

Descriptive analysis

Descriptive analysis and basic statistics in biomedical studies
using R and Markdown

Juan R Gonzalez
juanr.gonzalez@isglobal.org

BRGE - Bioinformatics Research Group in Epidemiology
ISGlobal - Barcelona Institute for Global Health
<http://brge.isglobal.org>

IACS - Instituto Aragonés de Ciencias de la Salud
Zaragoza, February 26th

Getting started

Installing R and RStudio

- ▶ R: <https://cran.rstudio.com/>
- ▶ RStudio:
<https://www.rstudio.com/products/rstudio/download/>

RStudio

Installing R and RStudio

Working with R is primarily text-based. The basic mode of use for R is that the user types in a command in the R language and presses enter, and then R computes and displays the result.

We will be working in RStudio. This surrounds the *console*, where one enters commands and views the results, with various conveniences. In addition to the console, RStudio provides panels containing:

- ▶ A *text editor*, where R commands can be recorded for future reference.
- ▶ A history of commands that have been typed on the console.
- ▶ An “environment” panel with a list of *variables*, which contain values that R has been told to save from previous commands.
- ▶ A file manager.
- ▶ Help on the functions available in R.
- ▶ A panel to show plots (graphs).

RStudio screen

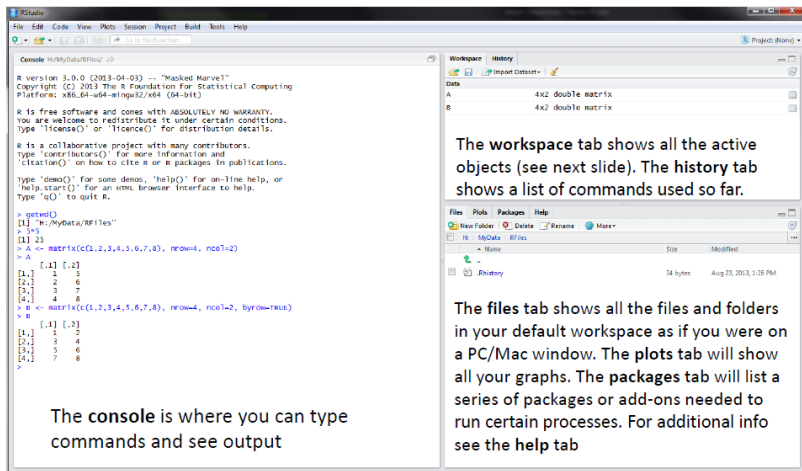


Figure 1: Rstudio screen

Workspace tab (1)

The workspace tab stores any object, value, function or anything you create during your R session. In the example below, if you click on the dotted squares you can see the data on a screen to the left.

The screenshot shows the RStudio interface. The top pane displays the R script editor with the following code:

```
1 betwd()
2 setwd("h:/mydata/Rfiles")
3 getwd()
4 5*5
5 A <- matrix(c(1,2,3,4,5,6,7,8), nrow=4, ncol=2)
6 A
7 B <- matrix(c(1,2,3,4,5,6,7,8), nrow=4, ncol=2, byrow=TRUE)
8 B
```

The bottom pane shows the Workspace tab with the following objects:

Object	Type
A	4x2 double matrix
B	4x2 double matrix
house.pets	3 obs. of 4 variables

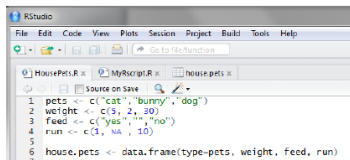
Red arrows indicate the relationship between the objects in the Workspace tab and the code in the script editor. One arrow points from the 'A' object to the code that creates matrix A. Another arrow points from the 'B' object to the code that creates matrix B. A third arrow points from the 'house.pets' object to the code that creates the 'house.pets' object.

Showing here matrix B. To see matrix A click on the respective tab.

Figure 2: Workspace tab

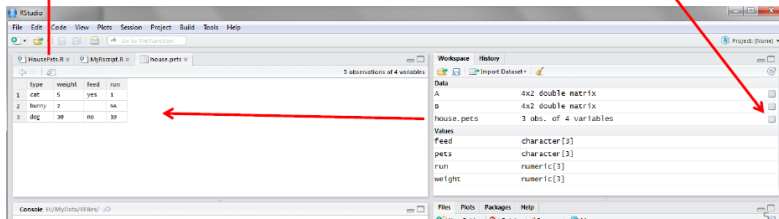
Workspace tab (2)

Here is another example on how the workspace looks like when more objects are added. Notice that the data frame `house.pets` is formed from different individual values or vectors.



```
1 pets <- c("cat", "bunny", "dog")
2 weight <- c(5, 2, 30)
3 feed <- c("yes", "", "no")
4 run <- c(1, NA, 10)
5
6 house.pets <- data.frame(type=pets, weight, feed, run)
7
```

Click on the dotted square to look at the dataset in a spreadsheet form.



The screenshot shows the RStudio interface with the workspace tab selected. The workspace contains the following objects:

Object	Type
A	4x2 double matrix
B	4x2 double matrix
house.pets	3 obs. of 4 variables
feed	character[3]
pets	character[3]
run	numeric[3]
weight	numeric[3]

The console shows the command: `My/MyData/Rfiles/`

A red arrow points from the text "Click on the dotted square to look at the dataset in a spreadsheet form." to the dotted square next to the `house.pets` object in the workspace. Another red arrow points from the `house.pets` object to the spreadsheet view of the data frame.

	type	weight	feed	run
1	cat	5	yes	1
2	bunny	2	NA	
3	dog	30	no	10

DSS/OTR

Figure 3: Workspace tab (cont.)

History tab

The history tab keeps a record of all previous commands. It helps when testing and running processes. Here you can either **save** the whole list or you can **select** the commands you want and send them to an R script to keep track of your work.

In this example, we select all and click on the “To Source” icon, a window on the left will open with the list of commands. Make sure to save the ‘untitled1’ file as an *.R script.

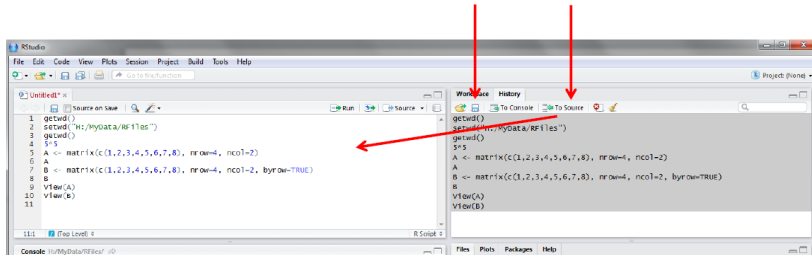


Figure 4: History tab

R basics

The console

Open RStudio, click on the “Console” panel, type `1+1` and press enter. R displays the result of the calculation. In this document, we will be showing such an interaction with R as below.

```
1+1
```

```
[1] 2
```

`+` is called an operator. R has the operators you would expect for basic mathematics: `+` `-` `*` `/` `^`. It also has operators that do more obscure things.

`*` has higher precedence than `+`. We can use brackets `()`, if necessary. Try `1+2*3` and `(1+2)*3`.

Spaces can be used to make code easier to read.

Comparisons

We can compare with `==` `<` `>` `<=` `>=`. This produces a “logical” value, `TRUE` or `FALSE`. Note the double equals, `==`, for equality comparison.

```
2 * 2 == 4
```

```
[1] TRUE
```

There are also character strings such as `"string"`.

Variables

A variable is a name for a value, such as `x`, `current_temperature`, or `subject.id`. We can create a new variable by assigning a value to it using `<-`.

```
weight_kg <- 55
```

RStudio helpfully shows us the variable in the “Environment” panel. We can also print it by typing the name of the variable and hitting enter. In general, R will print to the console any object returned by a function or operation *unless* we assign it to a variable.

```
weight_kg
```

```
[1] 55
```

Examples of valid variable names: `hello`, `hello_there`, `hello.there`, `value1`. Spaces aren't ok *inside* variable names. Dots (`.`) are ok, unlike in many other languages.

Aritmetics

We can do arithmetic with the variable:

```
# weight in pounds:  
2.2 * weight_kg
```

[1] 121

NOTE: *It is highly recommended writing in scripts (File -> New File -> R script) - not in the console. There are tabs and keys to facilitate code execution (Ctrl + Intro).*

We can add comments to our code using the # character. It is useful to document our code in this way so that others (and us the next time we read it) have an easier time following what the code is doing.

Vectors

A *vector* of numbers is a collection of numbers. We call the individual numbers *elements* of the vector. We can make vectors with `c()`, for example `c(1,2,3)`. `c` means “combine”. In R, numbers are just vectors of length one. R is a ‘vectorize’ language

```
myvec <- c(10,20,30,40,50)
myvec + 1
```

```
[1] 11 21 31 41 51
```

```
myvec + myvec
```

```
[1] 20 40 60 80 100
```

Vectors (2)

```
c(myvec, myvec)
```

```
[1] 10 20 30 40 50 10 20 30 40 50
```

```
c(60, myvec)
```

```
[1] 60 10 20 30 40 50
```

```
length(myvec)
```

```
[1] 5
```

When we talk about the length of a vector, we are talking about the number of numbers in the vector.

Types of vector

We will also encounter vectors of character strings, for example "hello" or c("hello", "world"). Also we will encounter “logical” vectors, which contain TRUE and FALSE values. R also has “factors”, which are categorical vectors, and behave very much like character vectors (think the factors in an experiment).

A categorical vector, where the elements can be one of several different “levels”. There will be more on these in other sections.

```
factor(c("mutant", "wildtype", "mutant"),  
       levels=c("wildtype", "mutant"))
```

```
[1] mutant  wildtype mutant  
Levels: wildtype mutant
```

Indexing vectors

Access elements of a vector with `[]`, for example `myvec[1]` to get the first element. You can also assign to a specific element of a vector.

```
myvec[1]
```

```
[1] 10
```

```
myvec[2]
```

```
[1] 20
```

```
myvec[2] <- 5  
myvec
```

```
[1] 10 5 30 40 50
```

Indexing vectors (2)

Can we use a vector to index another vector? Yes!

```
myind <- c(4,3,2)  
myvec[myind]
```

```
[1] 40 30 5
```

We could equivalently have written

```
myvec[c(4,3,2)]
```

```
[1] 40 30 5
```

Slicing

Sometimes we want a contiguous *slice* from a vector.

```
myvec[3:5]
```

```
[1] 30 40 50
```

- : here actually creates a vector, which is then used to index myvec.
- : is pretty useful on its own too.

```
3:5
```

```
[1] 3 4 5
```

```
1:50
```

```
[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25  
[26] 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50
```

```
numbers <- 1:10  
numbers*numbers
```

```
[1] 1 4 9 16 25 36 49 64 81 100
```

Slicing (2)

Now we can see why R always puts a `[1]` in its output: it is indicating that the first element of the vector can be accessed with `[1]`. Subsequent lines show the appropriate index to access the first number in the line.

matrix

A matrix is a two dimensional tabular data structure in which all the elements are the same type. We will typically be dealing with numeric matrices, but it is also possible to have character or logical matrices, etc. Matrix rows and columns may have names (`rownames`, `colnames`).

Access an element: `mat[3,5]` `mat["arowname","acolumnname"]`

Get a whole row: `mat[3,]`

Get a whole column: `mat[,5]`

Creation: `matrix()`

data.frame

A data frame is a two dimensional tabular data structure in which the columns may have different types, but all the elements in each column must have the same type. Data frame rows and columns may have names (`rownames`, `colnames`). However in typical usage columns are named but rows are not.

Accessing elements, rows, and columns is the same as for matrices, but we can also get a whole column using `$`.

Creation:

```
data.frame(colname1=values1,colname2=values2,...)
```

Functions

R has various *functions*, such as `sum()`. We can get help on a `sum` with `?sum`.

```
?sum
```

```
sum(myvec)
```

```
[1] 135
```

Here we have *called* the function `sum` with the *argument* `myvec`.

Because R is a language for statistics, it has many built in statistics-related functions. We will also be loading more specialized functions from “libraries” (also known as “packages”).

Functions (2)

Some functions take more than one argument. Let's look at the function `rep`, which means “repeat”, and which can take a variety of different arguments. In the simplest case, it takes a value and the number of times to repeat that value.

```
rep(42, 10)
```

```
[1] 42 42 42 42 42 42 42 42 42 42
```

As with many functions in R—which is obsessed with vectors—the thing to be repeated can be a vector with multiple elements.

```
rep(c(1,2,3), 10)
```

```
[1] 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3
```

Functions (3)

So far we have used *positional* arguments, where R determines which argument is which by the order in which they are given. We can also give arguments by *name*. For example, the above is equivalent to

```
rep(c(1,2,3), times=10)
```

```
[1] 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3
```

```
rep(x=c(1,2,3), 10)
```

```
[1] 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3
```

```
rep(x=c(1,2,3), times=10)
```

```
[1] 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3
```

Functions (4)

Arguments can have default values, and a function may have many different possible arguments that make it do obscure things. For example, `rep` can also take an argument `each=`. It's typical for a function to be invoked with some number of positional arguments, which are always given, plus some less commonly used arguments, typically given by name.

```
rep(c(1,2,3), each=3)
```

```
[1] 1 1 1 2 2 2 3 3 3
```

```
rep(c(1,2,3), each=3, times=5)
```

```
[1] 1 1 1 2 2 2 3 3 3 1 1 1 2 2 2 3 3 3 1 1 1 2 2 2 3 3 3 1 1 1 2 2 2 3 3 3 1 1  
[39] 1 2 2 2 3 3 3
```

Lists

Vectors contain all the same kind of thing. *Lists* can contain different kinds of thing. Lists can even contain vectors or other lists as elements.

We generally give the things in a list names. Try `list(num=42, greeting="hello")`. To access named elements we use `$`.

```
mylist <- list(num=42, greeting="Hello, world")  
mylist$greeting
```

```
[1] "Hello, world"
```

```
mylist[[2]]
```

```
[1] "Hello, world"
```

Functions that need to return multiple outputs often do so as a list.

Data types

We've seen several data types in this chapter, and will be seeing two more in the following chapters. This section serves as an overview of data types in R and their typical usage.

Each data type has various ways it can be created and various ways it can be accessed. If we have data in the wrong type, there are functions to “cast” it to the right type.

This will all make more sense once you have seen these data types in action.

If you're not sure what type of value you are dealing with you can use `class`, or for more detailed information `str` (structure). Try the following:

```
class(myvec)  
class(mylist)  
str(mylist)
```

Miscellanea

- One can be interested in looking at all the available objects

```
ls()
```

```
[1] "myind"      "mylist"     "myvec"      "numbers"    "weight_kg"
```

- or removing one object

```
rm(myvec)  
ls()
```

```
[1] "myind"      "mylist"     "numbers"    "weight_kg"
```

- or knowing the packages that are loaded

```
search()
```

```
[1] ".GlobalEnv"      "package:stats"    "package:graphics"  
[4] "package:grDevices" "package:utils"    "package:datasets"  
[7] "package:methods" "Autoloads"        "package:base"
```

Installing packages

Install and load packages

A package must be loaded before using a given function

```
> spss.get  
Error: object 'spss.get' not found
```

From CRAN (copy & paste - are required for the course):

```
install.packages(c("devtools", "foreign", "Hmisc", "readxl",  
                  "readstata13", "Epi", "car"))
```

From GitHub:

```
library(devtools)  
install_github("isglobal-brge/SNPassoc")
```

The functions can also be used without loading the package by:

```
devtools::install_github("isglobal-brge/SNPassoc")
```


Dealing with working directories

Working directories

```
getwd()
```

```
[1] "C:/Juan/CREAL/GitHub/R_course/Day1-Introduction"
```

```
setwd("c:/Juan/CREAL/GitHub/R_course")
```

```
ff <- "c:/Juan/CREAL/GitHub/R_course/Day1-Introduction"  
dir(ff)
```

```
[1] "data"                "Day1_descriptive.pdf"  "Day1_descriptive.R"  
[4] "Day1_descriptive.Rmd" "Day1_descriptive_cache" "figures"  
[7] "header.tex"
```

```
dir("data")
```

```
character(0)
```

```
file.path(ff, "data/parto2.dat")
```

```
[1] "c:/Juan/CREAL/GitHub/R_course/Day1-Introduction/data/parto2.dat"
```

Getting data into R - import data

Using data function

Most R packages contain `data.frames` to illustrative purposes. These data can be loaded into R using data function. For instance:

```
data(CO2, package="datasets")  
head(CO2)
```

	Plant	Type	Treatment	conc	uptake
1	Qn1	Quebec	nonchilled	95	16.0
2	Qn1	Quebec	nonchilled	175	30.4
3	Qn1	Quebec	nonchilled	250	34.8
4	Qn1	Quebec	nonchilled	350	37.2
5	Qn1	Quebec	nonchilled	500	35.3
6	Qn1	Quebec	nonchilled	675	39.2

Required packages

- ▶ `foreign`: ~ import/export from SPSS, STATA, SAS,...
- ▶ `RODBC`: ~ SQL or ACCESS data bases.
- ▶ `Hmisc`: ~ SPSS, Hmisc (64bits).
- ▶ `readxl`: ~ export/import Excel files.

```
library(foreign)
library(Hmisc)
library(readxl)
```

ASCII files

- ▶ sep: column/variable separator character
- ▶ header: first row contains variable names?
- ▶ as.is: convert character to factor variables?

```
df<-read.table("data/parto2.dat", sep=";", as.is=TRUE,  
               header=FALSE)  
head(df)
```

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11
1	1	GADI	14-JUN-2001	19-JUN-2001	26-JUL-2001	2	24	3.38	2	1	2
2	2	CAEL	15-JUN-2001	21-JUN-2001	15-FEB-2002	2	27	2.50	1	2	1
3	3	COMO	16-JUN-2001	01-JUL-2001	23-JUN-2001	1	44	3.15	2	2	1
4	4	VIMU	18-JUN-2001	23-JUN-2001	17-DEC-2001	2	25	2.74	1	1	1
5	5	PAVI	19-JUN-2001	25-JUN-2001	26-JUN-2001	1	27	3.60	2	2	1
6	6	PASA	20-JUN-2001	01-JUL-2001	27-JUN-2001	1	36	2.65	2	1	2

Excel

Use read_excel from readxl package.

```
df<-read_excel("data/mujeres.xlsx")  
class(df)
```

```
[1] "tbl_df"      "tbl"        "data.frame"
```

```
class(df) <- "data.frame"  
head(df)
```

	X__1	id	sexo	n_histo	an_diag	dondedx	dondectl	frecvisi
1	1	1	Mujer	GACA144012600	90	Ambulatorio	Ambulatorio	Cada 2-3 meses
2	3	3	Mujer	FOSA126052000	92	Ambulatorio	Ambulatorio	Cada 2-3 meses
3	5	5	Mujer	FEJI150053000	78	Hospital	Hospital	Cada 2-3 meses
4	6	6	Mujer	ORLO133102100	81	Ambulatorio	Ambulatorio	Mensual
5	7	7	Mujer	GRMA131110800	90	Ambulatorio	Ambulatorio	Cada 2-3 meses
6	16	16	Mujer	POFE121011400	71	Ambulatorio	Ambulatorio	Mensual

	tx_ab	tx_de	reflec	hbac_1	hbac_2	uso_re	uso_ok
1	ADO	ADO	Ninguno	8.57	5.95	No	<NA>
2	ADO	ADO	Ninguno	6.18	5.82	No	<NA>
3	ADO	ADO	Ninguno	8.33	6.23	No	<NA>
4	Dieta	Dieta	ACCUTREN SENSOR	5.27	10.42	Si	No
5	Dieta	ADO	Ninguno	7.40	6.81	No	<NA>
6	ADO	Insulina	Ninguno	6.90	8.33	No	<NA>

Stata

- To read Stata files (.dta), use read.dta function from foreign package

```
df <- read.dta("data/partoFin.dta",  
              convert.dates = TRUE, convert.factors = TRUE)  
head(df)
```

	id	ini	dia_nac	dia_entr	ulti_lac	tx	edad	peso	sexo	tip_par
1	1	GADI	2001-06-14	2001-06-19	2001-07-26	intensivo	24	3.38	niña	instrument.
2	2	CAEL	2001-06-15	2001-06-21	2002-02-15	intensivo	27	2.50	niño	no instrum.
3	3	COMO	2001-06-16	2001-07-01	2001-06-23	estándar	44	3.15	niña	no instrum.
4	4	VIMU	2001-06-18	2001-06-23	2001-12-17	intensivo	25	2.74	niño	instrument.
5	5	PAVI	2001-06-19	2001-06-25	2001-06-26	estándar	27	3.60	niña	no instrum.
6	6	PASA	2001-06-20	2001-07-01	2001-06-27	estándar	36	2.65	niña	instrument.

	hermanos	fuma_an	fuma_de	horas_an	horas_de	naci_ca	masde12	sem_lac
1	no	si	no	6	2	sudamérica	no	6
2	si	no	no	2	2	española	si	35
3	si	no	si	3	0	española	no	1
4	si	si	si	11	6	otras	si	26
5	si	si	no	10	22	española	no	1
6	no	no	no	9	9	española	no	1

- Stata version >12 are not supported. You can use readstata13

```
library(readstata13)
```


SPSS

- ▶ To read SPSS (.sav) files, use `spss.get` function from `Hmisc` package.
- ▶ `use.value.labels`: return the label instead of codes.
- ▶ `datevars`: specify date format variables.

```
df <- spss.get("data/parto2.sav", use.value.labels=TRUE, allow="_",  
              datevars=c("dia_nac", "dia_entr", "ulti_lac"))  
head(df)
```

	id	ini	dia_nac	dia_entr	ulti_lac	tx	edad	peso	sexo
1	10	JUNA	2001-06-23	2001-07-02	2001-09-29	Intensivo	32	2.10	niña
2	9	BEMI	2001-06-22	2001-07-05	2001-08-31	Estándar	40	2.40	niña
3	2	CAEL	2001-06-15	2001-06-21	2002-02-15	Intensivo	27	2.50	niño
4	6	PASA	2001-06-20	2001-07-01	2001-06-27	Estándar	36	2.65	niña
5	19	TDPO	2001-07-19	2001-07-26	2001-10-11	Estándar	29	2.65	niña
6	4	VIMU	2001-06-18	2001-06-23	2001-12-17	Intensivo	25	2.74	niño

	tip_par	hermanos
1	no instrum.	no
2	no instrum.	no
3	no instrum.	si
4	instrument.	no
5	no instrum.	no
6	instrument.	si

Read data using Rstudio

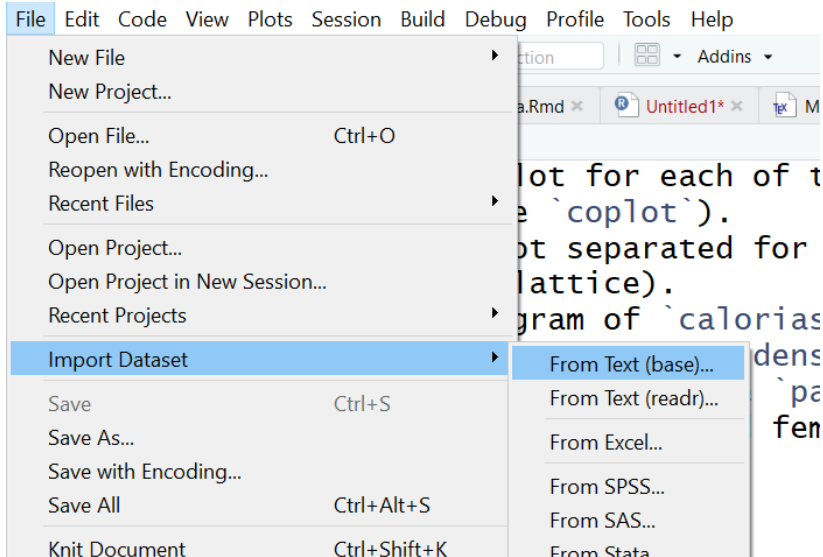


Figure 5: Read data

Import Dataset

Name: multicentric

Encoding: Automatic

Heading: ☐ Yes ☒ No

Row names: Automatic

Separator: Tab

Decimal: Period

Quote: Double quote (")

Comment: None

na.strings: NA

☒ Strings as factors

Input File

ident	pais	status	edad	niveledu	fumar	edad
10001	Brasil	Control	64	primaria	ex-fumador	
10002	Brasil	Caso	51	primaria	fumador	17
10003	Brasil	Control	48	ninguno	fumador	14
10004	Brasil	Caso	49	ninguno	no fumador	23
10005	Brasil	Control	41	primaria	no fumador	
10006	Brasil	Caso	45	primaria	fumador	18
10007	Brasil	Control	51	primaria	no fumador	
10008	Brasil	Caso	42	primaria	ex-fumador	1
10009	Brasil	Control	58	primaria	no fumador	
10010	Brasil	Caso	76	secundaria	no fumador	
10011	Brasil	Control	57	primaria	ex-fumador	
10012	Brasil	Caso	46	primaria	no fumador	2
10013	Brasil	Control	64	ninguno	ex-fumador	
10014	Brasil	Caso	39	ninguno	no fumador	10
10015	Brasil	Control	56	primaria	no fumador	
10016	Brasil	Control	41	primaria	fumador	1
10017	Brasil	Caso	52	ninguno	fumador	22

Data Frame

V1	V2	V3	V4	V5	V6	V7
ident	pais	status	edad	niveledu	fumar	edad
10001	Brasil	Control	64	primaria	ex-fumador	16
10002	Brasil	Caso	51	primaria	fumador	17

Figure 6: Read data

Export data

ASCII, Excel, Stata

► ASCII file

```
write.table(df, "parto2ex.dat")
```

► Stata

```
write.dta(df, file="c:/juan/data/bd.dta"), version=7L)  
save.dta13(df, file="c:/juan/data/bd.dta")
```

► Objects

Save:

```
save(df, file="c:/juan/data/bd.Rdata")) # or .rda
```

Load:

```
load("c:/juan/data/bd.Rdata")) # an object df will be in R
```

Descriptive analysis

Read the data

- ▶ Read the data from a SPSS data file
- ▶ Hmisc package is required

```
library(Hmisc)
df <- spss.get("data/partoFin.sav", allow="_",
              datevars=c("dia_nac", "dia_entr", "ulti_lac"))
```

- ▶ Let us look at first rows

```
head(df)
```

	id	ini	dia_nac	dia_entr	ulti_lac	tx	edad	peso	sexo
1	1	GADI	2001-06-14	2001-06-19	2001-07-26	Intensivo	24	3.38	niña
2	2	CAEL	2001-06-15	2001-06-21	2002-02-15	Intensivo	27	2.50	niño
3	3	COMO	2001-06-16	2001-07-01	2001-06-23	Estándar	44	3.15	niña
4	4	VIMU	2001-06-18	2001-06-23	2001-12-17	Intensivo	25	2.74	niño
5	5	PAVI	2001-06-19	2001-06-25	2001-06-26	Estándar	27	3.60	niña
6	6	PASA	2001-06-20	2001-07-01	2001-06-27	Estándar	36	2.65	niña

	tip_par	hermanos	fuma_an	fuma_de	horas_an	horas_de	naci_ca	masde12
1	instrument.	no	Si	No	6	2	Sudamérica	No
2	no instrum.	si	No	No	2	2	Española	Si
3	no instrum.	si	No	Si	3	0	Española	No
4	instrument.	si	Si	Si	11	6	Otras	Si
5	no instrum.	si	Si	No	10	22	Española	No
6	instrument.	no	No	No	9	9	Española	No

sem_lac

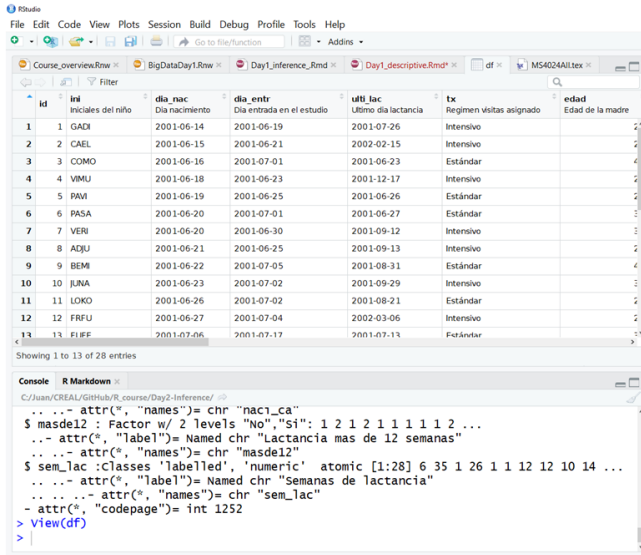
► Let us see the type of variables we have

`str(df)`

```
'data.frame': 28 obs. of 18 variables:
 $ id      : int  1 2 3 4 5 6 7 8 9 10 ...
 $ ini      : Factor w/ 28 levels "ADJU      ","ANZO      ",...: 11 5 8 28 21 20 27 1 3 14 ...
 ..- attr(*, "label")= Named chr "Iniciales del niño"
 .. ..- attr(*, "names")= chr "ini"
 $ dia_nac  : Date, format: "2001-06-14" "2001-06-15" ...
 $ dia_entr: Date, format: "2001-06-19" "2001-06-21" ...
 $ ulti_lac: Date, format: "2001-07-26" "2002-02-15" ...
 $ tx       : Factor w/ 2 levels "Estándar","Intensivo": 2 2 1 2 1 1 2 2 1 2 ...
 ..- attr(*, "label")= Named chr "Regimen visitas asignado"
 .. ..- attr(*, "names")= chr "tx"
 $ edad     :Classes 'labelled', 'numeric' atomic [1:28] 24 27 44 25 27 36 35 23 40 32 ...
 .. ..- attr(*, "label")= Named chr "Edad de la madre"
 .. ..- attr(*, "names")= chr "edad"
 $ peso     :Classes 'labelled', 'numeric' atomic [1:28] 3.38 2.5 3.15 2.74 3.6 2.65 2.97 3.2 2.4 ...
 .. ..- attr(*, "label")= Named chr "peso del niño"
 .. ..- attr(*, "names")= chr "peso"
 $ sexo     : Factor w/ 2 levels "niño","niña": 2 1 2 1 2 2 1 1 2 2 ...
 ..- attr(*, "label")= Named chr "sexo de la criatura"
 .. ..- attr(*, "names")= chr "sexo"
 $ tip_par  : Factor w/ 2 levels "instrument.",...: 1 2 2 1 2 1 1 2 2 2 ...
 ..- attr(*, "label")= Named chr "Tipo de parto"
 .. ..- attr(*, "names")= chr "tip_par"
 $ hermanos: Factor w/ 2 levels "si","no": 2 1 1 1 1 2 2 2 2 2 ...
 ..- attr(*, "label")= Named chr "Tiene hermanos"
 .. ..- attr(*, "names")= chr "hermanos"
 $ fuma_an  : Factor w/ 2 levels "No","Si": 2 1 1 2 2 1 1 1 2 2 ...
 ..- attr(*, "label")= Named chr "Fuma antes embarazo"
 .. ..- attr(*, "names")= chr "fuma_an"
```

- Also it is possible to visualize data like a 'spreadsheet'

View(df)



RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function Addins

Course_overview.Rnw BigDataDay1.Rnw Day1_inference.Rmd Day1_descriptive.Rmd df MS4024AllTex

Filter

	id	ini Iniciales del niño	dia_nac Dia nacimiento	dia_entr Dia entrada en el estudio	ulti_lac Ultimo dia lactancia	tx Regimen visitas asignado	edad Edad de la madre
1	1	GADI	2001-06-14	2001-06-19	2001-07-26	Intensivo	
2	2	CAEL	2001-06-15	2001-06-21	2002-02-15	Intensivo	
3	3	COMO	2001-06-16	2001-07-01	2001-06-23	Estándar	
4	4	VIMU	2001-06-18	2001-06-23	2001-12-17	Intensivo	
5	5	PAVI	2001-06-19	2001-06-25	2001-06-26	Estándar	
6	6	PASA	2001-06-20	2001-07-01	2001-06-27	Estándar	
7	7	VERI	2001-06-20	2001-06-30	2001-09-12	Intensivo	
8	8	ADJU	2001-06-21	2001-06-25	2001-09-13	Intensivo	
9	9	BEMI	2001-06-22	2001-07-05	2001-08-31	Estándar	
10	10	JUNA	2001-06-23	2001-07-02	2001-09-29	Intensivo	
11	11	LOKO	2001-06-26	2001-07-02	2001-08-21	Estándar	
12	12	FRFU	2001-06-27	2001-07-04	2002-03-06	Intensivo	
13	13	FIFF	2001-07-06	2001-07-17	2001-07-13	Estándar	

Showing 1 to 13 of 28 entries

Console R Markdown

C:/Juan/CREAL/GitHub/R_course/Day2-Inference/

```
.. ..- attr(*, "names")= chr "nac1_ca"
$ masde12 : Factor w/ 2 levels "No","Si": 1 2 1 2 1 1 1 1 1 2 ...
.. ..- attr(*, "label")= Named chr "Lactancia mas de 12 semanas"
.. ..- attr(*, "names")= chr "masde12"
$ sem_lac :Classes 'labelled', 'numeric' atomic [1:28] 6 35 1 26 1 1 12 12 10 14 ...
.. ..- attr(*, "label")= Named chr "Semanas de lactancia"
.. ..- attr(*, "names")= chr "sem_lac"
- attr(*, "codepage")= int 1252
> View(df)
>
```

Creating new variables

```
df$edad2 <- df$edad*df$edad
```

```
df$edad.c <- cut(df$edad, c(-Inf, 20, 25, Inf),  
                 labels=c("low", "med", "high"))  
table(df$edad.c)
```

```
low med high  
  2   7   19
```

```
df$edad.c2 <- cut(df$edad, seq(0,50,5))  
table(df$edad.c2)
```

```
(0,5] (5,10] (10,15] (15,20] (20,25] (25,30] (30,35] (35,40] (40,45] (45,50]  
    0      0      0      2      7      8      5      5      1      0
```

```
df$edad.c3 <- cut(df$edad, quantile(df$edad),  
                 label=c("1st", "2nd", "3rd", "4th"))  
table(df$edad.c3)
```

```
1st 2nd 3rd 4th  
  6   8   7   6
```

Dealing with factor variables

► Recode

```
table(df$naci_ca)
```

```
    Española      Otras Sudamérica  
         14          7          7
```

```
levels(df$naci_ca)
```

```
[1] "Española"  "Otras"     "Sudamérica"
```

```
df$naci_ca2 <- df$naci_ca  
levels(df$naci_ca2) <- c("Española", "Extranjero", "Extranjero")  
table(df$naci_ca2)
```

```
    Española Extranjero  
         14          14
```

► Relevel

```
df$naci_ca3 <- relevel(df$naci_ca2, 2)  
table(df$naci_ca3)
```

Extranjero	Española
14	14

Explore data

- How many rows and variables

```
nrow(df)
```

```
[1] 28
```

```
ncol(df)
```

```
[1] 24
```

- View names

```
names(df)
```

```
[1] "id"      "ini"      "dia_nac"  "dia_entr" "ulti_lac" "tx"
[7] "edad"    "peso"     "sexo"     "tip_par"  "hermanos" "fuma_an"
[13] "fuma_de" "horas_an" "horas_de" "naci_ca"  "masde12"  "sem_lac"
[19] "edad2"   "edad.c"   "edad.c2"  "edad.c3"  "naci_ca2" "naci_ca3"
```

► Summary of all variables

summary(df)

id	ini	dia_nac	dia_entr
Min. : 1.00	ADJU : 1	Min. :2001-06-14	Min. :2001-06-19
1st Qu.: 7.75	ANZO : 1	1st Qu.:2001-06-20	1st Qu.:2001-07-01
Median :14.50	BEMI : 1	Median :2001-07-13	Median :2001-07-20
Mean :14.50	BOPE : 1	Mean :2001-07-06	Mean :2001-07-14
3rd Qu.:21.25	CAEL : 1	3rd Qu.:2001-07-20	3rd Qu.:2001-07-27
Max. :28.00	CAGI : 1	Max. :2001-07-25	Max. :2001-08-03
	(Other) :22		

ulti_lac	tx	edad	peso	sexo
Min. :2001-06-23	Estándar :13	Min. :17.00	Min. :2.100	niño:12
1st Qu.:2001-08-05	Intensivo:15	1st Qu.:24.75	1st Qu.:2.938	niña:16
Median :2001-09-21		Median :27.00	Median :3.260	
Mean :2001-10-12		Mean :29.29	Mean :3.208	
3rd Qu.:2001-12-13		3rd Qu.:35.00	3rd Qu.:3.470	
Max. :2002-03-27		Max. :44.00	Max. :4.460	

tip_par	hermanos	fuma_an	fuma_de	horas_an	horas_de
instrument.: 5	si:12	No:14	No:18	Min. : 2.000	Min. : 0.000
no instrum.:23	no:16	Si:14	Si:10	1st Qu.: 5.000	1st Qu.: 2.000
				Median : 7.000	Median : 5.500
				Mean : 7.429	Mean : 6.536
				3rd Qu.:10.000	3rd Qu.: 9.250
				Max. :12.000	Max. :23.000

naci_ca	masde12	sem_lac	edad2	edad.c	edad.c2
Española :14	No:16	Min. : 1.00	Min. : 289.0	low : 2	(25,30]:8
Otras : 7	Si:12	1st Qu.: 2.75	1st Qu.: 612.8	med : 7	(20,25]:7
Sudamérica: 7		Median :12.00	Median : 729.0	high:19	(30,35]:5
		Mean :13.96	Mean : 901.5		(35,40]:5

Select variables

- Select a variable by its name

```
df$sexo
```

```
sexo de la criatura
```

```
[1] niña niño niña niño niña niña niño niño niña niña niño niña niña niño niño  
[16] niño niño niña niña niña niña niño niño niña niña niña niño niña  
Levels: niño niña
```

- Select a variable by its position

```
df[,2]
```

```
Iniciales del niño
```

```
[1] GADI    CAEL    COMO    VIMU    PAVI    PASA    VERI    ADJU  
[9] BEMI    JUNA    LOKO    FRFU    FUFE    POCA    LOLO    BOPE  
[17] ANZO    MEVE    TOPO    PUPI    ROPA    LOMA    CEMA    CAGI  
[25] GRSE    GUMA    PERI    MAPE  
28 Levels: ADJU ANZO BEMI BOPE CAEL CAGI ... VIMU
```


► Select some variables by names

```
df[,c("sexo", "peso", "edad")]
```

	sexo	peso	edad
1	niña	3.38	24
2	niño	2.50	27
3	niña	3.15	44
4	niño	2.74	25
5	niña	3.60	27
6	niña	2.65	36
7	niño	2.97	35
8	niño	3.20	23
9	niña	2.40	40
10	niña	2.10	32
11	niño	3.45	26
12	niña	3.45	29
13	niña	3.40	36
14	niño	3.05	36
15	niño	3.60	17
16	niño	3.40	40
17	niño	3.15	27
18	niña	3.32	32
19	niña	2.65	29
20	niña	4.46	21
21	niña	3.15	35
22	niño	3.70	27
23	niño	3.79	24
24	niña	3.75	18
25	niña	2.95	34
26	niña	2.90	27
27	niño	3.44	25
28	niña	3.53	24

► Select some variables by position

```
df[,c(1,3,5)]
```

	id	dia_nac	ulti_lac
1	1	2001-06-14	2001-07-26
2	2	2001-06-15	2002-02-15
3	3	2001-06-16	2001-06-23
4	4	2001-06-18	2001-12-17
5	5	2001-06-19	2001-06-26
6	6	2001-06-20	2001-06-27
7	7	2001-06-20	2001-09-12
8	8	2001-06-21	2001-09-13
9	9	2001-06-22	2001-08-31
10	10	2001-06-23	2001-09-29
11	11	2001-06-26	2001-08-21
12	12	2001-06-27	2002-03-06
13	13	2001-07-06	2001-07-13
14	14	2001-07-13	2001-11-09
15	15	2001-07-13	2001-07-20
16	16	2001-07-14	2002-01-19
17	17	2001-07-18	2001-12-05
18	18	2001-07-18	2002-03-27
19	19	2001-07-19	2001-10-11
20	20	2001-07-20	2001-10-12
21	21	2001-07-20	2001-08-17
22	22	2001-07-21	2002-03-02
23	23	2001-07-22	2001-08-12
24	24	2001-07-23	2001-07-30
25	25	2001-07-24	2001-08-07
26	26	2001-07-25	2001-12-12
27	27	2001-07-25	2002-01-16
28	28	2001-07-25	2001-11-14

Select rows

► Select a row

```
df[4,]
```

	id	ini	dia_nac	dia_entr	ulti_lac	tx	edad	peso	sexo
4	4	VIMU	2001-06-18	2001-06-23	2001-12-17	Intensivo	25	2.74	niño
		tip_par	hermanos	fuma_an	fuma_de	horas_an	horas_de	naci_ca	masde12
4	instrument.	si	Si	Si	11	6	Otras	Si	
	sem_lac	edad2	edad.c	edad.c2	edad.c3	naci_ca2	naci_ca3		
4	26	625	med (20,25]	2nd	Extranjero	Extranjero			

► Select rows

```
df[4:10,]
```

	id	ini	dia_nac	dia_entr	ulti_lac	tx	edad	peso	sexo
4	4	VIMU	2001-06-18	2001-06-23	2001-12-17	Intensivo	25	2.74	niño
5	5	PAVI	2001-06-19	2001-06-25	2001-06-26	Estándar	27	3.60	niña
6	6	PASA	2001-06-20	2001-07-01	2001-06-27	Estándar	36	2.65	niña
7	7	VERI	2001-06-20	2001-06-30	2001-09-12	Intensivo	35	2.97	niño
8	8	ADJU	2001-06-21	2001-06-25	2001-09-13	Intensivo	23	3.20	niño
9	9	BEMI	2001-06-22	2001-07-05	2001-08-31	Estándar	40	2.40	niña
10	10	JUNA	2001-06-23	2001-07-02	2001-09-29	Intensivo	32	2.10	niña
		tip_par	hermanos	fuma_an	fuma_de	horas_an	horas_de	naci_ca	masde12
4	instrument.	si	Si	Si	11	6	Otras	Si	
5	no instrument.	si	Si	No	10	22	Española	No	
6	instrument.	no	No	No	9	9	Española	No	

► Select rows by a condition, use subset

```
subset(df, sexo=="niña")
```

	id	ini	dia_nac	dia_entr	ulti_lac	tx	edad	peso	sexo
1	1	GADI	2001-06-14	2001-06-19	2001-07-26	Intensivo	24	3.38	niña
3	3	COMO	2001-06-16	2001-07-01	2001-06-23	Estándar	44	3.15	niña
5	5	PAVI	2001-06-19	2001-06-25	2001-06-26	Estándar	27	3.60	niña
6	6	PASA	2001-06-20	2001-07-01	2001-06-27	Estándar	36	2.65	niña
9	9	BEMI	2001-06-22	2001-07-05	2001-08-31	Estándar	40	2.40	niña
10	10	JUNA	2001-06-23	2001-07-02	2001-09-29	Intensivo	32	2.10	niña
12	12	FRFU	2001-06-27	2001-07-04	2002-03-06	Intensivo	29	3.45	niña
13	13	FUFE	2001-07-06	2001-07-17	2001-07-13	Estándar	36	3.40	niña
18	18	MEVE	2001-07-18	2001-07-27	2002-03-27	Intensivo	32	3.32	niña
19	19	TOPO	2001-07-19	2001-07-26	2001-10-11	Estándar	29	2.65	niña
20	20	PUPI	2001-07-20	2001-07-23	2001-10-12	Intensivo	21	4.46	niña
21	21	ROPA	2001-07-20	2001-07-30	2001-08-17	Estándar	35	3.15	niña
24	24	CAGI	2001-07-23	2001-07-25	2001-07-30	Intensivo	18	3.75	niña
25	25	GRSE	2001-07-24	2001-08-03	2001-08-07	Estándar	34	2.95	niña
26	26	GUMA	2001-07-25	2001-07-31	2001-12-12	Intensivo	27	2.90	niña
28	28	MAPE	2001-07-25	2001-07-30	2001-11-14	Estándar	24	3.53	niña

	tip_par	hermanos	fuma_an	fuma_de	horas_an	horas_de	naci_ca	masde12
1	instrument.	no	Si	No	6	2	Sudamérica	No
3	no instrum.	si	No	Si	3	0	Española	No
5	no instrum.	si	Si	No	10	22	Española	No
6	instrument.	no	No	No	9	9	Española	No
9	no instrum.	no	Si	Si	12	10	Española	No
10	no instrum.	no	Si	Si	7	0	Sudamérica	Si
12	no instrum.	si	Si	No	12	11	Sudamérica	Si
13	no instrum.	no	No	No	7	4	Española	No
18	no instrum.	no	Si	No	11	8	Otras	Si
19	no instrum.	no	No	Si	3	1	Española	No
20	no instrum.	no	Si	Si	7	0	Sudamérica	No

► More than one category

```
table(df$naci_ca)
```

```
Española      Otras Sudamérica  
14            7             7
```

```
subset(df, naci_ca%in%c("Española", "Otras"))
```

	id	ini	dia_nac	dia_entr	ulti_lac	tx	edad	peso	sexo
2	2	CAEL	2001-06-15	2001-06-21	2002-02-15	Intensivo	27	2.50	niño
3	3	COMO	2001-06-16	2001-07-01	2001-06-23	Estándar	44	3.15	niña
4	4	VIMU	2001-06-18	2001-06-23	2001-12-17	Intensivo	25	2.74	niño
5	5	PAVI	2001-06-19	2001-06-25	2001-06-26	Estándar	27	3.60	niña
6	6	PASA	2001-06-20	2001-07-01	2001-06-27	Estándar	36	2.65	niña
7	7	VERI	2001-06-20	2001-06-30	2001-09-12	Intensivo	35	2.97	niño
8	8	ADJU	2001-06-21	2001-06-25	2001-09-13	Intensivo	23	3.20	niño
9	9	BEMI	2001-06-22	2001-07-05	2001-08-31	Estándar	40	2.40	niña
13	13	FUFE	2001-07-06	2001-07-17	2001-07-13	Estándar	36	3.40	niña
14	14	POCA	2001-07-13	2001-07-24	2001-11-09	Intensivo	36	3.05	niño
16	16	BOPE	2001-07-14	2001-07-27	2002-01-19	Estándar	40	3.40	niño
17	17	ANZO	2001-07-18	2001-07-24	2001-12-05	Intensivo	27	3.15	niño
18	18	MEVE	2001-07-18	2001-07-27	2002-03-27	Intensivo	32	3.32	niña
19	19	TOPO	2001-07-19	2001-07-26	2001-10-11	Estándar	29	2.65	niña
21	21	ROPA	2001-07-20	2001-07-30	2001-08-17	Estándar	35	3.15	niña
22	22	LOMA	2001-07-21	2001-07-27	2002-03-02	Intensivo	27	3.70	niño
24	24	CAGI	2001-07-23	2001-07-25	2001-07-30	Intensivo	18	3.75	niña
25	25	GRSE	2001-07-24	2001-08-03	2001-08-07	Estándar	34	2.95	niña
26	26	GUMA	2001-07-25	2001-07-31	2001-12-12	Intensivo	27	2.90	niña
27	27	PERI	2001-07-25	2001-07-30	2002-01-16	Intensivo	25	3.44	niño

Descriptives

► Mean

```
mean(df$edad)
```

```
[1] 29.28571
```

► Standard deviation

```
sd(df$edad)
```

```
[1] 6.743211
```

► Median

```
median(df$edad)
```

```
[1] 27
```

► Others

```
var, quantile, range, ...
```

► Continuous variables

```
g <- function(x) c(Mean=mean(x,na.rm=TRUE),
                   Median=median(x,na.rm=TRUE),
                   Sd=sd(x,na.rm=TRUE))
summary(peso~sexo+tx, data=df, fun = g)
```

peso del niño N= 28

		N	Mean	Median	Sd
sexo de la criatura	niño	12	3.249167	3.300	0.3894859
	niña	16	3.177500	3.235	0.5722878
Regimen visitas asignado	Estándar	13	3.209231	3.400	0.4311701
	Intensivo	15	3.207333	3.200	0.5596997
Overall		28	3.208214	3.260	0.4950350

► Categorical variables

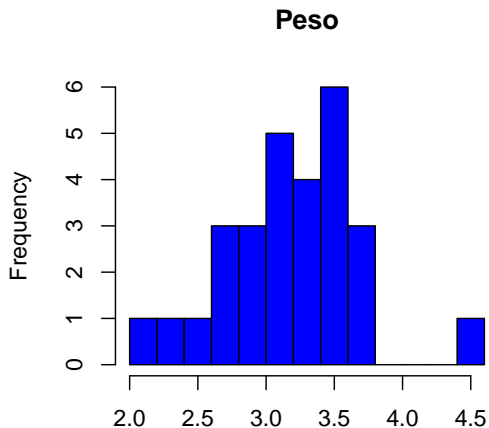
```
library(Epi)
stat.table(list(Sexo = sexo, Visitas = tx), list(N = count(),
  `%` = percent(tx)), data = df, margins = T)
```

Sexo	-----Visitas-----		
	Estándar	Intensivo	Total
niño	4 33.3	8 66.7	12 100.0
niña	9 56.2	7 43.8	16 100.0
Total	13 46.4	15 53.6	28 100.0

Plots

► Histogram

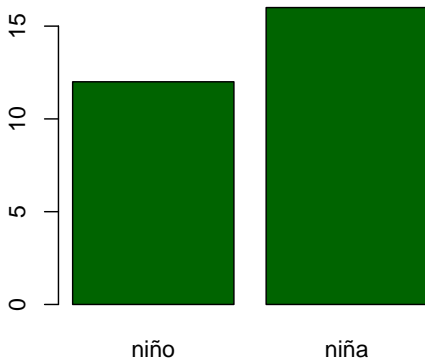
```
hist(df$peso, col="blue", breaks = 10,  
     main="Peso", xlab="")
```



► Barplot

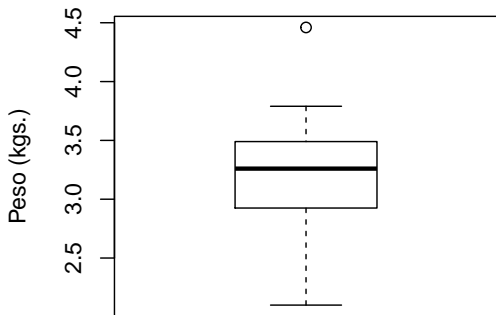
Note: The variable must be a factor or a character. If it is numeric (e.g. 0, 1) convert to a factor using `as.factor`.

```
plot(df$sexo, col="darkgreen")
```



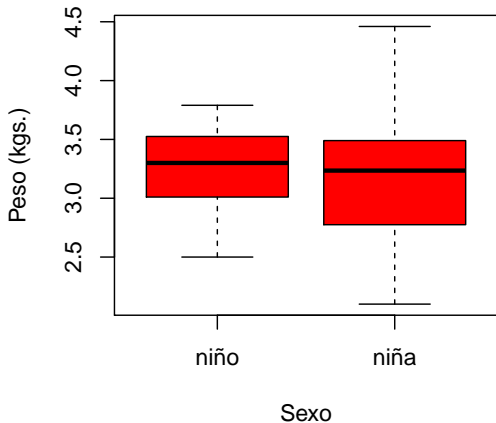
► Boxplot (I)

```
boxplot(df$peso, ylab="Peso (kgs.)")
```



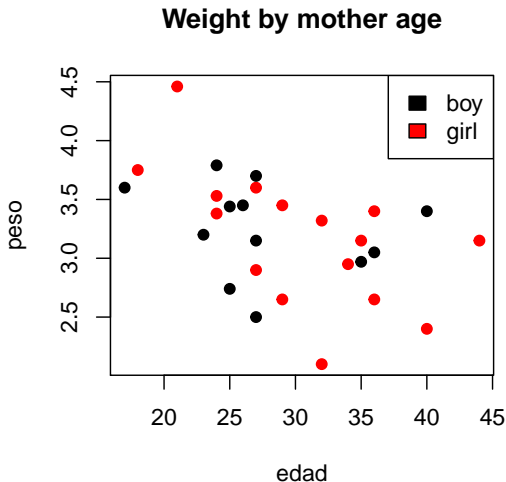
► Boxplot (II)

```
boxplot(peso ~ sexo , data=df, col="red",  
        ylab="Peso (kgs.)", xlab="Sexo")
```



► Scatterplot

```
plot(peso ~ edad, data=df, col=sexo, pch=19)  
title("Weight by mother age")  
legend("topright", c("boy", "girl"), fill=c(1,2))
```



Correlation

- ▶ Pearson correlation

```
with(df, cor(peso, edad))
```

```
[1] -0.4747143
```

- ▶ Spearman correlation

```
with(df, cor(peso, edad, method="spearman"))
```

```
[1] -0.5541522
```

Session info

sessionInfo()

R version 3.4.1 (2017-06-30)

Platform: x86_64-w64-mingw32/x64 (64-bit)

Running under: Windows 10 x64 (build 16299)

Matrix products: default

locale:

[1] LC_COLLATE=Spanish_Spain.1252 LC_CTYPE=Spanish_Spain.1252

[3] LC_MONETARY=Spanish_Spain.1252 LC_NUMERIC=C

[5] LC_TIME=Spanish_Spain.1252

attached base packages:

[1] stats graphics grDevices utils datasets methods base

other attached packages:

[1] Epi_2.24 readxl_1.0.0 Hmisc_4.0-3 ggplot2_2.2.1

[5] Formula_1.2-2 survival_2.41-3 lattice_0.20-35 foreign_0.8-69

loaded via a namespace (and not attached):

[1] Rcpp_0.12.12 cellranger_1.1.0 compiler_3.4.1
[4] pillar_1.1.0 RColorBrewer_1.1-2 plyr_1.8.4
[7] base64enc_0.1-3 tools_3.4.1 rpart_4.1-11
[10] digest_0.6.12 checkmate_1.8.3 evaluate_0.10.1
[13] tibble_1.4.2 gtable_0.2.0 htmlTable_1.9
[16] rlang_0.1.6 Matrix_1.2-10 cmprsk_2.2-7
[19] parallel_3.4.1 yaml_2.1.16 gridExtra_2.3
[22] stringr_1.3.0 knitr_1.20 cluster_2.0.6
[25] htmlwidgets_0.9 rprojroot_1.3-2 grid_3.4.1
[28] nnet_7.3-12 data.table_1.10.4 etm_0.6-2