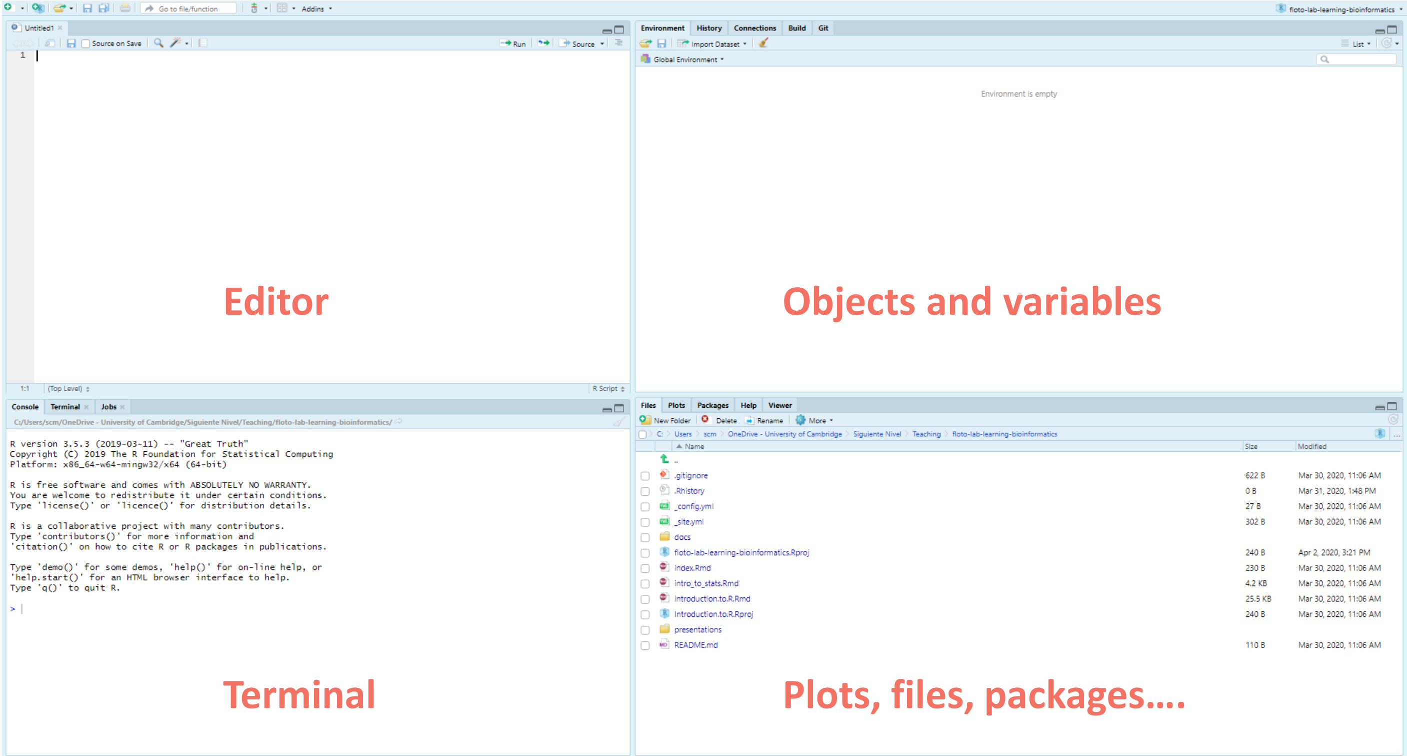
Data
File
Home
Share
View
OneDrive - University Of Cambridge › Siguiente Nivel › PhD_Data_files › Data

Name	Status	Date modified	Type	Size
Cloning_plates_optim	✓	12/06/2018 16:23	File folder	
Discarded datasets	✓	25/10/2017 22:41	File folder	
E1	✓	25/10/2017 22:29	File folder	
E2	✓	25/10/2017 22:35	File folder	
E2Extra	✓	12/03/2018 15:13	File folder	
☑ E3				
E3.3				
E3.4				
E4				
E5				
E5.5				
E5.X				
E7				
E9				
E11				
E12				
E13_limeCherry				
E14				
E15				
E16-M9-indolepulse				
E19				
E20				
E21				
E21_Slow				
E23				
E24				
E25				
E27				
E36_ion_sensors_treatment				
E39_ionSensor_LBtoAgarPad				
E42				
E42_02_pHmarker_interaction				
E43_O2Oscillations				

Name	Status	Date modifie
E3.1_t00_fum_trp-_004.fcs	✓	25/11/2016 0
E3.1_t00_fum_trp+_002.fcs	✓	25/11/2016 0
E3.1_t00_glu_trp-_003.fcs	✓	25/11/2016 0
E3.1_t00_glu_trp+_001.fcs	✓	25/11/2016 0
E3.1_t01_fum_trp-_008.fcs	✓	25/11/2016 0
E3.1_t01_fum_trp+_006.fcs	✓	25/11/2016 0
E3.1_t01_glu_trp-_007.fcs	✓	25/11/2016 0
E3.1_t01_glu_trp+_005.fcs	✓	25/11/2016 0
E3.1_t02_fum_trp+011.fcs	✓	25/11/2016 0
E3.1_t02_fum_trp-013.fcs	✓	25/11/2016 0
E3.1_t02_glu_trp+010.fcs	✓	25/11/2016 0
E3.1_t02_glu_trp-012.fcs	✓	25/11/2016 0
E3.1_t03_fum_trp+015.fcs	✓	25/11/2016 0
E3.1_t03_fum_trp-017.fcs	✓	25/11/2016 0
E3.1_t03_glu_trp+014.fcs	✓	25/11/2016 0
E3.1_t03_glu_trp-016.fcs	✓	25/11/2016 0
E3.1_t04_fum_trp+019.fcs	✓	25/11/2016 1
E3.1_t04_fum_trp-021.fcs	✓	25/11/2016 1
E3.1_t04_glu_trp+018.fcs	✓	25/11/2016 1
E3.1_t04_glu_trp-020.fcs	✓	25/11/2016 1
E3.1_t05_fum_trp+023.fcs	✓	25/11/2016 1
E3.1_t05_fum_trp-025.fcs	✓	25/11/2016 1

Inside R

RStudio



Editor

Objects and variables

C:/Users/scm/OneDrive - University of Cambridge/Siguiente Nivel/Teaching/floto-lab-learning-bioinformatics/

```
R version 3.5.3 (2019-03-11) -- "Great Truth"
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.

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.

> |
```

Terminal

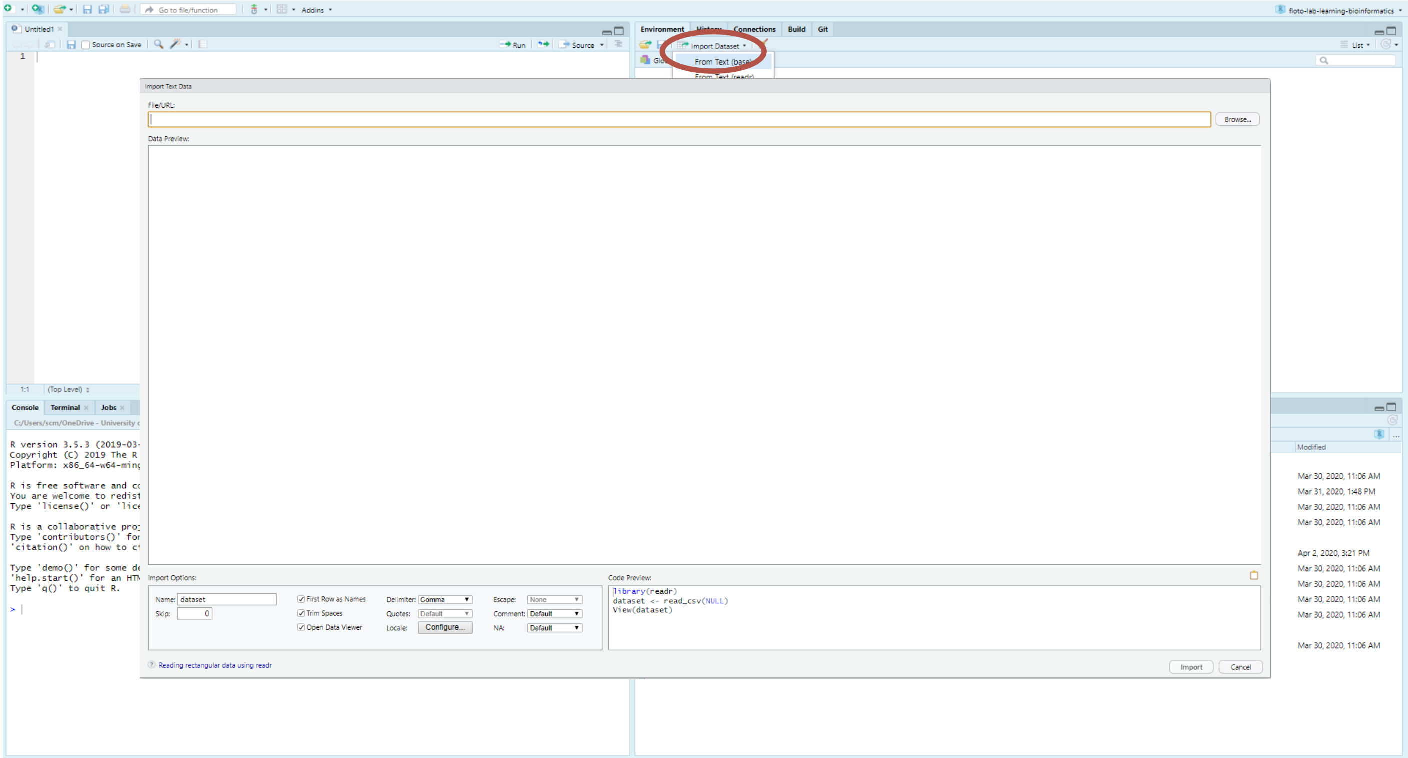
C: > Users > scm > OneDrive - University of Cambridge > Siguiente Nivel > Teaching > floto-lab-learning-bioinformatics

Name	Size	Modified
..		
.gitignore	622 B	Mar 30, 2020, 11:06 AM
.Rhistory	0 B	Mar 31, 2020, 1:48 PM
_config.yml	27 B	Mar 30, 2020, 11:06 AM
_site.yml	302 B	Mar 30, 2020, 11:06 AM
docs		
floto-lab-learning-bioinformatics.Rproj	240 B	Apr 2, 2020, 3:21 PM
index.Rmd	230 B	Mar 30, 2020, 11:06 AM
intro_to_stats.Rmd	4.2 KB	Mar 30, 2020, 11:06 AM
introduction.to.R.Rmd	25.5 KB	Mar 30, 2020, 11:06 AM
Introduction.to.R.Rproj	240 B	Mar 30, 2020, 11:06 AM
presentations		
README.md	110 B	Mar 30, 2020, 11:06 AM

Plots, files, packages….

Inside R

Importing data



```
R version 3.5.3 (2019-03-
Copyright (C) 2019 The R
Platform: x86_64-w64-ming

R is free software and co
You are welcome to redist
Type 'license()' or 'lice

R is a collaborative proj
Type 'contributors()' for
'citation()' on how to ci

Type 'demo()' for some de
'help.start()' for an HTM
Type 'q()' to quit R.

> 
```

Import Options:

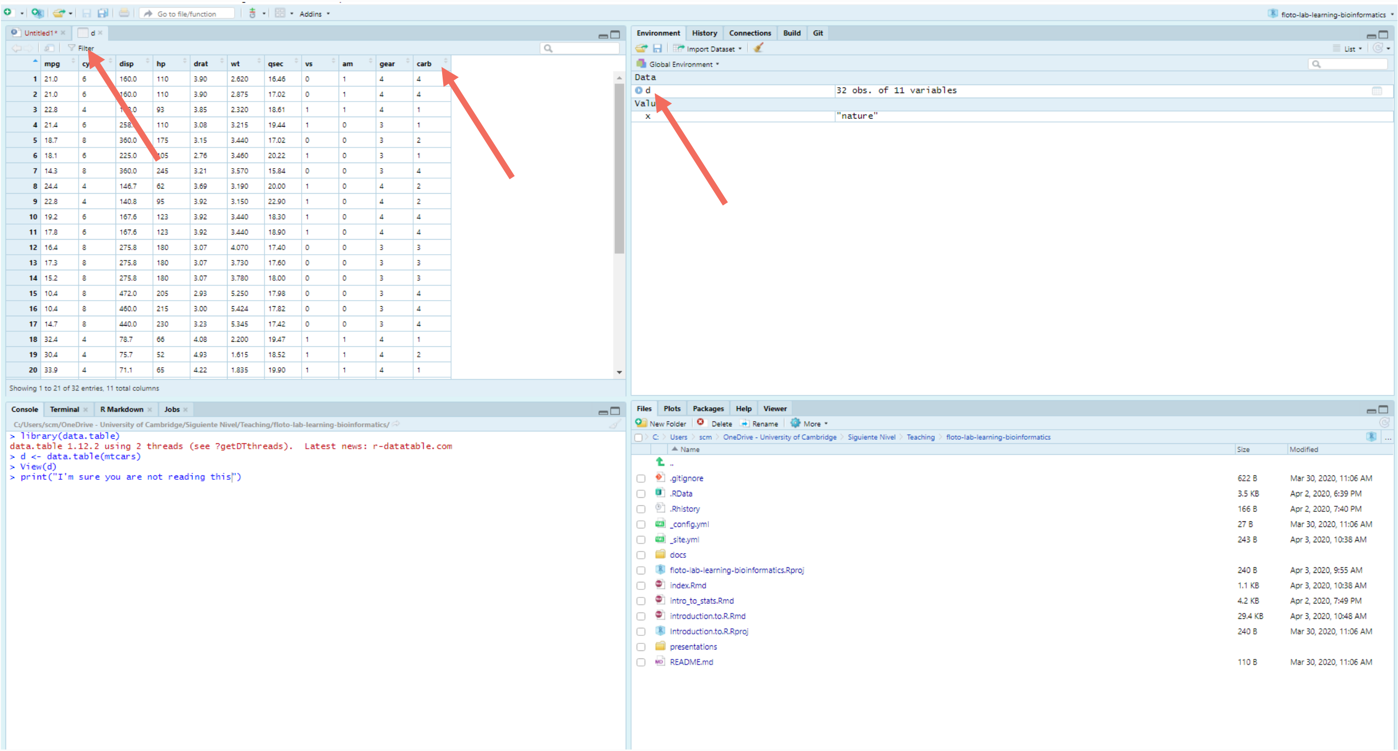
Name: dataset
Skip: 0
☑ First Row as Names
☑ Trim Spaces
☑ Open Data Viewer
Delimiter: Comma
Quotes: Default
Locale: Configure...
Escape: None
Comment: Default
NA: Default

Code Preview:

```
library(readr)
dataset <- read_csv(NULL)
View(dataset)
```

Inside R

Exploring datasets



	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
1	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
2	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
3	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
4	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
5	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
6	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
7	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
8	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
9	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
10	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
11	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
12	16.4	8	275.8	180	3.07	4.070	17.40	0	0	3	3
13	17.3	8	275.8	180	3.07	3.730	17.60	0	0	3	3
14	15.2	8	275.8	180	3.07	3.780	18.00	0	0	3	3
15	10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4
16	10.4	8	460.0	215	3.00	5.424	17.82	0	0	3	4
17	14.7	8	440.0	230	3.23	5.345	17.42	0	0	3	4
18	32.4	4	78.7	66	4.08	2.200	19.47	1	1	4	1
19	30.4	4	75.7	52	4.93	1.615	18.52	1	1	4	2
20	33.9	4	71.1	65	4.22	1.835	19.90	1	1	4	1

Showing 1 to 21 of 32 entries, 11 total columns

```
Data
d                     32 obs. of 11 variables
Valu
  x                   "nature"
```

C:/Users/scm/OneDrive - University of Cambridge/Siguiente Nivel/Teaching/floto-lab-learning-bioinformatics/

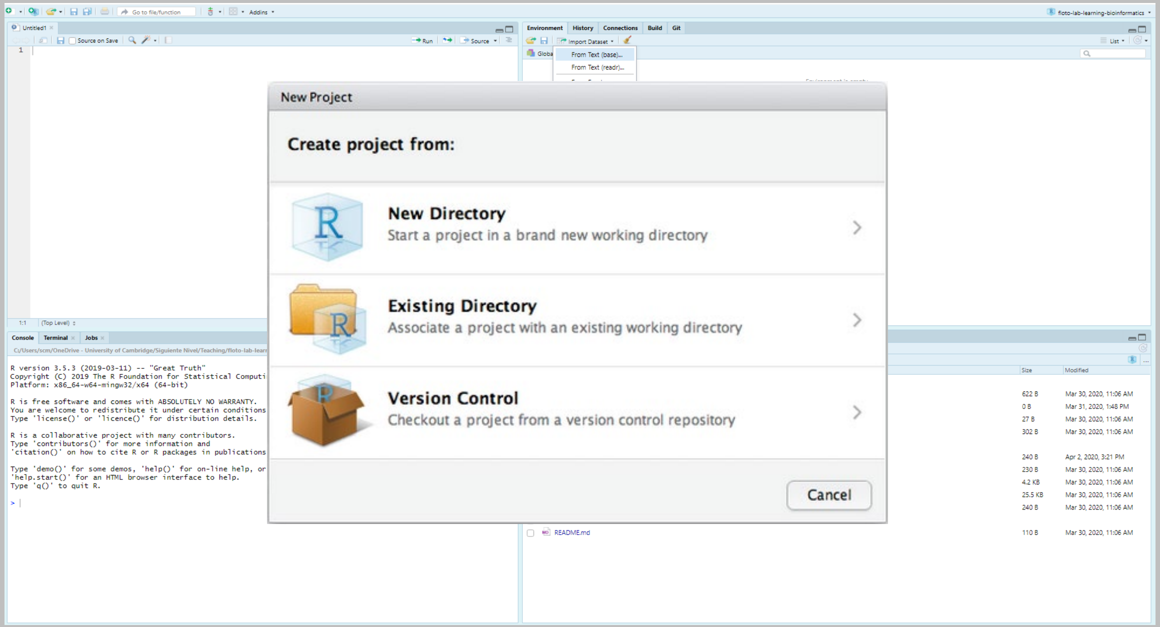
```
> library(data.table)
data.table 1.12.2 using 2 threads (see ?getDTthreads).  Latest news: r-datatable.com
> d <- data.table(mtcars)
> View(d)
> print("I'm sure you are not reading this")
```

C: > Users > scm > OneDrive - University of Cambridge > Siguiente Nivel > Teaching > floto-lab-learning-bioinformatics

Name	Size	Modified
..		
.gitignore	622 B	Mar 30, 2020, 11:06 AM
.RData	3.5 KB	Apr 2, 2020, 6:39 PM
.Rhistory	166 B	Apr 2, 2020, 7:40 PM
_config.yml	27 B	Mar 30, 2020, 11:06 AM
_site.yml	243 B	Apr 3, 2020, 10:38 AM
docs		
floto-lab-learning-bioinformatics.Rproj	240 B	Apr 3, 2020, 9:55 AM
index.Rmd	1.1 KB	Apr 3, 2020, 10:38 AM
intro_to_stats.Rmd	4.2 KB	Apr 2, 2020, 7:49 PM
introduction.to.R.Rmd	29.4 KB	Apr 3, 2020, 10:48 AM
Introduction.to.R.Rproj	240 B	Mar 30, 2020, 11:06 AM
presentations		
README.md	110 B	Mar 30, 2020, 11:06 AM

Time to program

For example



New Project

Create project from:

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

C:/Users/scm/OneDrive - University of Cambridge/Siguiente Nivel/Teaching/floto-lab-learn

```
R version 3.5.3 (2019-03-11) -- "Great Truth"
Copyright (C) 2019 The R Foundation for Statistical Computi
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.

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.

> 
```

	Size	Modified
	622 B	Mar 30, 2020, 11:06 AM
	0 B	Mar 31, 2020, 1:48 PM
	27 B	Mar 30, 2020, 11:06 AM
	302 B	Mar 30, 2020, 11:06 AM
	240 B	Apr 2, 2020, 3:21 PM
	230 B	Mar 30, 2020, 11:06 AM
	4.2 KB	Mar 30, 2020, 11:06 AM
	25.5 KB	Mar 30, 2020, 11:06 AM
	240 B	Mar 30, 2020, 11:06 AM
README.md	110 B	Mar 30, 2020, 11:06 AM

The foundation of R

The heir of mathematical notation

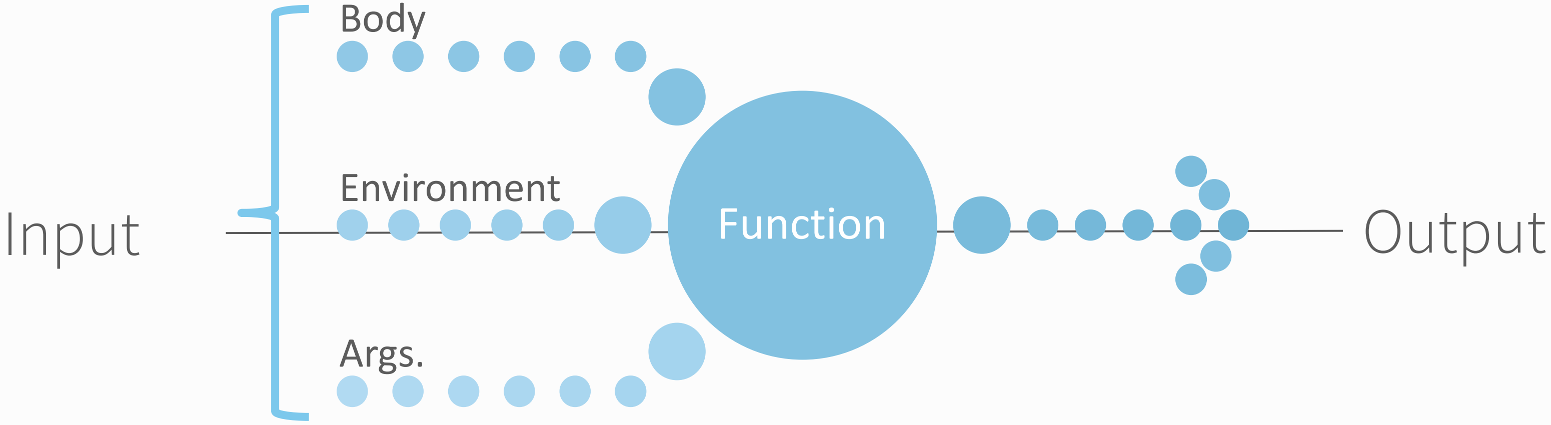
$$\text{Input} \xrightarrow{\quad f(x) \quad} \text{Output}$$

$$A = f(r) = \pi r^2$$

$$S = \frac{dA}{dr} = \frac{df(r)}{dr} = 2\pi r$$

The foundation of R

Functional programming



R like a super-calc on steroids

But… what's a function?

specify *"x"* and *"y"*

```
Action <- function(x, y = 0) {

        z <- x + y

  return(z)

}
```

Default "y" when we don't specify it

Add "x" e "y"

Return the result

The 3 elements

- **Args**: List of items, specified by **order** or **name**.
- **Body:** Transformations upon the arguments.
- **Env**: Variables accessible within the body.

R like a super-calc on steroids

But… what's a function?

```
# Intro to functions

runif(n = 10, min = 3, max = 10)    # This function generates 10 random numb between 3 and 10
runif(10, 3, 10)                    # Same but using argument position as reference

runif(n = 10)                       # Get 10 random numbers within the default range
runif(10)                           # 10 random numbers within the default range setting n by position

runif(min = -1, n = 3)              # Using the name allows the order to be alter

# Basic math functions
1 + 8

2 - 9

2 * 3

5/3
```

R like a super-calc on steroids

Logic operations

```r
x == y                  # Is x equal to y
identical(x, y)         # Same as == but using a function
!x                      # Negate (or logic inversión) of x
x & y                   # Is x AND y True?
x | y                   # Is x OR y True?
x < y                   # Is X Lower than y
x > y                   # Is X Bigger than y
x <= y                  # Is X Lower or equal than y
x >= y                  # Is X Bigger or equal than y
x != y                  # Is x different from y
xor()                   # Is only X or Y True?
isTrue()                # Is true?
```

R like a super-calc on steroids

In R you have a function for everything

```
A <- ...                            # Assigning a variable
c(object, object,...)               # Concatenate SAME type objects within a vector
length(object)                      # Number of elements in an object
str(object)                         # Show the internal structure of an object
class(object)                       # Show the class of an object
names(object)                       # Names of the elements
rm(object)                          # Remove the object
mean(vector)                        # Media
median(vector)                      # Median
sd(vector)                          # Standard deviation
sqrt(vector)                        # Square root
log(vector)                         # Natural logarithm
exp(vector)                         # E to the power of..

summary()                           # Summary
getwd()                             # Where we are working
read.delim()                        # Imporf files (also read.csv, read.csv2, read.txt, read.table)
```

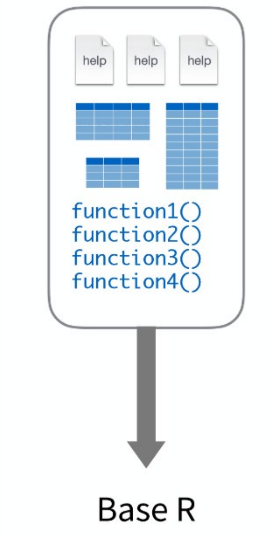
Time to program

For example

```r
mean(1:5, 0.1)
mean(x = 1:5, trim = 0.1)
mean(1:5, trim = 0.1)
mean(x = 1:5, 0.1)
```

The R packages

Families of functions comes in packages



[image taken from Ryan Wesslen presentation]

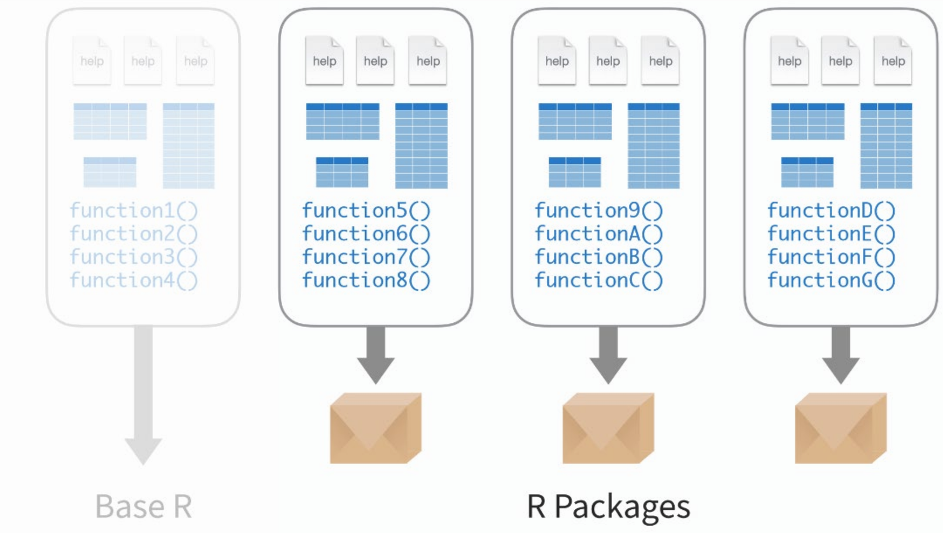
What they are

- The R community that develops related functions adds them in a library that other users can download and install.
- This allows us to start our work where others finished and focus our efforts on a single problem.
- They are usually accompanied by instruction books called " vignette " that explain how to use the functions of the library.

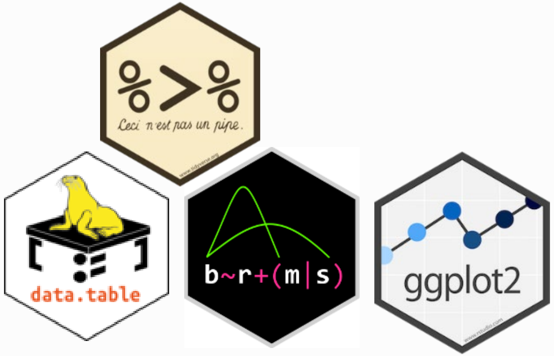
```
install.package(data.table)                      # Install library from CRAN
devtools::install_github("/paul-buerkner/brms") # Install from GitHub


library(tidyverse)                               # Load library
data.table::melt.data.table()                    # Calling a function from a
                                                 # library
```

The R packages

Families of functions comes in packages



[image taken from Ryan Wesslen presentation]



- In this workshop we will work with the following bookstores: data.table, magrittr, ggplot, glmmTMB, mgcv
- **Beware of using too many libraries:**
- Overlapping functions with the same name
- Reproducibility on other computers

First contact

Representing mathematical objects

Vectors

$$X = [x_1, x_2, x_3, x_i \dots \dots x_n]$$

Numeric: ℝ

- 0, -8, 3.14…

Logic: 1|0

- TRUE, FALSE, T, F

Character

- “Que”, “es”, “eso”, “Eso es Queso”

```
1 + 1                   # Hashtag to write down comments
Nombre_x <- valor       # Define objects with the operator <-
y <- 1
```

Principles for naming a variable

- Start with a character, no irregular caraters `(&, %, ^, …)`
- Brief and descriptive.
- For compound names connect with `_`

```
`<-`(x, 2)                # Assign
x <- y <- 1               # Assign multiple elements
y <- c(1, -2, 8, 5, 5e5)  # Manually define a vector
z <- c("A", "B", "C")     # Character vectors
x <- c(FALSE, TRUE, F, T) # Logic
x <- c(1L, 2L, 43L)       # Integer
```

First contact

Representing mathematical objects

Vectors

```
x + y
x * 3
x / y
x^3
1:5                                 # Dos puntos ":" para indicar secuencias
5:1
y <- c(1, -2, 8, 5, 5e5)
y[3]                                # Extraer elementos
[1] 8
y[-2]                               # Numeros negativos para excluir elementos
[1] 1  8  5  5e5
y[2:3]                              # Extraer secuencia de elementos

# Nombrar elementos para crear diccionarios
y <- c("a" = 1, "b" = -2, "c" = 8, "d" = 5, "e" = 5e5)
y[“d"]
[1] 5
y[y > 5]                            # Extraer elementos usando test lógicos ( también ==, >=, <= )
   c     e
8e+00 5e+05
```

First contact

Representing mathematical objects

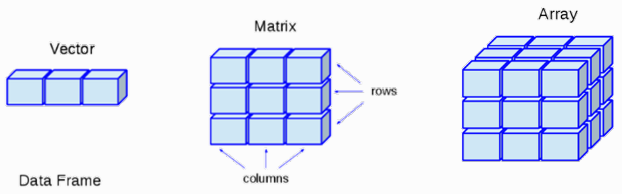
Recycling

```
# If two vectors have different length, the shorter one is recycled it will recycle to equal the
length of the longest with the longest vector.

x = c(10, 20, 30)
y = c(1, 2, 3, 4, 5, 6, 7, 8, 9)

> y + x
# R uses "y" as a 9-element vector and "x" will repeat it 3 times

[1] 11 22 33 14 25 36 17 28 39
```

First contact

Representing mathematical objects

Homogeneous structures



[source: University of Cantabria]

Principles for naming a variable

Within R, a matrix is a vector with two extra attributes:

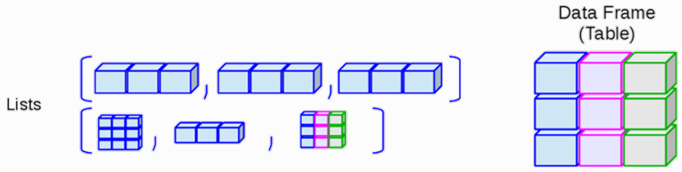
- Rows
- Columns

```
mat <- matrix(c(1, -2, 8, 5, 7, 0, 3, 6, 9),
                   nrow = 3, ncol = 3)

mat[4]                          # Use array as vector
[1] 5

mat[1, 2]                       # mat[fila, columna]
[1] 5

mat[ ,c(1, 3)]
     [,1] [,2]
[1,]   1    3
[2,]  -2    6
[3,]   8    9

A <- array(data = 1:27, dim = c(3, 3, 3))
A[ 2, 2, 3]
[1] 23
```

First contact

Representing mathematical objects

Heterogeneous structures



[source: University of Cantabria]

```
mat <- matrix(c(1, -2, 8, 5), nrow = 2, ncol = 2)
mat[1, 2]                            # mat[fila, columna]
[1] 8
Animals <- list("a" = c(1, 4, 5, 2),
              "b" = c("Cow", "Pig", “Chicken"))

Animals[["a"]][1]          # [[ to access each item
Animals$a[1]               # $ to access every element
```

Lists and tables

- It contains structures of different types, or even contain a list
- They can also be of different length

Time to program

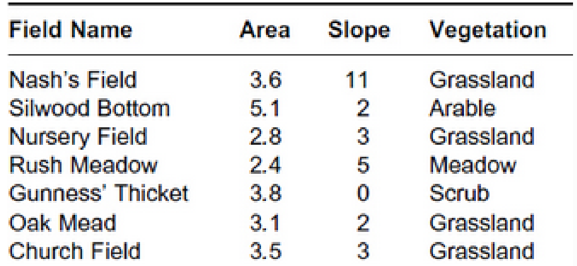
For example

```
x + y
x * 3
x / y
x^3
1:5                                 # Use colon ":" to indicate sequences
5:1
y <- c(1, -2, 8, 5, 5e5)
y[3]                                # Extract items
[1] 8
y[-2]                               # Negative numbers to delete elements
[1] 1  8  5  5e5
y[2:3]                              # Extract sequence of elements
# Name items to create “dictionaries”
y <- c("a" = 1, "b" = -2, "c" = 8, "d" = 5, "e" = 5e5)
y[“d"]
[1] 5
y[y > 5]                            # Extract elements using logical tests ( también ==, >=, <= )
   c     e
8e+00 5e+05
```

Tables

Data.fame

- Intuitively, `data.frame` is the classic way to represent information in our minds.
- Like excel sheet
- Each column represents a vector
- Each row a case



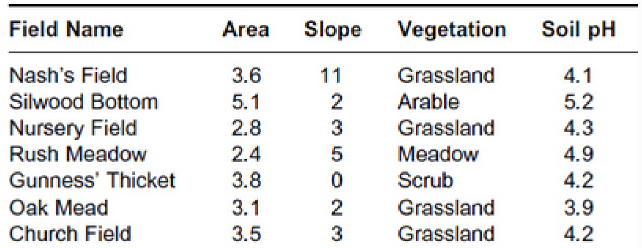
Field Name	Area	Slope	Vegetation
Nash's Field	3.6	11	Grassland
Silwood Bottom	5.1	2	Arable
Nursery Field	2.8	3	Grassland
Rush Meadow	2.4	5	Meadow
Gunness' Thicket	3.8	0	Scrub
Oak Mead	3.1	2	Grassland
Church Field	3.5	3	Grassland

```
table <- data.frame(Field_N = c(“Nash`s", “Silwood", “Nursery",
                   “Rush", “Gunness", “Oak Mead", “Church Field"),
                     Area = c(3.6, 5.1, 2.8, 2.4, 3.8, 3.1, 3.5),
                     Slope = c(11, 2, 3, 5, 0, 2 , 3),
                     Vegetation = c(“Grass", “Arabl", “Grass",
                    “Meadow", “Scrub", “Grass", “Grass"))

# Create by adding pre-existing vectors

table <- data.frame(Field_N, Area, Slope, Vegetation)

# Join tables

table <- cbind(Field_N, Area, Slope, Vegetation)
table <- rbind(table_1, table_2)

# Also via read.*

table <- read.csv(«path/to/my/file")
```

Tables

Data.fame

- Intuitively, `data.frame` is the classic way to represent information in our minds.
- Like excel sheet
- Each column represents a vector
- Each row a case



Field Name	Area	Slope	Vegetation	Soil pH
Nash's Field	3.6	11	Grassland	4.1
Silwood Bottom	5.1	2	Arable	5.2
Nursery Field	2.8	3	Grassland	4.3
Rush Meadow	2.4	5	Meadow	4.9
Gunness' Thicket	3.8	0	Scrub	4.2
Oak Mead	3.1	2	Grassland	3.9
Church Field	3.5	3	Grassland	4.2

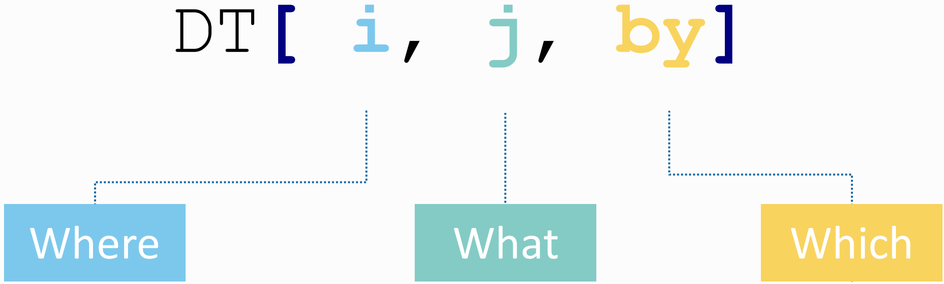
```
head(table)                # Exploring head/tail values
tail(table)
str(table)

table$Area                 # Expose values
mean(table$Slope)

summary(table)

table[1, 2]                # Access values as matrix
table[, 2:3]
table[, "nombre"]          # Access values via naming
```

Data.table

An expansion to data.frame

Properties

- Syntax and usage is similar to **`data.frame`**
- It's extremely fast
- It offers tools for:
- Data aggregate
  - Update cells
  - Join tables
- Allows elegant notation
- **No dependencies**



`DT[ i, j, by]`

Where

What

Which

Data.table

Continuation

```
matrix(data.table)
d <- data.table(mtcars, keep.rownames = TRUE)

 >                   rn  mpg cyl  disp  hp drat    wt  qsec vs am gear carb
 1:           Mazda RX4 21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
 2:       Mazda RX4 Wag 21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
 3:          Datsun 710 22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
 4:      Hornet 4 Drive 21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
 5:   Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
 6:             Valiant 18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
 7:          Duster 360 14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4

# How to perform simple extraction operations?

d[cyl > 4]                                       # Filter with a simple logical test

d[rn %in% c("Mazda RX4", "Hornet Sportabout")]   # Elementros within a vector

d[rn %like% "Mazda"]                             # Extract similar elements
```

Data.table

Continuation

```
# Summarise data

d[, .(g_mean = mean(cyl)),          # By group
            by = gear]

d[, .(g_mean2 = mean(cyl)),
            by = .(gear, vs)]       # Multiple groups

d[, Ref_col_mean := mean(hp),       # Multiple-groups and create a new column
            by = .(gear, vs)]

d[, .(“mean_hp", “mean_cyl"):=       # Multiple-groups and multiple columns
            .(mean(hp),
              mean(cyl),
            by = .(gear, vs)]

d[, mean(wt[vs == 0])/mean(wt[vs == 1])] # Vector within a column
```

Time to program

For example

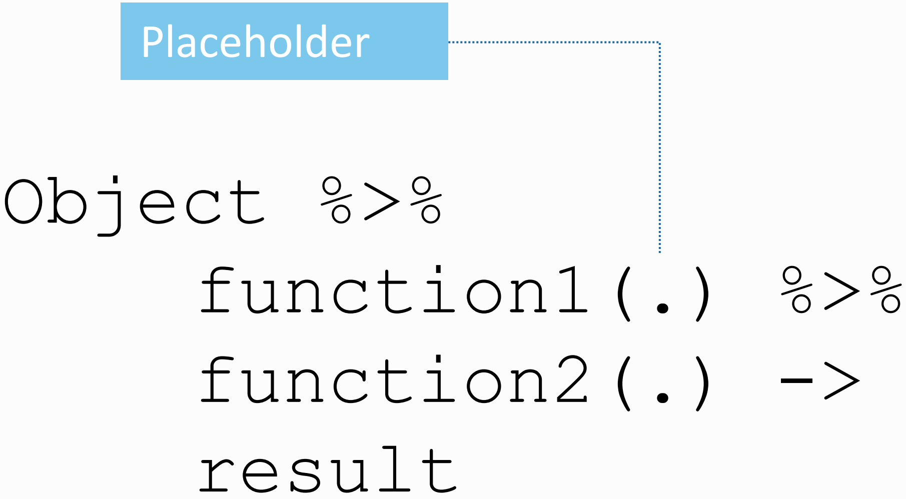
```
# Summarise data

d <- data.table(mtcars)
d[, .(m = mean(cyl)),                        # By group
            by = gear]

d[, .(m = mean(cyl)),
            by = .(gear, vs)]                # Multiple groups

d[, m_hp := mean(hp),                        # Multiple-groups and create a new column
            by = .(gear, vs)]

d[, c(“m_hp", “m_cyl"):=                     # Multiple-groups and multiple columns
            .(mean(hp),
              mean(cyl)),
            by = .(gear, vs)]

d[, mean(wt[vs == 0])/mean(wt[vs == 1])] # Vector within a column
```

Pipes

Moving A %>% B

The %>% operator

- One of the most useful and powerful elements of R.
- The operator ¨%>% helps structure the code and minimizes the creation of "transitional variables.
- Requires package `magrittr` o `tidyverse.`
- The basic idea:
  - `x %>% f` is `f(x)`
  - `x %>% f %>% g %>% h` is `h(g(f(x)))`



```
Object %>%
          function1(.) %>%
          function2(.) ->
          result
```

Pipes

Moving A %>% B

The %>% operator

- One of the most useful and powerful elements of R.
- The operator ¨%>% helps structure the code and minimizes the creation of "transitional variables.
- Requires package `magrittr` o `tidyverse.`
- The basic idea:
  - `x %>% f` is `f(x)`
  - `x %>% f %>% g %>% h` is `h(g(f(x)))`

Example: Representing averages by group

```r
# Extract values
Subset <- table[ table$a > "Value_x"]
# Sumity the information
mus <- aggregate(test ~ condition, Subset, FUN = mean)
stds <- aggregate(test ~ condition, Subset, FUN = sd)
summary_Ss <- merge(mus, stds, by = "condition")

 # Create the graph
colnames(summary_Ss)<- c("condition","mus","stds")
ggplot(summary_Ss, aes(x = condition, y = mus)) +
       geom_point() +
       geom_errorbar(aes(ymin = mus - stds,
                         ymax = mus +   stds)) +
       theme_bw()
```

Stack of lazy variables

Pipes

Moving A %>% B

The %>% operator

- One of the most useful and powerful elements of R.
- The operator ¨%>% helps structure the code and minimizes the creation of "transitional variables.
- Requires package `magrittr` o `tidyverse.`
- The basic idea:
  - `x %>% f` is `f(x)`
  - `x %>% f %>% g %>% h` is `h(g(f(x)))`

Example: Representing averages by group **Code Unreadable**

```
summary_Ss <- merge(aggregate(test ~ condition,
                              table[, table$a > "valor_x"],
                        FUN = mean),
                  aggregate(test ~ condition,
                            table[, table$a > "valor_x"],
                        FUN = sd), by = "condition")
 # Create the graph
colnames(summary_Ss)<- c("condition","mus","stds")
ggplot(summary_Ss, aes(x = condition, y = mus)) +
       geom_point() +
       geom_errorbar(aes(ymin = mus - stds,
                              ymax = mus +    stds)) +
       theme_bw()
```

Pipes

Moving A %>% B

The %>% operator

- One of the most useful and powerful elements of R.
- The operator ¨%>% helps structure the code and minimizes the creation of "transitional variables.
- Requires package `magrittr` o `tidyverse.`
- The basic idea:
  - `x %>% f` is `f(x)`
  - `x %>% f %>% g %>% h` is `h(g(f(x)))`

Example: Representing averages by group

```
table[a > “value_x" ] %>% # filter
    .[, .(mus = mean(test),
          stds = sd(test),
       by = condition] %>% # Calculate parameters

# Graphical representation
ggplot(., aes(x = condition, y = mus) +
       geom_point()
       geom_errorbar(aes(ymin = mus - stds,
                         ymax = mus + stds) +
       theme_bw()
```

Time to program

For example

```
install.packages(magrittr)
install.packages(ggplot2)
library(ggplot2)
library(magrittr)
library(data.table)
data("mtcars") # Load data
d <- data.table(mtcars, keep.rownames = T)

d[ cyl > 3] %>%
            ggplot(., aes(x = disp, y = wt)) +
            geom_point() +
            theme_bw()

d %>%
  lm(formula = disp ~ wt, data = .) %>%
  summary
```

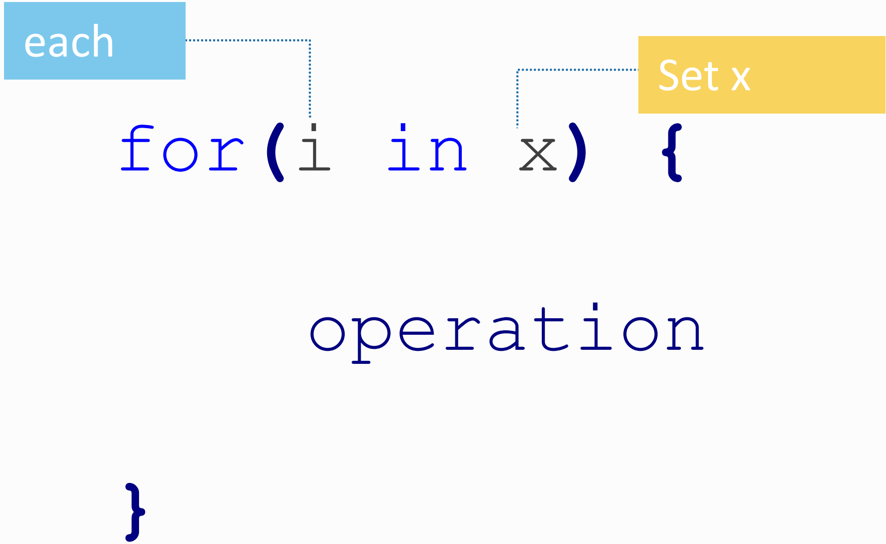
Loops and functionals

The essence of programming

What are they?

Computers are especially useful when the task requires repetition

- R provides us with three basic tools to repeat:
  - `For`
  - The of `*apply` family
- In addition, it is possible to filter with logical tests:
  - `if`
  - `ifelse`



each

Set x

```
for(i in x) {

        operation

}
```

Loops and functionals

An example

```
# Generate a sample dataset

set.seed(2018)

d <- data.frame(replicate(6, sample(c(1:10, -99), 100, rep = TRUE)))
names(d) <- letters[1:6]
head(d)
 a   b   c  d   e  f
1 4   7 -99  9   9  2
2 6   2   8 10   6 10
3 1 -99   9  3   4  1
4 3   7   7  7 -99  6

How do I calculate the mean of each column?
mean(d$a)
mean(d$b)
mean(d$b)
mean(d$d)
mean(d$e)
```

Loops and functionals

An example

```
# Looping with for

for (i in 1:ncol(d)) {                  # Using ":" to generate sequence of 1 to the number of columns

            x <- mean(d[[i]])
            print(x)
            }

# If we want to save the result, first we create an empty vector (or other format)

medias <- rep(NA, ncol(d))

for (i in seq_along(medias)) {

            x <- mean(d[[i]])           # We iterate by column
            medias[i] <- x              # Save the result in the vector "means", position "i"

}
```

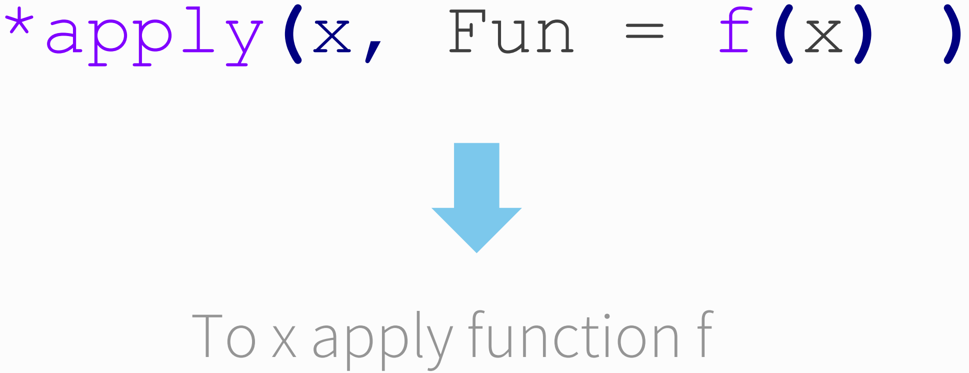
Loops and functionals

The essence of R

What are they?

Computers are especially useful when the task requires repetition

- R provides us with three basic tools to repeat:
  - `For`
  - The of `*apply` family
- In addition, it is possible to filter with logical tests:
  - `if`
  - `ifelse`



```
*apply(x, Fun = f(x) )
```

To x apply function f

Loops and functionals

An example

```
# Iteration-*apply

medias <- apply(X = d, MARGIN = 2, FUN = mean)

> medias
    a     b     c     d     e    f
-4.26 -4.75 -1.85 -5.92 -4.43 1.40

# If we also want to specify other arguments, we can indicate them at the end

apply(X = d, MARGIN = 2, FUN = quantile, probs = c(0.1, 0.5, 0.9))

>   a  b c   d  e  f
10% 1 -9 1 -99 -9  1
50% 4  5 5   5  6  5
90% 9  9 9   9 10 10
```

Loops and functionals

Basic programming

What are they?

Computers are especially useful when the task requires repetition

- R provides us with three basic tools to repeat:
  - `For`
  - The of `*apply` family
- In addition, it is possible to filter with logical tests:
  - `if`
  - `ifelse`

```
ifelse(test = ***,
       yes = Accion_A,
       no = Accion_B)
```

Loops and functionals

An example

```
# Basic if loop
if (paper_acepted == TRUE) {          # Notice, it only accepts 1 element at a time
             print( “We are the best!")

            }
# Nested if loop
if (publisher == “science”) {
  print (“We are the best”)
} else if ( publisher == “arXiV” ){     # else if allow us to make another question
  print (“Andres will kill you” )
} else {                                # Only else resolve all the remaining cases
  print (“If you look for me, I am in the crying room”)
}
# Iterate throught if-test throught a vector

all_papers <- c(“accepted”, “accepted”, “accepted”, “rejected”)

ifelse(all_papers == “accepted”,
            print(“Fantastic, it’s a science?”),
            print(“If you look for me, I am in the crying room”)
            )
```

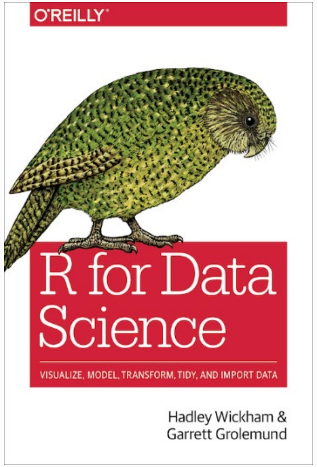
Loops and functionals

An example

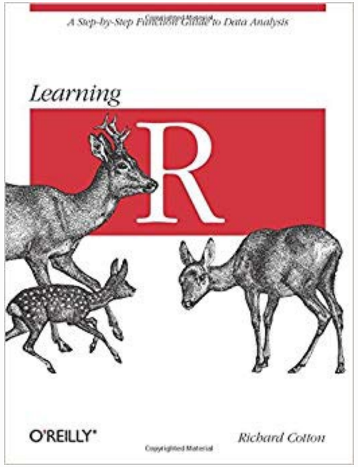
```
Looping with *apply
d <- apply(d, 2, function(x) {ifelse(x == -99, NA, x)})
medias <- apply(X = d, MARGIN = 2, FUN = mean)

# Looping with *apply and the pipe operator

medias <- apply(d, 2, function(x) {tmp <- ifelse(x == -99, NA, x)}) %>%
            apply(X = ., MARGIN = 2, FUN = mean, na.rm = TRUE)
medias

# looping with*apply

medias <- apply(d, 2, function(x) {tmp <- ifelse(x == -99, NA, x)
                                   mean(tmp, na.rm = TRUE)})
> medias
      X1       X2       X3       X4       X5       X6
5.585106 5.369565 5.263736 5.866667 5.423913 5.217391
```

Support channels

Online resources


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

R for Data Science,
H. Wickham &
G. Grolemund



Learning R,
R. Cotton

R in a nutshell,
J. Adler

Aprender a
programar en R
P. García Montero

¡Gracias por vuestro tiempo!

¿Preguntas?