Introducción a R

Econometría I

Paula Pereda (ppereda@correo.um.edu.uy)
13 de agosto de 2021

Preliminares

Sobre los talleres en sí

- **Viernes** de 08:00 a 09:40 | 📍 Sede Sinergia Ejido (Ejido 1275) - Salón K301
- Talleres: ejercicios teóricos + aplicaciones en R (intuición & práctica).
- Cada dos semanas apróximadamente hay que hacer un trabajo domiciliario.
- **Trabajo final:**
  - Hasta 3 integrantes
  - 29/10: Entrega tema de trabajo de investigación.
  - 21/11: Entrega trabajo de investigación (*paper* o presentación, código + datos).
  - Mi mail está al inicio de cada presentación, **por cualquier consulta siempre me pueden escribir.**

Sobre los talleres en sí

- **Composición de la nota final:**
  - Parcial - 40%
  - Trabajo de investigación - 30%
  - Prácticos - 25%
  - Actitud y participación - 5%

Econometría

Un econometrista aplicado[†] necesita un conocimiento sólido de (al menos) tres áreas:

1. La **teoría** de la econometría subyacente (supuestos, resultados, fortalezas, debilidades).

2. Sabér cómo **aplicar métodos teóricos** a datos reales.

3. Métodos eficientes para **trabajar con datos** — limpiar, agregar, unir, visualizar.

Este curso tiene como objetivo profundizar conocimientos en cada una de estas tres áreas.

- 1: Como antes.
- 2–3: **R**

[†]: *Econometrista aplicado* = Practicante de econometría, *e.g.*, analista, consultor, científico de datos.

R

¿Qué es R?

Citando a la web del proyecto R:

> R es un entorno de software libre para gráficos y computación estadística. Se compila y se ejecuta en una amplia variedad de plataformas UNIX, Windows y MacOS.

¿Qué significa esto?

- R fue creado para el trabajo estadístico y gráfico requerido por la econometría.
- R tiene una comunidad en línea bastante activa y útil. (Stack Overflow, R-Ladies, Grupos de usuarios de R)
- Además, es **gratis** y **código abierto**.

¿Por qué usar R?

1. R es **gratis** y de **código abierto**.

2. R es **flexible y poderoso**— es adaptable a casi cualquier tarea, *e.g.*, estadística, análisis de datos espaciales, machine learning, web scraping, limpieza de datos, creación de sitios web, estas notas de clase.

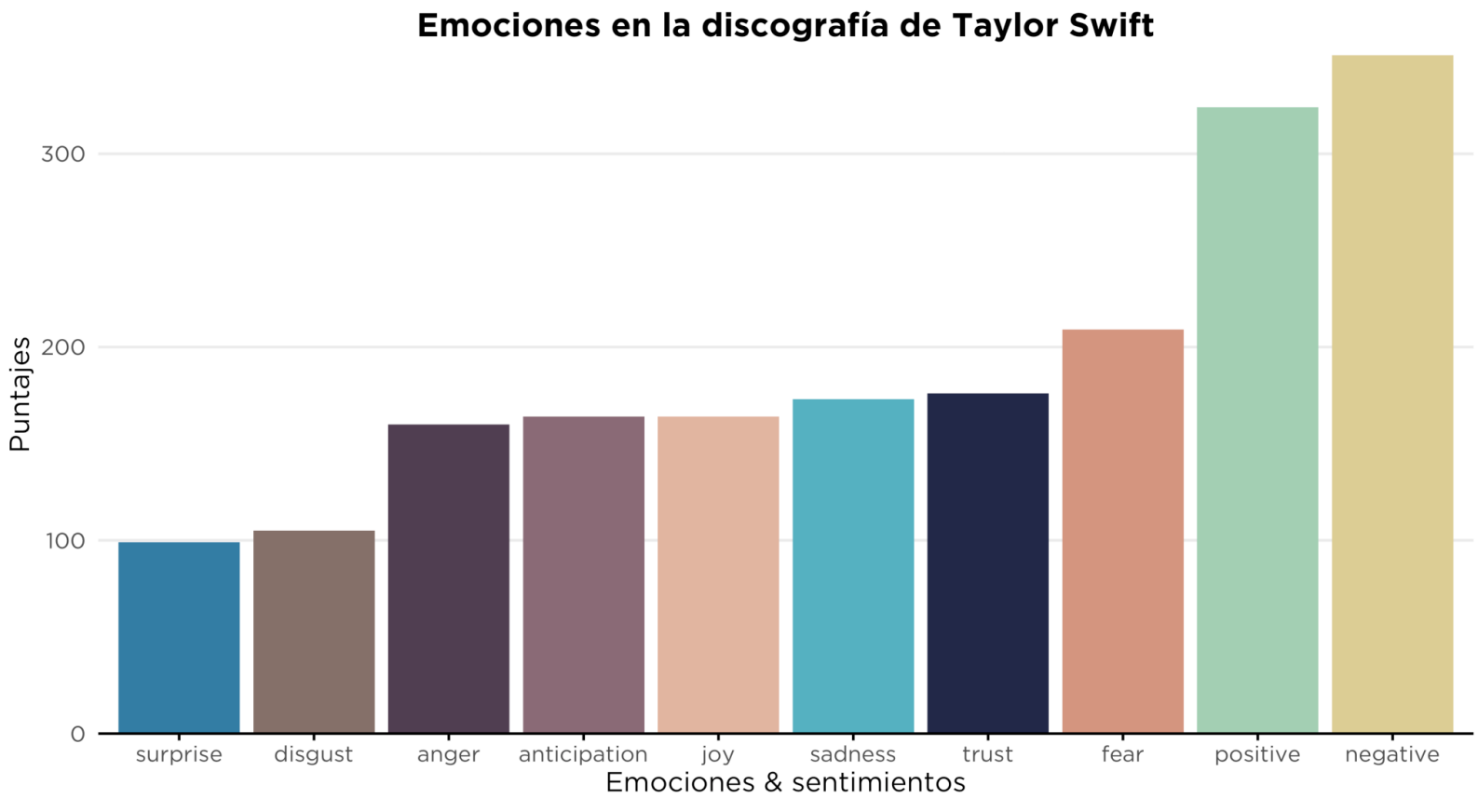
3. *Relacionado:* R **no impone** en la cantidad de observaciones, memoria o poder de procesamiento.

4. Con dedicación † pueden obtener una herramienta **valiosa y comercial**.

5. 💖 **R**

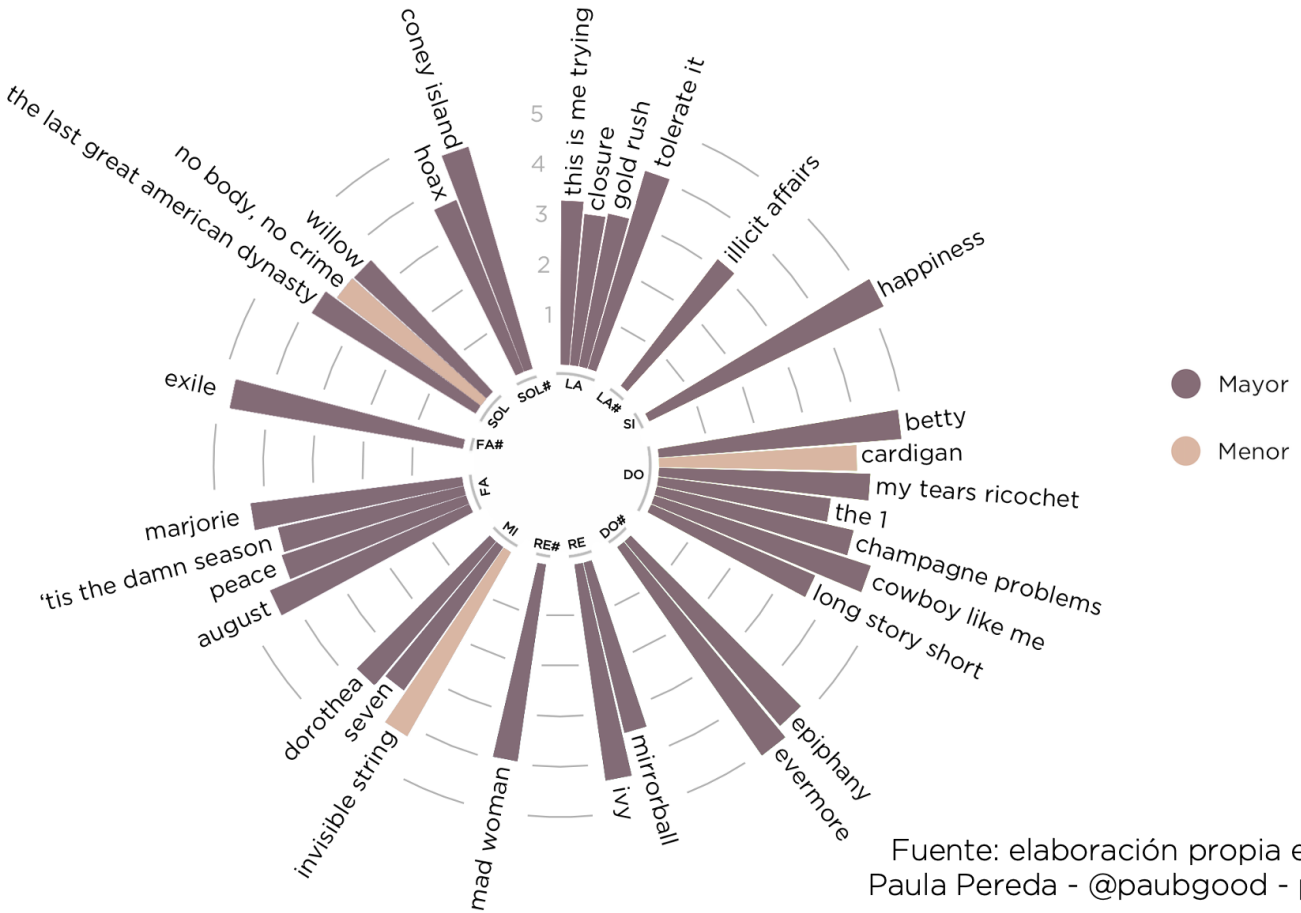
[†]: Aprender R definitivamente requiere tiempo y esfuerzo.


Al principio, era del tipo...
...pero ahora es del tipo...
Horst '18



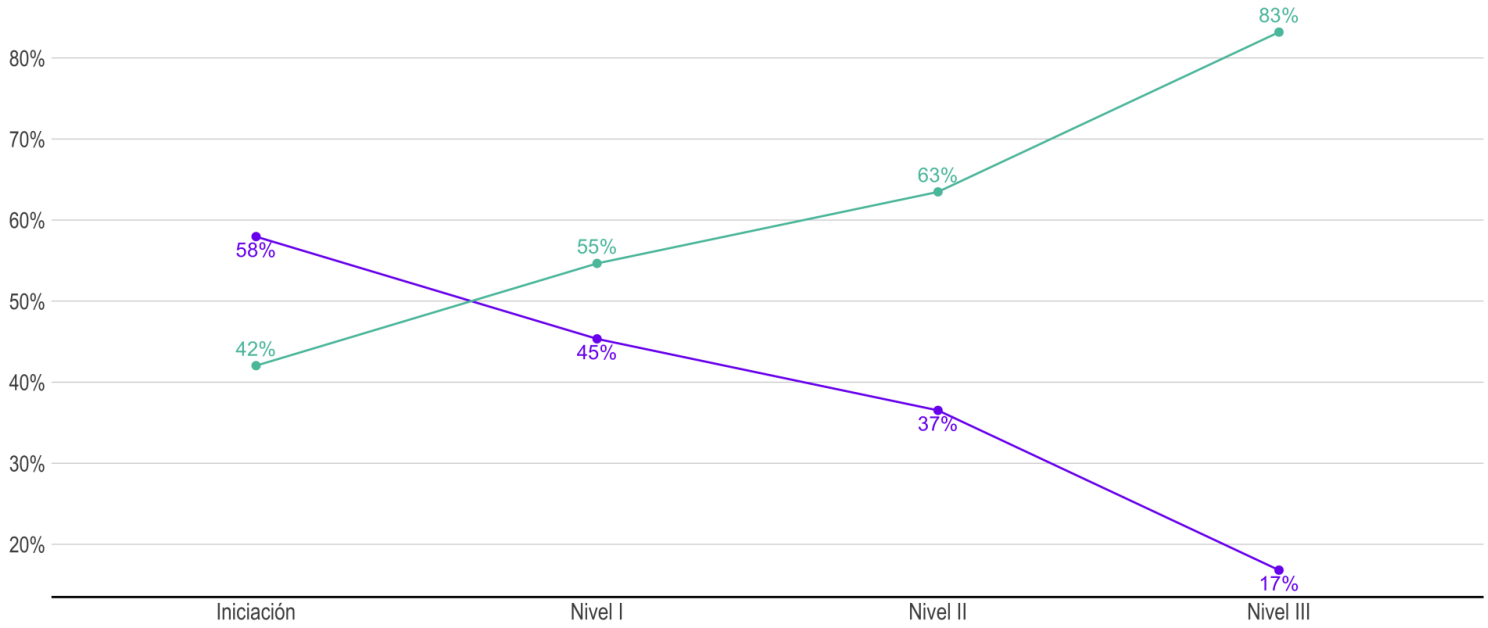
Fuente: elaboración propia en base a Genius.
Paula Pereda - @paubgood - paulapereda.com

Tonalidades de las canciones de folklore y evermore

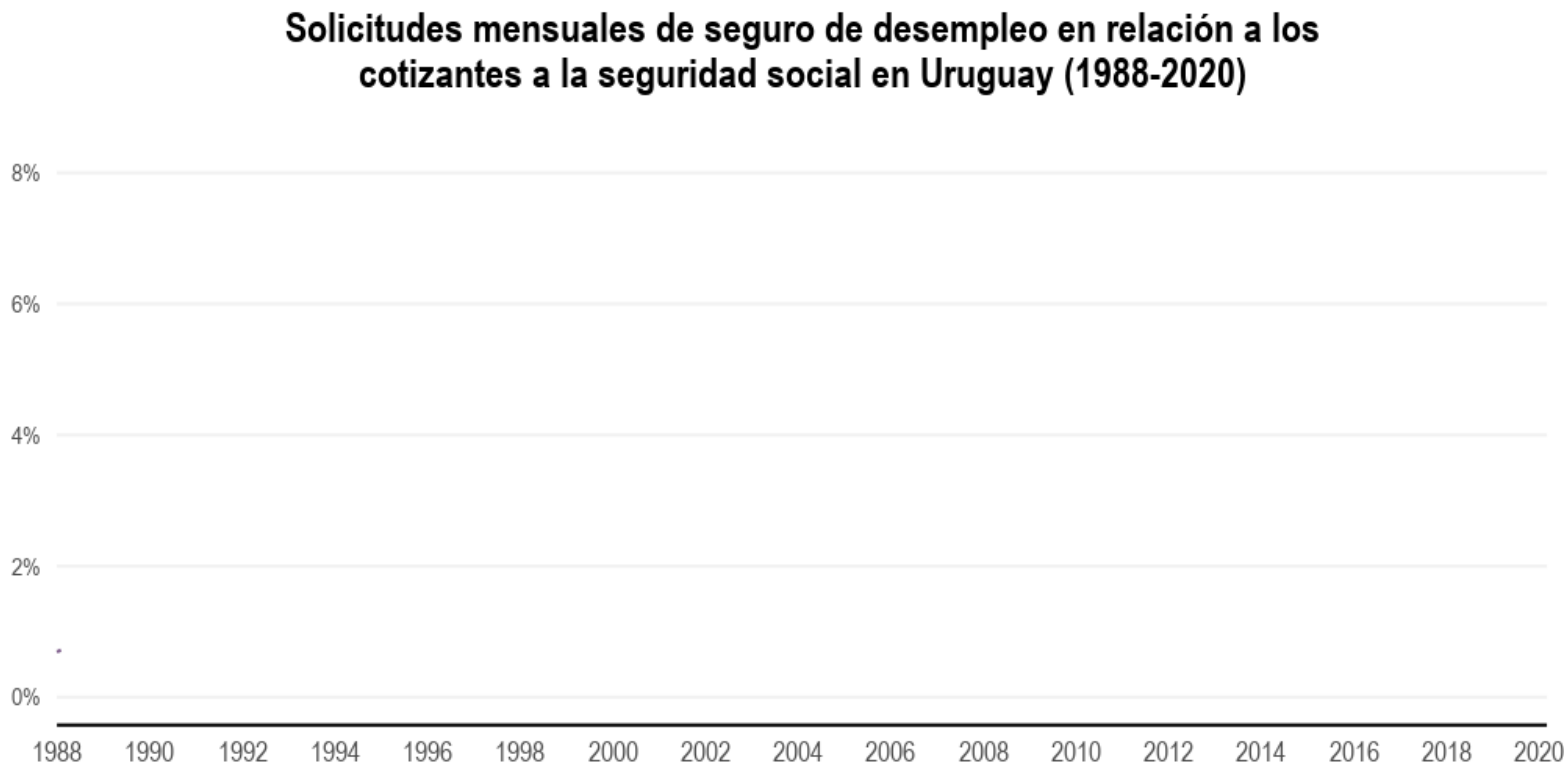


Fuente: elaboración propia en base a Spotify.
Paula Pereda - @paubgood - paulapereda.com

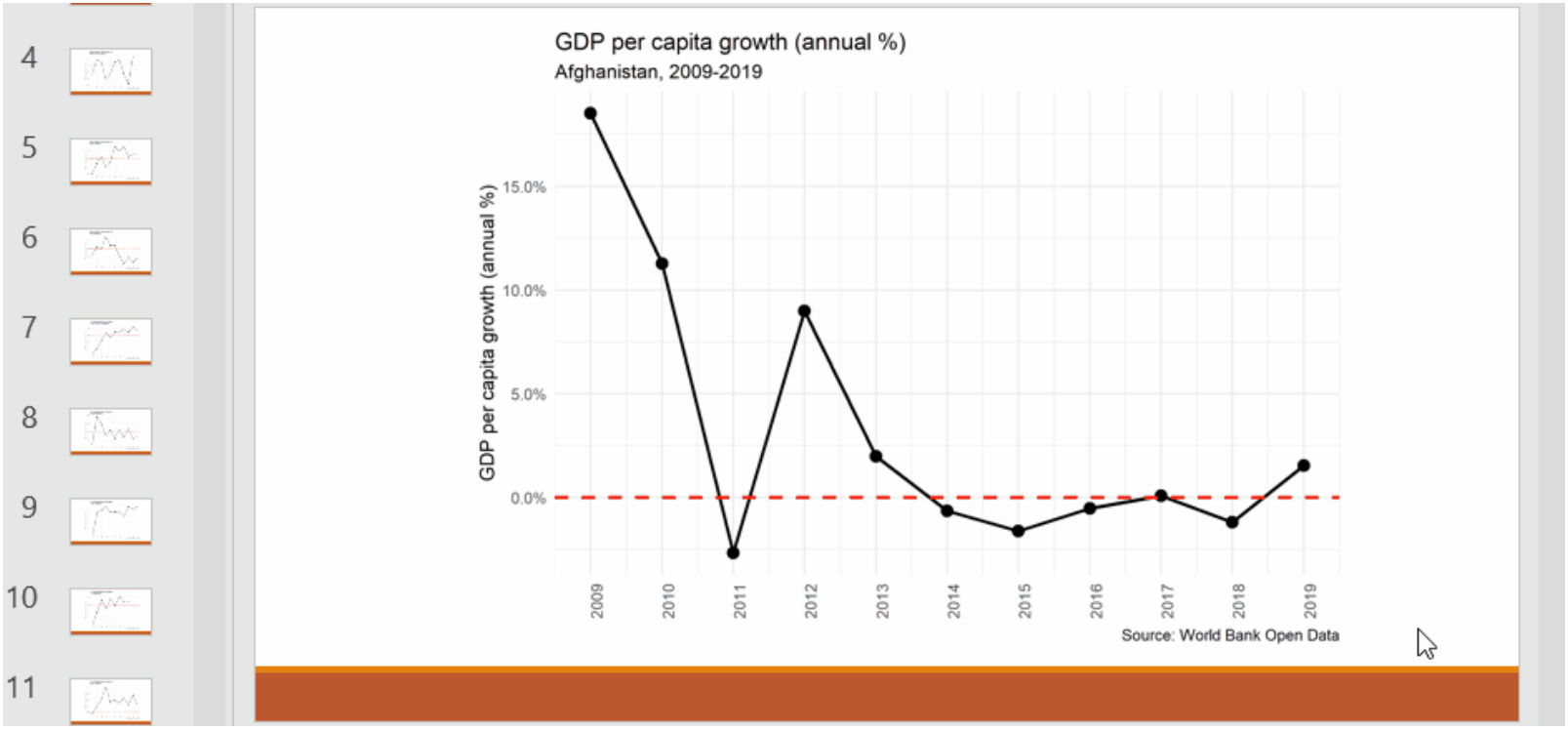
Composición del Sistema Nacional de Investigadores según nivel y sexo: mujeres y varones. Año 2020.



Fuente: elaboración propia en base a datos del SNI.
Paula Pereda - @paubgood



Fuente: elaboración propia en base a datos del BPS.
@paubgood - Paula Pereda


4
5
6
7
8
9
10
11
GDP per capita growth (annual %)
Afghanistan, 2009-2019
GDP per capita growth (annual %)
15.0%
10.0%
5.0%
0.0%
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
Source: World Bank Open Data

R

- Aplicaciones web: https://bancodedatos-fcs.shinyapps.io/OMIF-UNICEF/
- Libros: https://www.econometrics-with-r.org/

`R` + [Conceptos claves]

Descarga, actualización y R en línea

1. Descarga

- La descarga e instalación de **R** se hace desde este link: https://cran.r-project.org/
- La descarga e instalación de RStudio se hace desde este link: https://www.rstudio.com

2. Actualización de **R**

- Una manera simple desde Windows:

```
install.packages("installr")
installr::updateR()
```

- Una manera simple desde Mac:

```
devtools::install_github("AndreaCirilloAC/updateR")
updateR::updateR(admin_password = "PASSWORD") # "PASSWORD" es la contraseña del sistema
```

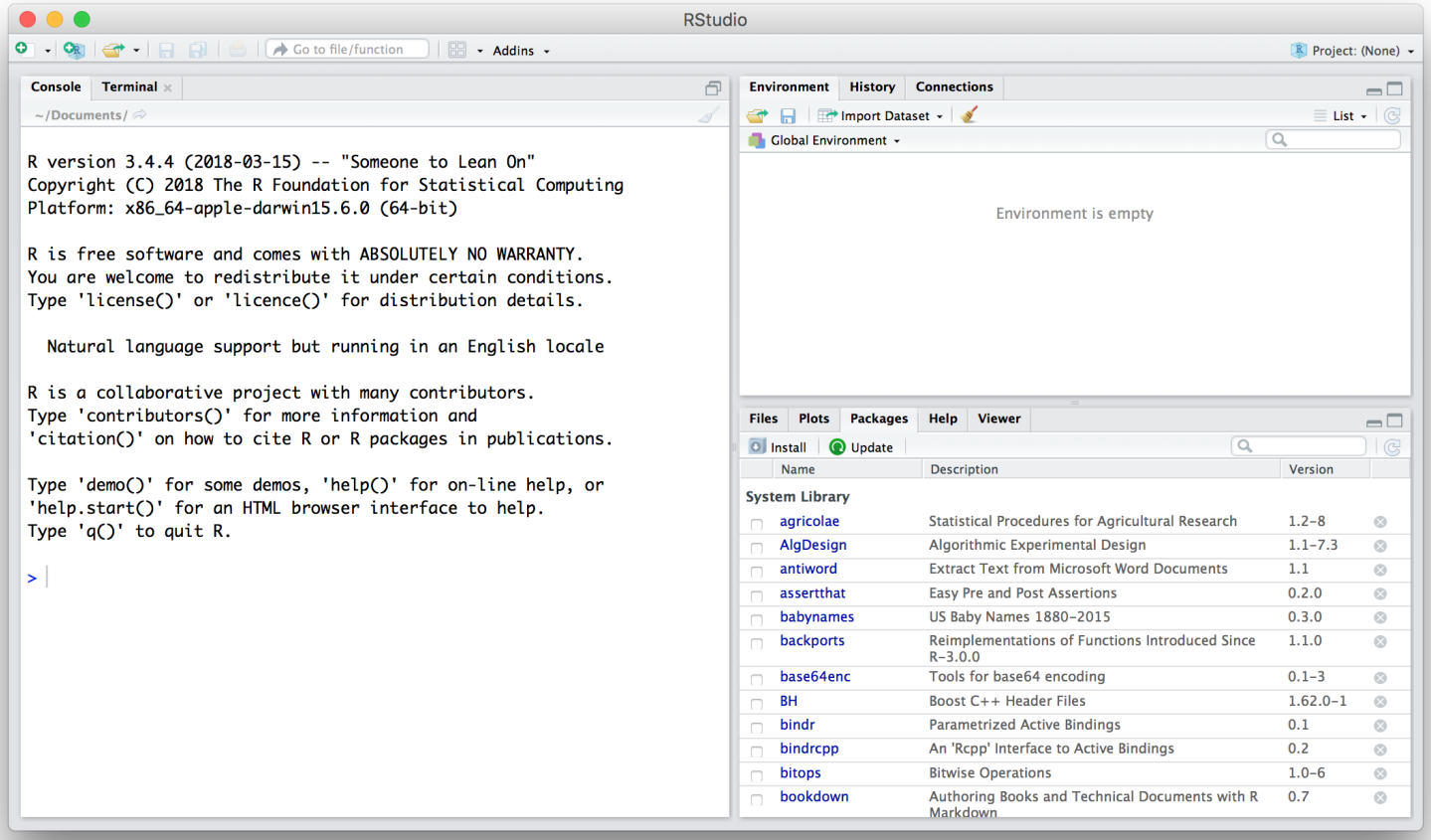
Descarga, actualización y R en línea

3. R Online

Es un RStudio que se ejecuta lento pero sirve para salir de apuros:
https://rstudio.cloud/plans/free

R

RStudio


RStudio
Go to file/function
Addins
Project: (None)
Console
Terminal
~/Documents/
R version 3.4.4 (2018-03-15) -- "Someone to Lean On"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin15.6.0 (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.
>
Environment
History
Connections
Import Dataset
List
Global Environment
Environment is empty
Files
Plots
Packages
Help
Viewer
Install
Update
Name
Description
Version
System Library
agricolae Statistical Procedures for Agricultural Research 1.2-8
AlgDesign Algorithmic Experimental Design 1.1-7.3
antiword Extract Text from Microsoft Word Documents 1.1
assertthat Easy Pre and Post Assertions 0.2.0
babynames US Baby Names 1880-2015 0.3.0
backports Reimplementations of Functions Introduced Since R-3.0.0 1.1.0
base64enc Tools for base64 encoding 0.1-3
BH Boost C++ Header Files 1.62.0-1
bindr Parametrized Active Bindings 0.1
bindrcpp An 'Rcpp' Interface to Active Bindings 0.2
bitops Bitwise Operations 1.0-6
bookdown Authoring Books and Technical Documents with R Markdown 0.7

Guía para las clases

- Un block de color gris en una diapositiva significa que se inicia código **R**. Es como si hubiera una consola de **R** en la diapositiva.
- Dentro del block gris un signo de numeral ('#') significa que se inicia un comentario.
- Dentro del block gris un signo de numeral y el signo '>' ('>') es un resultado del intérprete.

Ejemplo:

```
9 + 12 # esto es una suma
```

```
> [1] 21
```

Resultados del intérprete

```
9 + 12
```

```
> [1] 21
```

```
rnorm(15)
```

```
>  [1]  0.02952340  0.50052069 -0.82161919  0.88029585 -0.08414981  0.22278819
>  [7]  0.42358822 -0.94386505 -0.09737997 -0.03789364  0.91706278  0.68956134
> [13]  0.21660320  0.28755817  0.49089587
```

Los números que aparecen entre paréntesis rectos ([]) luego de '#>' indican la posición del resultado.

Código incompleto

Ejemplo:

```
> objeto_1 ← seq(1, 30, 0.5)
> plot(objeto_1
+
+ )
>
```

- En R el signo '>' (prompt) indica que el intérprete "está listo" para recibir órdenes.
- Si luego de dar una orden, en lugar de aparecer nuevamente el prompt, aparece el signo de '+' indica que hay una orden incompleta. En el ejemplo de arriba lo que falta es cerrar el paréntesis de la función plot().
- Si doy 'Enter' con una función incompleta va a aparecer nuevamente el signo de '+', para salir de este ciclo debo apretar 'Esc'.

Consejos para aprender

Dos consejos de Hadley Wickham para aprender y mejorar la programación en R:

1. Lean código fuente. Busquen los paquetes o las funciones que usan con más frecuencia y miren cómo están escritos.

2. Adopten una mentalidad científica. Si no comprenden cómo funciona algo, desarrollen una hipótesis, diseñen algunos experimentos, ejecútenlos y registren los resultados.

Importancia del uso de scripts

Un script es un archivo en el que se escriben los comandos y las funciones que se desean guardar de una sesión de trabajo.

- Confección: crear estos archivos es importante para tener un historial de trabajo.
- Guardado: El nombre del archivo debe ser lo sufcientemente claro como para refejar el contenido y encontrarlo con facilidad. En lo posible debe ser corto.
- Documentación: Es muy importante comentar las operaciones o creación de funciones que se realicen para poder replicar con facilidad el trabajo. Los comentarios deben iniciar con '#'

```
x ← c(1, 8, 19)
# El objeto x es un vector de tres números. Y esto es un comentario :)
```

Paquetes

Un paquete de **R** es un conjunto de funciones que pertenecen a un mismo ambiente y que -por lo común- tienen una estructura general que justifica que estén juntas.

Descarga:

```
# Descargar un paquete desde CRAN
install.packages("wooldridge")

# la función
install.packages("...")
# descarga el paquete a la computadora
# (funciones y documentación de las mismas)

# Descargar un paquete desde GitHub (versión del paquete en desarrollo)
install.packages("devtools")
devtools::install_github("JustinMShea/wooldridge")
```

Uso (cargar un paquete):

```r
# Para usar las funciones de un paquete ya descargado hay que usar la función
library( ... )
library("wooldridge")
# se puede prescindir del uso de las comillas.
# En el caso de la descarga el paquete debe estar entrecomillado
library(wooldridge)
```

Ayuda

```
help(lm) #Comando básico de búsqueda
?lm      #ídem anterior
help("+") #Ayuda sobre un operador
help.search("norm") # Busca entre los paquetes instalados las
                    # funciones que contienen el termino "norm"
??norm # ídem anterior
apropos("mean") #Listado de funciones que contienen el termino "norm"
help(rlm, package = "MASS") # Busca la ayudad de una función específica
                            # de un paquete
RSiteSearch("glm") # Busca en los manuales y ficheros de ayuda de la web
                   # R-project
find("mean") #Devuelve el paquete al que pertenece la función.
             # (el paquete debe estar cargado: library())
example(contour) # Ejecuta los ejemplos disponibles de la función
browseVignettes() # Muestra en web las viñetas disponibles y el acceso
                  # a ellas. De un paquete: (package="package-name")
vignette(all = TRUE) # Muestra todas las viñetas disponibles en pantalla
```

Ejemplo:

```
help(mean)
```

Ficha técnica de una función

- Name: El nombre de la función, entre llaves '{}' el paquete al que pertenece la función
- Description: Breve descripción de la función
- Usage: Sintaxis de la función
- Arguments: Explicación de los argumentos de la función
- Value: Características de la salida de la función (puede ser un objeto como un valor)
- References: Bibliografía relacionada con la función
- See Also: Funciones relacionadas.
- Examples: Ejemplos de cómo se usa la función

Uso de funciones

Una función en R es una o varias sentencias de código que realizan una operación determinada.

Una función tiene dos características fundamentales para su correcto uso: un nombre y argumentos que siempre van entre paréntesis.

Ejemplo: función para obtener el promedio de un conjunto de datos

mean(x, trim = 0, na.rm = FALSE, ...)

- Nombre de la función: mean
- Argumentos de la función: x, trim, na.rm, ...

De los 4 argumentos hay dos que tienen un valor asignado mediante el signo '='. Estos son valores por defecto. Si el usuario no los modifica la función siempre va a asumir esos valores.

Uso de funciones

Cuando se usa una función:

- Los argumentos que tienen valores por defecto pueden no llamarse cuando se ejecuta la función.
- Los argumentos tienen un orden que resulta importante únicamente si se opta por no nombrarlos.
- Los argumentos pueden ir en distinto orden siempre que sean nombrados y asignados.

Ejemplo: todas estas opciones generan el mismo resultado

```
mean(x = 1:10)
mean(1:10)
mean(x = c(1:10, NA), trim = 0, na.rm = TRUE)
mean(trim = 0, x = c(1:10, NA), na.rm = TRUE)
mean(c(1:10, NA), 0, TRUE)
```

Objetos

En R se crea un objeto con el asignador '<-'. También se puede usar '=' pero no es recomendable.

nombre_del_objeto <- contenido_del_objeto

Palabras reservadas: if, break, next, TRUE, NaN, else, for, NULL, Inf, NA, FALSE, function, repeat, while, in

- **Importante:** el nombre de un objeto no puede iniciar con números, ni signos, ni puede tener espacios en blanco.
- Relevante: R es sensible a minúsculas y mayúsculas.
- Sugerencia: evitar nombrar objetos con nombres de funciones existentes.

Tipos de objetos: typeof()

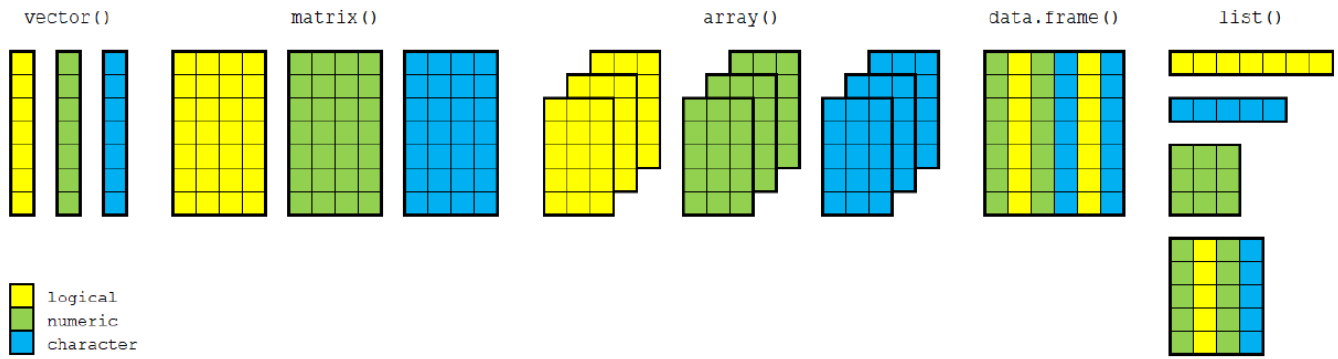
Los tipos de objetos más frecuentes son estos:

typeof*	Descripción	Ejemplo
logical	un vector que contiene valores lógicos	a <- c(TRUE, FALSE, TRUE)
double	un vector que contiene valores reales	a <- seq(1:10, 0.3)
character	un vector que contiene valores de caracteres	a <- letters[1:5]

Estructuras de datos en R

La estructura fundamental en R es el vector. El resto de las estructuras se pueden pensar como combinaciones de vectores.

- matrix() y array(): una matriz o un arreglo es un vector con el atributo dim() (dimensión).
- data.frame(): uno o más vectores del mismo largo (length()) que en el caso de un marco de datos sería con la misma cantidad de las (nrow())
- list(): un lista es un vector genérico. Cada elemento de una lista puede ser de distinto tipo.



R + [Aplicaciones]