

R para análisis científicos reproducibles

elige tu propio camino

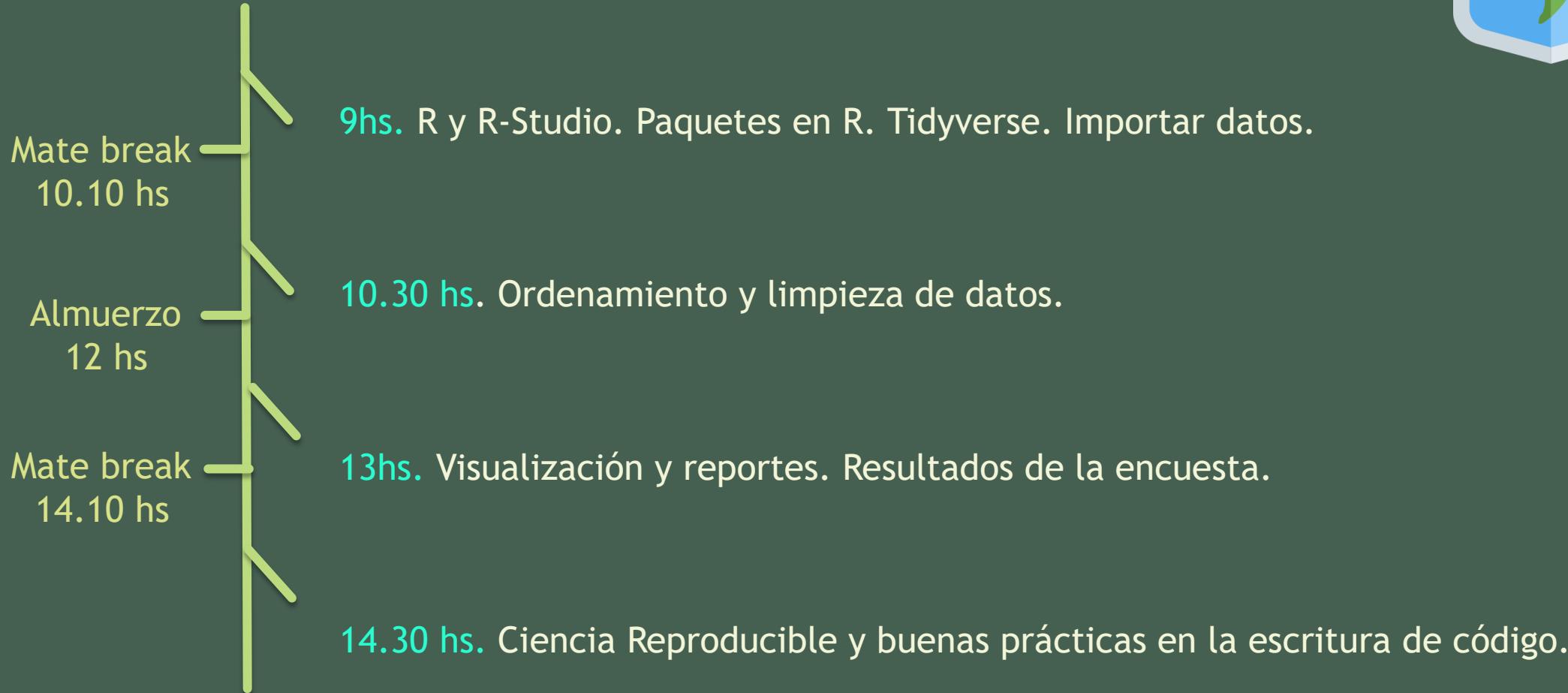
Mg. Yanina Bellini Saibene - INTA Anguil

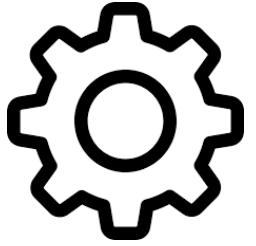
Dra. María Florencia D'Andrea - IRB - CNIA



Instituto Nacional
de Tecnología Agropecuaria

Mapa de ruta





Configuración



Modo: on

- No comer ni tomar mate en la sala
- Evitar descartar los vasos de telgopor

El material del curso estará disponible en
https://flor14.github.io/r_inta/

- ▶ Estas filminas se encuentran bajo licencia [Creative Commons Attribution-ShareAlike 4.0 International License](#)

- ▶ Para leer sobre esta licencia:
https://creativecommons.org/licenses/by-sa/4.0/deed.es_ES

Obteniendo datos

Campo

Obtención de datos a campo – Planillas

Ej: Planilla con valores de presencia – ausencia de especies



Laboratorio

Procesamiento de muestras de experimentos – Mediciones

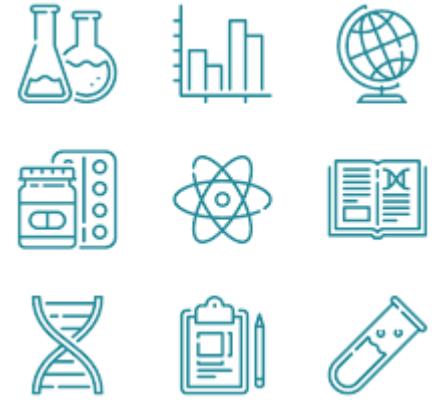
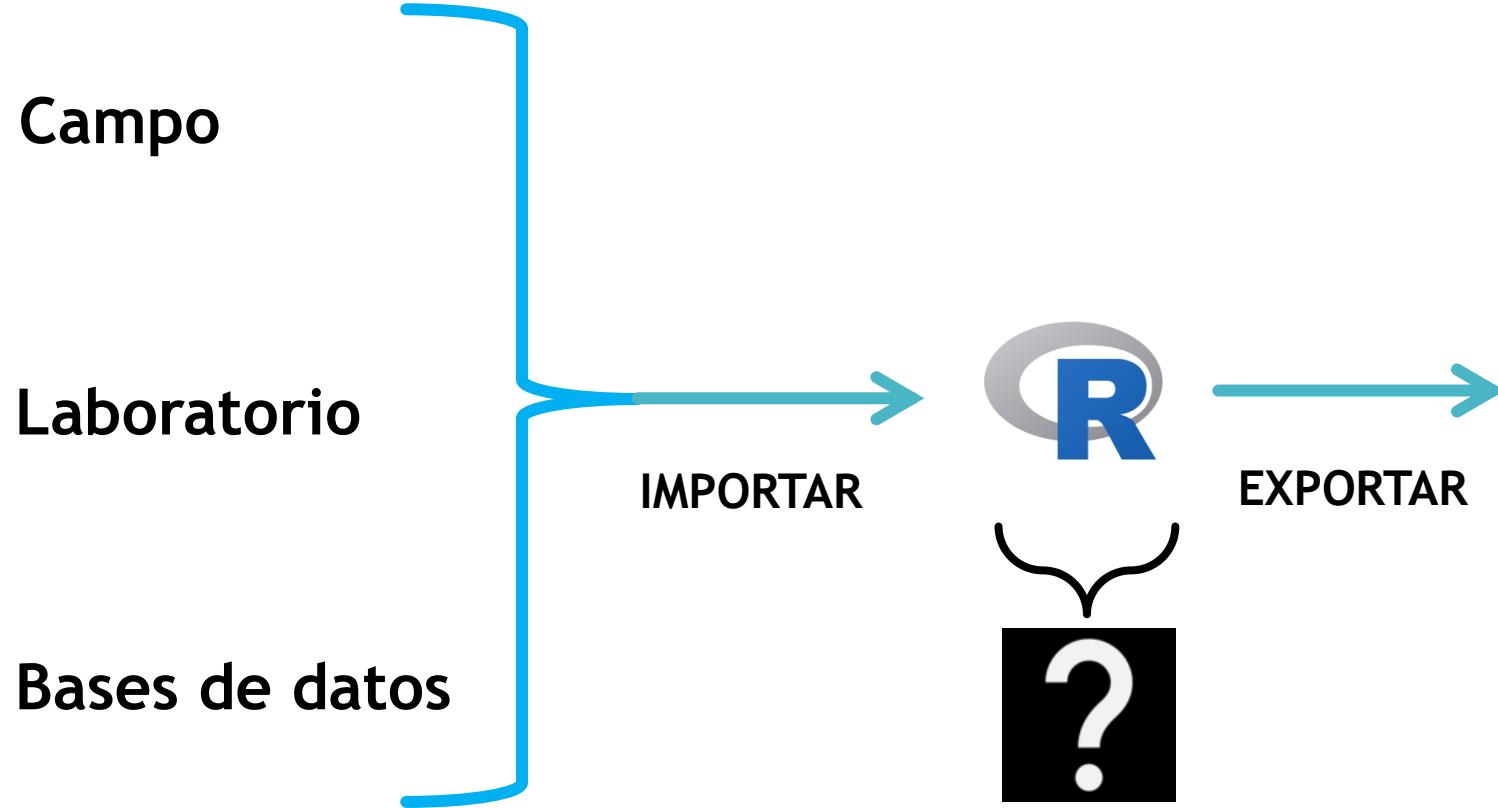
Ej: Planilla con valores de mediciones enzimáticas.

Bases de datos

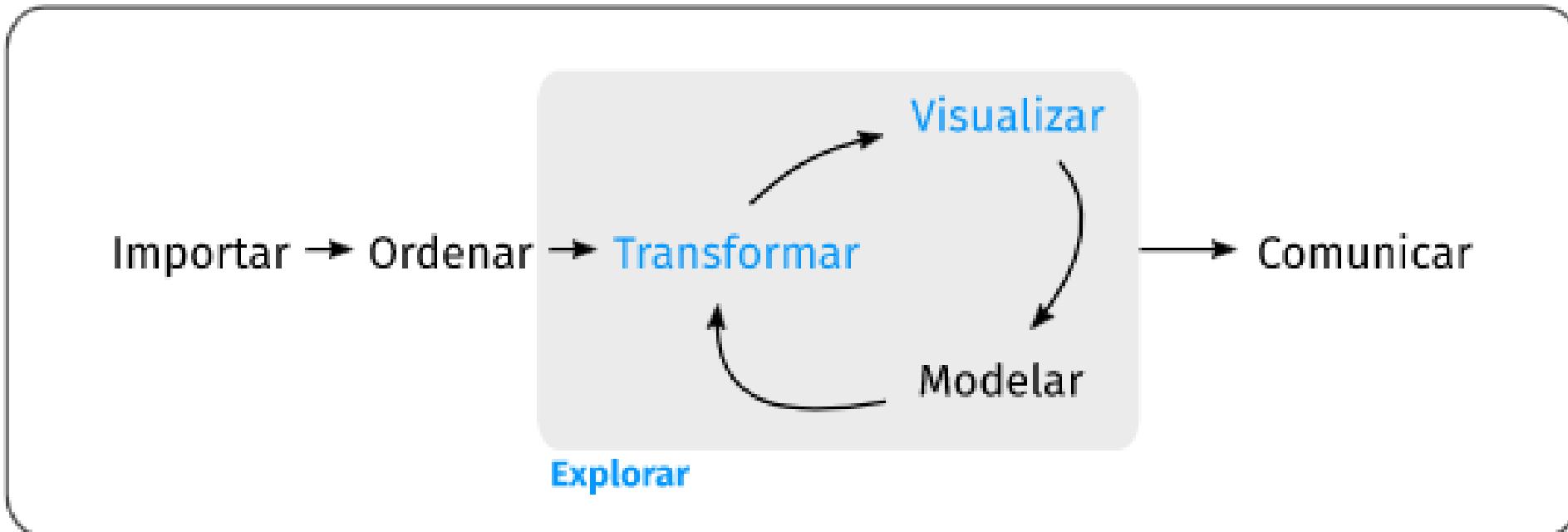
Trabajo sobre datos ya obtenidos

Ej: Datos de estaciones meteorológicas.

Procesamiento de datos

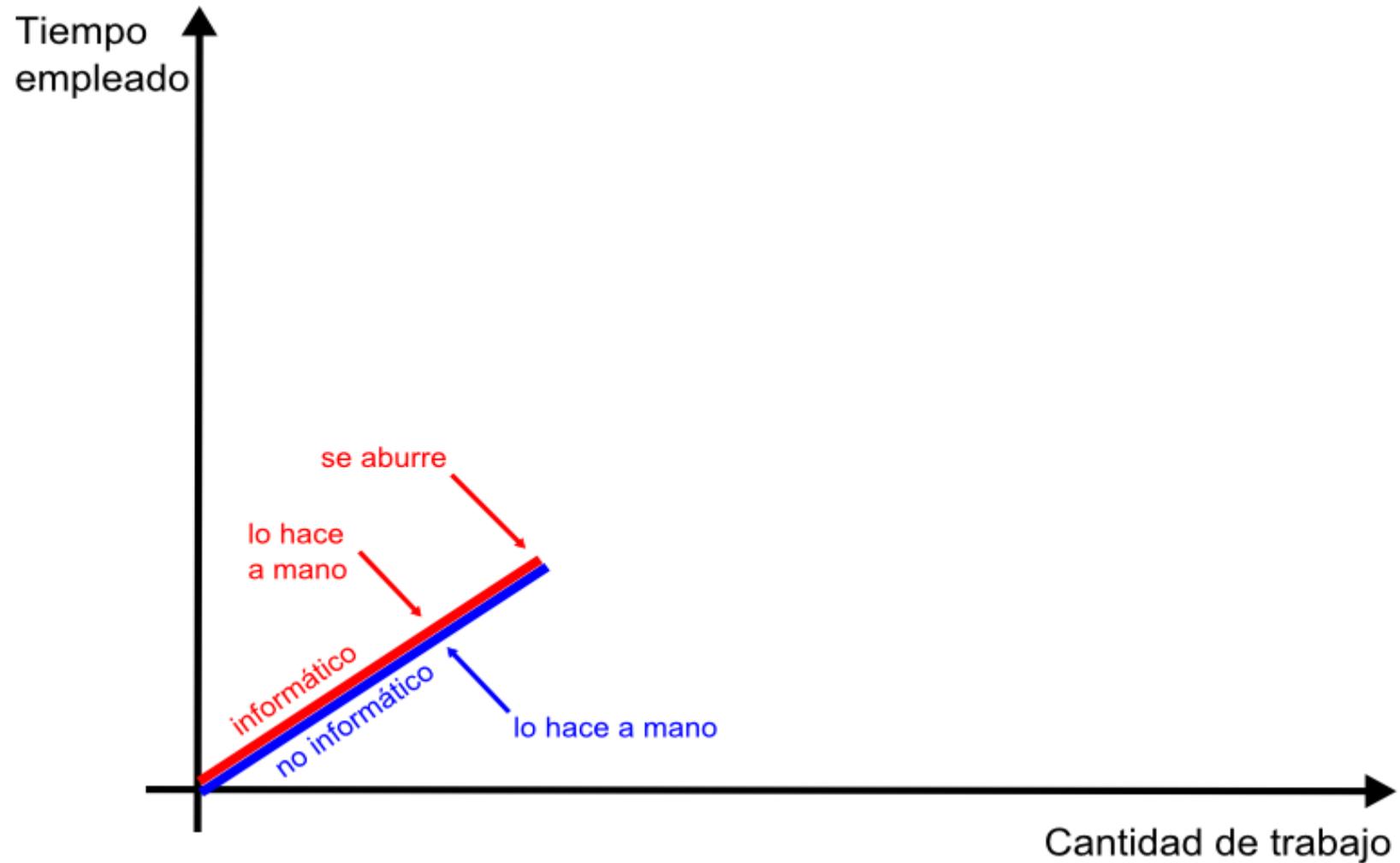


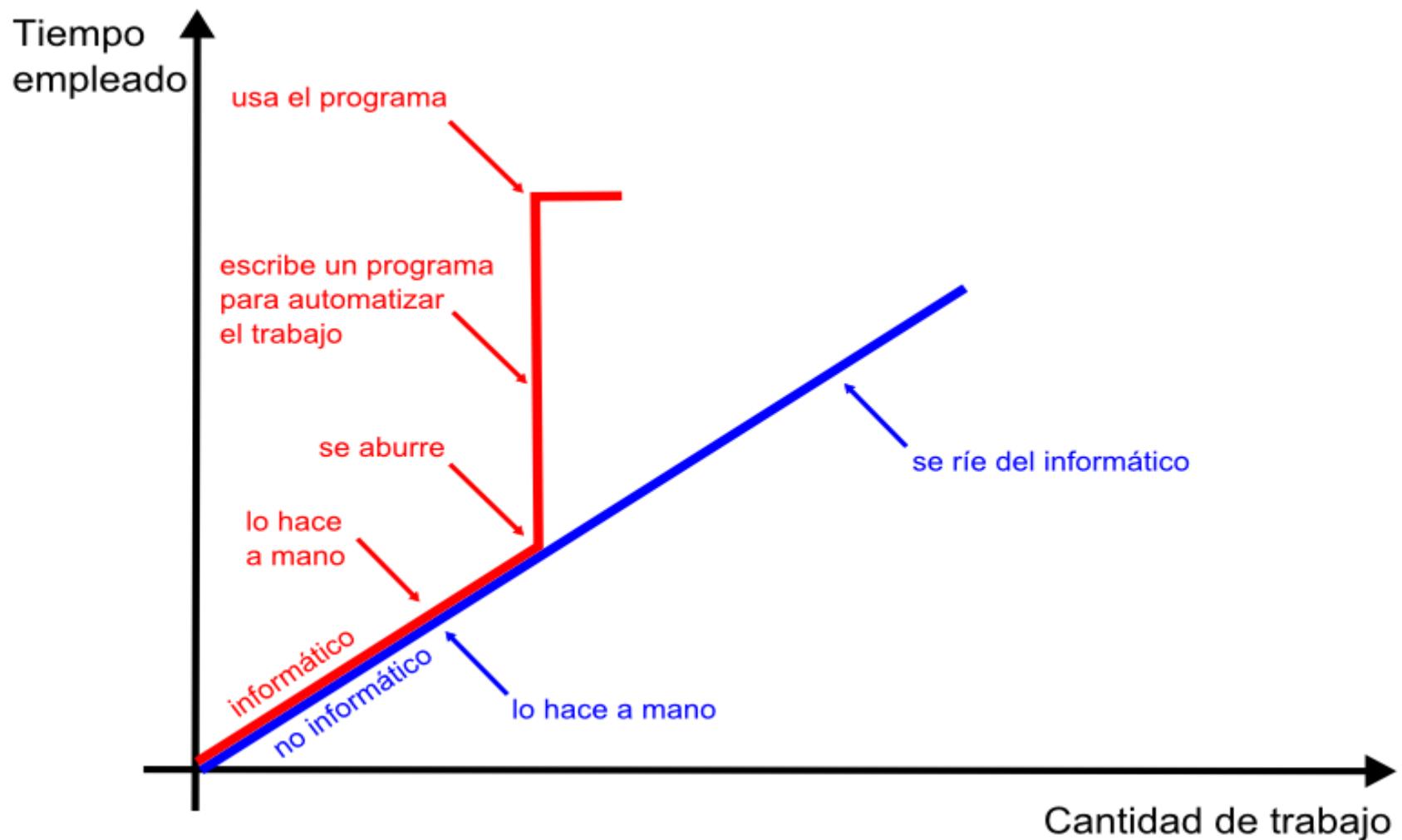
Ciencia de datos



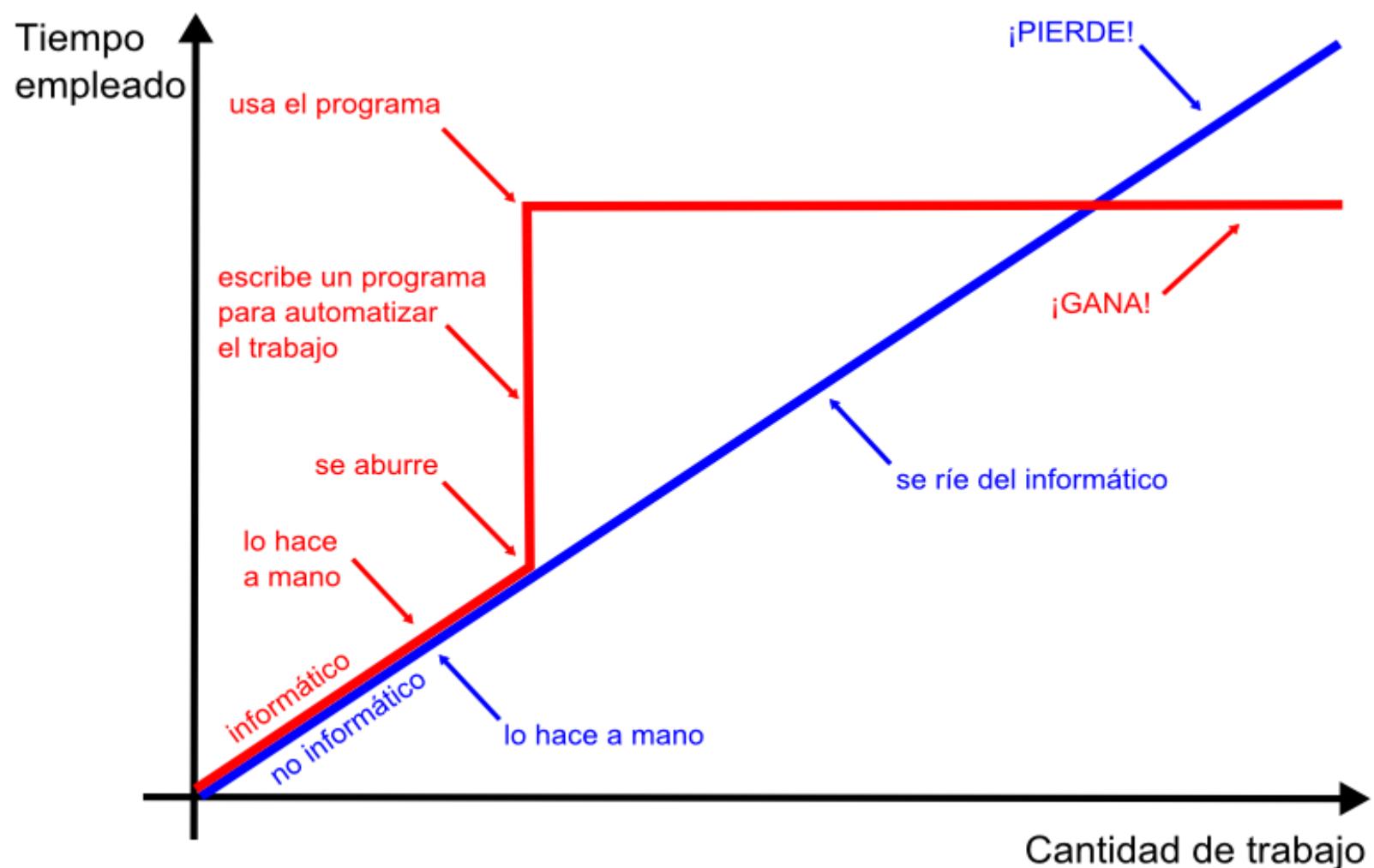
¿Por qué programar?

Cuando un trabajo se hace a manualmente, el tiempo empleado para realizar un trabajo suele ser directamente proporcional a la cantidad de trabajo





Al final, el informático puede hacer más trabajo sin necesidad de invertir más tiempo, mientras que el no informático descubre que no puede competir.



La programación nos permite
automatizar tareas



Tim Schoof
@drtimschoof

⌄

When you've written the same code 3 times, write a function.

When you've given the same in-person advice 3 times, write a blog post.

When you've written the same function 3 times, make a package.

Great advice from [@drob](#) & [@robinson_es](#)

What are you planning to do next?

Nos permite manejar **bases de datos más grandes**

Los paquetes permiten explorar **varias funcionalidades** de forma relativamente sencilla

Múltiples formas de **comunicar resultados**:
con R podes crear reportes en distintos formatos, paginas web,
mapas, gráficos, libros, posters, presentaciones, etc..

Es software libre

R lingua franca de la estadística

R es un lenguaje y entorno para computación estadística y gráficos.

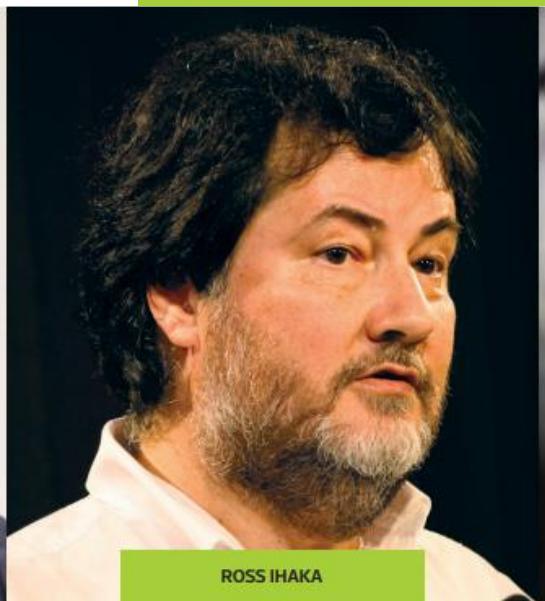


R cumplió 25 años

1993 <- Nace

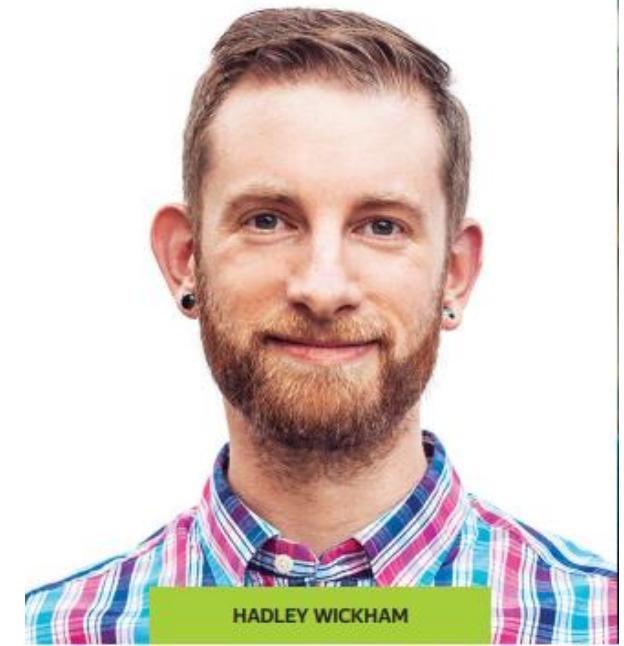
2000 <- Se libera la versión 1.0.0,
la primera de uso público

>y<- 25
>y R_{generation}
[1] 25



Nace una nueva generación de usuarios menos preocupados en la mecánica de R y más en las posibilidades que lo que R les permite hacer.

Serie de paquetes conocidos como «[tidyverse](#)» que vuelve «más simple», la limpieza y visualización de los datos.



R y RStudio

R y R-Studio



R: motor



RStudio: tablero





R es un lenguaje de programación que ejecuta cálculos.

Software gratuito y de código abierto

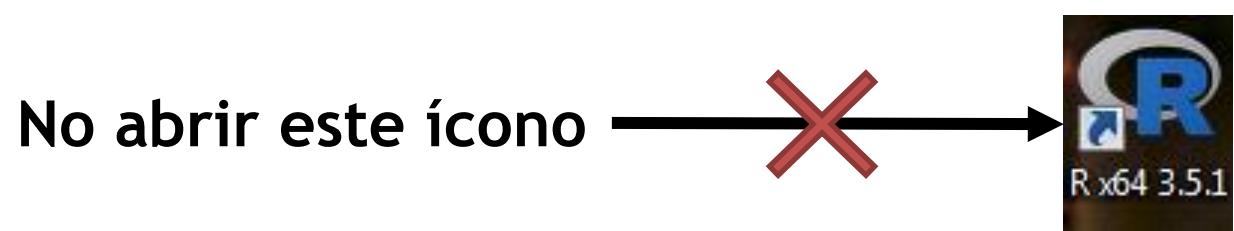
RStudio es un **entorno de desarrollo integrado (IDE)** que proporciona una interfaz al agregar muchas funciones y herramientas convenientes



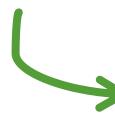
Vamos a usar **R desde RStudio**

Para instalar R y R-Studio pueden usar el [link](#) que esta en la página del curso.

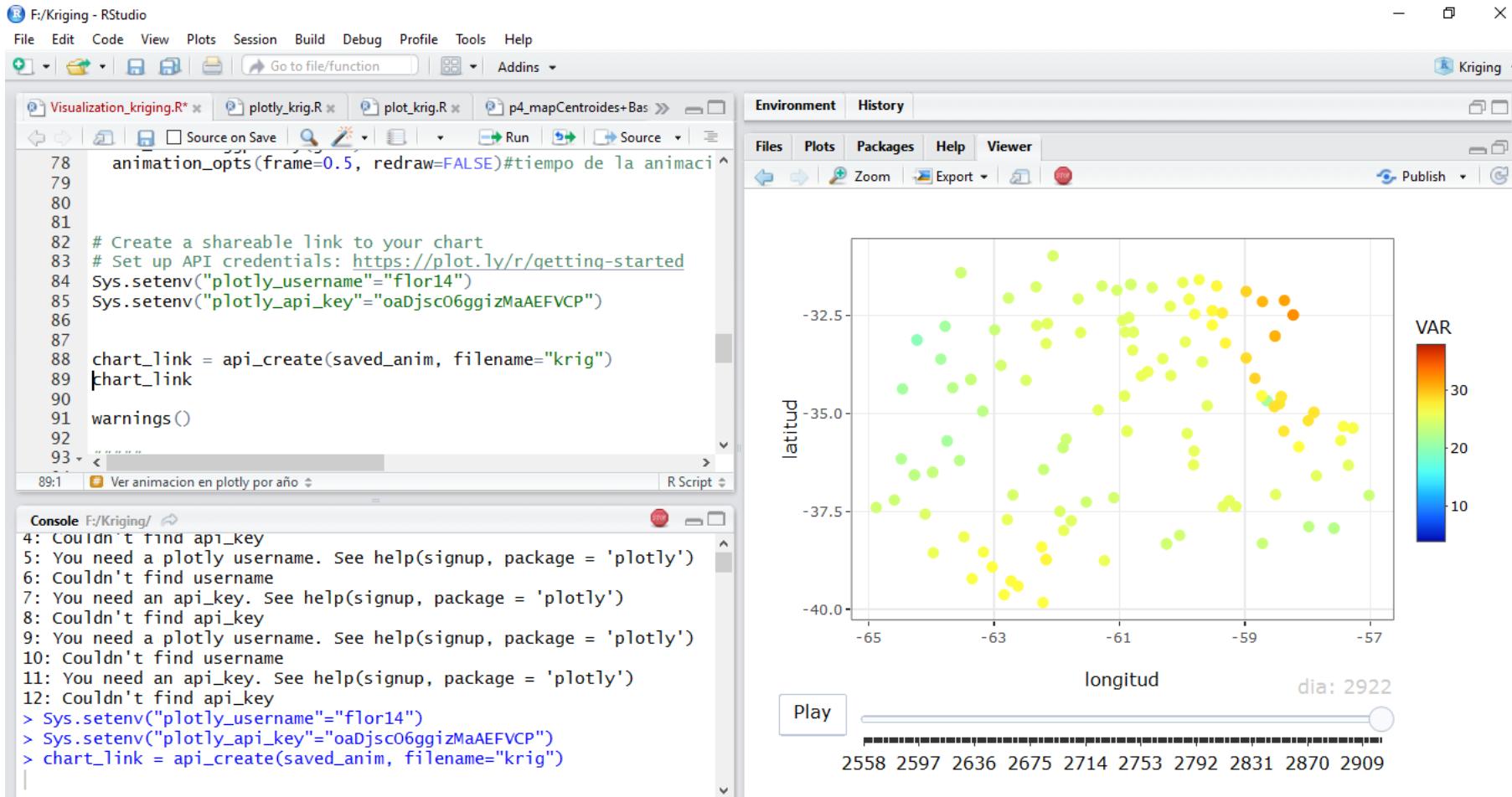
Cuando lo instalamos en nuestra máquina, se presentan dos íconos



R-Studio (IDE)



entorno de desarrollo integrado para R.



Panel de Vista/Editor

```

library(readxl)
library(dplyr)
library(tidyr)

#Graficos
library(ggplot2)

#Mapas
library(maps)
library(rworldmap)
library(viridis)

#Importar los datos
BDCAI20082018 <- read_excel("D:/EJS/BDCAI20082018.xlsx")

#Cantidad de casos
BDCAI20082018 %>%
  summarise(n(),na.rm=TRUE)

#Cantidad de casos por género
BDCAI20082018 %>%
  group_by(Genero) %>%
  summarise(cantidad=n())

BDCAI20082018 %>%
  distinct(IdPaper) %>%
  summarise(cantidadPapers=n())
  
```

Consola

```

R version 3.5.1 (2018-07-02) -- "Feather Spray"
Copyright (C) 2018 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.

[workspace loaded from D:/EJS/.RData]

> |
```

Panel Ambiente Historial

Panel Archivos Gráficos Paquetes Ayuda Visor

The screenshot shows the RStudio interface. In the top-left, a script editor window displays R code for reading data from Excel files and summarizing it. The code includes imports for `readxl`, `dplyr`, and `tidyverse`, and operations like `read_excel`, `summarise`, and `group_by`. The console window at the bottom shows the R startup message and the loading of the workspace from `D:/EJS/.RData`.

```
library(readxl)
library(dplyr)
library(tidyverse)
#GeoJson
library(ggplot2)
#Maps
library(maps)
library(rwandelmap)
library(maps)
#Importar los datos
BDCAI20082018 <- read_excel("D:/EJS/BDCAI20082018.xlsx")
#Cantidad de casos
BDCAI20082018 %>%
  summarise(n(),na.rm=TRUE)
#Cantidad de casos por género
BDCAI20082018 %>%
  group_by(Genero) %>%
  summarise(cantidad=n())
BDCAI20082018 %>%
  distinct(IdPaper) %>%
  summarise(cantidadPapers=n())

```

R version 3.5.1 (2018-07-02) -- "Feather Spray"
Copyright (C) 2018 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.

[workspace loaded from D:/EJS/.RData]

> |

Resultados y salidas del código

Objetos en memoria (tablas, variables)

Aquí se ven archivos, gráficos, ayuda, paquetes instalados

Paquetes

Paquetes

Los paquetes son las unidades fundamentales de código reproducible en R

- ▶ Típicamente, un paquete incluirá **código** (¡no solo de R!)
- ▶ **Documentación** del paquete y de las funciones internas,
- ▶ Algunas pruebas o **tests** para verificar que todo funcione como debería
- ▶ Conjuntos de **datos**



Funciones

Grupo de sentencias bajo un **nombre** que realizan una tarea específica



doblar(izquierda, 90 grados)

Nombre

Argumentos

¿Cuál es la diferencia entre un paquete y una librería?



Una biblioteca o *library* es simplemente un directorio que contiene paquetes instalados.

Puedes tener múltiples bibliotecas en tu computadora.



paquete



Usas la función **library()** para llamar un paquete de la librería

Analogía en tweet de H. Wickham

Desarrollo de paquetes

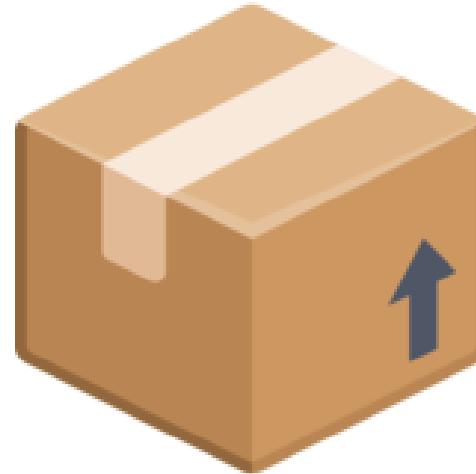
Rol

Autores

Mantenimiento

Colaboradores

Los paquetes que no se mantienen pueden quedar obsoletos o huérfanos



ggplot2: Create Elegant Data Visualisations Using the Grammar of Graphics

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.

Version: 3.2.0
Depends: R (≥ 3.2)
Imports: [digest](#), grDevices, grid, [gtable](#) (≥ 0.1.1), [lazyeval](#), MASS, mgcv, [reshape2](#), [rlang](#) (≥ 0.3.0), [scales](#) (≥ 0.5.0), stats, [tibble](#), [viridisLite](#), [withr](#) (≥ 2.0.0)
Suggests: [covr](#), [dplyr](#), [ggplot2movies](#), [hexbin](#), [Hmisc](#), [knitr](#), [lattice](#), [mapproj](#), [maps](#), [maptools](#), [multcomp](#), [munsell](#), [nlme](#), [profvis](#), [quantreg](#), [rgeos](#), [rmarkdown](#), [rpart](#), [sf](#) (≥ 0.7-3), [svglite](#) (≥ 1.2.0.9001), [testthat](#) (≥ 0.11.0), [vdiffr](#) (≥ 0.3.0)

Enhances: [sp](#)

Published: 2019-06-16

Author: Hadley Wickham [aut, cre], Winston Chang [aut], Lionel Henry [aut], Thomas Lin Pedersen [aut], Kohske Takahashi [aut], Claus Wilke [aut], Kara Woo [aut], Hiroaki Yutani [aut], RStudio [cph]

Maintainer: Hadley Wickham <hadley at rstudio.com>

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

License: [GPL-2](#) | file [LICENSE](#)

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

NeedsCompilation: no

Citation: [ggplot2 citation info](#)

Materials: [README](#) [NEWS](#)

In views: [Graphics](#), [Phylogenetics](#), [TeachingStatistics](#)

CRAN checks: [ggplot2 results](#)

Downloads:

Reference manual: [ggplot2.pdf](#)

Vignettes: [Extending ggplot2](#)
[Aesthetic specifications](#)
[Profiling Performance](#)

¿Qué es un repositorio?



Un repositorio es un lugar donde se encuentran los paquetes para que puedas instalarlos desde allí.

Están en línea y son accesibles para todos.

CRAN

Bioconductor

GitHub

Específico de R

No específico

General

Bioinformática

General

Con proceso de revisión

Sin revisión

Desarrollo de paquetes por la
comunidad de usuarios

- > Paquetes muy específicos

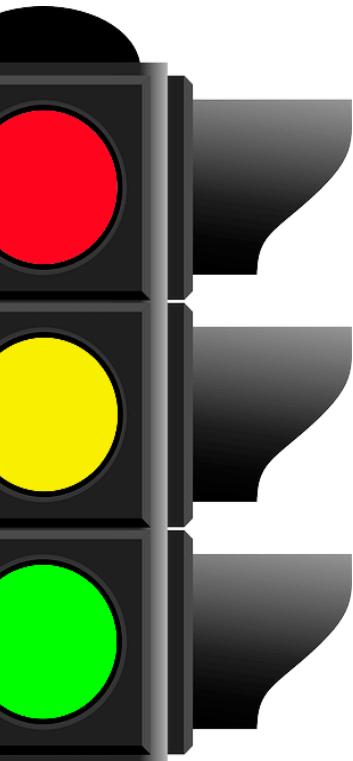


Error in xy.coords(x, y, xlabel, ylabel, log) : object '...



¿Abandono R?
¡NO!

Mensajes de R



Errores:

Aparecerá “**Error in...**” y tratará de explicar qué fue lo que salió mal.
Generalmente cuando hay un error, **el código no se ejecutará**.

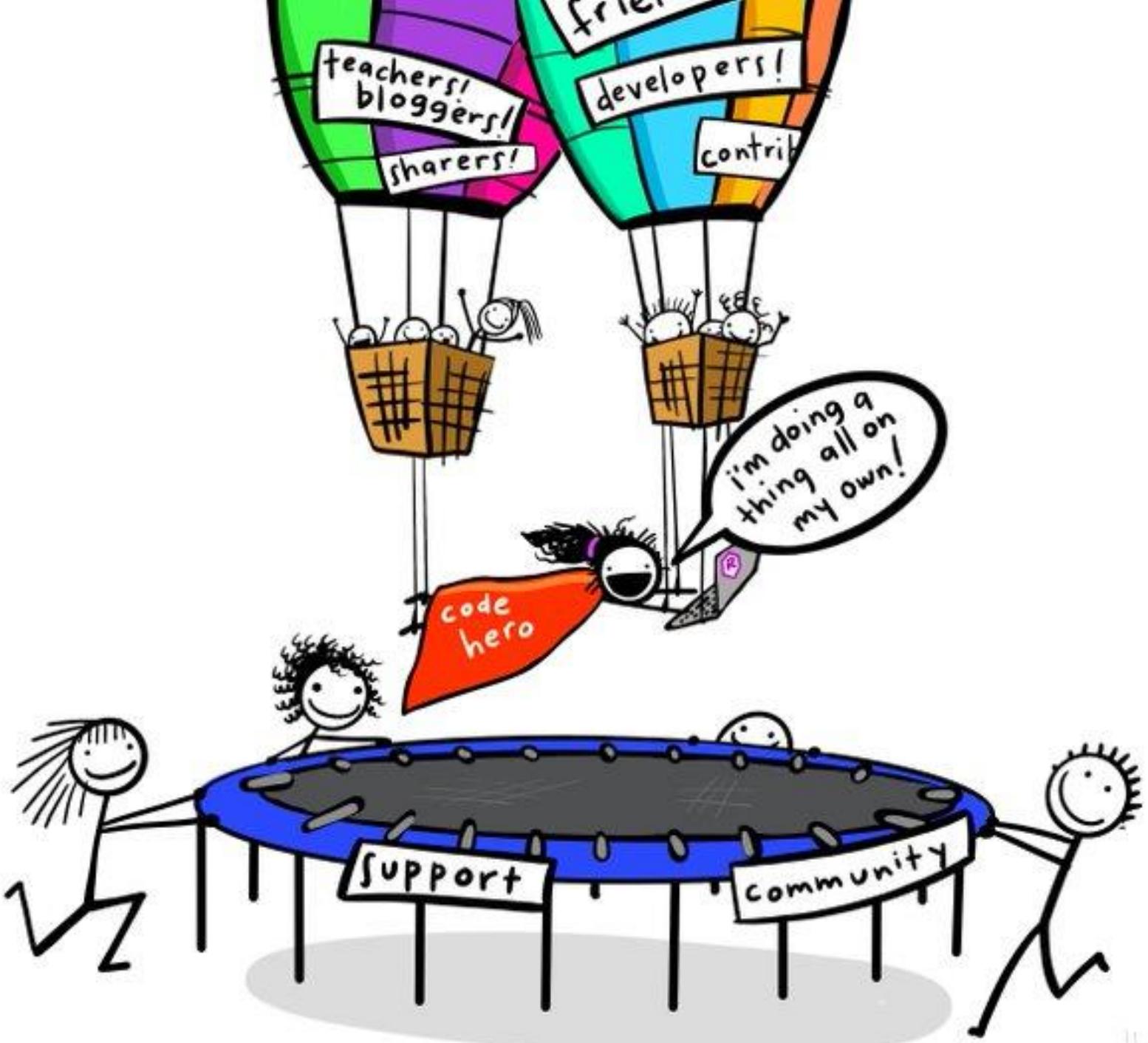
Advertencias:

Aparecerá “**Warning**” y tratará de explicar por qué hay una advertencia.
En general, **el código seguirá funcionando, pero hay algo a revisar o estar atentos**.

Mensajes:

Cuando el texto en rojo **no comienza con "Error in" o «Warning** mensaje amistoso.
Son mensajes de diagnóstico útiles y no impiden que el código funcione.

Imagen por Allison Horst
twitter.com/allison_horst





RStudio Community

FORO

<https://community.rstudio.com/>



Sign Up

Log In



[all categories ▾](#) [all tags ▾](#)

Latest

Categories Top

Topic

Category

Users

Replies

Views

Activity

Welcome to the RStudio Community!

Welcome to community.rstudio.com — we're glad to have you! This welcome page will give you some advice on how to get the most out of the site if you're getting or giving help. We want this to be a friendly, inclusive com... [read more](#)

meta



0

2.7k

Jul 22

Get column from table

[rstudio](#) [datatable](#)



1

3

1m

How to update with content containing multiple lines using updateTextAreainput

shiny



0

2

5m

Strange locale problems in R after update to Mojave

RStudio IDE



21

229

11m

Is it possible to save edits made in a DT table?

General



3

236

2h

Not able to get an image into word/pdf from URL in RMarkdown

[rmarkdown](#) [rstudio](#) [pandoc](#)

R Markdown



3

17

2h

create MCA ggplot

[ggplot2](#)

tidyverse



1

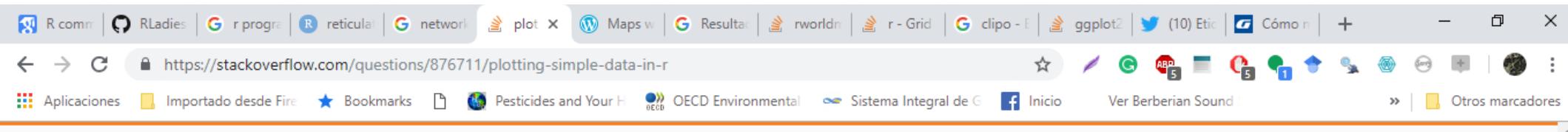
9

2h

StackOverflow (no es solo de R)

FORO

<https://stackoverflow.com/>



Plotting Simple Data in R

I have a comma separated file named `foo.csv` containing the following data:

```
scale, serial, spawn, for, worker
5, 0.000178, 0.000288, 0.000292, 0.000300
10, 0.156986, 0.297926, 0.064509, 0.066297
12, 2.658998, 6.059502, 0.912733, 0.923606
15, 188.023411, 719.463264, 164.111459, 161.687982
```

I essentially have two questions:

- How do I plot the first column (x-axis) versus the second column (y-axis)? I'm trying this (from reading [this site](#)):

```
data <- read.table("foo.csv", header=T, sep=",")  
attach(data)  
scale <- data[1]  
serial <- data[2]  
plot(scale, serial)
```

But I get this error back:

```
Error in stripchart.default(x1, ...) : invalid plotting method
```

Any idea what I'm doing wrong? A [quick Google search](#) reveals someone else with the same problem, but no solution. UPDATE: It's working fine! Had to change my code to:

asked 9 years, 4 months ago
viewed 68,988 times
active 1 year, 7 months ago

Looking for a job?

Software Architect
MuleSoft Buenos Aires, Argentina
RELOCATION
architecture

Engineering Manager
MuleSoft Argentina Buenos Aires, Argentina
RELOCATION
javascript java

Machine Learning Information Engineer
Elastic No office location
REMOTE

Twitter

Seguir cuentas de Twitter sobre **#rstatsES** o
#rstats

- Nuevos paquetes
- Actualizaciones
- Funciones útiles
- Conferencias (UseRs y RStudioConf)

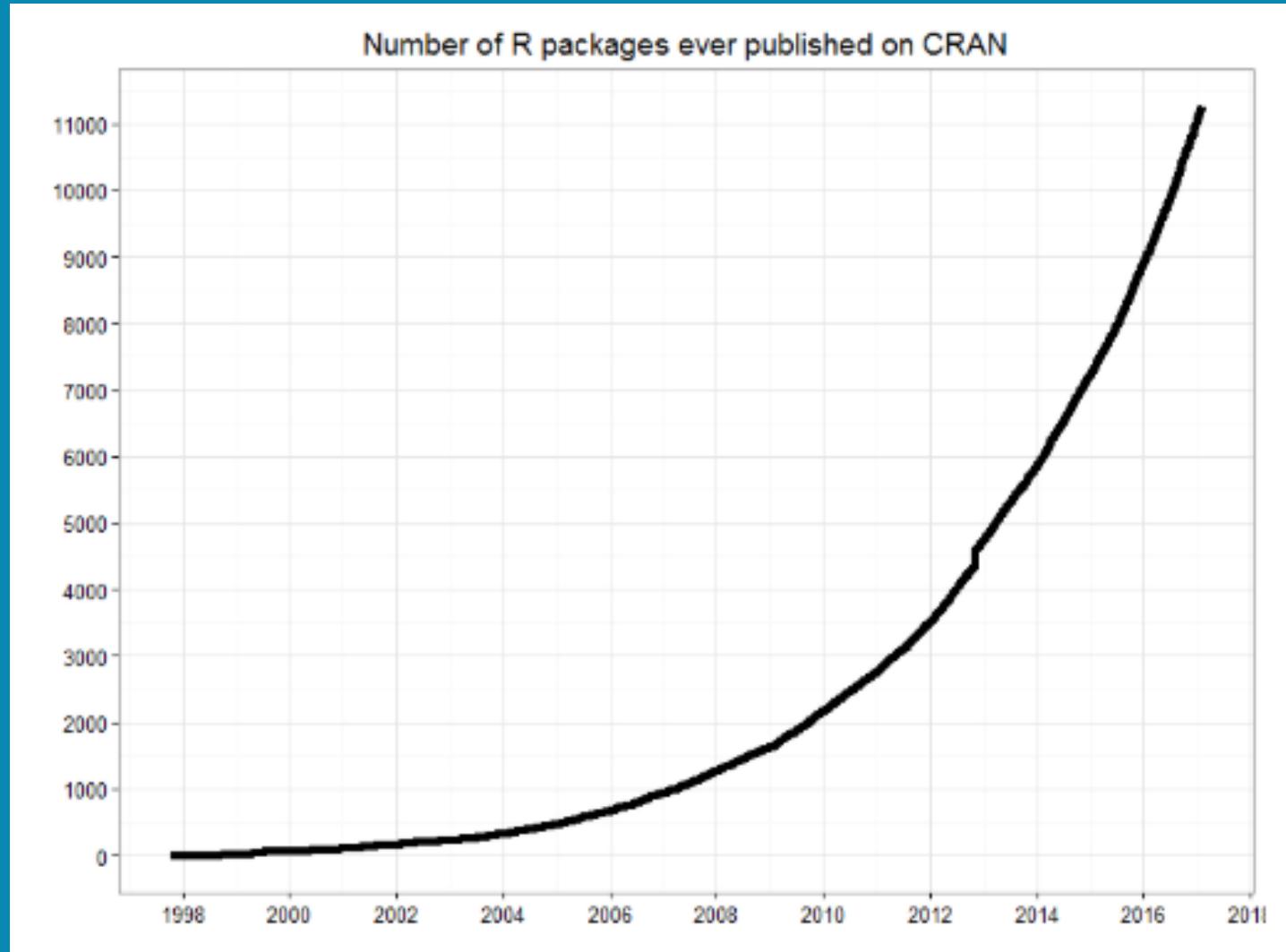


https://twitter.com/rweekly_org

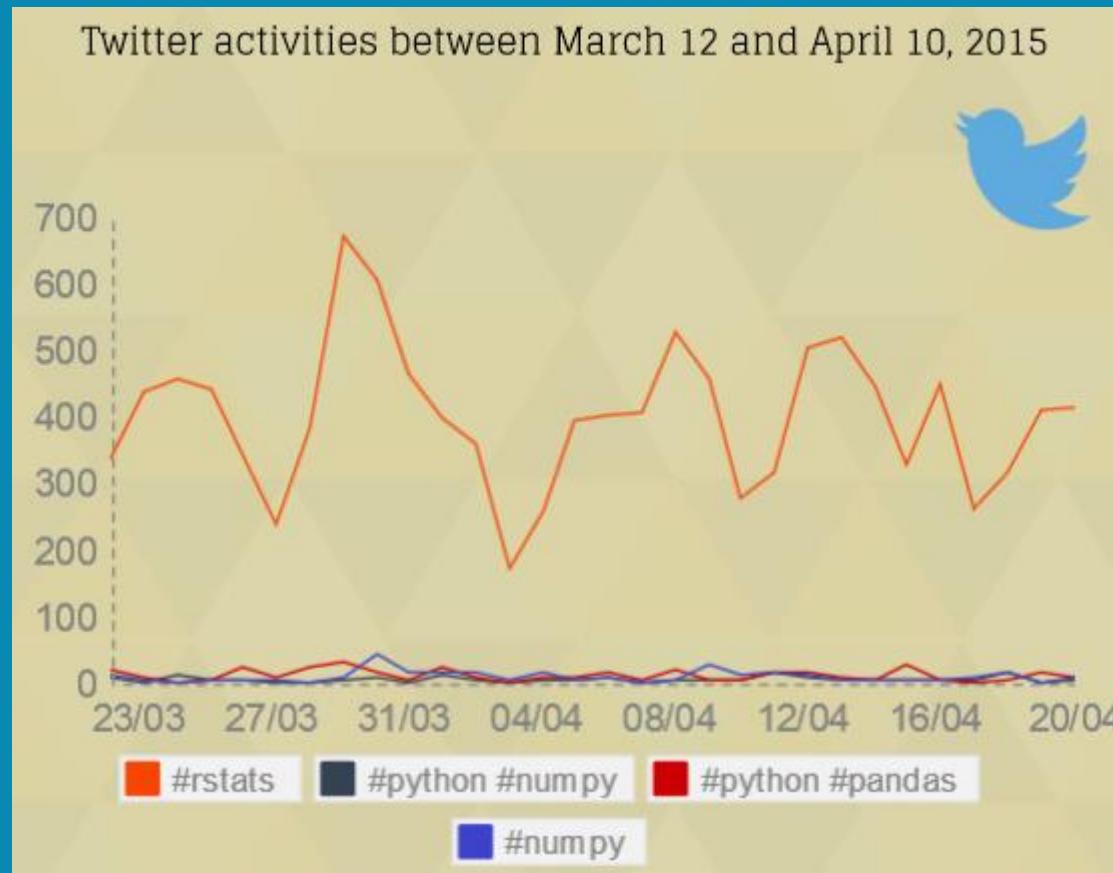
<https://twitter.com/dataandme>

Información

Avanza MUY rápido



La comunidad de R es twittera



Tidyverse

Untitled1* diamantes diamonds

Source on Save Run Source

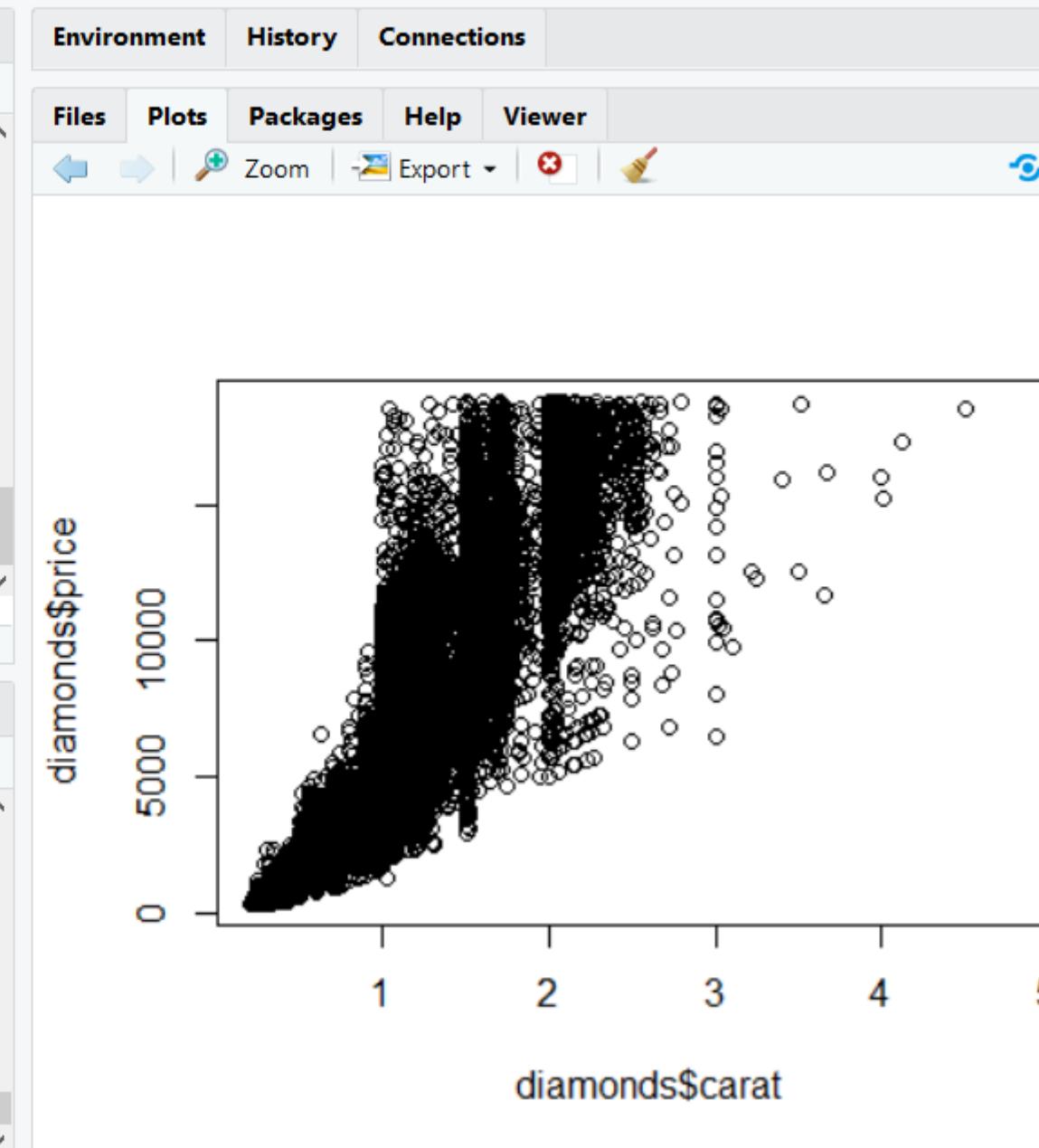
```
52
53
54
55 # transparencia -----
56
57 plot(diamonds$carat, diamonds$price)
58
59
60
61
62
63
64
65
```

transparencia

R Script

Console Terminal Markers

```
~/.25 Good E VS1 0.9 0.5 327 4.00 4.01 2.31
4 0.290 Premium I VS2 62.4 58 334 4.2 4.23 2.63
5 0.31 Good J SI2 63.3 58 335 4.34 4.35 2.75
6 0.24 Very ~ J VVS2 62.8 57 336 3.94 3.96 2.48
> write.table(head(diamonds), "D:/Desktop/diamantes.txt", sep = " ")
)
> diamantes <- read.csv("D:/Desktop/diamantes.txt", sep="")
> View(diamantes)
> plot(diamonds$carat, diamonds$price)
>
```



[Home](#)

PUBLIC

 Stack Overflow[Tags](#)[Users](#)

Esta pregunta se
acerca a lo que
quiero resolver

Any way to make plot points in scatterplot more transparent in R?



44



6

I have a 3 column matrix; plots are made by points based on column 1 and column 2 values, but colored based on column 2 (6 different groups). I can successfully plot all points, however, the last plot group (group 6) which was assigned the color purple, masks the plots of the other groups. Is there a way to make the plot points more transparent?

```
s <- read.table("~/.../parse-output.txt", sep="\t")
dim(s)
[1] 67124      3
x <- s[,1]
y <- s[,2]
z <- s[,3]
cols <- cut(z, 6, labels = c("pink", "red", "yellow", "blue", "green", "purple"))
plot(x, y, main= "Fragment recruitment plot - FR-HIT", ylab = "Percent identity", xlab = "Bas
<
```

[r](#) [plot](#) [ggplot2](#) [alpha](#)[share](#) [improve this question](#)[add a comment](#)

edited Oct 22 '12 at 0:59



mnel

88.6k • 14 • 212 • 219

asked Oct 21 '12 at 6:58

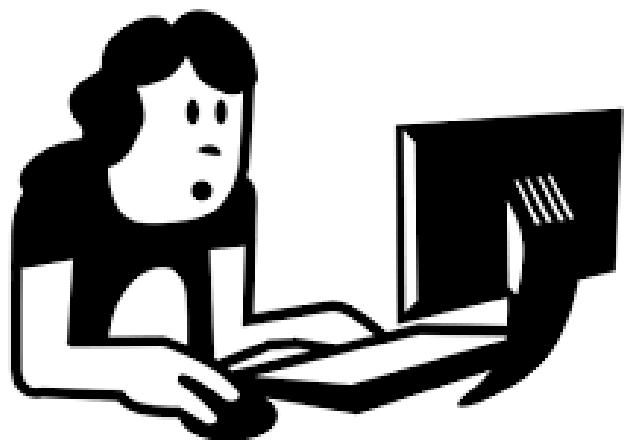


Steve

387 • 1 • 5 • 12



Creo que
encontré una
possible solución



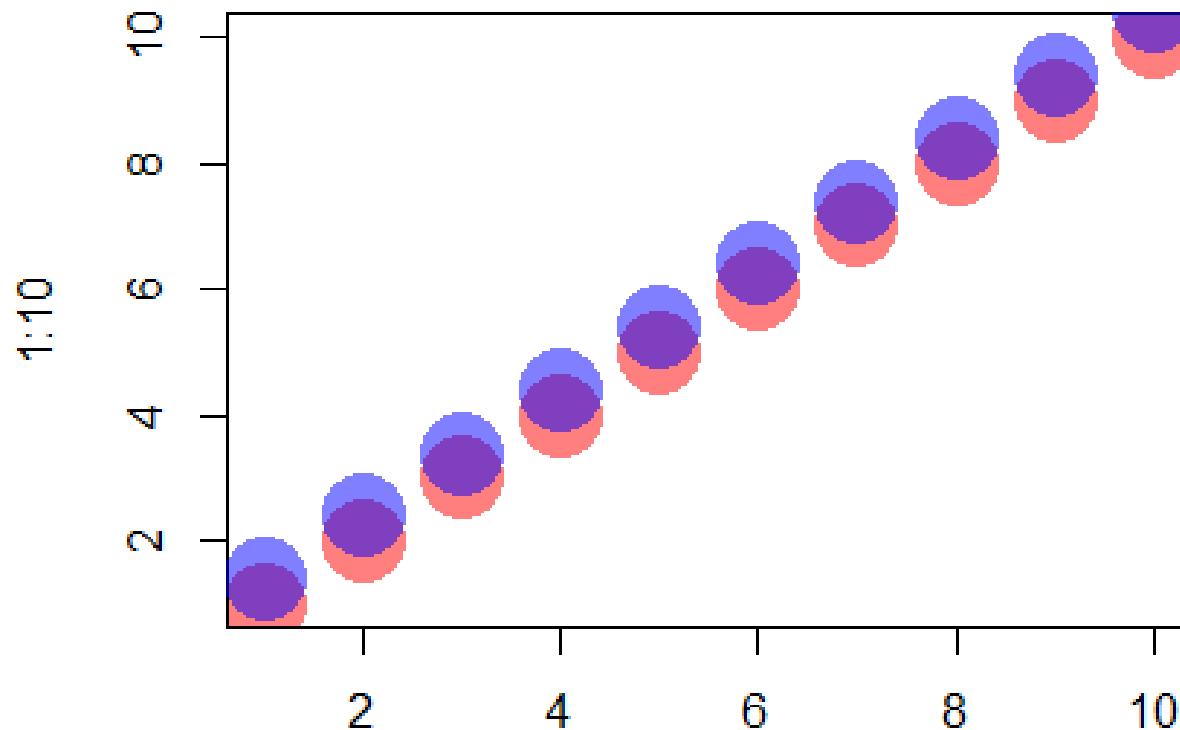
Respuesta 1



When creating the colors, you may use `rgb` and set its `alpha` argument:

32

```
plot(1:10, col = rgb(red = 1, green = 0, blue = 0, alpha = 0.5),  
     pch = 16, cex = 4)  
points((1:10) + 0.4, col = rgb(red = 0, green = 0, blue = 1, alpha = 0.5),  
       pch = 16, cex = 4)
```



Respuesta 2



10



If you decide to use `ggplot2`, you can set transparency of overlapping points using the `alpha` argument.

e.g.

```
library(ggplot2)
ggplot(diamonds, aes(carat, price)) + geom_point(alpha = 1/40)
```

[share](#) [improve this answer](#)

answered Oct 21 '12 at 7:44



Maiasaura

16.9k • 16 • 83 • 99

`plot()`

R base



`ggplot()`

tidyverse

Mismo gráfico... escrito diferente

R base

```
plot(x = diamonds$carat, y = diamonds$price)
```

tidyverse

```
library(ggplot2)  
ggplot(diamonds) aes(x = carat, y = price) +  
  geom_point()
```



Mismo gráfico... escrito diferente

R base

```
plot(diamonds$carat, diamonds$price)
```

tidyverse

```
library(ggplot2)  
ggplot(diamonds, aes(x = carat, y = price)) +  
  geom_point()
```



Tidyverse

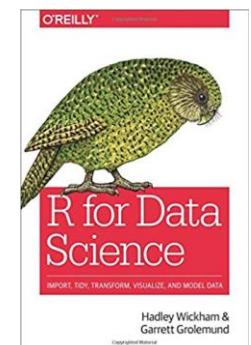
```
install.packages("tidyverse")
```





El **tidyverse** es una colección de paquetes R diseñados para ciencia de datos.

Todos los paquetes comparten una filosofía de diseño subyacente, gramática y estructuras de datos.



<http://r4ds.had.co.nz/>

Tidyverse



Tidyverse y r base
pueden coexistir



- **ggplot2**, 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.



- **tidyr** 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.

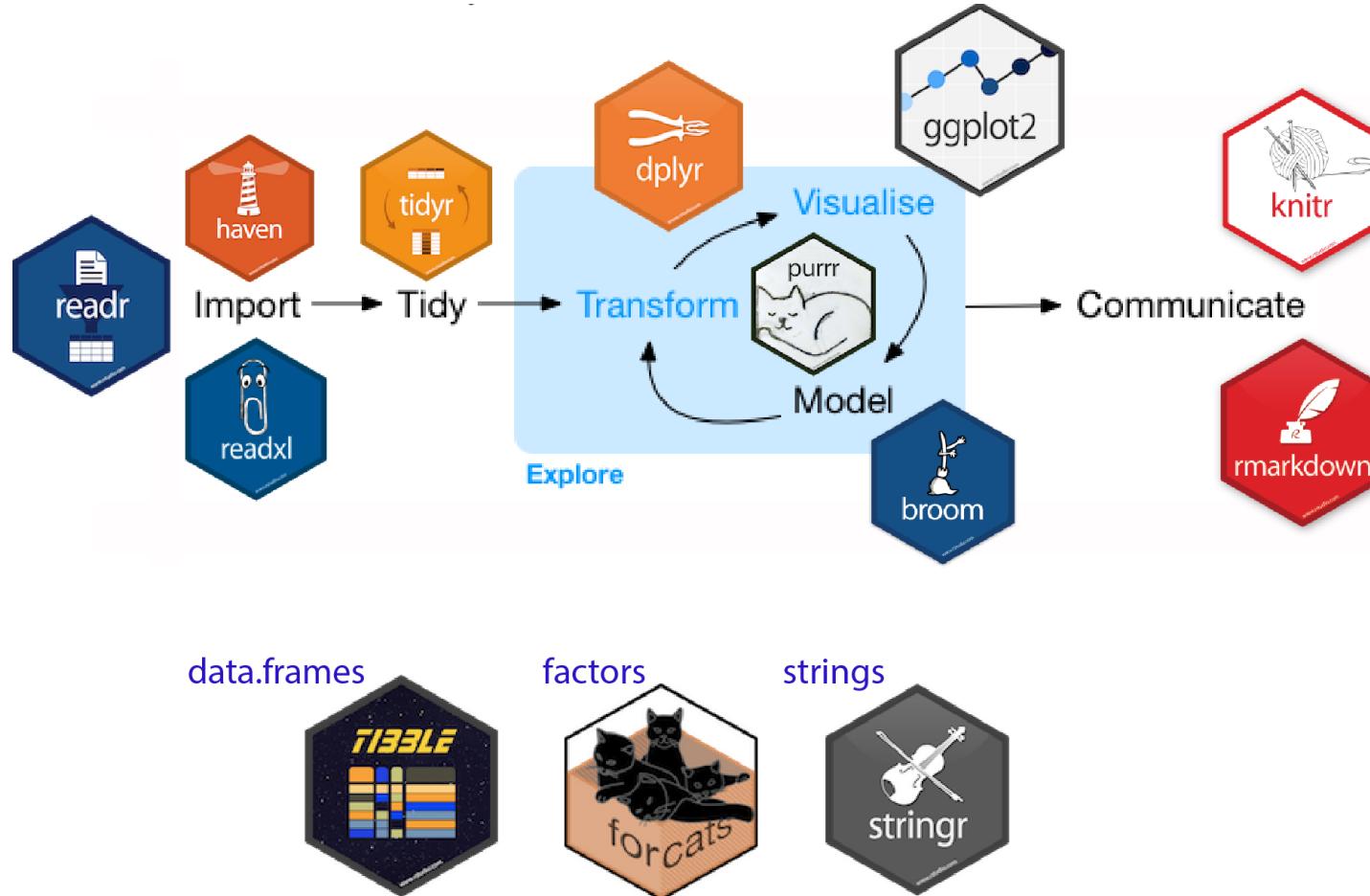


- **stringr** provides a cohesive set of functions designed to make working with strings as easy as possible



- **forcats** provide a suite of useful tools that solve common problems with factors.

Tidyverse ejemplo



Los paquetes del tidyverse se escriben diferente,
tienen una **sintaxis** particular

R Syntax Comparison :: CHEAT SHEET

Dollar sign syntax

```
goal(data$x, data$y)
```

SUMMARY STATISTICS:

one continuous variable:

```
mean(mtcars$mpg)
```

one categorical variable:

```
table(mtcars$cyl)
```

two categorical variables:

```
table(mtcars$cyl, mtcars$am)
```

one continuous, one categorical:

```
mean(mtcars$mpg[mtcars$cyl==4])
```

```
mean(mtcars$mpg[mtcars$cyl==6])
```

```
mean(mtcars$mpg[mtcars$cyl==8])
```

PLOTTING:

one continuous variable:

```
hist(mtcars$disp)
```

```
boxplot(mtcars$disp)
```

one categorical variable:

```
barplot(table(mtcars$cyl))
```

two continuous variables:

```
plot(mtcars$disp, mtcars$mpg)
```

two categorical variables:

```
mosaicplot(table(mtcars$am, mtcars$cyl))
```

one continuous, one categorical:

```
histogram(mtcars$disp[mtcars$cyl==4])
```

```
histogram(mtcars$disp[mtcars$cyl==6])
```

```
histogram(mtcars$disp[mtcars$cyl==8])
```

```
boxplot(mtcars$disp[mtcars$cyl==4])
```

```
boxplot(mtcars$disp[mtcars$cyl==6])
```

```
boxplot(mtcars$disp[mtcars$cyl==8])
```

WRANGLING:

subsetting:

```
mtcars[mtcars$mpg>30, ]
```

making a new variable:

```
mtcars$eff1cient[mtcars$mpg>30] <- TRUE
```

```
mtcars$efficient[mtcars$mpg<30] <- FALSE
```

Formula syntax

```
goal(y~x|z, data=data, group=w)
```

SUMMARY STATISTICS:

one continuous variable:

```
mosaic::mean(~mpg, data=mtcars)
```

one categorical variable:

```
mosaic::tally(~cyl, data=mtcars)
```

two categorical variables:

```
mosaic::tally(cyl~am, data=mtcars)
```

one continuous, one categorical:

```
mosaic::mean(mpg~cyl, data=mtcars)
```

tilde

PLOTTING:

one continuous variable:

```
lattice::histogram(~disp, data=mtcars)
```

```
lattice::bwplot(~disp, data=mtcars)
```

one categorical variable:

```
mosaic::bargraph(~cyl, data=mtcars)
```

two continuous variables:

```
lattice::xyplot(mpg~disp, data=mtcars)
```

two categorical variables:

```
mosaic::bargraph(~am, data=mtcars, group=cyl)
```

one continuous, one categorical:

```
lattice::histogram(~disp|cyl, data=mtcars)
```

```
lattice::bwplot(cyl~disp, data=mtcars)
```

The variety of R syntaxes give you many ways to “say” the same thing

read across the cheatsheet to see how different syntaxes approach the same problem

Tidyverse syntax

```
data %>% goal(x)
```

SUMMARY STATISTICS:

one continuous variable:

```
mtcars %>% dplyr::summarize(mean(mpg))
```

one categorical variable:

```
mtcars %>% dplyr::group_by(cyl) %>%  
dplyr::summarize(n())
```

the pipe

two categorical variables:

```
mtcars %>% dplyr::group_by(cyl, am) %>%  
dplyr::summarize(n())
```

one continuous, one categorical:

```
mtcars %>% dplyr::group_by(cyl) %>%  
dplyr::summarize(mean(mpg))
```

PLOTTING:

one continuous variable:

```
ggplot2::qplot(x=mpg, data=mtcars, geom = "histogram")
```

```
ggplot2::qplot(y=disp, x=1, data=mtcars, geom="boxplot")
```

one categorical variable:

```
ggplot2::qplot(x=cyl, data=mtcars, geom="bar")
```

two continuous variables:

```
ggplot2::qplot(x=disp, y=mpg, data=mtcars, geom="point")
```

two categorical variables:

```
ggplot2::qplot(x=factor(cyl), data=mtcars, geom="bar")  
facet_grid(.~am)
```

one continuous, one categorical:

```
ggplot2::qplot(x=disp, data=mtcars, geom = "histogram")  
facet_grid(.~cyl)
```

```
ggplot2::qplot(y=disp, x=factor(cyl), data=mtcars,  
geom="boxplot")
```

WRANGLING:

subsetting:

```
mtcars %>% dplyr::filter(mpg>30)
```

making a new variable:

```
mtcars <- mtcars %>%  
dplyr::mutate(efficient = if_else(mpg>30, TRUE, FALSE))
```

Algunos paquetes usan pipes

`%>%`

es un operador que nos permite encadenar las acciones que queremos aplicar

La mayoría de los paquetes del **tidyverse** usa pipes



Secuencias de Operaciones

Pasos intermedios

```
foo_foo_1 <- saltar(foo_foo, a_traves_del = bosque)
foo_foo_2 <- recoger(foo_foo_1, que = ratones_del_campo)
foo_foo_3 <- golpear(foo_foo_2, en_la = cabeza)
```

Sobre-escribir el original

```
foo_foo <- saltar(foo_foo, a_traves_del = bosque)
foo_foo <- recoger(foo_foo, que = ratones_del_campo)
foo_foo <- golpear(foo_foo, en_la = cabeza)
```

Ejemplo detallado en

<https://es.r4ds.hadley.nz/pipes.html>

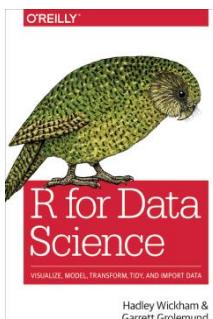
Pipes

```
foo_foo %>%  
  saltar(a_través_del = bosque) %>%  
  recoger(arriba = ratones_campo) %>%  
  golpear(en = la_cabeza)
```



Los pipes son mayoritariamente usados para reescribir una secuencia lineal bastante corta de operaciones.

Podría usar pipes con funciones de r base usando el paquete *magrittr*



Entonces

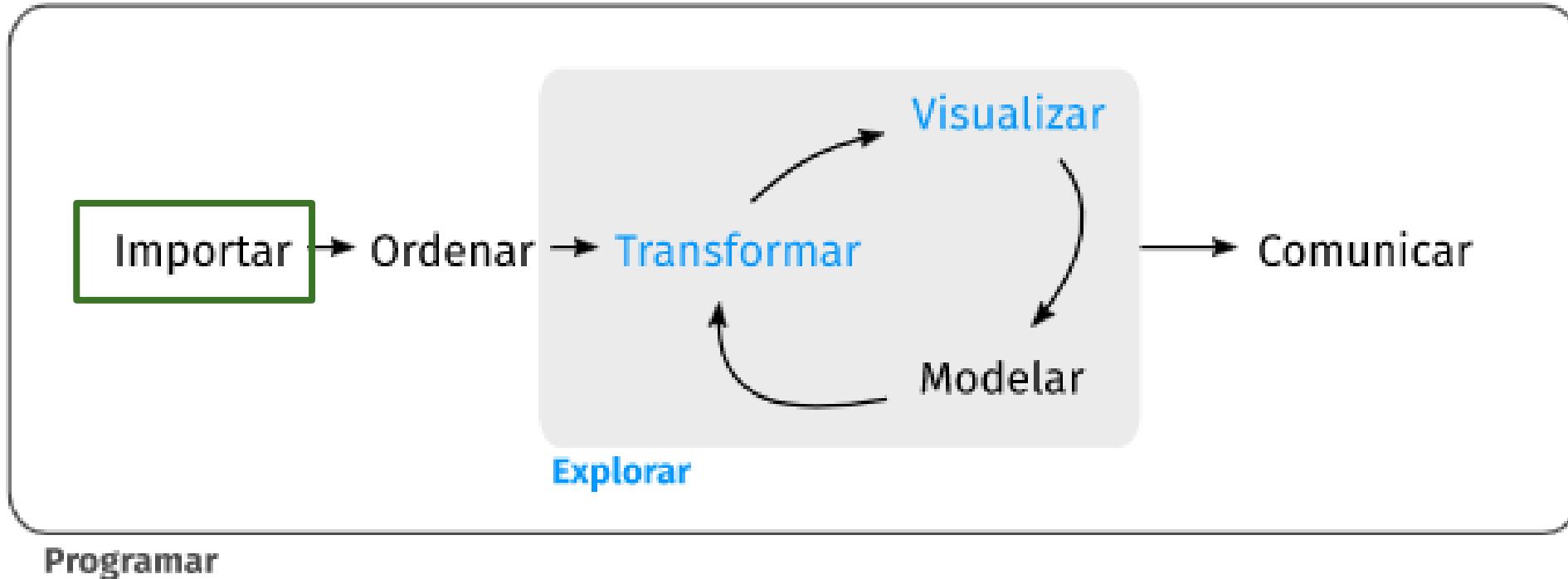
- ▶ ¿Tidyverse? -> incluir explícitamente en la búsqueda (ej: ggplot2) o seleccionar las respuestas que encuentro
- ▶ Tener en cuenta la **fecha** de la respuesta
- ▶ Buscar en **inglés** va a dar más resultados de búsqueda
- ▶ Busca ayudar en las comunidades, foros y twitter



Google

Importar datos

Importar datos



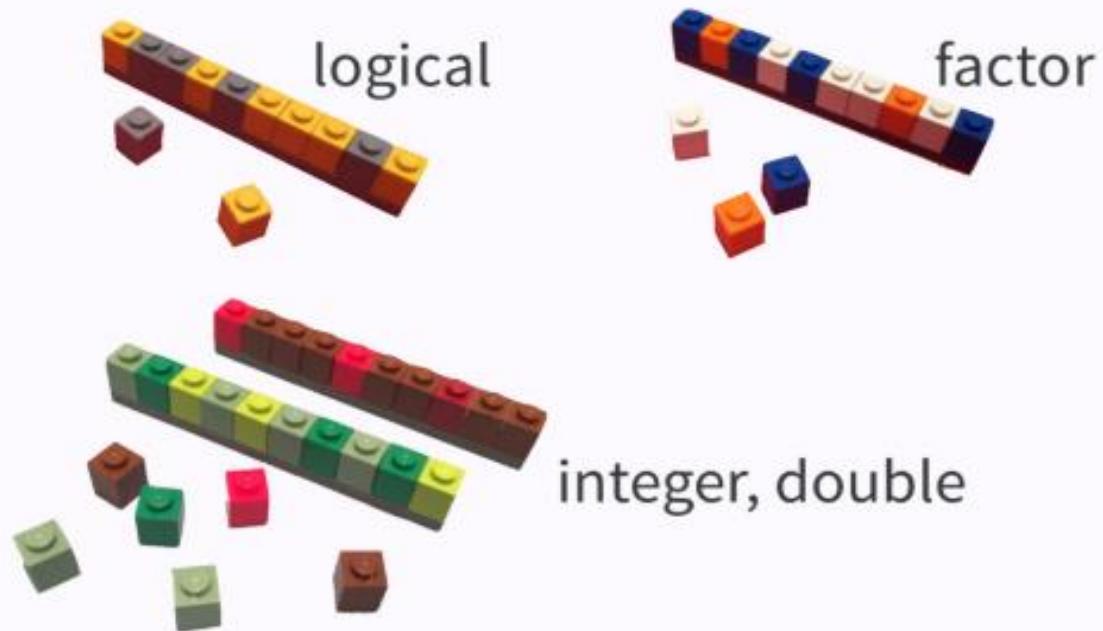
Vector

- Un vector es esencialmente una lista ordenada de cosas

Condición especial:

- Todo en el vector debe ser el mismo tipo de datos (double, integer, complex, logical o character)

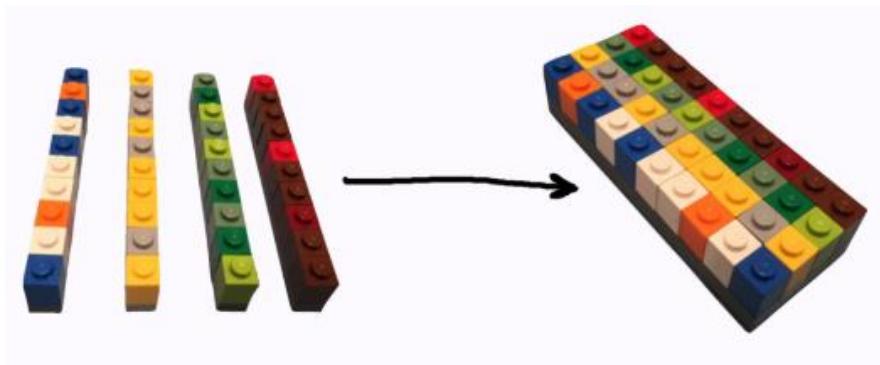
atomic vectors



data.frame

ESTRUCTURA DE DATOS

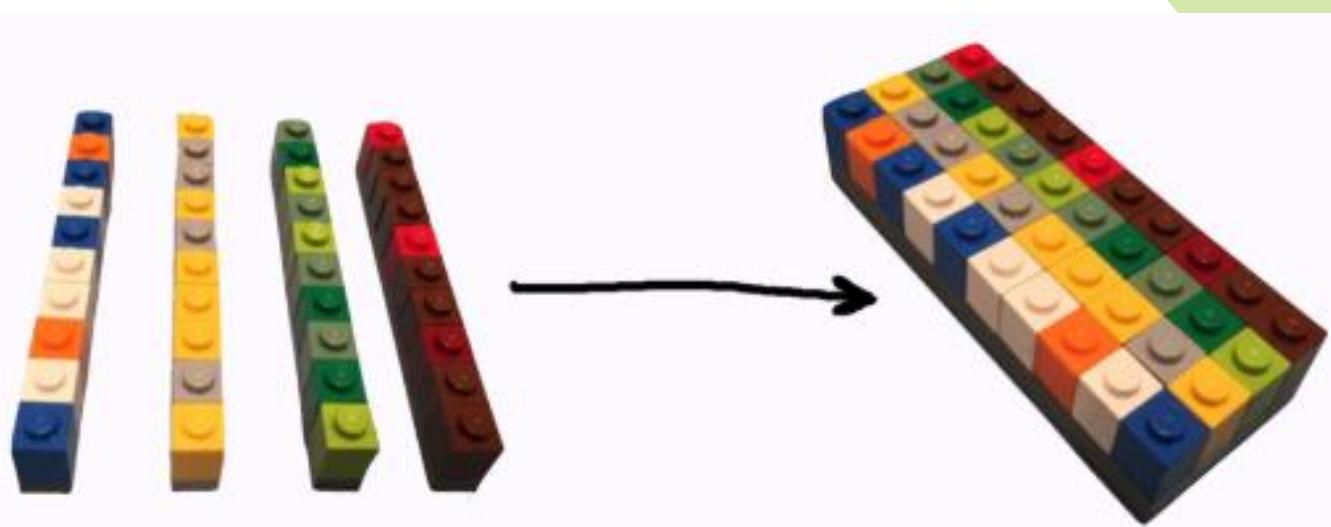
estructura que R sabe cómo construir a partir de los tipos de datos básicos.



data.frame

las columnas de los data.frames
son todos vectores

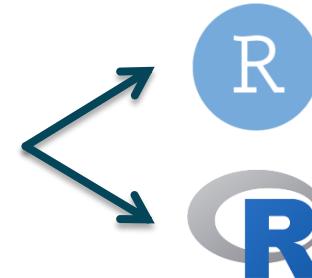
Por eso R cada columna tiene
un mismo tipo de datos.



vectors of same length? **DATA FRAME!**

Trabajando con datos tabulares

- ▶ Leer datos desde archivos externos
- ▶ Generar datos tabulares con R -> `data.frame()`
- ▶ Los paquetes pueden tener datasets precargados



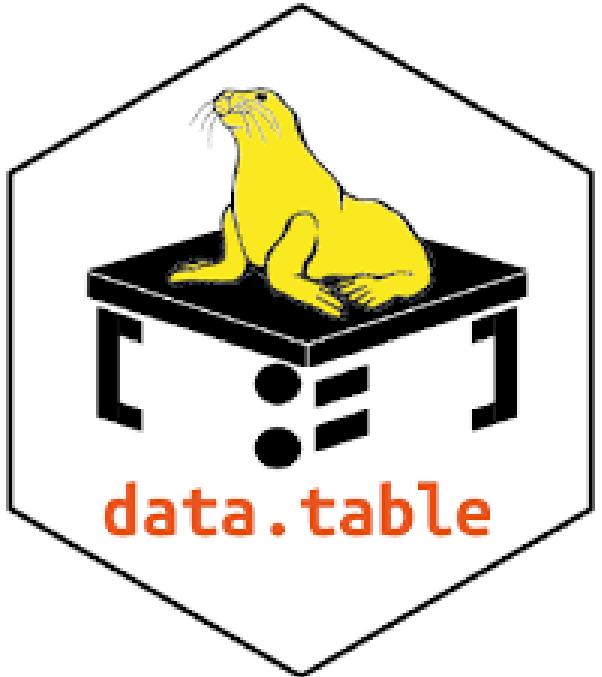
Tengo mis archivos en formato .xlsx



El paquete **readxl** facilita la obtención de datos desde Excel hacia R.

R Studio IDE tiene integrado este paquete

Otras consideraciones



Si tengo que importar muchos datos
puedo usar *data.table::fread()*. Es más
rápido.

¡Manos a la obra!

- ▶ Mostrar R y RStudio
- ▶ Uso de proyectos
- ▶ Cargar un paquete
- ▶ Escritura Paquete::funcion()
- ▶ Ayuda
- ▶ Importar una base de datos

https://flor14.github.io/r_inta -> `R_inta_LC1_2019.R`



Mate break

Foto: gentiliza Mauro Lepore