Guia TP 1

April 14, 2020

1 Guía TP 1

1.1 Primeros pasos con R

Alumno: Leandro Carreira

LU: 669/18

El objetivo de esta guía es repasar algunos conceptos relativos a la sintaxis y manejo de datos usando R, siguiendo la propuesta presentada en *Foundations of Statistics with R*, de Darrin Speegle y Bryan Clair.

Haciendo clicke aquí podra acceder al libro. Buena suerte!

1.1.1 Ejercicio 1.a)

Sea

```
x <- c(1,2,3)
y <- c(6,5,4)
```

Predecir el resultado de correr cada una de las siguientes líneas y **después** chequear las respuestas.

```
x * 2
x * y
x[1] * y[2]
1/x
(1:10) * x[2]
rep(c(1,1,2), times = 2)
seq(from = 0, to = 10, length.out = 5) + 1:10
```

1.1.2 Solución 1.a

Predicciones antes de ejecutar el codigo:

`x * 2` $\xrightarrow{\text{devuelve}}$ el vector: `2 4 6`

`x * y` $\xrightarrow{\text{devuelve}}$ el vector: `6 10 12`

Justificación: Producto de los elementos de cada vector uno a uno.

`x[1] * y[2]` $\xrightarrow{\text{devuelve}}$ el valor numérico: `5`

`1/x` $\xrightarrow{\text{devuelve}}$ el vector: `1 0.5 0.333...`

Justificación: Con cantidad de `3`'s correspondiente a una variale numerica de doble precisión, 16 bits totales entre parte entera y parte decimal.

En este caso el `0.` ocupa 1 bit, por lo que quedan 15 `3`'s detrás de la coma.

`(1:10) * x[2]` $\xrightarrow{\text{devuelve}}$ el vector: `2 4 6 8 10 12 14 16 18 20`

Justificación: Similar a la primera línea, cada uno de los elementos del vector secuencia de 1 a 10 se multiplica por el segundo elemento del vector x

`rep(c(1,1,2), times = 2)` $\xrightarrow{\text{devuelve}}$ el vector: `1 1 2 1 1 2`

Justificación: Se repite *times* veces y concatena de izquierda a derecha el vector `1 1 2`

`seq(from = 0, to = 10, length.out = 5) + 1:10` $\xrightarrow{\text{devuelve}}$ el vector: `1 4.5 8 11.5 15 6 9.5 13 16.5 20`

Justificación: Se genera una secuencia de **5 elementos** equidistantes entre 0 y 10 incluídos `0 2.5 5 7.5 10` y se suma a la secuencia de **10 elementos** de los naturales entre 1 y 10.

Al ser de **longitudes diferentes**, por *recycling* (o *broadcasting* en python), **R repite la operación del elemento más corto sobre el más largo la cantidad de veces necesaria** para que el **resultado final** tenga la **longitud del elemento más largo**.

En particular, a los primeros 5 elementos de la secuencia del 1 al 10 `1 2 3 4 5` le suma la menor secuencia `0 2.5 5 7.5 10`, y a los segundos 5 elementos de la secuencia del 1 al 10 `6 7 8 9 10`, vuelve a sumarle la misma menor secuencia `0 2.5 5 7.5 10`.

El resultado son estas dos secuencias resultantes concatenadas.

1.1.3 Chequeo de respuestas (ejecución de código)

[2]:
```r
# Variables de datos
x <- c(1,2,3)
y <- c(6,5,4)
```

[3]:
```
# Operaciones sobre x e y
x * 2
x * y
x[1] * y[2]
1/x
(1:10) * x[2]
rep(c(1,1,2), times = 2)
seq(from = 0, to = 10, length.out = 5) + 1:10
```

1. 2 2. 4 3. 6

1. 6 2. 10 3. 12

5

1. 1 2. 0.5 3. 0.333333333333333

1. 2 2. 4 3. 6 4. 8 5. 10 6. 12 7. 14 8. 16 9. 18 10. 20

1. 1 2. 1 3. 2 4. 1 5. 1 6. 2

1. 1 2. 4.5 3. 8 4. 11.5 5. 15 6. 6 7. 9.5 8. 13 9. 16.5 10. 20

Se verifica que los resultados se corresponden con las respuestas anteriores.

1.1.4 Ejercicio 1.b

¿Qué ocurre si se multiplican 2 vectores de **distinta** longitud?

1.1.5 Respuesta 1.b

Similar a lo que sucedió con la suma de vectores de distintas longitudes, R resuelve estos casos haciendo uso de *recycling*, realizando la operación **elemento a elemento**, **repitiendo el más corto la cantidad de veces que sea necesaria**.

Si la cantidad de elementos del más largo es proporcional a la longitud del más corto, la operación se ejecuta normalmente.

Pero si la cantidad de elementos **no** es proporcional, R avisa con un mensaje de advertencia que éste es el caso, aunque devuelve el resultado anteriormente descripto.

1.1.6 Ejemplo 1.b

[4]:
```
c(2,3,4)*c(1,1,1,1,1)
```

```
Warning message in c(2, 3, 4) * c(1, 1, 1, 1, 1):
"longer object length is not a multiple of shorter object length"
```

1. 2 2. 3 3. 4 4. 2 5. 3

1.1.7 Ejercicio 2.a

Crear un vector llamado `tratamiento` con coordenadas A, B y C de manera que A aparezca 20 veces, B 18 y C 22.

1.1.8 Solución 2.a

DUDA:

No estoy muy seguro de entender lo que pide el enunciado cuando se refiere a coordenadas A,B,C que se repiten:

Lo interpreto como un vector de 60 coordenadas donde existen 3 valores distintos (que no se especifican), repetidos la cantidad de veces solicitadas.

Mi duda está en que cuando dice *"con coordenadas A, B y C"* pienso en un vector de 3 coordenadas, pero luego éstas se repiten, entonces no son 3, sino que 60, donde sus valores se repiten 20, 18 y 22 veces (no necesariamente ordenados).

[5]:
```
A <- 1
B <- 2
C <- 3
```

[6]:
```
# Forma 1
tratamiento <- c(rep(A, 20), rep(B, 18), rep(C, 22))
tratamiento
```

1. 1 2. 1 3. 1 4. 1 5. 1 6. 1 7. 1 8. 1 9. 1 10. 1 11. 1 12. 1 13. 1 14. 1 15. 1 16. 1 17. 1 18. 1 19. 1 20. 1 21. 2 22. 2 23. 2 24. 2 25. 2 26. 2 27. 2 28. 2 29. 2 30. 2 31. 2 32. 2 33. 2 34. 2 35. 2 36. 2 37. 2 38. 2 39. 3 40. 3 41. 3 42. 3 43. 3 44. 3 45. 3 46. 3 47. 3 48. 3 49. 3 50. 3 51. 3 52. 3 53. 3 54. 3 55. 3 56. 3 57. 3 58. 3 59. 3 60. 3

[7]:
```
# Forma 2. Con funciones que NO vimos todavia, pero al ser escalable, lo agrego
coordenadas  <- c(A,B,C)
repeticiones <- c(20,18,22)
# mapply: aplica FUN entre los elementos de los vectores uno a uno, en orden
tratamiento  <- unlist(mapply(FUN=rep, coordenadas, repeticiones))
tratamiento
```

1. 1 2. 1 3. 1 4. 1 5. 1 6. 1 7. 1 8. 1 9. 1 10. 1 11. 1 12. 1 13. 1 14. 1 15. 1 16. 1 17. 1 18. 1 19. 1 20. 1 21. 2 22. 2 23. 2 24. 2 25. 2 26. 2 27. 2 28. 2 29. 2 30. 2 31. 2 32. 2 33. 2 34. 2 35. 2 36. 2 37. 2 38. 2 39. 3 40. 3 41. 3 42. 3 43. 3 44. 3 45. 3 46. 3 47. 3 48. 3 49. 3 50. 3 51. 3 52. 3 53. 3 54. 3 55. 3 56. 3 57. 3 58. 3 59. 3 60. 3

1.1.9 Ejercicio 2.b

Definir un vector J que sea una secuencia de 1 a 30 con un incremento de 2 y luego sumar la primera y la octava coordenadas.

[8]:
```
J <- seq(1, 30, by=2)
sum(J[1], J[8])
```

16

1.1.10 Ejercicio 3

Calcular la suma de los números naturales del 1 al 100 usando R.

1.1.11 Solución 3

[9]:
```
sum(1:100)
```

5050

1.1.12 Ejercicio 4

Calcular la suma de los cuadrados de los números naturales del 1 al 100 usando R.

1.1.13 Solución 4

[10]:
```
sum(c(1:100)^2)
```

338350

1.1.14 Ejercicio 5

Considerar el conjunto de datos *airquality*, incluído en la librería *datasets*.

Pida ayuda con el comando `?airquality` para obtener información.

[11]:
```
#?airquality
head(airquality)
```

Ozone	Solar.R	Wind	Temp	Month	Day
41	190	7.4	67	5	1
36	118	8.0	72	5	2
12	149	12.6	74	5	3
18	313	11.5	62	5	4
NA	NA	14.3	56	5	5
28	NA	14.9	66	5	6

1.1.15 Ejercicio 5.a

¿Cuántas **observaciones** tiene el conjunto de datos? ¿Cuántas **variables**?

1.1.16 Respuesta 5.a

La cantidad de **observaciones** corresponde con la totalidad de datos observados, donde **cada dato observado**, incluye la totalidad de las **variables** que lo caracterizan.

En otras palabras, las **observaciones** se corresponden con las **filas**, y la **variables** (*características*) se corresponden con las **columnas**.

De la documentación: > *A data frame with 154 observations on 6 variables.*

Podemos obtener que:

La cantidad de **observaciones** es **154**

La cantidad de **variables** es **6**

1.1.17 Ejercicio 5.b

¿Cuáles son los **nombres de las variables**?

1.1.18 Respuesta 5.b

Usando nuevamente la documentación, los **nombres de las variables** son:

Ozone

Solar.R

Wind

Temp

Month

Day

1.1.19 Ejercicio 5.c

¿Qué variables tienen **datos faltantes**?

1.1.20 Solución 5.c

Usando el comando `is.na` podemos obtener una matriz de valores booleanos en la que habrá **verdaderos** donde **existan datos faltantes** (*NA*).

Sumando todos estos valores por columnas, podemos determinar qué **variables** (*columnas*) poseen NAs, donde cada NA contará como un **1** (verdadero).

[12]:
```
colSums(is.na(airquality))
```

Ozone 37 **Solar.R** 7 **Wind** 0 **Temp** 0 **Month** 0 **Day** 0

Las variables que poseen **valores faltantes** son **Ozone** y **Solar.R**

1.1.21 Ejercicio 5.d

d) ¿Cuántas observaciones corresponden al **mes de mayo**?

1.1.22 Solución 5.d

Podemos filtrar la columna `Month` usando `airquality$Month`, y generar una máscara booleana a partir de ella, para luego sumar sus componentes (los `TRUE` valdrán 1, los `FALSE` valdrán 0)

[13]:
```
sum(airquality$Month == 5)
```

31

Existen **31** observaciones en el mes de Mayo

1.1.23 Ejercicio 6

Considerar el conjunto de datos `mtcars` en R.

(Ver `help(mtcars)`)

[14]:
```
#help(mtcars)
head(mtcars)
```

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225	105	2.76	3.460	20.22	1	0	3	1

1.1.24 Ejercicio 6.a

¿Qué autos tienen **4 velocidades**?

1.1.25 Solución 6.a

Podemos filtrar la columna de velocidades (`gear`) para obtener todas las observaciones que coincidan, y extraer de ellas solo los **nombres de las filas**, que corresponden a los **modelos de los vehículos**.

[15]:
```
as.list(row.names(mtcars[mtcars$gear == 4,]))
```

1. 'Mazda RX4'

2. 'Mazda RX4 Wag'
3. 'Datsun 710'
4. 'Merc 240D'
5. 'Merc 230'
6. 'Merc 280'
7. 'Merc 280C'
8. 'Fiat 128'
9. 'Honda Civic'
10. 'Toyota Corolla'
11. 'Fiat X1-9'
12. 'Volvo 142E'

Los modelos de automóviles con **4 velocidades** son los **listados anteriormente**.

1.1.26 Ejercicio 6.b

¿Qué subconjunto de `mtcars` es `mtcars[mtcars$disp > 150 & mtcars$mpg > 20,]`?

1.1.27 Solucion 6.b

El código dado devuelve solo los automóviles con **cilindrada** (*engine displacement*) **mayor a 150 pulgadas cúbicas** ($\approx$ 2458 cm^3) **y que además** consuman más de **20 galones por milla** recorrida ($\approx 47.043\ L/Km$) .

[16]:
```
as.list(row.names(mtcars[mtcars$disp > 150 & mtcars$mpg > 20,]))
```

1. 'Mazda RX4'
2. 'Mazda RX4 Wag'
3. 'Hornet 4 Drive'

Los 3 automóviles en cuestión son los **listados anteriormente**.

1.1.28 Ejercicio 6.c

¿Qué autos tienen **4 velocidades** y **transmisión manual**?

1.1.29 Solución 6.c

La variable binaria `am` indica cuándo un automóvil tiene transmisión manual (`1`) o automática (`0`)

[17]:
```
as.list(row.names(mtcars[mtcars$gear == 4 & mtcars$am == 1,]))
```

1. 'Mazda RX4'
2. 'Mazda RX4 Wag'
3. 'Datsun 710'
4. 'Fiat 128'
5. 'Honda Civic'
6. 'Toyota Corolla'
7. 'Fiat X1-9'
8. 'Volvo 142E'

Los automóviles con **4 velocidades** y **transmisión manual** están **listados anteriormente**.

1.1.30 Ejercicio 6.d

Hallar la **cantidad media de millas por galón** de los autos con **2 carburadores**.

1.1.31 Solución 6.d

[18]:
```
mean(mtcars[mtcars$carb == 2, 'mpg'])
```

22.4

La **cantidad media de millas por galón** de autos con **2 carburadores** es de **22.4**

1.1.32 Ejercicio 7

El conjunto de datos ***arbolado-en-espacios-verdes.csv*** contiene datos de 2011 de **todos los árboles de los espacios verdes de la ciudad de Buenos Aires**.

Los datos actualizados pueden encontrarse aquí, junto con una descripción de los mismos.

1.1.33 Ejercicio 7.a

¿Cuántas **observaciones** tiene el conjunto de datos? ¿Cuántas **variables**?

1.1.34 Solución 7.a

Cargamos los datos del .csv en el mismo directorio que esta notebook, agrego `colClasses = "character"` para prevenir la conversiona factores, que luego son devueltos en cada impresión de valores.

[47]:
```
data <- read.csv('./arbolado-en-espacios-verdes.csv', colClasses = "character")
```

Bajo la misma descripción que en el ejercicio **5.a**, podemos establecer la **cantidad de observaciones** a partir de la **cantidad de filas**, y la **cantidad de variables** a partir de las **columnas**:

[48]:
```
nrow(data)
ncol(data)
```

51502

17

Podemos ver que los datos importados poseen **51.502 observaciones**, y **17 variables**

1.1.35 Ejercicio 7.b

¿Cuáles son los **nombres de las variables**?

1.1.36 Solución 7.b

Usando el comando `colnames` podemos mostrar los nombres de cada una de las columnas del data frame:

[49]:
```
as.list(colnames(data))
```

1. 'long'
2. 'lat'
3. 'id_arbol'
4. 'altura_tot'
5. 'diametro'
6. 'inclinacio'
7. 'id_especie'
8. 'nombre_com'
9. 'nombre_cie'
10. 'tipo_folla'
11. 'espacio_ve'
12. 'ubicacion'
13. 'nombre_fam'
14. 'nombre_gen'
15. 'origen'

16. 'coord_x'
17. 'coord_y'

Los nombres de las **17 variables** están **listados anteriormente**.

1.1.37 Ejercicio 7.c

Calcular la **altura promedio de los árboles** de la ciudad.

1.1.38 Solución 7.c

Podemos promediar la altura total de cada uno de los árboles (`altura_tot`), convertiendo de caracteres a tipo numérico:

[52]:
```
mean(as.numeric(data$altura_tot))
```

12.1671003067842

La **altura promedio** es de **12.167 metros**

1.1.39 Ejercicio 7.d

¿Cuántos árboles había en 2011 en la plaza Arenales?

1.1.40 Solución 7.d

Dado que las observaciones corresponden a **TODOS** los árboles de la ciudad, podemos contar la **cantidad de observaciones** en plaza Arenales, que se corresponderá con la **cantidad de árboles** en esa plaza.

[55]:
```
sum(data$espacio_ve == "ARENALES")
```

198

Cantidad de árboles en plaza Arenales (en 2011): **198**

1.1.41 Ejercicio 7.e

Construya un data.frame llamado `arboles_cercanos` que contenga sólo las filas correspondientes a su espacio verde favorito.

Puede utilizar el comando sort para ordenar alfabéticamente un vector de caracteres.

1.1.42 Solución 7.e

Elijo Plaza Holanda, cercano al Jardín Japonés. Un espacio verde amplio y con algunos árboles muy antíguos.

[56]:
```
arboles_cercanos <- data.frame(data[data$espacio_ve == "HOLANDA",])
nrow(arboles_cercanos)
#head(arboles_cercanos)
```

253

1.1.43 Ejercicio 7.f

> Utilizando el comando `unique` averigüe los **nombres de los árboles** presentes en su espacio verde favorito en 2011.

1.1.44 Solución 7.f

Los datos tienen tanto los nombres comunes de los árboles, como los nombres científicos, de la familia, y el género.

Elijo los **nombres comunes** ya que en su gran parte son de conocimiento general.

[60]:
```
unique(arboles_cercanos$nombre_com)
```

1. 'El chañar' 2. 'No Determinado' 3. 'Eucalipto' 4. 'No Determinable' 5. 'Palmito' 6. 'Fenix' 7. 'Jacarandá' 8. 'Plátano' 9. 'Coculus, Cóculo' 10. 'Washingtonia' 11. 'Celtis tala' 12. 'Cedro del Himalaya variedad aurea' 13. 'Bunya-bunya (Araucaria de Bidwill)' 14. 'Laurel' 15. 'Ligustro disciplinado (Ligustro variegado)' 16. 'Tipa blanca' 17. 'Palo borracho rosado' 18. 'Magnolia' 19. 'Almez (Almecino o Almecina)' 20. 'Liquidambar' 21. 'Árbol del cielo (Ailanto o Árbol de los dioses)' 22. 'Naranjo amargo' 23. 'Lapacho amarillo' 24. 'Sófora japónica' 25. 'Álamo plateado' 26. 'Pino del Paraná (Pino de Misiones o Pino de Brasil)' 27. 'Ginkgo' 28. 'Morera blanca' 29. 'Encina' 30. 'Pino de las canarias' 31. 'Falso alerce' 32. 'Ombú' 33. 'Coronillo' 34. 'Robusta' 35. 'Washingtonia (Palmera washingtonia)' 36. 'Palma de california' 37. 'Pindó' 38. 'Arrayán (Anacahuita)' 39. 'Tejo chino' 40. 'Quebracho colorado' 41. 'Roble palustre' 42. 'Crespón (Àrbol de Júpiter)' 43. 'Arbol de las orquideas' 44. 'Álamo carolina' 45. 'Eucalipto sideroxylon' 46. 'Ligustro' 47. 'Cefalotaxus' 48. 'Cedro del Himalaya' 49. 'Fotinia' 50. 'Cica' 51. 'Roble sedoso (Grevillea)' 52. 'Eucalipto (Eucalipto común)' 53. 'Ciprés'

Listado de **árboles en plaza Holanda** (ignorando arboles no identificados):

> El chañar, Eucalipto, Palmito, Fenix, Jacarandá, Plátano, Coculus, Cóculo, Washingtonia, Celtis tala, Cedro del Himalaya variedad aurea, Bunya-bunya (Araucaria de Bidwill), Laurel, Ligustro disciplinado (Ligustro variegado), Tipa blanca, Palo borracho rosado, Magnolia Almez (Almecino o Almecina), Liquidambar, Árbol del cielo (Ailanto o Árbol de los dioses) ,Naranjo amargo, Lapacho amarillo, Sófora japónica, Álamo plateado ,Pino del Paraná (Pino de Misiones o Pino de Brasil) ,Ginkgo, Morera blanca, Encina, Pino de

las canarias, Falso alerce, Ombú Coronillo, Robusta Washingtonia (Palmera washingtonia), Palma de california, Pindó Arrayán (Anacahuita), Tejo chino, Quebracho colorado, Roble palustre, Crespón (Àrbol de Júpiter), Arbol de las orquideas, Álamo carolina, Eucalipto sideroxylon, Ligustro Cefalotaxus, Cedro del Himalaya, Fotinia Cica, Roble sedoso (Grevillea), Eucalipto (Eucalipto común) y Ciprés

1.1.45 Ejercicio 8

Los datos `TITANIC3` del paquete `PASWR2` contienen información sobre los pasajeros del Titanic, incluyendo clase, sexo y si sobrevivieron o no, entre otras características.

1.1.46 Ejercicio 8.a

Determine la **proporción de sobrevivientes por clase**.

1.1.47 Solución 8.a

[75]:
```
# Se necesita para ejercicios 8 y 9
install.packages("PASWR2")
```

```
Updating HTML index of packages in '.Library'
Making 'packages.html' … done
```

[76]:
```
# Importo librería PASWR2
require(PASWR2)
```

[63]:
```
# Verifico que cargue los datos
head(TITANIC3)
```

pclass	survived	name	sex	age	sibsp	parch	ticket	fare	ca
1st	1	Allen, Miss. Elisabeth Walton	female	29.0000	0	0	24160	211.3375	B5
1st	1	Allison, Master. Hudson Trevor	male	0.9167	1	2	113781	151.5500	C2
1st	0	Allison, Miss. Helen Loraine	female	2.0000	1	2	113781	151.5500	C2
1st	0	Allison, Mr. Hudson Joshua Crei	male	30.0000	1	2	113781	151.5500	C2
1st	0	Allison, Mrs. Hudson J C (Bessi	female	25.0000	1	2	113781	151.5500	C2
1st	1	Anderson, Mr. Harry	male	48.0000	0	0	19952	26.5500	E1

1.1.48 Ejercicio 8.b

Calcule la **proporción de sobrevivientes por clase y sexo**.

¿Quién tuvo una tasa **más alta** de supervencia: los **varones de 1ra clase** o las **mujeres de 3ra**?

1.1.49 Solución 8.b

Filtro pasajeros por **varones de 1ra clase** (`sex=="male" & pclass=="1st"`) o **mujeres de 3ra** (`sex=="female" & pclass=="3rd"`), y ademas cuento sobrevivientes en cada uno (`survived==1`), para calcular la tasa:

[64]:
```
# Males 1st class
total_males    <- nrow(TITANIC3[TITANIC3$sex=="male" & TITANIC3$pclass=="1st",])
survived_males <- nrow(TITANIC3[TITANIC3$sex=="male" & TITANIC3$pclass=="1st" &TITANIC3$survived==1 ,])
rate_males     <- survived_males/total_males
# Females 3rd class
total_females    <- nrow(TITANIC3[TITANIC3$sex=="female" &TITANIC3$pclass=="3rd",])
survived_females <- nrow(TITANIC3[TITANIC3$sex=="female" &TITANIC3$pclass=="3rd" & TITANIC3$survived==1 ,])
rate_females     <- survived_females/total_females
```

[65]:
```
rate_males
rate_females
```

0.340782122905028

0.490740740740741

Los datos muestran que los **varones de 1era clase** tuvieron una tasa se supervivencia del ≈**34.08%**, mientras que las **mujeres de 3era clase** tuvieron una tasa de supervivencia de ≈**49.07%**.

Por lo tanto, la **tasa más alta de supervivencia** la tuvo el grupo de **mujeres de 3era clase**.

1.1.50 Ejercicio 8.c

¿Cuál era la **edad** de la **mujer más grande** que **sobrevivió**?

1.1.51 Solución 8.c

Dado que hay **muchos valores nulos** (*NA*) entre las edades de los pasajeros en los registros, es necesario hacer explícita la opción `na.rm` del comando `max`, que **calcula el máximo valor solo entre los valores no-nulos**.

[66]:
```
max(TITANIC3[TITANIC3$sex=="female" & TITANIC3$survived==1 , "age"], na.rm=TRUE)
```

76

La edad de la mujer más grande que sobrevivió era de **76 años**

1.1.52 Ejercicio 9.

El conjunto de datos `CARS2004` del paquete `PASWR2` contiene datos de automóviles en Europa del año 2004.

(ver `help(CARS2004)`)

1.1.53 Ejercicio 9.a

Calcular **cantidad total de autos** en cada país.

1.1.54 Solución 9.a

El conjunto de datos `CARS2004` posee la cantidad de autos organizados por países entre sus variables, aunque **por cada 1000 habitantes**.

La población también está normalizada sobre 1000, por lo que para obtener la **cantidad total** de automóviles, debemos multiplicar `cars` por `population`.

De la documentación:

> *cars (number of cars per 1000 inhabitants)*
>
> *population (country population/1000)*

[119]:
```
total_cars <- CARS2004$cars * CARS2004$population
cars_per_country <- data.frame(CARS2004$country, total_cars)
colnames(cars_per_country) <- c("country", "cars")
cars_per_country
```

country	cars
Belgium	4854932
Czech Republic	3809076
Denmark	1910892
Germany	45062472
Estonia	472850
Greece	3842268
Spain	19224630
France	29411391
Ireland	1550780
Italy	33632928
Cyprus	327040
Latvia	688743
Lithuania	1323264
Luxembourg	297868
Hungary	2832760
Malta	210000
Netherlands	6974682
Austria	4065114
Poland	11991974
Portugal	5991700
Slovenia	910176
Slovakia	1194360
Finland	2338560
Sweden	4093056
United Kingdom	27618876

La **cantidad total de automóviles por países** se encuentra en la tabla anterior.

1.1.55 Ejercicio 9.b

Calcular la **tasa de mortalidad de automovilistas** para cada país como el **número total de muertes de automovilistas dividido el número total de autos**.

1.1.56 Solución 9.b

Puedo extraer las variables necesarias de la tabla en forma de **vector**, y operar directamente con ellos ya que mantienen el orden original y son **operaciones elemento a elemento**.

[120]:
```
tasas_mortalidad <- CARS2004$deaths / total_cars
# Agrego columna de países y renombro
tasa <- data.frame(CARS2004$country, tasas_mortalidad)
colnames(tasa) <- c("country","dead_rate")
tasa
```

country	dead_rate
Belgium	2.306932e-05
Czech Republic	3.544167e-05
Denmark	3.558548e-05
Germany	1.575590e-06
Estonia	2.664693e-04
Greece	3.825865e-05
Spain	5.825860e-06
France	3.128040e-06
Ireland	6.061466e-05
Italy	2.884078e-06
Cyprus	4.892368e-04
Latvia	3.223263e-04
Lithuania	1.647441e-04
Luxembourg	3.659339e-04
Hungary	4.518561e-05
Malta	1.571429e-04
Netherlands	7.025410e-06
Austria	2.656752e-05
Poland	1.250837e-05
Portugal	2.069530e-05
Slovenia	1.505203e-04
Slovakia	9.377407e-05
Finland	3.078818e-05
Sweden	1.294876e-05
United Kingdom	2.027599e-06

1.1.57 Ejercicio 9.c

Hacer un gráfico de barras que indique la **tasa de mortalidad en accidentes de tránsito** de cada país de la Unión Europea.

Ordene las barras de forma **creciente**.

1.1.58 Solución 9.c

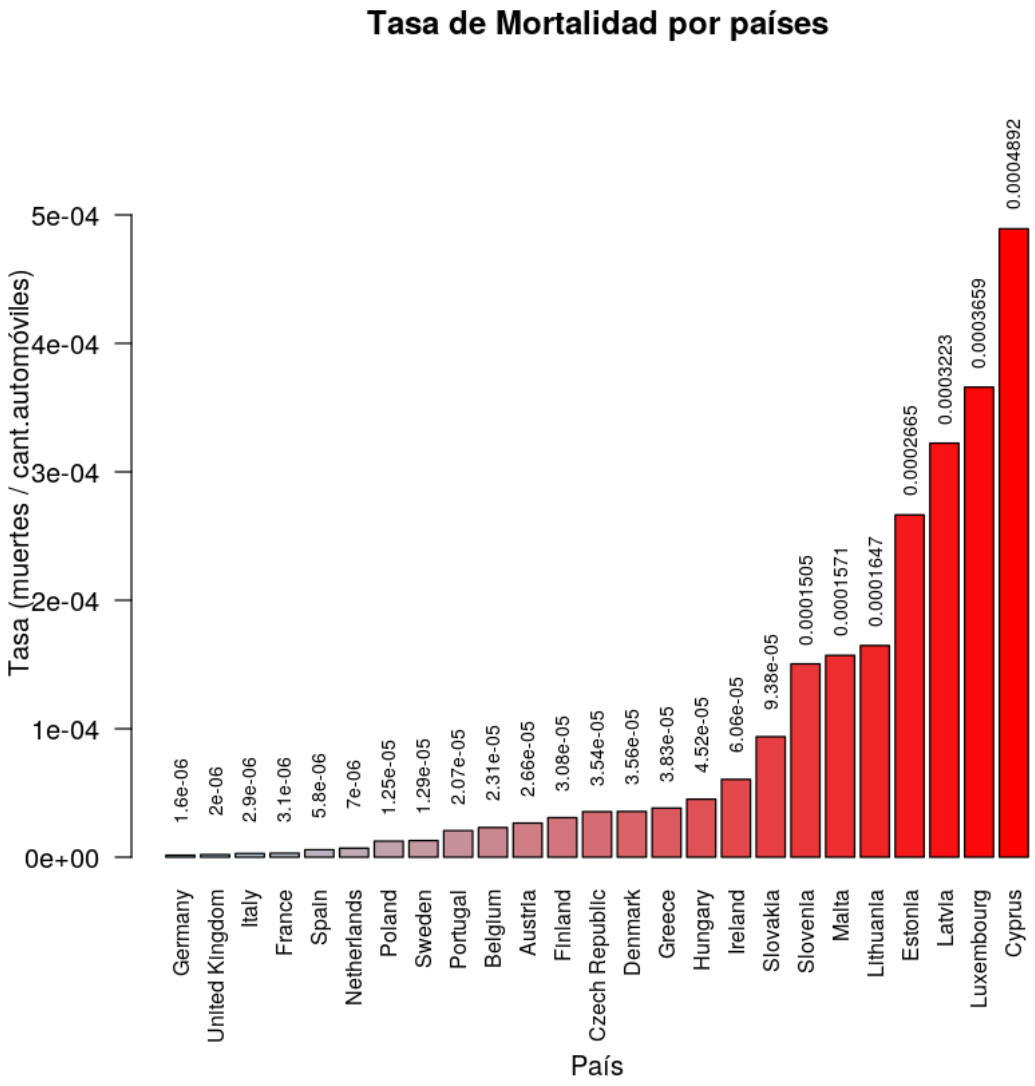
[121]:
```
tasa <- tasa[order(tasa$dead_rate),]

# Agrando margen inferior
par(mar=c(7,4,4,2))

# Degrade de colores
# NOTA: NO proporcionales a las alturas!
color.function <- colorRampPalette( c("lightblue" , "red") )
color.ramp <- color.function( n=nrow(tasa) )

# Grafico de barras
```

```r
bb <- barplot(height=tasa$dead_rate, names.arg=tasa$country,
              las=2, cex.names=0.8,
              ylim=c(0, 0.0006), yaxt = "n",
              main="Tasa de Mortalidad por países",
              ylab="Tasa (muertes / cant.automóviles)",
              col=color.ramp)
text(bb, round(tasa$dead_rate, 7)+0.00005, round(tasa$dead_rate, 7), srt=90,␣
 ↪cex=0.7)
# Ticks en eje y
axis(side=2, at=seq(0,0.0005,0.0001), las=2)
# Eje x
mtext("País", side=1, line=6)
```



1.1.59 Ejercicio 9.d

¿Qué país tiene la **menor tasa de mortalidad** de automovilistas y qué país la más alta?

1.1.60 Solución 9.d

Viendo el gráfico podemos determinar que **Alemania** lidera la **menor tasa**, y **Cyprus** la **más alta**.

Extrayendolo directamente de los datos:

[122]:
```
print("País con tasa mínima:")
tasa[which.min(tasa$dead_rate), c("country", "dead_rate")]
print("País con tasa máxima:")
tasa[which.max(tasa$dead_rate), c("country", "dead_rate")]
```

```
[1] "País con tasa mínima:"
```

	country	dead_rate
4	Germany	1.57559e-06

```
[1] "País con tasa máxima:"
```

	country	dead_rate
11	Cyprus	0.0004892368

Menor tasa de mortalidad: **Alemania**

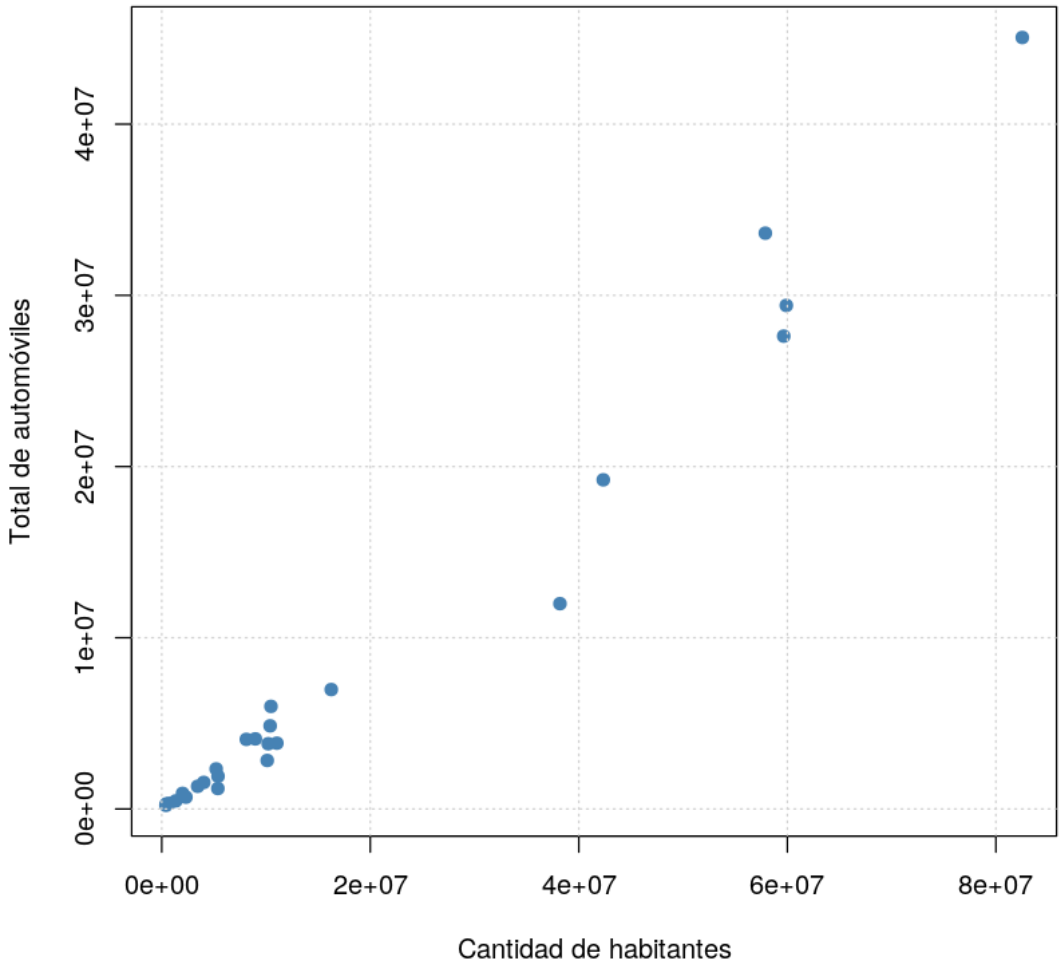
Mayor tasa de mortalidad: **Cyprus**

1.1.61 Ejercicio 9.e

Haga un gráfico de **cantidad total de autos** vs **población**. ¿Cómo describiría la relación?

1.1.62 Solución 9.e

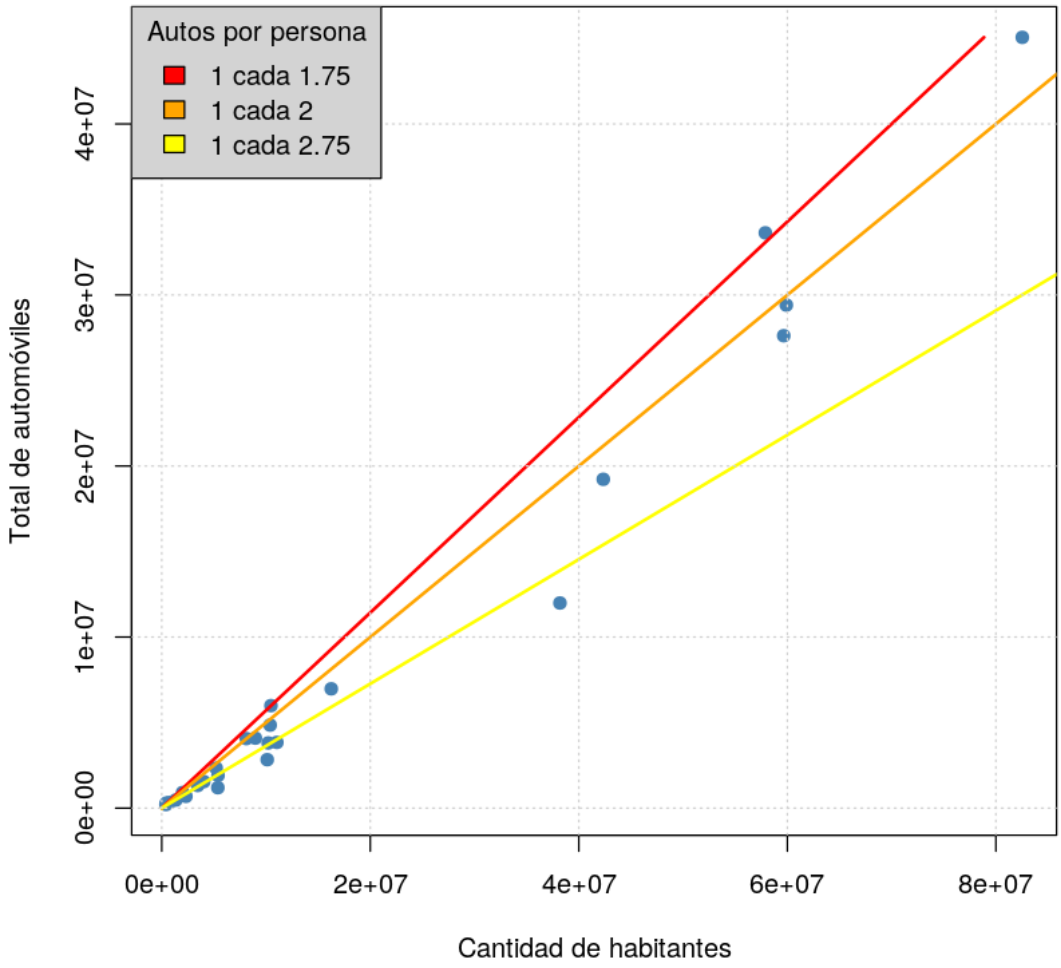
[123]:
```
total_cars <- CARS2004$cars * CARS2004$population
total_pobl <- CARS2004$population * 1000
plot(total_pobl, total_cars,
     xlab="Cantidad de habitantes", ylab="Total de automóviles",
     col="steelblue", pch=19)
grid()
```



Se observa una **relación aproximadamente lineal**, donde habría **1 automóvil por cada 2/2.75 habitantes**, graficada toscamente a continuación:

[124]:
```
plot(total_pobl, total_cars,
     xlab="Cantidad de habitantes", ylab="Total de automóviles",
     col="steelblue", pch=19)
lines(c(0, 1.75*max(total_cars)), c(0, max(total_cars)), col="red", lwd=2)
lines(c(0, 2*max(total_cars)), c(0, max(total_cars)), col="orange", lwd=2)
lines(c(0, 2.75*max(total_cars)), c(0, max(total_cars)), col="yellow", lwd=2)
grid()
legend("topleft", title="Autos por persona",
       legend=c("1 cada 1.75","1 cada 2","1 cada 2.75"),
       fill=c("red","orange","yellow"),
```

```
        bg="lightgray")
```



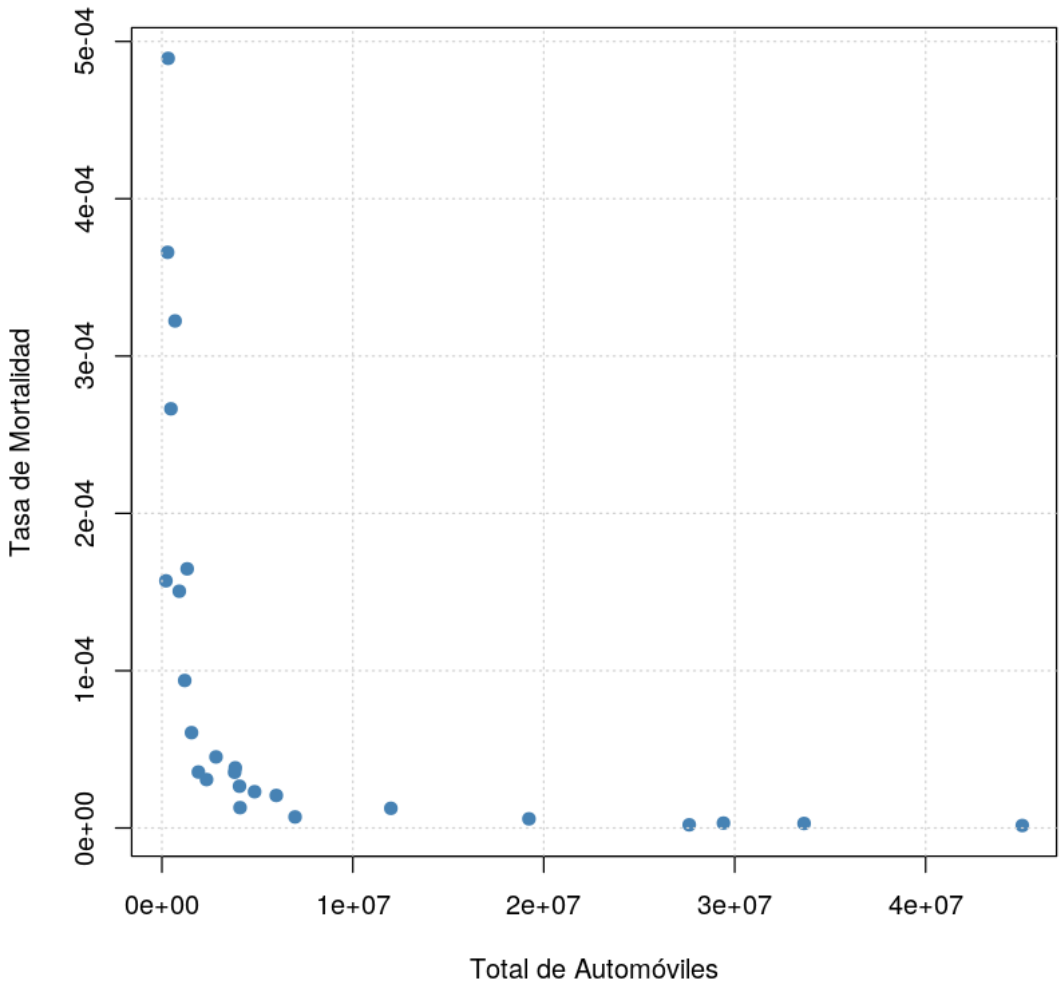
1.1.63 Ejercicio 9.f

Haga un gráfico de **cantidad total de autos** vs **tasa de mortalidad de automovilistas**. ¿Cómo describiría la relación?

1.1.64 Solución 9.f

[125]:
```
cars_vs_deadrate <- data.frame(total_cars, tasas_mortalidad)
colnames(cars_vs_deadrate) <- c("cars","dead_rate")
```

[126]:
```
plot(cars_vs_deadrate,
     xlab="Total de Automóviles", ylab="Tasa de Mortalidad",
     col="steelblue", pch=19)
grid()
```



Se observa una relación poco intuitiva en cuanto a las dos variables.

Por un lado, a partir de los 10.000 automóviles, la tasa de mortalidad se mantiene muy baja y casi constante hasta llegar a casi 50.000 (5 veces más!)

Por otro lado, para valores menores a 10.000 automoviles, la tasa de mortalidad crece rápidamente, inversamente proporcional a la misma, alcanzando valores máximos con menos de 2000 automóviles.

A partir de estos datos, uno *podría* (de manera **incorrecta!**) interpretar que

> ***"a mayor cantidad de automóviles en un país, menor la tasa de mortalidad, y a menor cantidad, mayor tasa de mortalidad"***

Y a partir de ello, **reducir la tasa de mortalidad mundial muy facilmente**: *Fabricar miles de millones de automóviles y reducir el precio de los mismos!.*

Es claro que este no es el caso, y analisis deja afuera muchas otras variables que influyen de forma directa e indirecta en las causas y de la tasa de mortalidad:

- Tamaño del país
- Densidad de población
- Estructura y planeamiento vial
- Transporte público
- Principales actividades laborales
- Nivel de educación de la población

fin