

Introducción a R

Libro de texto

Ana Cecilia Plavan, Andrés Andruskevicius



Sección 1: Introducción a R

- Definición de objetos, realización de operaciones lógicas y aritmética básica.
- Utilizar funciones pre definidas para realizar operaciones sobre objetos
- Distinguir entre diferentes tipos de datos



El curso tiene variadas prácticas y un trabajo final en cada sección para evaluar las habilidades de codificación adquiridas.

Instalar R

- Descargar e instalar R:

<https://cran.r-project.org/>

Descargar la última versión del subdirectorio Básico:

R for Windows

Subdirectories:

base	Binaries for base distribution. This is what you want to install R for the first time .
contrib	Binaries of contributed CRAN packages (for R >= 2.13.x; managed by Uwe Ligges). There is also information on third party software available for CRAN Windows services and corresponding environment and make variables.
old contrib	Binaries of contributed CRAN packages for outdated versions of R (for R < 2.13.x; managed by Uwe Ligges).
Rtools	Tools to build R and R packages. This is what you want to build your own packages on Windows, or to build R itself.

Please do not submit binaries to CRAN. Package developers might want to contact Uwe Ligges directly in case of questions / suggestions related to Windows binaries.

You may also want to read the [R FAQ](#) and [R for Windows FAQ](#).

Note: CRAN does some checks on these binaries for viruses, but cannot give guarantees. Use the normal precautions with downloaded executables.

base subdirectory

Probar R

- Hay muchos paquetes para utilizar en R
- Para el curso utilizaremos los siguientes
- Para descargar el paquete dslabs desde la consola ingresamos:

```
> install.packages("dslabs") # instalar solo 1  
paquete
```

```
➤ install.packages(c("tidyverse", "dslabs"))
```

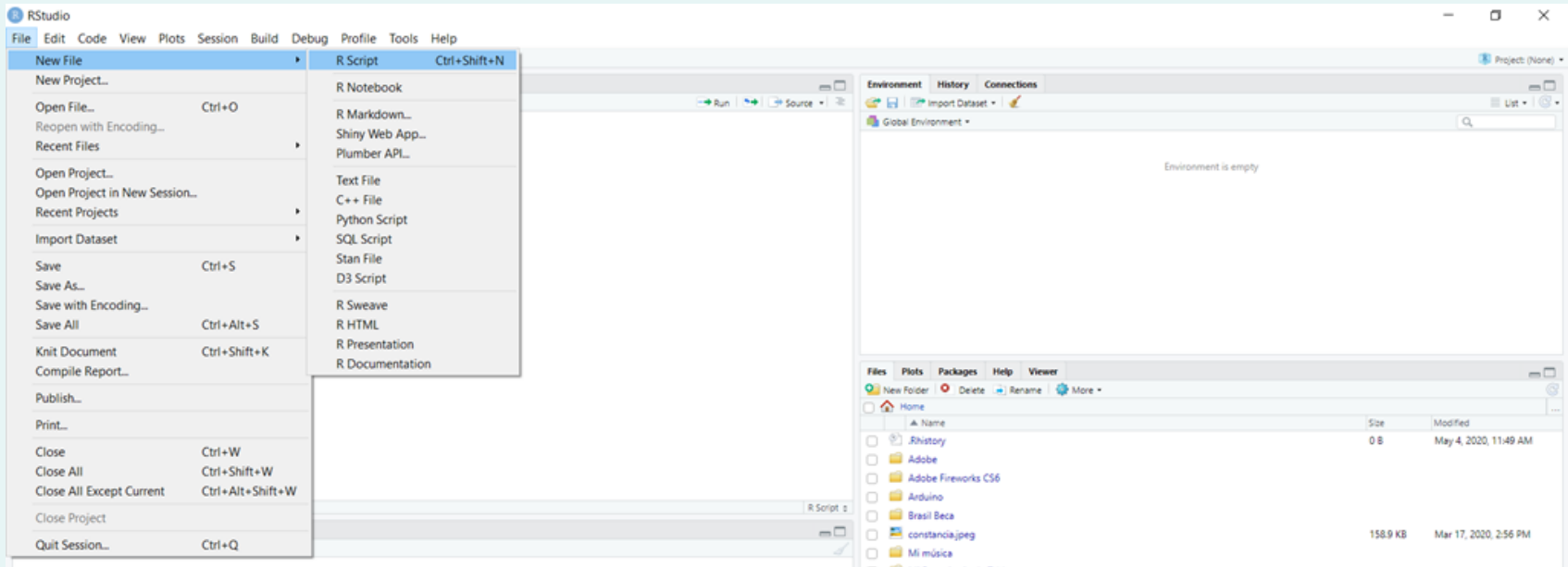
```
➤ # instalar dos paquetes al mismo tiempo
```

```
> installed.packages() # ver la lista de paquetes  
instalados
```

Instalar RStudio

- Descargar la opción gratis de escritorio:
<https://rstudio.com/products/rstudio/download/>
- Seleccionar el sistema operativo en la descarga
- Seguir la instalación dejando los parámetros por defecto.

Probar RStudio



- También podemos descargar un paquete en RStudio desde Tools > Install Packages (permite autocompletar)
- Para cargar en Rstudio y poder utilizar un paquete descargado debemos ejecutar:

```
library(dslabs)
```

Primera prueba en RStudio: Escribir comandos y editar scripts

- Creamos un nuevo script
- Lo guardamos para ingresarle nombre
- Cargamos para utilizar las librerías descargadas:

```
library(tidyverse)
```

```
library(dslabs)
```

- Luego seguimos escribiendo código, por ejemplo vamos a hacer mostrar un gráfico de total de asesinatos versus población por estado

Ejecutar el código

Para ejecutar y poder visualizar el gráfico programado clickeamos en el botón Run:

```
library(tidyverse)
library(dslabs)
data(murders)

murders %>%
  ggplot(aes(population, total, label=abb, color=region))
+
  geom_label()
```

Archivo abierto

```
codigoEjemplo1.R x
Source on Save
1 library(tidyverse)
2 library(dslabs)
3 murders %>%
4   ggplot(aes(population, total,
5             geom_label()
6
7
```

Código

Run the current line
or selection
(Ctrl+Enter)

```
Console Terminal x Jobs x
~/
+ geom_label()
> source('~/.active-rstudio-document', echo=TRUE)
> library(tidyverse)
> library(dslabs)
> murders %>%
+   ggplot(aes(population, total, label=abb,
+             color=region)) +
+   geom_label()
> source('~/.active-rstudio-document')
>
```

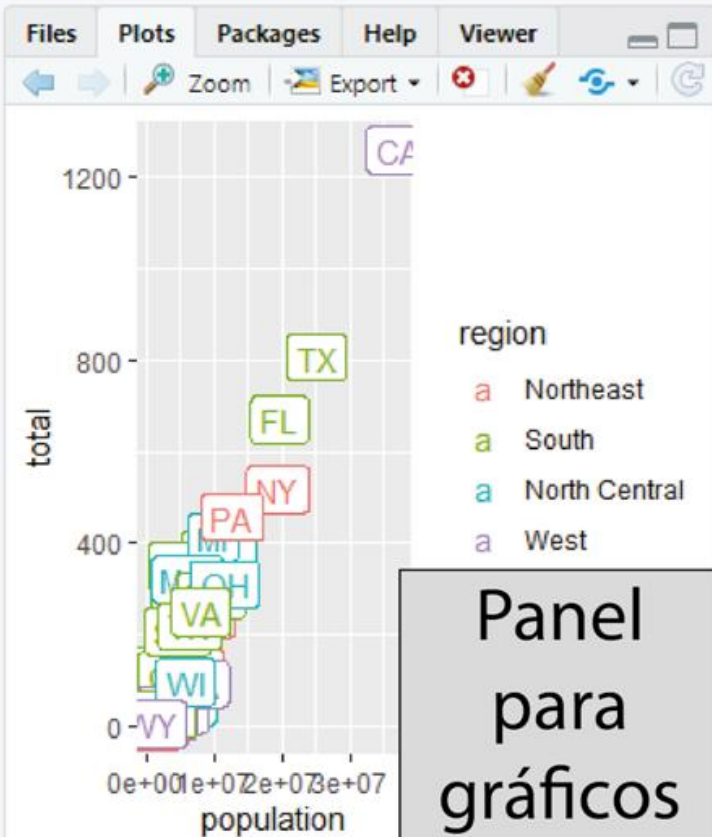
Consola

Environment History Connections

Import Dataset

Environment

Environment is empty



Panel
para
gráficos

Conceptos básicos - Variables

Cuando resolvemos ecuaciones como:

$$x^2+bx+c=0$$

Sabemos que la solución es: $x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$

Asignamos valores a variables de la forma:

```
a <- 1
```

```
b <- 1
```

```
c <- -1
```

Luego resolvemos la ecuación escribiendo:

```
(-b + sqrt(b^2 - 4*a*c)) / (2*a)
```

```
(-b - sqrt(b^2 - 4*a*c)) / (2*a)
```

Nombre de variables

- Deben comenzar con una letra y no contener espacios
- No podemos crear variables con nombres ya utilizados en funciones o objetos predeterminadas en R por ejemplo no podemos nombrar una variable `install.packages`
- Como consejo elegir nombre de variables intuitivos, asociados al contenido que estamos calculando

```
solucion1<-(-b + sqrt(b^2 - 4*a*c)) / (2*a)
```

```
solucion2<-(-b + sqrt(b^2 - 4*a*c)) / (2*a)
```

Funciones

- Un proceso de análisis de datos es una serie de funciones aplicada a los datos
- Para evaluar una función escribimos el nombre de la función con los valores con la cual la queremos evaluar entre paréntesis como argumento:

```
> log(8)  
[1] 2.079442
```

- Si no agregamos argumento vemos el código que define a la función:

```
> log  
function (x, base = exp(1)) .Primitive("log")
```

The screenshot shows the RStudio environment. The top menu bar includes 'Files', 'Plots', 'Packages', 'Help', and 'Viewer'. The 'Help' menu is open, displaying 'R: Logarithms and Exponentials' and a search bar. The main pane shows the 'log {base}' documentation page, which includes a 'Description' section explaining the `log` function and its variants (`log1p`, `exp`, `expm1`). The left pane shows the 'Console' window with the command `> help("log")` entered, and a prompt character `>` on the next line.

```
> help("log")
> |
```

R Documentation

Logarithms and Exponentials

Description

`log` computes logarithms, by default natural logarithms, `log10` computes common (i.e., base 10) logarithms, and `log2` computes binary (i.e., base 2) logarithms. The general form `log(x, base)` computes logarithms with base `base`.

`log1p(x)` computes $\log(1+x)$ accurately also for $|x| \ll 1$.

`exp` computes the exponential function.

`expm1(x)` computes $\exp(x) - 1$ accurately also for $|x| \ll 1$.

Usage

- En caso de no terminar de escribir una función la consola mostrará un símbolo de + en espera del contenido faltante.

```
> help(
+ |
```

Argumentos de funciones

- Las funciones reciben como entrada argumentos o datos para realizar el proceso o cálculo

```
> args(log)
function (x, base = exp(1))
NULL
```

- Algunos son obligatorios y otros opcionales

```
> help("log")
```

Arguments

x a numeric or complex vector.

base a positive or complex number: the base with respect to which logarithms are computed. Defaults to `e=exp(1)`.

Por defecto si no hay segundo argumento, la función log calcula el logaritmo natural

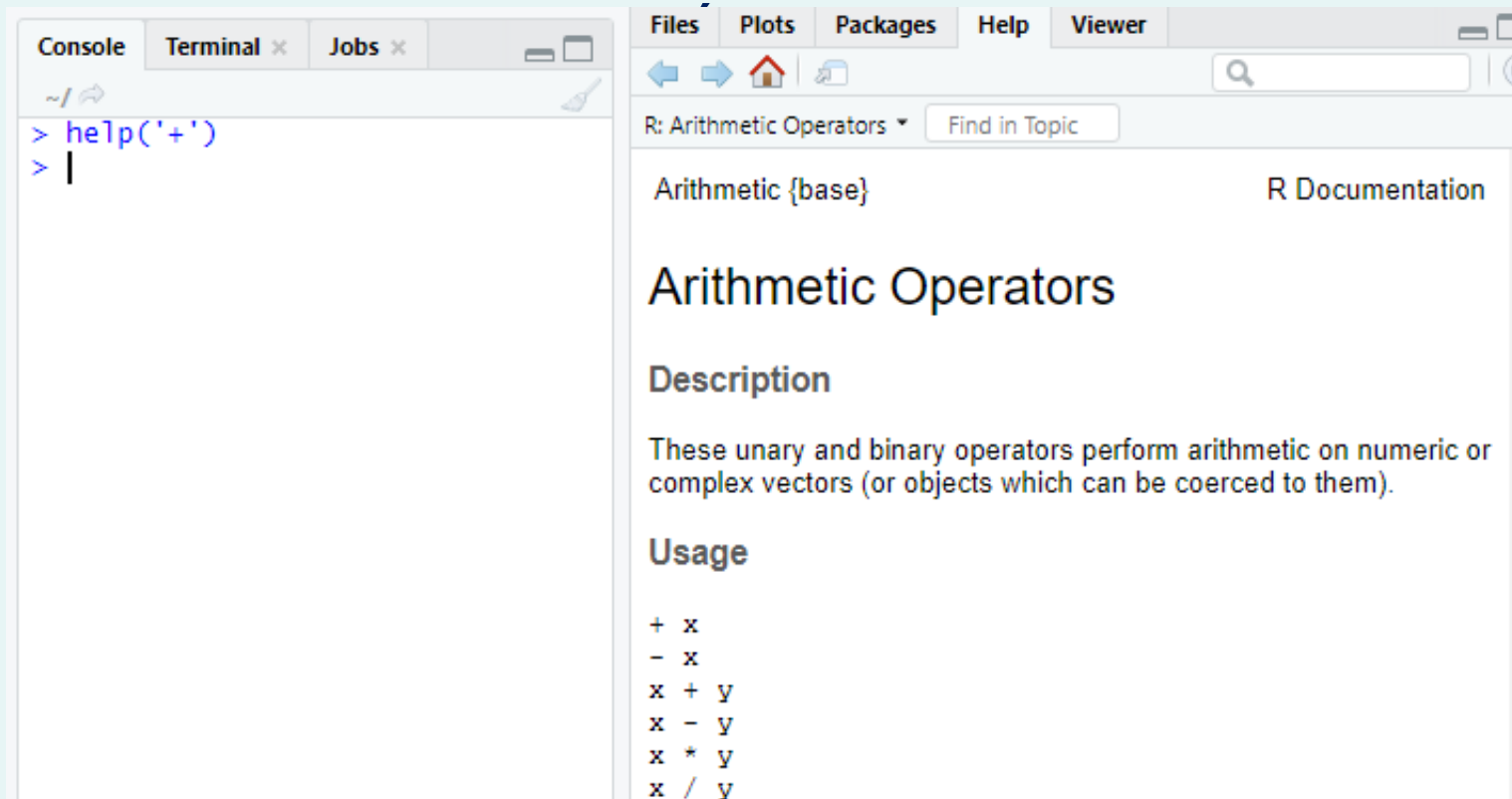
Funciones anidadas

- Una función está anidada si llama a otra función para obtener el parámetro que necesita como argumento.

```
> exp(1)
[1] 2.718282
> log(2.718282)
[1] 1
> log(exp(1))
[1] 1
```


Funciones especiales

- Algunas funciones no necesitan los () para ejecutarse y recibir la entrada de datos, por ejemplo si calculamos 2^3
- Podemos ver todos los operadores



The screenshot shows the R Studio interface. On the left, the Console pane displays the command `> help('+')` followed by a prompt `> |`. On the right, the Help pane shows the documentation for 'Arithmetic Operators' from the 'Arithmetic {base}' package. The documentation includes a description of unary and binary operators and a list of usage examples.

Console:

```
> help('+')
> |
```

Help Pane: R: Arithmetic Operators

Arithmetic {base} R Documentation

Arithmetic Operators

Description

These unary and binary operators perform arithmetic on numeric or complex vectors (or objects which can be coerced to them).

Usage

```
+ x
- x
x + y
x - y
x * y
x / y
```

Data sets

- Además de funciones hay más objetos definidos por defecto en R, por ejemplo data sets
- Data sets son objetos predefinidos a los cuales accedemos solo al escribir su nombre

```
> co2
```

	Jan	Feb	Mar
1959	315.42	316.31	316.50
1960	316.27	316.81	317.42
1961	316.73	317.54	318.38
1962	317.78	318.40	319.53
1963	318.58	318.92	319.70
1964	319.41	320.07	320.74
1965	319.27	320.28	320.73

```
> Inf  
[1] Inf  
> pi  
[1] 3.141593
```

Comentarios

- Asociar nombres de variables intuitivos no es solo la única manera de simplificar la lectura del código.
- Podemos agregar texto con comentarios que no se ejecuten en el programa con el símbolo #

```
# Asignando valores a variables
a <- 1
b <- 1
c <- -1
# Resolviendo la ecuación
solucion_1<- (-b + sqrt(b^2 - 4*a*c)) / (2*a)
solucion_2<- (-b - sqrt(b^2 - 4*a*c)) / (2*a)
```