

Introducción a R

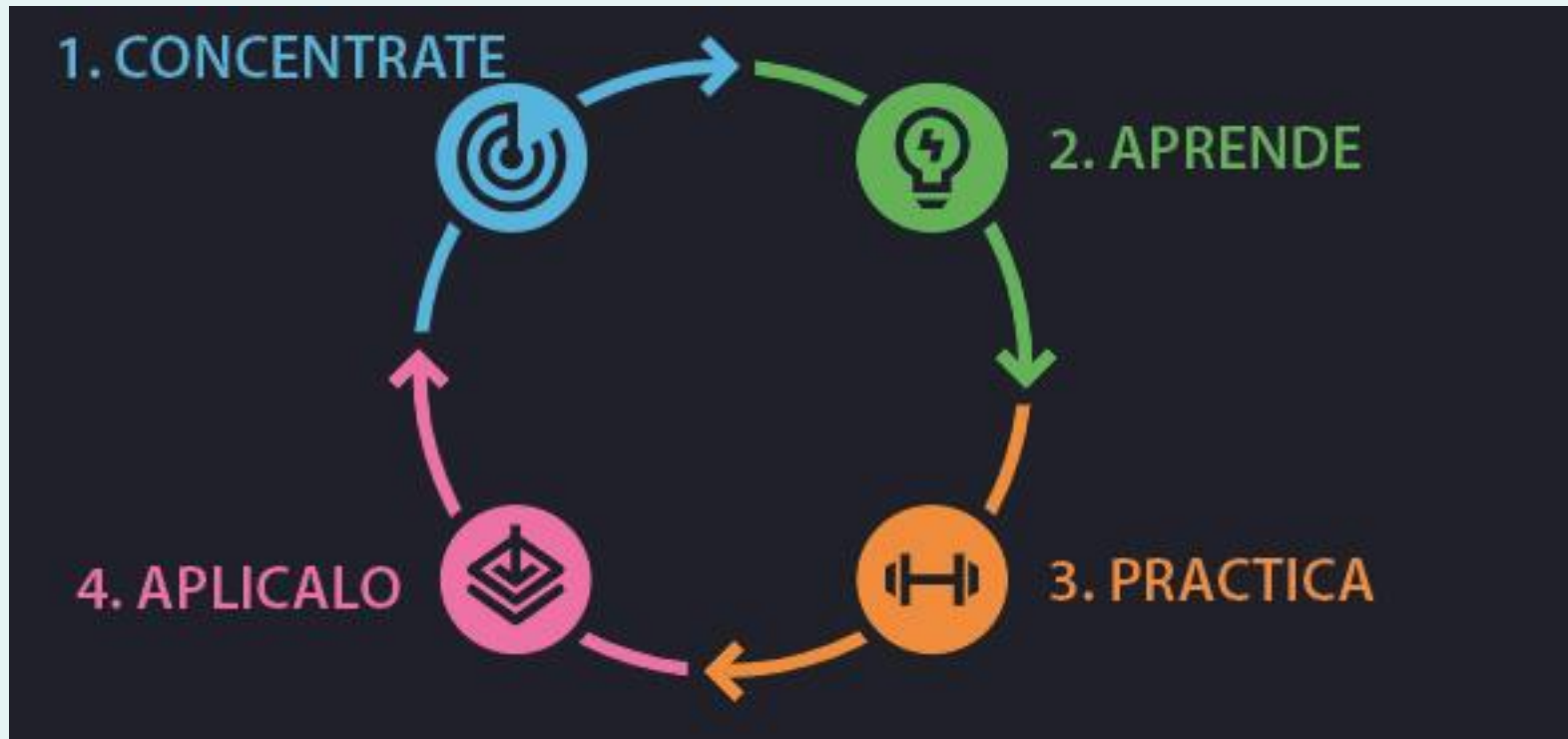
Libro de texto

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](#)

[contrib](#)

Binaries for base distribution. This is what you want to [install R for the first time](#).

Binaries of contributed CRAN packages (for R \geq 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Binaries of contributed CRAN packages for outdated versions of R (for R 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:

```
> # instalar solo 1 paquete  
> install.packages("dslabs")  
> # instalar dos paquetes al mismo tiempo  
> install.packages(c("tidyverse", "dslabs"))  
> # ver la lista de paquetes instalados  
> installed.packages()
```

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.

Posit Cloud

posit.cloud/spaces/262515/content/4266823

Prueba / pruebas

RAM

⚙️

⋮

AC

⬆️

RStudio cloud

FileEditCodeViewPlotsSessionBuildDebugProfileToolsHelp

Go to file/function

Addins

R 4.2.3

Untitled3*ej4_sol.Rej2.Rej1.RejercicioN1y2S3

RunSource

```
1 #Carga las librerías a utilizar
2 library("stringr")
3 library("htmltools")
4 #Para este ejercicio probaremos el carácter especial.
5 #En el lenguaje regex este elemento puede machear cualquier elem
6 #es como el concepto de un comodín en un mazo de cartas, el cual
7 #sustituir a cualquier carta del mazo.
8 #El metacaracter . puede representar a cualquier letra, símbolo,
9
10 #Si creamos una variable con el contenido:
11 prueba<-c('hola.','533.','0$%&.','#$56.')
12 #Y buscamos utilizando el patrón:
13 str_view_all(prueba, ".")
14
```

10:31 (Top Level)

R Script

ConsoleTerminalBackground Jobs

R 4.2.3 . /cloud/project/

1969 323.83 324.26 325.47 326.50 327.21 326.54 325.72 323.50 322.22 3
21.62 322.69 323.95
1970 324.89 325.82 326.77 327.97 327.91 327.50 326.18 324.53 322.93 3
22.90 323.85 324.96
1971 326.01 326.51 327.01 327.62 328.76 328.40 327.20 325.27 323.20 3
23.40 324.63 325.85
1972 326.60 327.47 327.58 329.56 329.90 328.92 327.88 326.16 324.68 3
25.04 326.34 327.39
1973 328.37 329.40 330.14 331.33 332.31 331.90 330.70 329.15 327.35 3
27.02 327.99 328.48

EnvironmentHistoryConnectionsGitT

415 MiB

List

RGlobal Environment

Data

admissions	12 obs. of 4 variables
avg	75 obs. of 2 variables
AwardsPla...	80 obs. of 6 variables
AwardsPla...	80 obs. of 4 variables
co2_tidy	468 obs. of 3 variables
co2_wide	39 obs. of 13 variables

FilesPlotsPackagesHelpViewerPres

R: Logarithms and ExponentialsFind in Topic

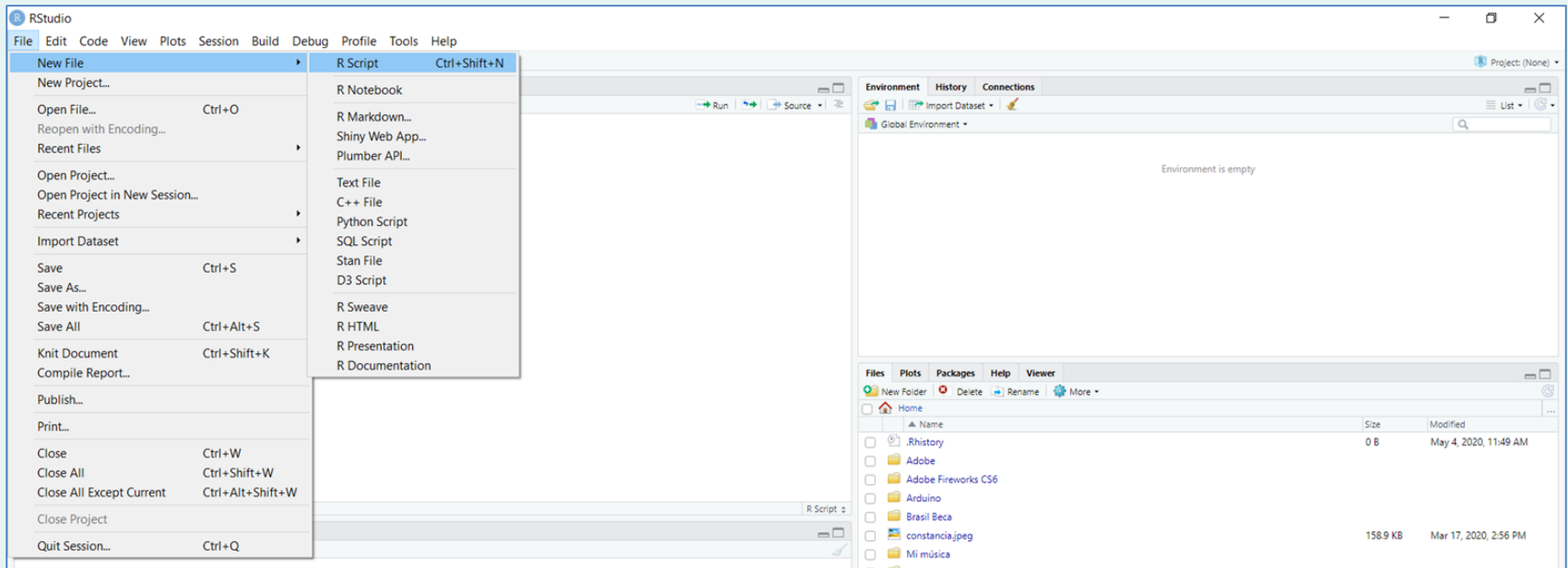
log {base}R Documentation

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

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
- Para utilizar las librerías descargadas, escribimos:

```
library(tidyverse)  
library(dslabs)
```

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

Ejecutar el código

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

```
library(dslabs)  
data(murders)  
plot(murders$population,murders$total)
```

Archivo abierto

```
codigoEjemplo1.R x
1 library(tidyverse)
2 library(dslabs)
3 murders %>%
4   ggplot(aes(population, total, label=abb,
5             color=region)) +
6   geom_label()
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')
> |
```

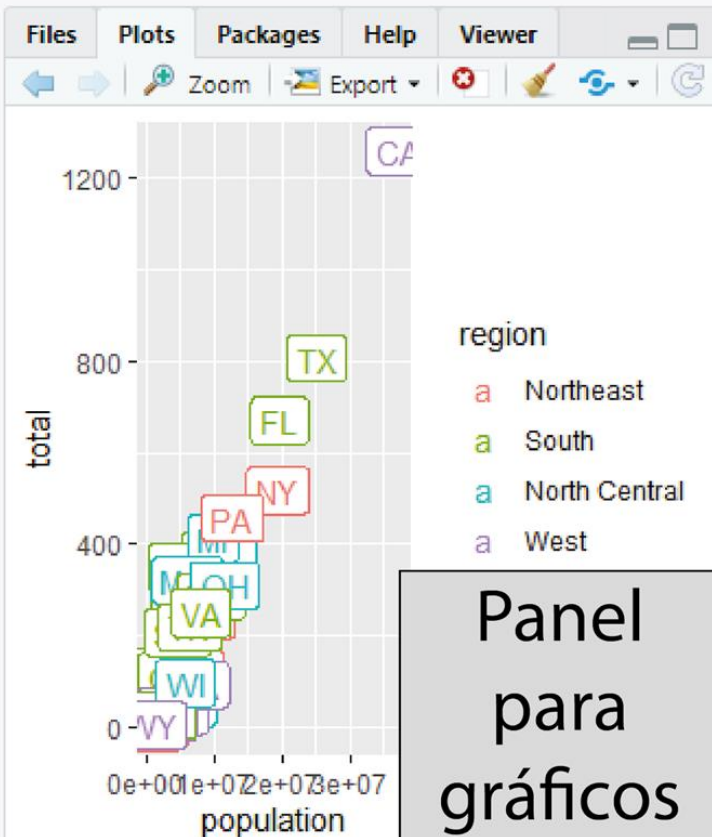
Consola

Environment History Connections

Import Dataset

Environment

Environment is empty



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 interface. The top menu bar includes 'Files', 'Plots', 'Packages', 'Help', and 'Viewer'. The 'Console' pane on the left shows the command `> help("log")` followed by a prompt `> |`. The 'Viewer' pane on the right displays the R documentation for the `log` function, titled 'Logarithms and Exponentials'. The documentation includes a 'Description' section with the following text: `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 $|x| \ll 1$.
The 'Usage' section is also visible.

- 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

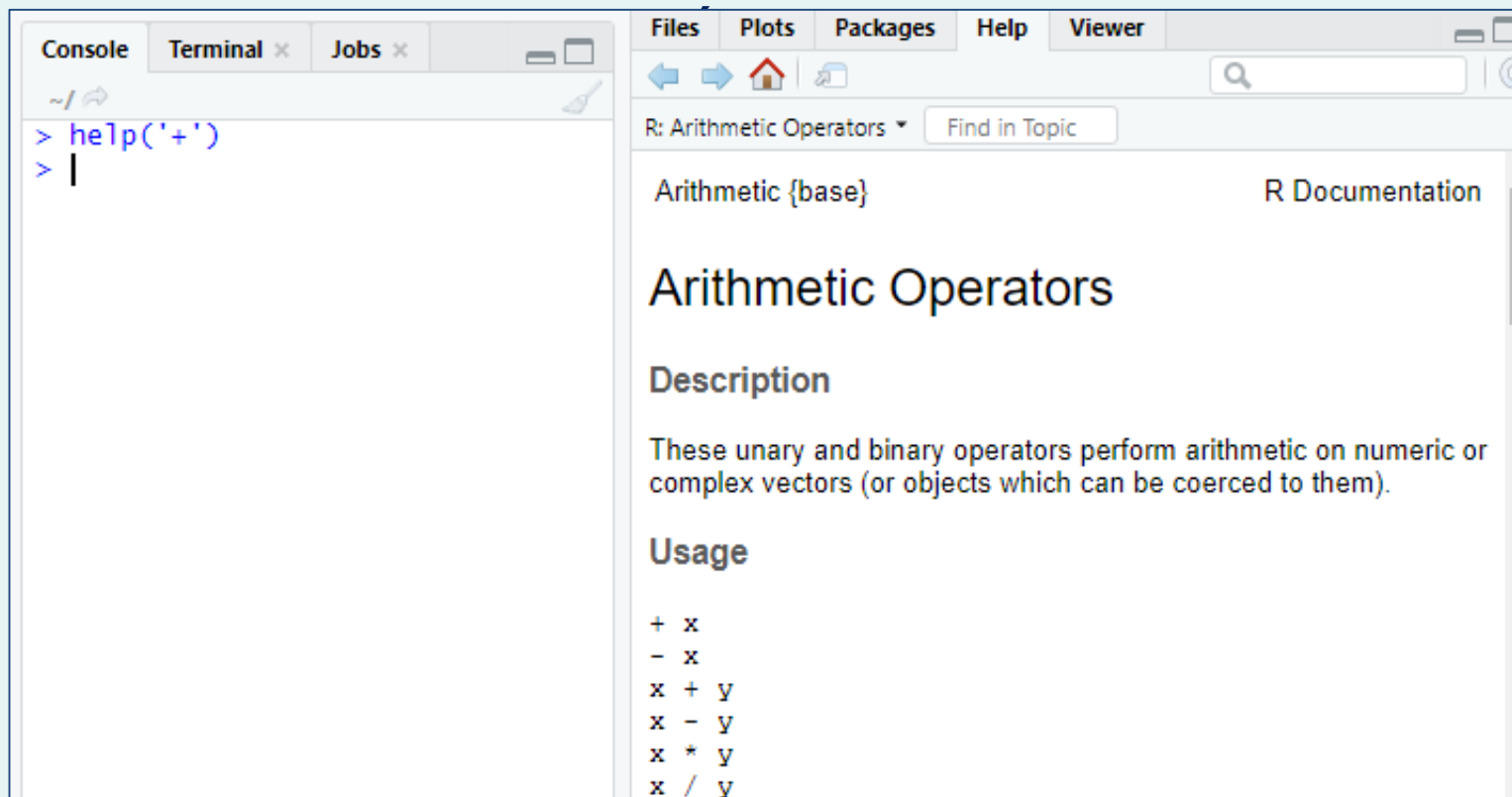
Funciones anidadas

- 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 RStudio interface with the 'Console' tab selected. In the console, the command `> help('+')` has been entered, and the cursor is on the next line. The 'Viewer' tab is also open, displaying the 'R: Arithmetic Operators' help page. The page title is 'Arithmetic {base}' and 'R Documentation'. The main heading is 'Arithmetic Operators'. Under the 'Description' section, it states: 'These unary and binary operators perform arithmetic on numeric or complex vectors (or objects which can be coerced to them)'. Under the 'Usage' section, the following operators are listed:

```
+ 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)
```