Estadística Computacional

Juan Carlos Martínez Ovando
juan.martinez[at]banxico.org.mx
JC.Martinez.Ovando[at]gmail.com

agosto--diciembre, 2014

Introducción a R: Instalación de paquetes

```
# Instalación de paquetes

pkgspath <- "C:/JCMO.Research/Academia/Cursos/2014_EstadisticaComputacional/paquetesR/"

install.packages(c(
   paste(pkgspath, "ade4_1.5-0.zip", sep = ""),
   paste(pkgspath, "bayesm_2.2-5.zip", sep = ""),
   paste(pkgspath, "bayesm_2.2-5.zip", sep = ""),
   paste(pkgspath, "BMA_3.15.2.zip", sep = ""),
   paste(pkgspath, "coda_0.14-7.zip", sep = ""),
   paste(pkgspath, "dlm_1.1-2.zip", sep = ""),
   paste(pkgspath, "dma_1.1.zip", sep = ""),
   paste(pkgspath, "dynlm_0.3-1.zip", sep = ""),
   paste(pkgspath, "ggplot2_0.9.3.1.zip", sep = ""),
   paste(pkgspath, "predfinitepop_1.0.zip", sep = "")
   ),
 repos = NULL,
 lib = "C:/JCMO.Research/Coding/_r/libs",
 type="source")
```

Contenido

Una breve descripción de R

R es un lenguaje de programación orientada a objetos (véase Müller (1997)), desarrollado con particular énfasis en realizar análisis estadístico. R trabaja a través de una infraestructura en la que se integran diferentes colecciones o librerías de funciones, conocidos como 'paquetes' (packages), que contribuyen a producir, cargar y almacenar software de diferentes autores y fuentes.

El manual de Venables et al. (2004) es la principal referencia para aprender R. Esta nota no pretende ser un manual de usuario de R, su propósito es el de mostrar y resumir algunas características y funciones de R que son de utilidad para aquellos principiantes en este lenguaje.

¿Qué PUEDE hacer?

- Manipulación y almacenamiento de datos, de dos tipos: numéricos y de caracteres o texto.
- Realiza cálculos y operaciones matriciales
- Manipula tablas y expresiones regulares
- Es posible realizar análisis de datos y procedimientos estadísticos (algunos de ellos sofisticados)
- Permite definir y manipular clases de objetos ("OO")
- Gráficas
- Provee un ambiente de programación, con: loops, branching, subrutinas, condiciones, etc.

¿Qué NO PUEDE hacer?

- No es un administrador de base de datos, pero EXISTEN paquetes que permiten conectar R a algunos administradores de bases de datos (DBMSs)
- No es una interface gráfica, pero permite conexiones a Java, TclTk
- No es un interprete de lenguajes muy eficiente, de hecho puede ser bastante lento, pero permite invocar rutinas de C/C++, Matlab, y de algunos otros lenguajes
- No permite ver hojas de datos, pero permite conexiones a Excel/MsOffice
- No provee soporte comercial o profesional, pero existe un grupo de discusión muy activo en el que usuarios y miembros del grupo de desarrollo presentan y aclaran problemas específicos

Miscelánea

Usualmente los paréntesis, '()', son usados para delimitar los argumentos de funciones, mientras que las llaves, '{}' son usadas para delimitar las operaciones de funciones, de ciclos, o de condiciones. Los corchetes, '[]', se utilizan para indicar la posición de algunas de las entradas en arreglos vectoriales o matriciales.

`q()`

Salida de la aplicación R. Si se desea salir directamente contestando no a la pregunta de almacenamiento de las variables sesión, escriba q(``no") en la ventana de comandos.

`<-`

Asignación (siempre de izquierda a derecha)

- INSTALL package1
Instalación del paquete 'package1' a través de la conexión URL de CRAN. también puede emplearse la instrucción install.packages('package1').

NA
Esta es la etiqueta que da R a los datos nulos.

Ayuda

help(command1)
Obtiene ayuda del comando o función command1. Esta instrucción puede abreviarse como ?command1.

help.start()
Inicia el navegador de ayuda

help(package=mva)
Despliega la ayuda del paquete mva, junto con la lista de contenido.

apropos("topic1")
Despliega los comandos relevantes asociados (o invocados) a topic1.

example(command1)
Despliega los ejemplos de la documentaci´on del comando command1.

1. Ejemplos de Objetos

```
# Una sola variable

y <- 39
y

[1] 39

43 -> y
y

[1] 43

z <- 5
w <- z^2
w

[1] 25

i <- (z*2 + 45)/2
i

[1] 27.5

(w <- z^2)
```

```
[1] 25

(34 + 90)/12.5

[1] 9.92

ls()

[1] "apiclus1" "apiclus2" "apipop"   "apistrat" "dmu284"   "i"
[7] "mu284"    "w"        "y"         "ytotal"   "z"

# la función 'rm' remueve o elimina 'objetos' del ambiente
rm(y)

rm(z,w,i)

# así...
ls()

[1] "apiclus1" "apiclus2" "apipop"   "apistrat" "dmu284"   "mu284"
"ytotal"
```

2. Ejemplos de Vectores

```
v <- c(4,7,23.5,76.2,80)
v
```

```
[1]  4.0  7.0 23.5 76.2 80.0
```

```
# Longitud de un vector
length(v)
```

```
[1] 5
```

```
# La función 'mode' regresa el tipo de 'objeto', es decir, el formato con el
# que está almacenado en la memoria de R
mode(v)
```

```
[1] "numeric"
```

```
# Se define un vector de caracteres
v <- c(4,7,23.5,76.2,80,"rrt")
v
```

```
[1] "4"    "7"    "23.5" "76.2" "80"   "rrt"
```

```
# Se define un vector de caracteres con una entrada "nula" o "vacía"
v <- c(NA,"rrr")
v
```

```
[1] NA    "rrr"

# Se define un vector numérico con una entrada “nula” o “vacía”
u <- c(4,6,NA,2)
u

[1]  4  6 NA  2

# Se define un vector lógico (FALSO o VERDADERO) con una entrada “nula” o
“vacía”
k <- c(T,F,NA,TRUE)
k

[1]  TRUE FALSE    NA  TRUE

# Se despliega la segunda entrada del vector “v”
v[2]

[1] "rrr"

# modificamos la primera entrada del vector “v”
v[1] <- 'hello'
v

[1] "hello" "rrr"
```

```
# creamos un vector "v" con atributos abiertos (para ser modificados
# arbitrariamente)
v <- vector()
v

logical(0)

mode(v)

[1] "logical"

# modificamos la tercera entrada del vector "v", y por consiguiente definimos
# automáticamente sus atributos
v[3] <- 45
v

[1] NA NA 45

mode(v)

[1] "numeric"

# Otro vector
v <- c(45,243,78,343,445,645,2,44,56,77)
v

[1]  45 243  78 343 445 645   2  44  56  77
```

```
# Redefinimos el vector a partir del vector mismo
v <- c(v[5],v[7])
v

[1] 445   2
```

3. Operaciones con vectores

```
#   Raíz cuadrada entrada por entrada
v <- c(4,7,23.5,76.2,80)
x <- sqrt(v)
x

[1] 2.000000 2.645751 4.847680 8.729261 8.944272

#   Suma directa
v1 <- c(4,6,87)
v2 <- c(34,32.4,12)
v1+v2

[1] 38.0 38.4 99.0

#   Suma por ciclos
v1 <- c(4,6,8,24)
v2 <- c(10,2)
```

```
v1+v2

[1] 14  8 18 26

#   Otra suma por ciclos
v1 <- c(4,6,8,24)
v2 <- c(10,2,4)
v1+v2

[1] 14  8 12 34
Warning message:
longer object length
        is not a multiple of shorter object length in: v1 + v2

#   Producto escalar
v1 <- c(4,6,8,24)
2*v1

[1]  8 12 16 48
```

4. Factores

```
# Definición de un vector de categorías (caracteres)
g <- c('f','m','m','m','f','m','f','m','f','f')
g
```

```
[1] "f" "m" "m" "m" "f" "m" "f" "m" "f" "f"

# Se convierte en vector "g" en un vector de factores (categorías)
g <- factor(g)
g

[1] f m m m f m f m f f
Levels: f m

# otra forma de definir un vector "g" en un vector de factores (categorías)
other.g <- factor(c('m','m','m','m','m'),levels=c('f','m'))
other.g

[1] m m m m m
Levels: f m

# Calcula una tabla con las frecuencias de los factores en "g"
table(g)

g
f m
5 5

# Calcula una tabla con las frecuencias de los factores en "other.g"
table(other.g)

other.g
```

```
f m
0 5

# Se define un vector de factores de la misma longitud de “g”
a <- factor(c('adulto','adulto','joven','joven','adulto','adulto',
       'adulto','joven','adulto','joven'))

# Se calcula la tabla de tabulaciones cruzadas de “a” y “g”
table(a,g)

         g
a         f m
  adulto 4 2
  joven  1 3

# Se calcula la tabla de tabulaciones cruzadas de “a” y “g” en “t”
t <- table(a,g)

# Se calculan las tablas marginales de “t”
margin.table(t,1)

a
adulto  joven
4

margin.table(t,2)
```

```
g
f m
5 5

# Se calculan las tablas de proporciones marginales de "t" respecto a la
primera
# variable
prop.table(t,1)

         g
a         f         m
  adulto 0.6666667 0.3333333
  joven  0.2500000 0.7500000

# Se calculan las tablas de proporciones marginales de "t" respecto a la
primera
# variable
prop.table(t,2)

g
f m
5 5

# Se calcula la tabla cruzada de proporciones "t"
prop.table(t)

         g
```

```
a         f   m
  adulto 0.4 0.2
  joven  0.1 0.3
```

5. Generación de sucesiones

```
x <- 1:1000
```

```
10:15-1
```

```
[1]  9 10 11 12 13 14
```

```
10:(15-1)
```

```
[1] 10 11 12 13 14
```

```
# una sucesión decreciente
5:0
```

```
[1] 5 4 3 2 1 0
```

```
# sucesión con salto fijo
# una sucesión que inicia en -4, termina en 1, con saltos de 0.5 unidades
seq(-4,1,0.5)
```

```
[1] -4.0 -3.5 -3.0 -2.5 -2.0 -1.5 -1.0 -0.5  0.0  0.5  1.0
```

```
# sucesión con longitud fija
seq(from=1,to=5,length=4)

[1] 1.000000 2.333333 3.666667 5.000000seq(from=1,to=5,length=2)

# sucesión con salto fijo
seq(length=10,from=-2,by=.2)

[1] -2.0 -1.8 -1.6 -1.4 -1.2 -1.0 -0.8 -0.6 -0.4 -0.2

# sucesión con repeticiones
rep(5,10)

[1] 5 5 5 5 5 5 5 5 5 5

rep('hi',3)

[1] "hi" "hi" "hi"


# sucesión con subsucesiones repetidas
rep(1:3,2)

[1] 1 2 3 1 2 3

# sucesión de factores
```

```
gl(3,5)

[1] 1 1 1 1 1 2 2 2 2 2 3 3 3 3 3
Levels: 1 2 3

gl(2,5,labels=c('female','male'))

[1] female female female female female male   male   male   male   male
Levels: female male

# sucesión de número pseudoaleatorios de N(0,1)
rnorm(10)

[1]  0.9073465  0.2460138  0.1156773 -0.4120160  1.1834779 -0.9577176
[7]  0.6039924 -0.3861979  0.6380648  0.1405475

# sucesión de número pseudoaleatorios de N(mean,sd^2)
rnorm(10,mean=10,sd=3)

[1] 17.156659  6.407150 12.861801 12.123106  6.391464 12.122609 14.266385
[8] 18.472419  9.927460 13.944400

# sucesión de número pseudoaleatorios de t(gl=df)
rt(5,df=10)

[1]  0.4348921  0.1308199  1.6649053 -1.1361502  0.8136867
```

```
# Manipulación con índices

x <- c(0,-3,4,-1,45,90,-5)
x

[1]  0 -3  4 -1 45 90 -5

# Operación lógica
x > 0

[1] FALSE FALSE  TRUE FALSE  TRUE  TRUE FALSE

# Operación lógica
y <- x>0
y

[1] FALSE FALSE  TRUE FALSE  TRUE  TRUE FALSE

# extraction de “x” de los indices en “y” verdaderos
x[y]

[1]  4 45 90

# idem.
x[x>0]

[1]  4 45 90
```

```
# Semejante pero con uniones de condiciones
x[x <= -2 | x > 5]

[1] -3 45 90 -5

# Semejante pero con intersecciones de condiciones
x[x > 40 & x < 100]

[1] 45 90

# extracción de las entradas 4 y 6 de "x"
x[c(4,6)]

[1] -1 90

x[1:3]

[1]  0 -3  4

# "x" pero sin la l-ésima entrada
x[-1]

[1] -3  4 -1 45 90 -5

x[-c(4,6)]
```

```
[1]  0 -3  4 45 -5

x[-(1:3)]

[1] -1 45 90 -5

# Asignación de nombres a los casos
pH <- c(4.5,7,7.3,8.2,6.3)
names(pH) <- c('area1','area2','mud','dam','middle')
pH

area1  area2    mud    dam middle
   4.5    7.0    7.3    8.2    6.3

pH['mud']

mud
7.3

pH[c('area1','dam')]

area1   dam
  4.5 8.2
```

6. Arreglos y Matrices

```
# Construimos una matriz
m <- c(45,23,66,77,33,44,56,12,78,23)
m

[1] 45 23 66 77 33 44 56 12 78 23

# definimos las dimensiones de "m"
dim(m) <- c(2,5)
m


     [,1] [,2] [,3] [,4] [,5]
[1,]   45   66   33   56   78
[2,]   23   77   44   12   23

# otra forma de definir las misma matriz
m <- matrix(c(45,23,66,77,33,44,56,12,78,23),2,5)
m

     [,1] [,2] [,3] [,4] [,5]
[1,]   45   66   33   56   78
[2,]   23   77   44   12   23

# los mismos datos, pero leídos en orden distinto
m <- matrix(c(45,23,66,77,33,44,56,12,78,23),2,5,byrow=T)
```

```
m
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]   45   23   66   77   33
[2,]   44   56   12   78   23
```

```
# extracciones
m[-2,2]
```

```
[1] 23
```

```
m[1,-c(3,5)]
```

```
[1] 45 23 77
```

```
# renglón uno
m[1,]
```

```
[1] 45 23 66 77 33
```

```
# columna uno
m[,4]
```

```
[1] 77 78
```

```
# otra matriz
m1 <- matrix(c(45,23,66,77,33,44,56,12,78,23),2,5)
```

```
m1
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]   45   66   33   56   78
[2,]   23   77   44   12   23
```

```
# combinación de columnas
cbind(c(4,76),m1[,4])
```

```
     [,1] [,2]
[1,]    4   56
[2,]   76   12
```

```
# matriz de réplicas de datos
m2 <- matrix(rep(10,50),10,5)
m2
```

```
      [,1] [,2] [,3] [,4] [,5]
 [1,]   10   10   10   10   10
 [2,]   10   10   10   10   10
 [3,]   10   10   10   10   10
 [4,]   10   10   10   10   10
 [5,]   10   10   10   10   10
 [6,]   10   10   10   10   10
 [7,]   10   10   10   10   10
 [8,]   10   10   10   10   10
 [9,]   10   10   10   10   10
```

```
[10,]   10   10   10   10   10

# combinación de renglones
m3 <- rbind(m1[1,],m2[5,])
m3

     [,1] [,2] [,3] [,4] [,5]
[1,]   45   66   33   56   78
[2,]   10   10   10   10   10

# arreglos multidimensionales
a <- array(1:50,dim=c(2,5,5))
a

, , 1

     [,1] [,2] [,3] [,4] [,5]
[1,]    1    3    5    7    9
[2,]    2    4    6    8   10

, , 2

     [,1] [,2] [,3] [,4] [,5]
[1,]   11   13   15   17   19
[2,]   12   14   16   18   20

, , 3
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,]   21   23   25   27   29
[2,]   22   24   26   28   30

, , 4

     [,1] [,2] [,3] [,4] [,5]
[1,]   31   33   35   37   39
[2,]   32   34   36   38   40

, , 5

     [,1] [,2] [,3] [,4] [,5]
[1,]   41   43   45   47   49
[2,]   42   44   46   48   50
```

```
a[1,5,2]
```

```
[1] 19
```

```
a[1,,4]
```

```
[1] 31 33 35 37 39
```

```
a[1,3,]
```

```
[1]  5 15 25 35 45
```

```
a[,c(3,4),-4]
```

```
, , 1

     [,1] [,2]
[1,]    5    7
[2,]    6    8

, , 2

     [,1] [,2]
[1,]   15   17
[2,]   16   18

, , 3

     [,1] [,2]
[1,]   25   27
[2,]   26   28

, , 4

     [,1] [,2]
[1,]   45   47
```

```
[2,]   46   48

a[1,c(1,5),-c(4,5)]

     [,1] [,2] [,3]
[1,]    1   11   21
[2,]    9   19   29

# Operaciones matriciales
m <- matrix(c(45,23,66,77,33,44,56,12,78,23),2,5)
m

     [,1] [,2] [,3] [,4] [,5]
[1,]   45   66   33   56   78
[2,]   23   77   44   12   23

m*3

     [,1] [,2] [,3] [,4] [,5]
[1,]  135  198   99  168  234
[2,]   69  231  132   36   69

m1 <- matrix(c(45,23,66,77,33,44),2,3)
m1

     [,1] [,2] [,3]
[1,]   45   66   33
```

```
[2,]   23   77   44
```

```
m2 <- matrix(c(12,65,32,7,4,78),2,3)
m2
```

```
     [,1] [,2] [,3]
[1,]   12   32    4
[2,]   65    7   78
```

```
m1+m2
```

```
     [,1] [,2] [,3]
[1,]   57   98   37
[2,]   88   84  122
```

7. Listas

```
# creación de una lista
my.lst <- list(stud.id=34453, stud.name="John", stud.marks=c(14.3,12,15,19))
my.lst
```

```
$stud.id
[1] 34453
```

```
$stud.name
[1] "John"

$stud.marks
[1] 14.3 12.0 15.0 19.0

my.lst[[1]]

[1] 34453

my.lst[[3]]

[1] 14.3 12.0 15.0 19.0

my.lst[1]

$stud.id
[1] 34453

my.lst$stud.id

[1] 34453

names(my.lst)

[1] "stud.id"    "stud.name"  "stud.marks"
```

```
# otra forma de crear una lista
names(my.lst) <- c('id','name','marks')
my.lst

$id
[1] 34453

$name
[1] "John"

$marks
[1] 14.3 12.0 15.0 19.0

# edición de un elemento de una lista
my.lst$parents.names <- c("Ana","Mike")
my.lst

$id
[1] 34453

$name
[1] "John"

$marks
[1] 14.3 12.0 15.0 19.0
```

```
$parents.names
[1] "Ana"  "Mike"

# longitud (número de elementos) de una lista
length(my.lst)

[1] 4

# otra lista
other <- list(age=19,sex='male')
other

$age
[1] 19

$sex
[1] "male"

# concatenación de listas
lst <- c(my.lst,other)
lst

$id
[1] 34453

$name
[1] "John"
```

```
$marks
[1] 14.3 12.0 15.0 19.0

$parents.names
[1] "Ana"  "Mike"

$age
[1] 19

$sex
[1] "male"

# enlista las entradas de todos los elementos de una lista, entrada por
entrada
unlist(my.lst)

           id          name         marks1         marks2         marks3
      "34453"        "John"         "14.3"           "12"           "15"
       marks4 parents.names1 parents.names2
         "19"         "Ana"         "Mike"
```

8. 'Data Frames'

```
#definimos un conjunto de datos (en forma de tabla)
```

```
my.dataset <-
data.frame(site=c('A','B','A','A','B'),season=c('Winter','Summer','Summer','S
pring','Fall'),pH = c(7.4,6.3,8.6,7.2,8.9))
my.dataset
```

```
  site season  pH
1    A Winter 7.4
2    B Summer 6.3
3    A Summer 8.6
4    A Spring 7.2
5    B   Fall 8.9
```

```
# exploramos la tabla por entrada
my.dataset[3,2]
```

```
[1] Summer
Levels: Fall Spring Summer Winter
```

```
# exploramos la tabla por columnas
my.dataset$pH
```

```
[1] 7.4 6.3 8.6 7.2 8.9
```

```
# extraemos una subtabla, de acuerdo a una condición
my.dataset[my.dataset$pH > 7,]
```

```
  site season  pH
1    A Winter 7.4
3    A Summer 8.6
4    A Spring 7.2
5    B   Fall 8.9

# idem
my.dataset[my.dataset$site == 'A','pH']

[1] 7.4 8.6 7.2

# idem
my.dataset[my.dataset$season == 'Summer',c('site','pH')]

  site  pH
2    B 6.3
3    A 8.6

# asigna atributos (caracteres o numéricos) a cada una de las columnas de una
# tabla; nombres de columnas, etc.
attach(my.dataset)

# resumen de una tabla
summary(my.dataset)

site     season        pH
```

```
 A:3   Fall  :1   Min.   :6.30
 B:2   Spring:1   1st Qu.:7.20
       Summer:2   Median :7.40
       Winter:1   Mean   :7.68
                  3rd Qu.:8.60
                  Max.   :8.90
```

```
# exploración de una tabla por medio de los nombres de las columnas,
cumpliendo
# una condición - después de 'attach'
my.dataset[pH > 8,]
```

```
  site season  pH
3    A Summer 8.6
5    B   Fall 8.9
```

```
# exploración de una tabla por medio de los nombres de las columnas,
cumpliendo
# una condición - después de 'attach'
season
```

```
[1] Winter Summer Summer Spring Fall
Levels: Fall Spring Summer Winter
```

```
detach(my.dataset)
```

```
season

[1] Winter Summer Summer Spring Fall
Levels: Fall Spring Summer Winter

# combinación de tables y listas
my.dataset$NO3 <- c(234.5,256.6,654.1,356.7,776.4)
my.dataset

  site season  pH   NO3
1    A Winter 7.4 234.5
2    B Summer 6.3 256.6
3    A Summer 8.6 654.1
4    A Spring 7.2 356.7
5    B   Fall 8.9 776.4

nrow(my.dataset)

[1] 5

ncol(my.dataset)

[1] 4

# despliega la tabla en un ambiente de hoja de trabajo tipo Excel,
# para su edición
my.dataset <- edit(my.dataset)
```

```
# idem., para crear una tabla
new.data <- edit(data.frame())

# despliega los nombres de la tabla
names(my.dataset)

[1] "site"   "season" "pH"     "NO3"

# asigna los nombres de la tabla
names(my.dataset) <- c("area","season","pH","NO3" )
my.dataset

  area season  pH   NO3
1    A Winter 7.4 234.5
2    B Summer 6.3 256.6
3    A Summer 8.6 654.1
4    A Spring 7.2 356.7
5    B   Fall 8.9 776.4


# idem.
names(my.dataset)[4] <- "PO4"
my.dataset
```

```
  area season  pH   PO4
1    A Winter 7.4 234.5
2    B Summer 6.3 256.6
3    A Summer 8.6 654.1
4    A Spring 7.2 356.7
5    B   Fall 8.9 776.4
```

```
# despliega todos los datos almacenados en R
data()

# carga los datos "USArrests"
data(USArrests)
```

9. Funciones

```
# creación de funciones
# calcula la varianza
se <- function(x)
  {
  v <- var(x)
  n <- length(x)
  return(sqrt(v/n))
  }

# aplica la function a un vector de datos
```

```
se(c(45,2,3,5,76,2,4))

[1] 11.10310

# funciones dentro de funciones
# la función calcula las estadísticas básicas de un vector
# condicionando si se desea una versión extendida o no
basic.stats <- function(x,more=F)
  {
  stats <- list()
  clean.x <- x[!is.na(x)]
  stats$n <- length(x)
  stats$nNAs <- stats$n-length(clean.x)
  stats$mean <- mean(clean.x)
  stats$std <- sd(clean.x)
  stats$med <- median(clean.x)
  if (more)
    {
    stats$skew <- sum(((clean.x-stats$mean)/stats$std)^3)/length(clean.x)
    stats$kurt <- sum(((clean.x-stats$mean)/stats$std)^4)/length(clean.x) - 3
    }
  stats
 }

#aplica la function a un vector de datos
basic.stats(c(45,2,4,46,43,65,NA,6,-213,-3,-45))
```

```
$n
[1] 11

$nNAs
[1] 1

$mean
[1] -5


$std
[1] 79.87768

$med
[1] 5

# idem, pero en una version extendida
basic.stats(c(45,2,4,46,43,65,NA,6,-213,-3,-45),more=T)

$n
[1] 11

$nNAs
[1] 1

$mean
[1] -5
```

```
$std
[1] 79.87768

$med
[1] 5

$skew
[1] -1.638217

$kurt
[1] 1.708149


# otra function con ciclos
f <- function(x)
     {
      for(i in 1:10)
         {
          res <- x*i
          cat(x,'*',i,'=',res,'\n')
          }
      }

# aplica la función a un vector
f(c(45,2,4,NA,6,-213))
```

```
45 2 4 NA 6 -213 * 1 = 45 2 4 NA 6 -213
45 2 4 NA 6 -213 * 2 = 90 4 8 NA 12 -426
45 2 4 NA 6 -213 * 3 = 135 6 12 NA 18 -639
45 2 4 NA 6 -213 * 4 = 180 8 16 NA 24 -852
45 2 4 NA 6 -213 * 5 = 225 10 20 NA 30 -1065
45 2 4 NA 6 -213 * 6 = 270 12 24 NA 36 -1278
45 2 4 NA 6 -213 * 7 = 315 14 28 NA 42 -1491
45 2 4 NA 6 -213 * 8 = 360 16 32 NA 48 -1704
45 2 4 NA 6 -213 * 9 = 405 18 36 NA 54 -1917
45 2 4 NA 6 -213 * 10 = 450 20 40 NA 60 -2130
```

10. Importación y exportación de datos (paquete 'foreign')

```
# Ejecuta los comandos del archivo de comandos file1.r. Si file1.r no está
# almacenado en la carpeta de trabajo de R, puede ejecutarse incluyendo
# dentro del argumento de la función su ruta física,
# e.g. source("D://Mis ocumentos//...//file1").

source("file1")

# Carga un conjunto de datos desde archivo file1. Los datos
# no admiten campos vac´ıos, si existen, deben ser recodificados por fuera de
# R, e.g. con valor '-1' y al cargarlo en R,
# incluir el argumento na.strings = 1".
# Esta función es semejante a scan() para vectores.
```

```
read.table(file, header = FALSE, sep = "", quote = "\"'", dec = ".",
                 row.names, col.names, as.is = FALSE, na.strings = "NA",
                 colClasses = NA, nrows = -1,
                 skip = 0, check.names = TRUE, fill = !blank.lines.skip,
                 strip.white = FALSE, blank.lines.skip = TRUE,
                 comment.char = "#")


# Esta función despliega los datos de x (que debe ser un arreglo, tabla
# o matriz) en una hoja de trabajo tipo Excel, para modificar manualmente
#alguno de los registros.

data.entry(x)

# Lee un vector desde el archivo x1.

scan(x1)

# Descarga un archivo de un sitio Web url1 en la direcci´on
especificada.

download.file(url1)

# Despliega archivos desde un servidor remoto o una carpeta Web.

url.show(url1)
```

```
# Escribe un objeto en el archivo file1; puede incluirse la
# ruta donde está almacenado el archivo.

write(object, "file1")
# Escribe la tabla 'dataframe1' en el archivo file1, con un formato
predefinido,
# usualmente o DAT.

x=c(1,2,3)
write.table(x, file = "foo.csv", sep = ",", col.names = NA)

read.table("foo.csv", header = TRUE, sep = ",", row.names=1)

  x
1 1
2 2
3 3

# con 'foreign', podemos leer archivos de diferentes formatos, e.g.
#   DBF

x <- read.dbf(system.file("files/sids.dbf", package="foreign")[1])
x
     AREA PERIMETER CNTY_ CNTY_ID         NAME  FIPS FIPSNO CRESS_ID BIR74
1   0.114     1.442  1825    1825         Ashe 37009  37009        5  1091
2   0.061     1.231  1827    1827    Alleghany 37005  37005        3   487
```

```
3   0.143     1.630  1828    1828        Surry 37171  37171       86  3188
4   0.070     2.968  1831    1831    Currituck 37053  37053       27   508
5   0.153     2.206  1832    1832  Northampton 37131  37131       66  1421
6   0.097     1.670  1833    1833     Hertford 37091  37091       46  1452

# se preserva la estructura de la tabla original

str(x)
`data.frame':   100 obs. of  14 variables:
 $ AREA     : num  0.114 0.061 0.143 0.07 0.153 0.097 0.062 0.091 0.118 0.124 ...
 $ PERIMETER: num  1.44 1.23 1.63 2.97 2.21 ...
 $ CNTY_    : int  1825 1827 1828 1831 1832 1833 1834 1835 1836 1837 ...
 $ CNTY_ID  : int  1825 1827 1828 1831 1832 1833 1834 1835 1836 1837 ...
 $ NAME     : Factor w/ 100 levels "Alamance","Alex..",..: 5 3 86 27 66 46 15 37 93 85 ...
 $ FIPS     : Factor w/ 100 levels "37001","37003",..: 5 3 86 27 66 46 15 37 93 85 ...
 $ FIPSNO   : int  37009 37005 37171 37053 37131 37091 37029 37073 37185 37169 ...
 $ CRESS_ID : int  5 3 86 27 66 46 15 37 93 85 ...
 $ BIR74    : num  1091  487 3188  508 1421 ...
 $ SID74    : num  1 0 5 1 9 7 0 0 4 1 ...
 $ NWBIR74  : num    10   10  208  123 1066 ...
 $ BIR79    : num  1364  542 3616  830 1606 ...
 $ SID79    : num  0 3 6 2 3 5 2 2 2 5 ...
 $ NWBIR79  : num    19   12  260  145 1197 ...
 - attr(*, "data_types")= chr  "N" "N" "N" "N" ...



# CSV o dleimitados
# Las intrucciones son iguales que para 'read.table', solo se debe sustituir
el comando
```

```
# por:   CSV      - read.csv
#        Delim    - read.delim
```

11. Administración de variables y objetos

```
# Enlista todos los objetos activos de la sesión

ls()

# Agrega el objeto x1 en la ruta de trabajo.

attach(x1)

# Remueve al objeto x1 de la ruta de trabajo.

detach(x1)

# Remueve el objeto object1 de la sesión de trabajo.

rm(object1)

# Calcula las dimensiones de la la matriz o arreglo matrix1.

dim(matrix1)
```

```
# Despliega los nombres de las dimensiones de x1.

dimnames(x1)

# Calcula la longitud del vector (columna o rengl´on) vector1.

length(vector1)

# Combina columnas de los objetos (vectores o matrices) a1, a2 y
# cuando los tres son de la misma longitud (en el n´umero de casos).

xx <- data.frame(I=rep(0,2))
cbind(xx, X=rbind(a=1, b=1:3))    # named differently

  I X.1 X.2 X.3
1 0   1   1   1
2 0   1   2   3

# Idem. pero con los renglones de los objetos.

rbind(a1,b1,c1)

# Consolida dos tablas de datos, df1 y df2, considerando dos variables para
# ligarlas (una por cada tabla); by.x=``Var1" denota la columna Var1 del
# objeto df1, y by.y=``Otra1" denota la variable Otra1 del objeto df2.
merge(df1,df2, by.x = ``Var1", by.y = ``Otra1")
```

```
authors <- data.frame(
          surname = c("Tukey", "Venables", "Tierney", "Ripley", "McNeil"),
          nationality = c("US", "Australia", "US", "UK", "Australia"),
          deceased = c("yes", rep("no", 4)))

books <- data.frame(
          name = c("Tukey", "Venables", "Tierney",
                   "Ripley", "Ripley", "McNeil", "R Core"),
          title = c("Exploratory Data Analysis",
                    "Modern Applied Statistics ...",
                    "LISP-STAT",
                    "Spatial Statistics", "Stochastic Simulation",
                    "Interactive Data Analysis",
                    "An Introduction to R"),
          other.author = c(NA, "Ripley", NA, NA, NA, NA,
                           "Venables & Smith"))
```

```
> authors
   surname nationality deceased
1    Tukey          US      yes
2 Venables   Australia       no
3  Tierney          US       no
4   Ripley          UK       no
5   McNeil   Australia       no
```

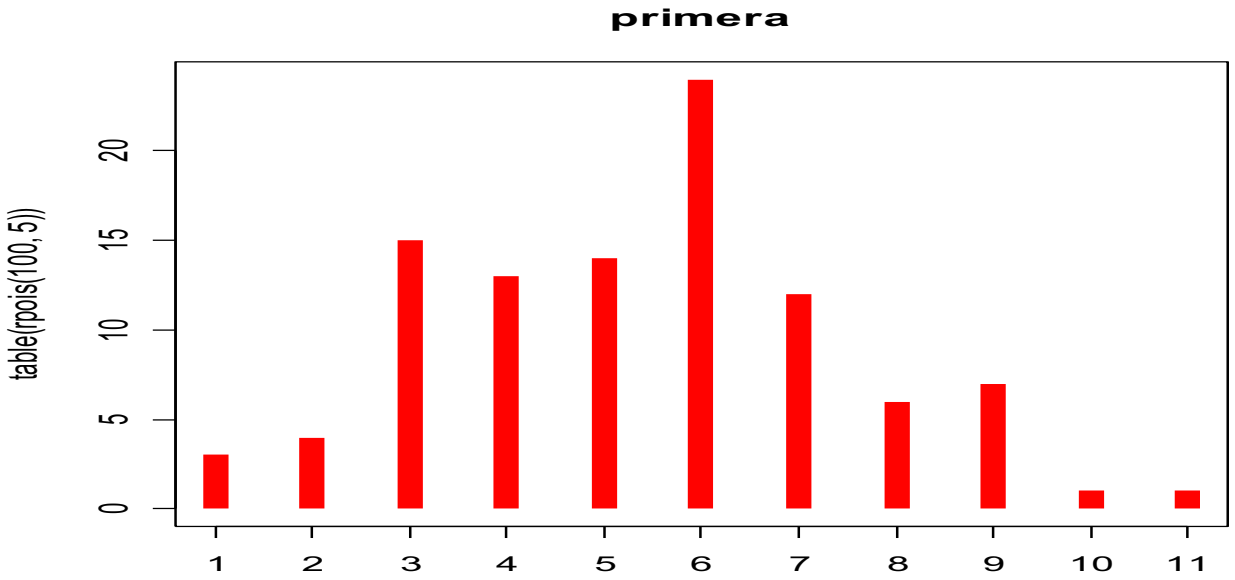
```
> books
      name                           title     other.author
1    Tukey     Exploratory Data Analysis              <NA>
2 Venables Modern Applied Statistics ...           Ripley
3  Tierney                      LISP-STAT              <NA>
4   Ripley             Spatial Statistics              <NA>
5   Ripley          Stochastic Simulation              <NA>
6   McNeil     Interactive Data Analysis              <NA>
7   R Core           An Introduction to R Venables & Smith

m1 <- merge(authors, books, by.x = "surname", by.y = "name")
m1
   surname nationality deceased                          title other.author
1   McNeil   Australia       no     Interactive Data Analysis         <NA>
2   Ripley          UK       no              Spatial Statistics         <NA>
3   Ripley          UK       no           Stochastic Simulation         <NA>
4  Tierney          US       no                       LISP-STAT         <NA>
5    Tukey          US      yes     Exploratory Data Analysis         <NA>
6 Venables   Australia       no Modern Applied Statistics ...       Ripley

m2 <- merge(books, authors, by.x = "name", by.y = "surname")
m2
      name                           title other.author nationality deceased
1   McNeil     Interactive Data Analysis         <NA>   Australia       no
2   Ripley             Spatial Statistics         <NA>          UK       no
3   Ripley          Stochastic Simulation         <NA>          UK       no
4  Tierney                      LISP-STAT         <NA>          US       no
```

```
5    Tukey     Exploratory Data Analysis          <NA>          US      yes
6 Venables Modern Applied Statistics ...       Ripley   Australia      no
```

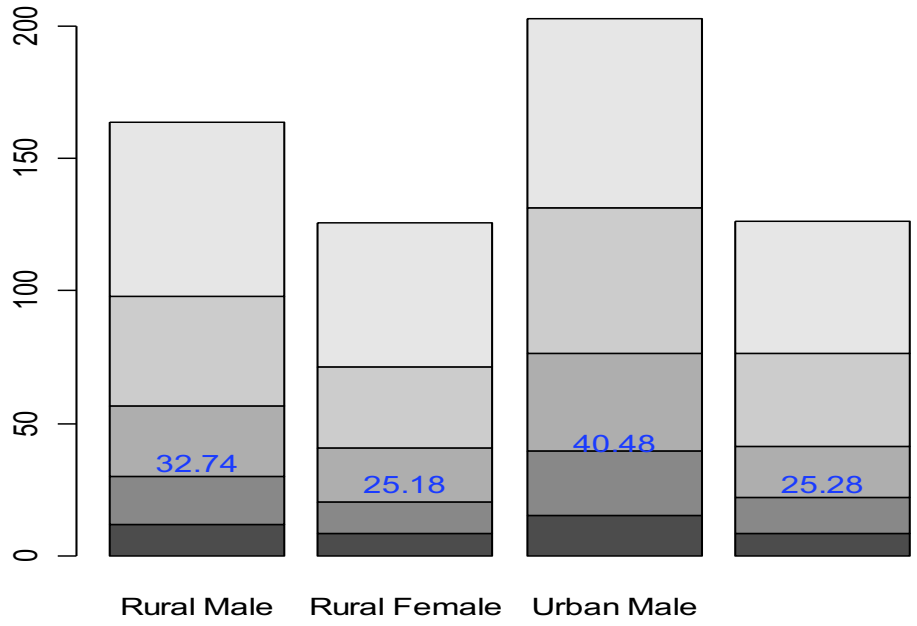
```
# Convierte el vector vector1 en una matriz con r1 renglones y c1 columnas,
# entrada por entrada vector1.

matrix(vector1,r1,c1)

# Esta es una función de localización con restricciones, regresa un arreglo
# con los índices de x1 que son iguales (==) a a1. El rastreo puede definirse
# con cualquiera de los operadores lógicos que se describieron anteriormente.

which(x1==a1)
```

12. Gráficas

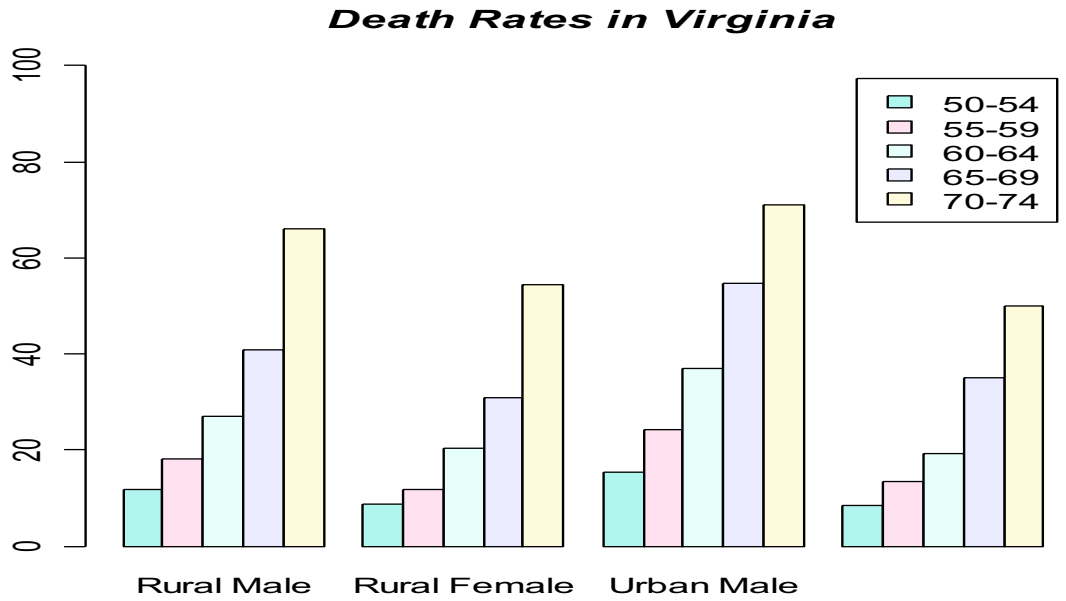
```
# Gráfica simple de dos dimensiones

plot(table(rpois(100,5)),
     type = "h",
     col = "red",
     lwd=10,
     main="primera")
```




```
# Gráfica de barras

mp <- barplot(VADeaths)
tot <- colMeans(VADeaths)
text(mp, tot + 3,
     format(tot),
     xpd = TRUE,
     col = "blue")
```



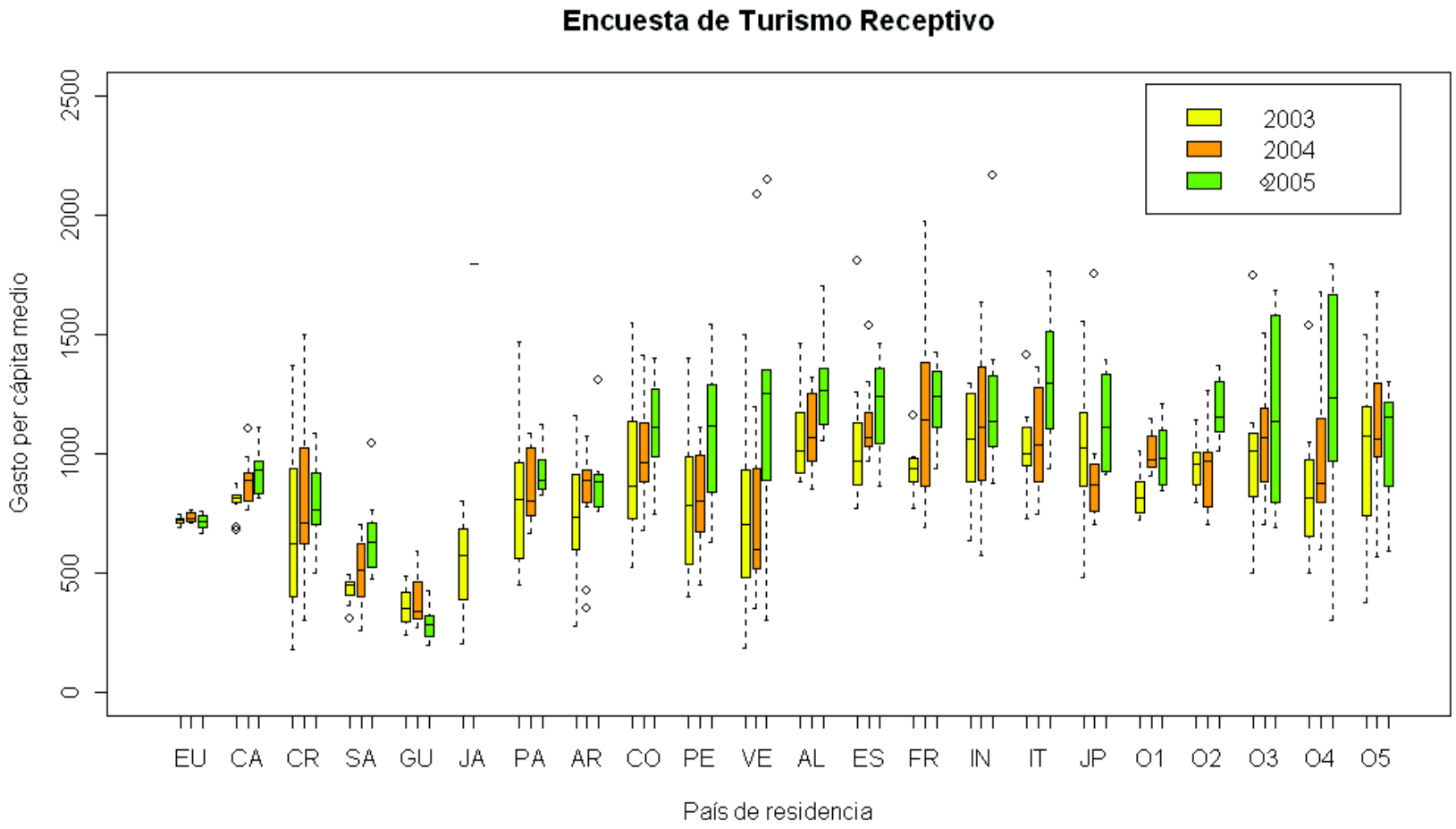

```
barplot(VADeaths,
        beside = TRUE,
        col = c("lightblue", "mistyrose", "lightcyan",
                "lavender", "cornsilk"),
        legend = rownames(VADeaths), ylim = c(0, 100))
title(main = "Death Rates in Virginia", font.main = 4)
```




```
# Gráfica de caja

#    EU
png(file="D://JCMO.BANXICO//Encuestas//Turismo//SIOM//Distrib. muestra
          turismo receptivo 2003-2005//Cuestionarios//Dist-GPC-EU.png",
bg="white")

boxplot(SerieTR[which(SerieTR[,"ANIO"]==2003),"GPC_M"]
        ~ SerieTR[which(SerieTR[,"ANIO"]==2003),"PA_RES"],
        data = SerieTR[which(SerieTR[,"ANIO"]==2003),],
```

```
          boxwex = 0.15,
          at = 1:22 - 0.1,
          col="yellow",
          main="Encuesta de Turismo Receptivo",
          xlab="País de residencia",
          ylab="Gasto per cápita medio",
          ylim = c(0, 2500),
          names=c("","","","","","","","","","","",
                  "","","","","","","","","","","",
                  "",""))
boxplot(SerieTR[which(SerieTR[,"ANIO"]==2004),"GPC_M"]
          ~ SerieTR[which(SerieTR[,"ANIO"]==2004),"PA_RES"],
          data = SerieTR[which(SerieTR[,"ANIO"]==2004),],
          add = TRUE,
          boxwex = 0.15,
          at = 1:22 + 0.1,
          col="orange",
          ylim = c(0, 2500),
          names=c("EU","CA","CR","SA","GU","JA","PA","AR","CO","PE",
                  "VE","AL","ES","FR","IN","IT","JP","O1","O2","O3",
                  "O4","O5"))
boxplot(SerieTR[which(SerieTR[,"ANIO"]==2005),"GPC_M"]
          ~ SerieTR[which(SerieTR[,"ANIO"]==2005),"PA_RES"],
          data = SerieTR[which(SerieTR[,"ANIO"]==2005),],
                  boxwex = 0.15,
          add = TRUE,
          at = c(1:5,7:22) + 0.3,
```

```
          col="green",
          ylim = c(0, 2500),
          names=c("","","","","","","","","","","",
                  "","","","","","","","","","","",
                  ""))
legend(18, 2550, c("2003", "2004", "2005"),
             fill = c("yellow", "orange", "green"))

dev.off()
```


Encuesta de Turismo Receptivo
Gasto per cápita medio
0
500
1000
1500
2000
2500
2003
2004
2005
EU
CA
CR
SA
GU
JA
PA
AR
CO
PE
VE
AL
ES
FR
IN
IT
JP
O1
O2
O3
O4
O5
País de residencia

14. Hmisc package

```
# entre otras cosas, tiene una suite para imputar datos perdidos

age <- c(1,2,NA,4)
age
[1]  1  2 NA  4

age.i <- impute(age)

impute(age,"random")
 1  2  3  4
 1  2 4*  4

age.i
 1  2  3  4
 1  2 2*  4

summary(age.i)
 1 values imputed to 2

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1.00    1.75    2.00    2.25    2.50    4.00
is.imputed(age.i)
[1] FALSE FALSE  TRUE FALSE
```

13. Bootstrap package

```
# Jackknife

xdata <- matrix(rnorm(30),ncol=2)
xdata
              [,1]       [,2]
 [1,]  1.53833227  0.7081109
 [2,]  0.20991345 -0.9654372
 [3,]  0.72726094 -0.4315130
 [4,] -2.13615011 -1.1172184
 [5,]  0.34502692 -0.6559252
 [6,]  0.30654026  0.2444141
 [7,]  0.05590595 -0.3165479
 [8,] -0.56899440  1.2843492
 [9,] -0.77656846 -0.3510983
[10,] -0.12509251 -0.2568242
[11,] -0.46372506 -1.5888879
[12,] -1.45913285  0.3188510
[13,] -0.16175351  2.4704858
[14,] -1.28205951 -0.1718442
[15,] -0.75468302  0.5059210

n <- 15
theta <- function(x,xdata)
             { cor(xdata[x,1],xdata[x,2]) }
```

```
results1 <- jackknife(1:n,theta,xdata)
results1
$jack.se
[1] 0.2155114

$jack.bias
[1] -0.02565586

$jack.values
 [1]  0.05052276  0.20066010  0.19733241 -0.01688548  0.19206404  0.14114532
 [7]  0.16189744  0.19365237  0.14088938  0.15570850  0.14519632  0.19716510
[13]  0.16802599  0.14620808  0.17474200

$call
jackknife(x = 1:n, theta = theta, xdata)

# Bootstrap

results <- bootstrap(1:n,20,theta,xdata)
results

$thetastar
 [1]  0.178747452  0.235091679  0.047228603 -0.229630742 -0.127644217
 [6] -0.171870252 -0.203918576  0.327533747  0.245267231  0.006881872
[11]  0.180600024  0.048535666  0.280927731 -0.766454883  0.135711810
[16]  0.318686721 -0.119563588  0.130890771  0.042781654  0.335438540
```

```
$func.thetastar
NULL

$jack.boot.val
NULL

$jack.boot.se
NULL

$call
bootstrap(x = 1:n, nboot = 20, theta = theta, xdata)
```

14. Sitios Web

Sitio Web oficial: http://www.r-project.org/

Manuales: http://cran.r-project.org/manuals.html

Funciones básicas:
http://www.maths.lth.se/help/R/.R/library/base/html/00Index.html

Grupos de discusión: http://www.r-project.org/mail.html

Referencias

Müller, Peter (1997) Introduction to Object-Oriented Programming Using C++.

The R Development Core Team (2004) R: A Language and Environment for Statistical Computing.

Venables, W. N., Smith, D. M. and the R Development Core Team (2004) An Introduction to R. También existe una versión en español (liga).