TALLER DE INTRODUCCIÓN A R

José Miguel Contreras

Talleres de Software Libre de la Universidad de Granada

Índice

Índice

R es un lenguaje de programación, creado por Ross Ihaka y Robert Gentleman, cuya característica principal es que forma un entorno de análisis estadístico para la manipulación de datos, su cálculo y creación de gráficos.

La página principal del proyecto "*R – project*" es *http : //www.r – project.org*, en ella podremos conseguir gratuitamente el programa, manuales, paquetes y demás elementos que forman la gran familia que es *R*.

R es un proyecto vivo, ya que los usuarios pueden contribuir al proyecto implementando funciones, librerias,... Ningún otro programa estadístico en la actualidad reúne la cantidad de recursos y manejabilidad que posee R.

R es un lenguaje de programación, creado por Ross Ihaka y Robert Gentleman, cuya característica principal es que forma un entorno de análisis estadístico para la manipulación de datos, su cálculo y creación de gráficos.

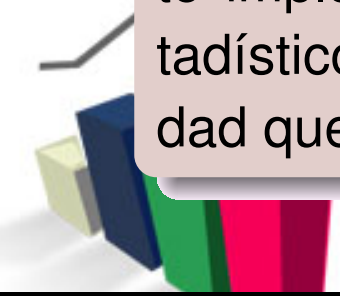
La página principal del proyecto "$R-project$" es $http://www.r-project.org$, en ella podremos conseguir gratuitamente el programa, manuales, paquetes y demás elementos que forman la gran familia que es *R*.

R es un proyecto vivo, ya que los usuarios pueden contribuir al proyecto implementando funciones, librerias,... Ningún otro programa estadístico en la actualidad reúne la cantidad de recursos y manejabilidad que posee R.

R es un lenguaje de programación, creado por Ross Ihaka y Robert Gentleman, cuya característica principal es que forma un entorno de análisis estadístico para la manipulación de datos, su cálculo y creación de gráficos.

La página principal del proyecto "$R-project$" es $http://www.r-project.org$, en ella podremos conseguir gratuitamente el programa, manuales, paquetes y demás elementos que forman la gran familia que es *R*.

R es un proyecto vivo, ya que los usuarios pueden contribuir al proyecto implementando funciones, librerias,... Ningún otro programa estadístico en la actualidad reúne la cantidad de recursos y manejabilidad que posee R.



Ayuda

Función help

```
> help( )
> help(sum)
> help(fix)
```

Ayuda

Función ?

```
> ?
> ?sum
> ?fix
```

Ayuda

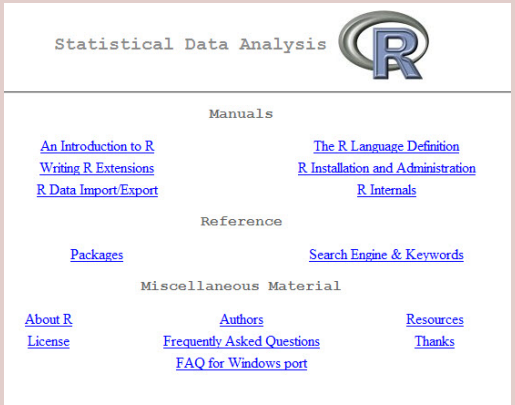
Función help.search

Cuando queremos consultar funciones especificadas por caracteres especiales, el argumento deberá ir entre comillas para transformarlo en una cadena de caracteres. Otra forma de realizar la misma acción es con la función "*help.search*("*median*")"

```
> help("[[")
> help([[)
> help("for")
> help(for)
```

AYUDA A TRAVÉS DE INTERNET

```
> help.start(  )
```



buscador: "*www.rseek.org*"

OTROS:

- *example*( ): Muestra los resultados propuestos en el ejemplo.
- *demo*( ): Muestra la relación de demos disponibles.
- *data*( ): Muestra la relación de datos de ejemplo disponibles.
- *apropos*( ): Muestra la relación de objetos disponibles con una cadena dada.

```
> ?example
> example(InsectSprays)
```

Asignaciones

Una opción muy común de *R* es la de asignar nombres a las operaciones. Esto lo conseguiremos mediante el símbolo "$<-$", "$->$" o "$=$" (solo de izquierda a derecha).
Para poder visualizar un dato renombrado hay de escribir el nombre después de haberlo asignado.

```
> rnorm(6)->x
> x
[1]  0.1364489 -0.5514473  0.2333607  1.1652428  0.4284878  0.5159063
> X<-rnorm(5)
> X
[1] -0.482520557 -0.951955735  1.537756423  0.839669367  0.772915316
> y<-1:15; y
```

Asignaciones

R utiliza determinados términos para referirse a algunas funciones por lo que lo mejor es evitar esos nombres a la hora de las asignaciones.

```
> a=c(1,2,3,4,3)
> b=t(a)
> t(b)
```

Comentarios

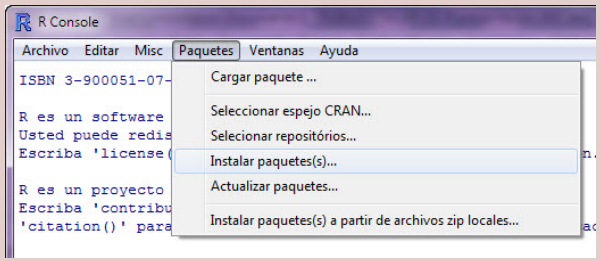
En el caso que necesitemos poner notaciones o comentarios en *R*, estos se realizan poniendo delante del comentario el símbolo "#".

```
> #Hola a todos!!!
> #Espero que todo esto os sea útil
```

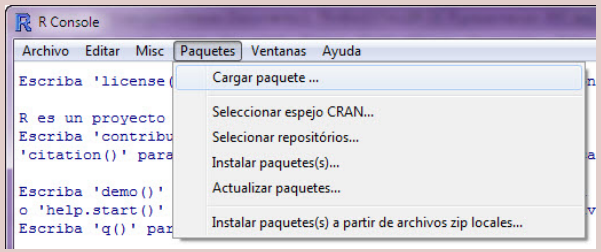
PAQUETES

Descarga de paquetes:

- http://www.cran.r-project.org/web/packages/
- Mediante la ventana desplegable
- install.packages("sampling")



Instalación del paquete:



Ejemplo con paquete sampling

Visitamos la web donde está alojado el paquete:

http://www.cran.r-project.org/web/packages/sampling/index.html

Descargamos el archivo sampling.pdf
Veamos las opciones belgianmunicipalities (datos) y inclusionprobabilities:

```
> library(sampling)
Loading required package: MASS
Loading required package: lpSolve
> library(MASS)
> library(lpSolve)
> #datos que contiene el paquete
> data(belgianmunicipalities)
> hist(belgianmunicipalities$medianincome)
> # La siguiente función calcula las probabilidades de inclusión
> # para una muestra de tamaño 20
> # de la variable Tot04 (población total a 1 de Julio de 2004)
> pik<-inclusionprobabilities(belgianmunicipalities$Tot04,20)
```

R COMO CALCULADORA

+, -, *, / suma, resta, producto, cociente

%%, %/%, ^ módulo, cociente entero, potencia

==, !=, ! igual, distinto, no

>, >=, <, <= mayor que, mayor o igual que, menor que, menor o igual que

|, ||, &, && y, y, o, o

: generar una serie

Ejemplos

```
> 2+3
[1] 5
> 2+3*5-(6/2)*2
[1] 11
> 31%%7
[1] 3
> 31%/%7
[1] 4
> 2^3
[1] 8
> x<-1:5
> y<-c(2,4,3,6,5)
> x==y
[1] FALSE FALSE  TRUE FALSE  TRUE
> x!=y
[1]  TRUE  TRUE FALSE  TRUE FALSE
> x[x==y]
[1] 3 5
> x[x!=y]
[1] 1 2 4
```

Ejemplos

```
> x<y
[1]  TRUE  TRUE FALSE  TRUE FALSE
> x<=y
[1] TRUE TRUE TRUE TRUE TRUE
> x>=y
[1] FALSE FALSE  TRUE FALSE  TRUE
> x>y
[1] FALSE FALSE FALSE FALSE FALSE
> x[x>y]
integer(0)
> x[x<y]
[1] 1 2 4
> y[x>y]
numeric(0)
> x[x<=y]
[1] 1 2 3 4 5
```

FUNCIONES LS Y RM:

ls saca en pantalla los objetos almacenados en la memoria por el usuario

```
> nombre<-"pepe"; x1<-5; x2<-100; b<-0.5
> ls()
[1] "b"      "nombre" "x"      "x1"     "x2"     "y"
# solo los elementos que contengan una b en el nombre
> ls(pat="b")
[1] "b"      "nombre"
# solo los elementos cuyo el nombres comienzen con una b
> ls(pat="^b")
[1] "b"
# características de los datos guardados
> ls.str()
b :  num 0.5
nombre :  chr "pepe"
x :  int [1:5] 1 2 3 4 5
x1 :  num 5
x2 :  num 100
y :  num [1:5] 2 4 3 6 5
# creo una tabla con los datos
> B <- data.frame(x1,x2,b)
> ls.str(pat="B")
B : 'data.frame':       1 obs. of  3 variables:
 $ x1: num 5
 $ x2: num 100
 $ b : num 0.5
```

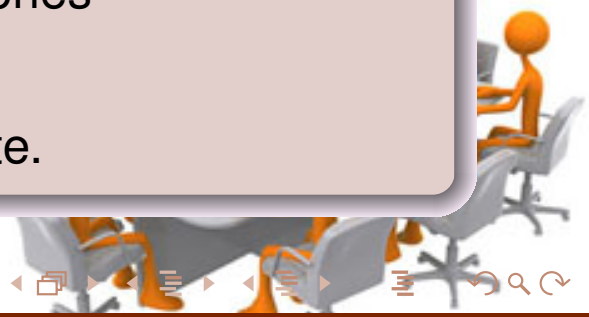
FUNCIONES LS Y RM:

rm borra objetos almacenados en la memoria, por ejemplo $rm(x)$ elimina el objeto x; $rm(x, y)$ elimina ambos objetos x e y, y "$rm(list = ls(\ ))$" elimina todos los objetos que estén en la memoria.

```
> rm(nombre)
> ls()
[1] "b"  "B"  "x"  "x1" "x2" "y"
> #Hemos borrano "nombre"
> rm(list=ls(pat="x"))
> ls()
[1] "b" "B" "y"
> #Hemos borrano las asignaciones que empiezan por x
> rm(list=ls())
> ls()
character(0)
> #Hemos borrano todo
```

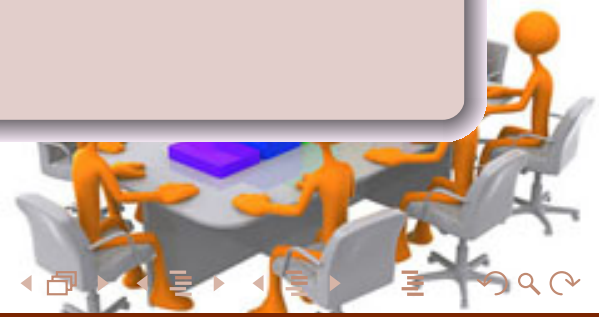
FUNCIONES BÁSICAS

- *seq*( ) Generar una secuencia numérica.
- *rep*( ) Generar un conjunto de valores repetidos.
- *t*( ) Transponer una matriz.
- *sqrt*( ) Raíz cuadrada.
- *abs*( ) Valor absoluto.
- *sin*( ), *cos*( ), . . . Funciones trigonométricas.
- *log*( ), *exp*( ) Logaritmo y exponencial.
- *round*( ), *signif*( ) Redondeo de valores numéricos.
- *for*( ), *while*( ) Evalúa una o un conjunto de expresiones repetitivamente.
- *if*( ), *ifelse*( ) Evalúa una expresión condicionalmente.



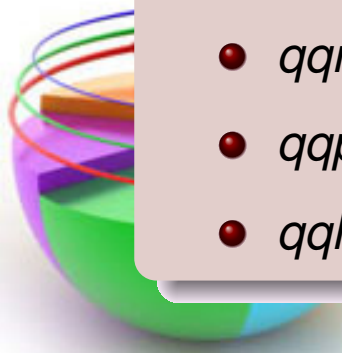
EJEMPLOS

```
> seq(1, 9, by = 2)
[1] 1 3 5 7 9
> rep(1:3,3)
[1] 1 2 3 1 2 3 1 2 3
> sqrt(81)
[1] 9
> abs(-9)
[1] 9
> sin(-2*pi)
[1] 2.449213e-16
> log(100)
[1] 4.60517
> exp(3)
[1] 20.08554
> round(2.345632)
[1] 2
> round(2.3456432, 3)
[1] 2.346
> signif(2.345432, 3)
[1] 2.35
```



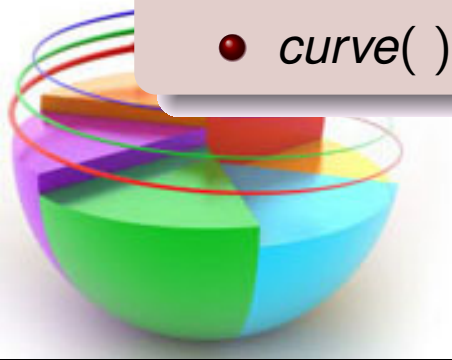
FUNCIONES GRÁFICAS BÁSICAS

- *plot*( ) Función genérica para representar en el plano xy puntos, lineas, etc.
- *barplot*( ) Diagramas de barras.
- *piechart*( ) Diagramas de sectores.
- *hist*( ) Histogramas.
- *boxplot*( ) Diagramas de box-and-whisker.
- *stripplot*( ) Similares a *boxplot*( ) con puntos.
- *sunflowerplot*( ) Representación en el plano xy de diagramas de girasol.
- *qqnor*( ) Diagramas de cuantil a cuantil frente a la distribución normal.
- *qqplot*( ) Diagramas de cuantil a cuantil de dos muestras.
- *qqline*( ) Representa la línea que pasa por el primer y el tercer cuartil.



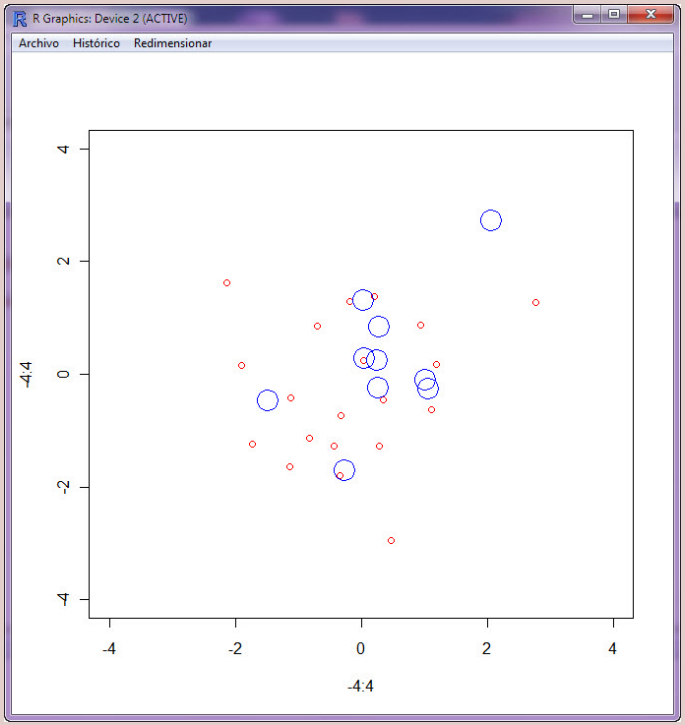
FUNCIONES GRÁFICAS

- *lines*( ) Añade líneas a un gráfico.
- *points*( ) Añade puntos a un gráfico.
- *segments*( ) Añade segmentos a un gráfico.
- *arrows*( ) Añade flechas a un gráfico.
- *polygons*( ) Añade polígonos a un gráfico.
- *rect*( ) Añade rectángulos a un gráfico.
- *abline*( ) Añade una recta de pendiente e intersección dada.
- *curve*( ) Representa una función dada.



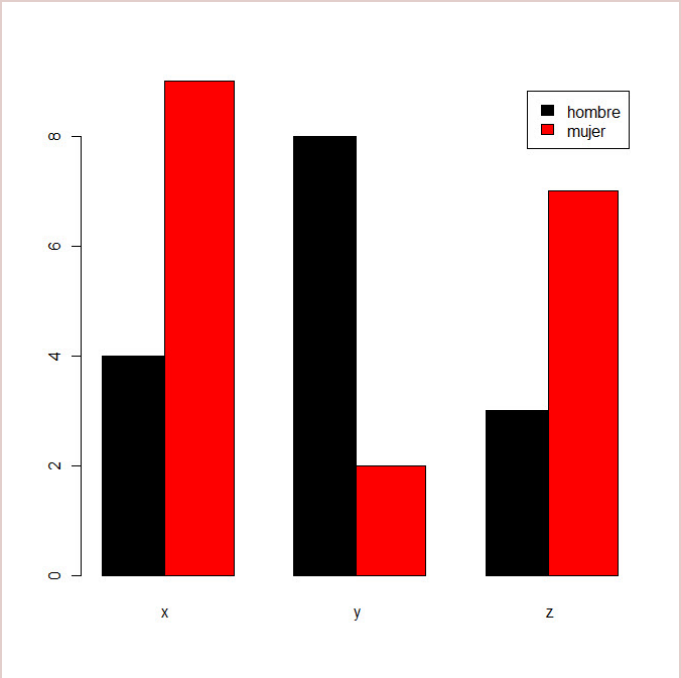
EJEMPLOS

```
>#  generamos una gráfica
> plot(-4:4, -4:4, type = "n")
>#  representamos 20 datos aleatorios de una distribución normal en rojo
> points(rnorm(20), rnorm(20), col = "red")
>#  representamos 10 datos aleatorios de una distribución normal en azul y más grandes
> points(rnorm(10), rnorm(10), col = "blue", cex = 3)
```



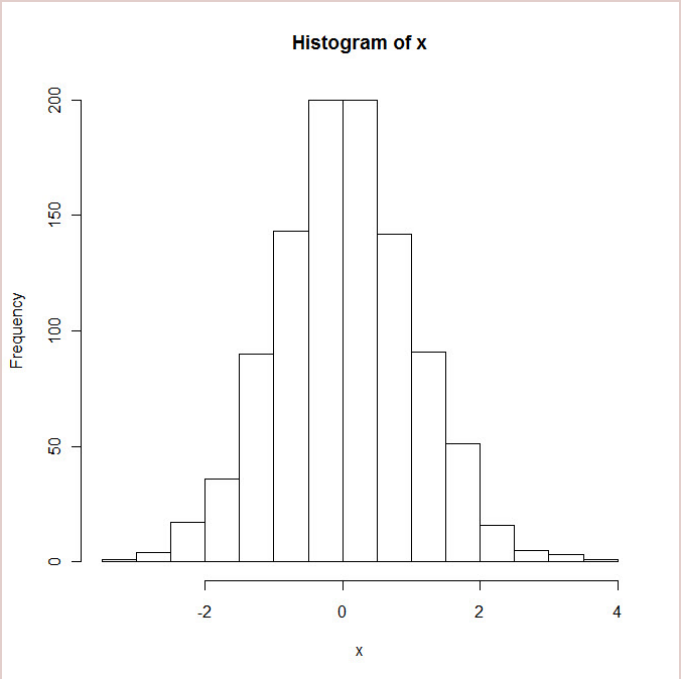
EJEMPLOS

```
>#  Representamos un diagrama de barras con las variables x,y y z,
>#  agrupadas una al lado de la otra (para juntarlas cambiamos TRUE por FALSE),
>#  de grosor 45,50 y35,
>#  con los colores 1 y 2 (negro y rojo) y
>#  con la leyenda hombre y mujer
> barplot(height = cbind(x = c(4, 9),  y = c(8, 2),  z = c(3, 7) ),  beside = TRUE,
+ width = c(45, 50, 35),  col = c(1, 2), legend.text = c("hombre", "mujer"))
```



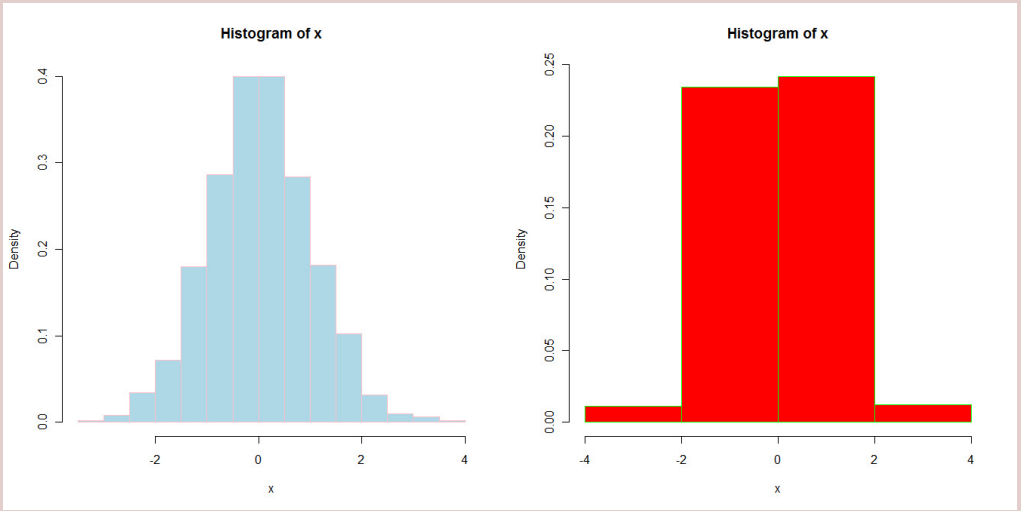
EJEMPLOS

```
>#  Representamos un histograma de la variable x, con x 1000 datos normales
> x<-rnorm(1000)
> hist(x)
```



EJEMPLOS

```
>#  creamos 13 intervalos (12 puntos de corte), de color azul y lineas rosas
> hist(x, freq = FALSE, breaks = 12, col="lightblue", border="pink")
>#  creamos 4 intervalos (3 puntos de corte), de color azul y lineas rosas
> hist(x, freq = FALSE, breaks = 3, col="red", border="green")
```

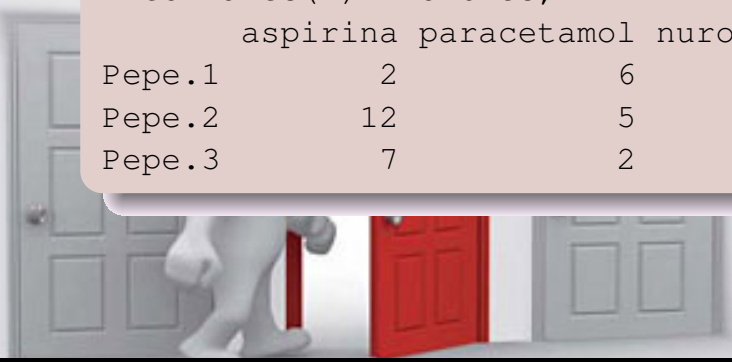


Matrices y Data frame

```
> X<-matrix(c(1,0,0,0,1,0,0,0,1),nrow=3); X
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
[3,]    0    0    1
> class(X)
[1] "matrix"
> attributes(X)
$dim
[1] 3 3
> v<-c(1,2,3,4,4,3,2,1)
> V<-matrix(v,nrow=2); V
     [,1] [,2] [,3] [,4]
[1,]    1    3    4    2
[2,]    2    4    3    1
> V<-matrix(v,byrow=T,nrow=2); V
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    4    3    2    1
> dim(v)<-c(4,2); v
     [,1] [,2]
[1,]    1    4
[2,]    2    3
[3,]    3    2
[4,]    4    1
```

Matrices y Data frame

```
> # Matriz de 15 datos de una dist. de Poisson de media 3
> Y<-matrix(rpois(15,5),nrow=3); Y
     [,1] [,2] [,3] [,4] [,5]
[1,]    2    6    4    9    6
[2,]   12    5    9    1    2
[3,]    7    2    6    5    4
> rownames(Y)<-rownames(Y,prefix="Pepe."); Y
     [,1] [,2] [,3] [,4] [,5]
[1,]    2    6    4    9    6
[2,]   12    5    9    1    2
[3,]    7    2    6    5    4
> rownames(Y)<-rownames(Y,do.NULL=FALSE,prefix="Pepe."); Y
       [,1] [,2] [,3] [,4] [,5]
Pepe.1    2    6    4    9    6
Pepe.2   12    5    9    1    2
Pepe.3    7    2    6    5    4
> nombres<-c("aspirina", "paracetamol", "nurofen", "ebuprofeno", "placebo")
> colnames(Y)<-nombres; Y
       aspirina paracetamol nurofen ebuprofeno placebo
Pepe.1        2           6       4          9       6
Pepe.2       12           5       9          1       2
Pepe.3        7           2       6          5       4
```



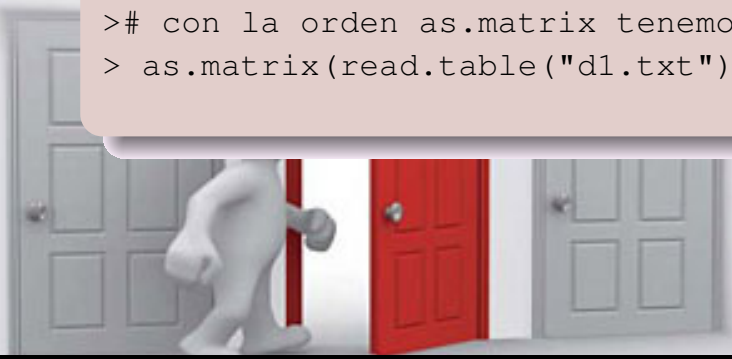
MATRICES Y DATA FRAME

Para trabajar en una carpeta concreta utilizamos *setwd*("*C* : /.../*datos*") o con el menú Archivo -> Cambiar dir...

```
> # leer datos de un fichero
> datos1<-read.table("datos1.txt")
> # leemos datos de un fichero (con nombre de variables, separadas por espacios,...)
> datos2<-read.table("datos2.txt", header=TRUE, sep="", na.strings="NA", dec=".")
> # calculamos la media de los datos de la columna 5
> mean(datos1[,5])
[1] 3.766791
> # calculamos la varianza de la columna 9
> var(datos1[,9])
[1] 388391.8
> # sumamos todas las varibles
> colSums(datos1)
     V1      V2      V3      V4      V5      V6      V7      V8      V9     V10     V11
 800525    3156    5153    3415    6057    2657    2277    9553 1549610 2983950 1549610
 > colMeans(datos1)
         V1          V2          V3          V4          V5          V6          V7
                 V8          V9         V10         V11
 497.838930    1.962687    3.204602    2.123756    3.766791    1.652363    1.416045
  5.940920  963.687811 1855.690299  963.687811
 >#  por filas sería rowSums( ) y rowMeans( )
```

EJEMPLOS

```
> m<-1000
> n<-100
> s<- sample(m,n); s
> # ordeno los datos
> s<-sort(s); s
> # Elijo solo los elementos de datos1 que coinciden con la muestra y creo otro fichero
> for (i in s) { subset(datos1,datos1[,1]==i)->d1
+ write.table(d1,file="d1.txt",append=TRUE,row.names=FALSE,col.names=FALSE,
quote = FALSE)
+ }
> # la opción append=TRUE hace que se anexe al documento la información nueva,
> # si es FALSE borra lo anterior
> d1
      V1 V2 V3 V4 V5 V6 V7 V8  V9 V10 V11
1602 997  2  1  1  6  1  1  6 463 790 395
1603 997  2  6  1  6  1  1  6 327 790 395
> # solo muestra los últimos datos
>#  con la orden as.matrix tenemos todos los datos
> as.matrix(read.table("d1.txt"))->xd1; xd1
```



FUNCIONES

if, else

```
> x <- rnorm(1)
> if(x > 2) print(" Mayor que 2 ") else print("Menor que 2")
[1] "< percentil 97.72"
```

FUNCIONES

for

```
> for(i in 1:5){
+ print(i)}
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
```

EJEMPLOS

while

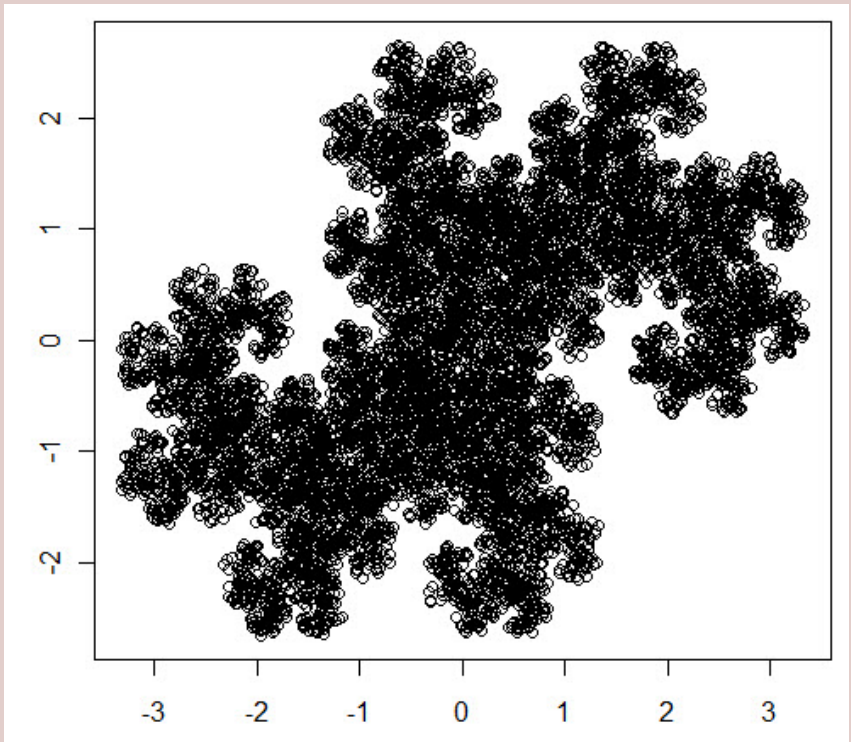
```
> valor <- 1;  n <- 0 # inicializamos los valores
> while(valor<5) {
+   n <- n + 1
+   x <- runif(1,-1,1) # un nº aleatorio entre -1 y 1
+   y <- runif(1,-1,1)
+       if(x^2 + y^2 < 1) valor <- valor + 1
+   }
> n
[1] 6
```

Arte con R

```
> f1=function(numero = 100)
{
x = vector(mode = "numeric", length = numero)
y = vector(mode = "numeric", length = numero)
x[1] = 1
y[1] = 1
for(i in 2:numero)
{
if(sample(2,1) == 2)
{m = 1}
else
{m = -1}
x[i] = 0.5 * x[i - 1] + 0.5 * y[i - 1] + m
y[i] = -0.5 * x[i - 1] + 0.5 * y[i - 1] + m
}
return(list(x = x[2:numero], y = y[2:numero]))
}
```

Arte con R

```
> xx<-f1(10000)
> plot(xx)
```



Estadística descriptiva

```
> dado<-c(1,2,3,4,5,6)
> frecuencia<-c(3,4,1,2,6,4)
> tabla<-data.frame(dado,frecuencia)
> # Frecuencias acumuladas
> cumsum(tabla[,2])
[1]  3  7  8 10 16 20
> # Media aritmética
> mean(tabla[,2])
[1] 3.333333
> # Mediana
> median(tabla[,2])
[1] 3.5
> # Cuantiles
> quantile(tabla[,2])
  0%  25%  50%  75% 100%
1.00 2.25 3.50 4.00 6.00
> quantile(tabla[,2], 0.65)
65%
  4
```

Regresión

```
> t<-c(1,1.2,1.3,1,1.2,1.3,1.4)
> g<-c(5,4.5,5.5,4.7,5.1,4.4,4.8)
> # gráfica
> plot(g,t)
> # creamos un modelo sin termino indep
> rl1<-lm(t~g-1)
> # creamos un modelo con termino indep
> rl2<-lm(t~g)
> # coeficiente de correlación de Pearson
> cor(t,g)
> # test del coeficiente de correlación de Pearson
> cor.test(t,g)
> # Resumen de la regresión
> summary(rl1)
> summary(rl2)
```

Test de Hipótesis

T de Student's

```
> t.test(x=1:10,y=c(7:20))

        Welch Two Sample t-test

data:  1:10 and c(7:20)
t = -5.4349, df = 21.982, p-value = 1.855e-05
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -11.052802  -4.947198
sample estimates:
mean of x mean of y
      5.5      13.5
> # opciones de la función
> ?t.test
```

EJEMPLO

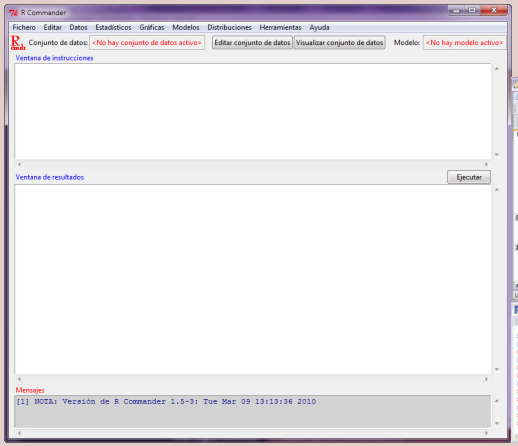
```
> alt<-c(1.77,1.80,1.65,1.69,1.86,1.75,1.58,1.79,1.76,1.81,1.91,1.78,1.80,1.69,1.81)
> t.test(alt)
        One Sample t-test

data:  alt
t = 82.3296, df = 14, p-value < 2.2e-16
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 1.717396 1.809270
sample estimates:
mean of x
 1.763333
> # Queremos estimar si la altura media es 1.78
> t.test(alt,mu=1.78)
        One Sample t-test
data:  alt
t = -0.7782, df = 14, p-value = 0.4494
alternative hypothesis: true mean is not equal to 1.78
95 percent confidence interval:
 1.717396 1.809270
sample estimates:
mean of x
 1.763333
```

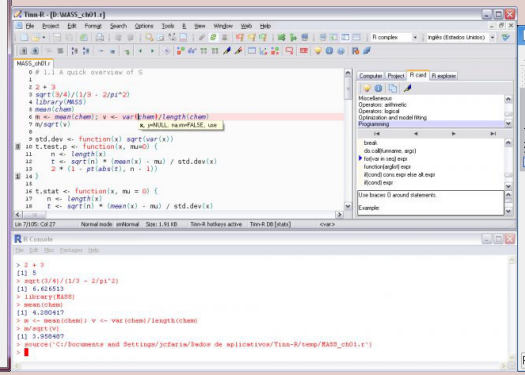
Test de Hipótesis

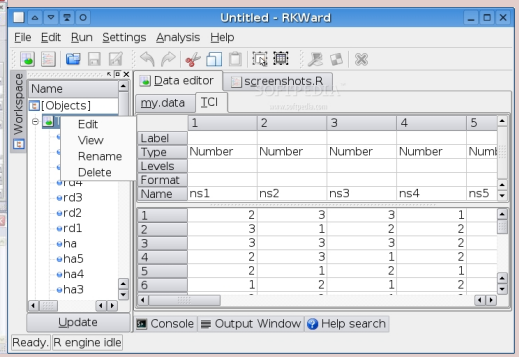
```
> # F para igualdad de varianzas
> var.test(x=1:10,y=c(7:20))

        F test to compare two variances

data:  1:10 and c(7:20)
F = 0.5238, num df = 9, denom df = 13, p-value = 0.3343
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
 0.1581535 2.0065024
sample estimates:
ratio of variances
         0.5238095
```

Entornos gráficos para trabajar con R

- Rcomander (todos los sistemas operativos)
- Tinn-R (windows)
- Rkaguar (linux y windows)
- ...

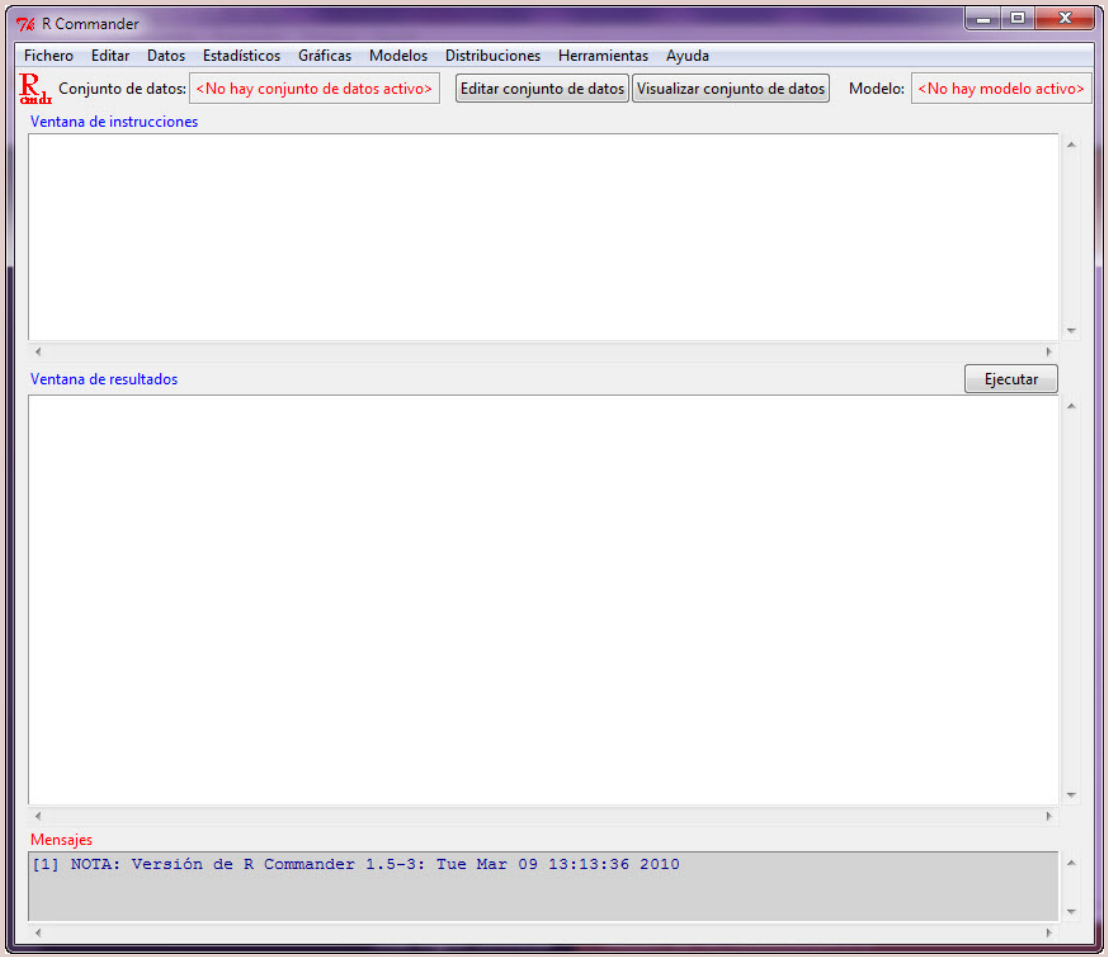




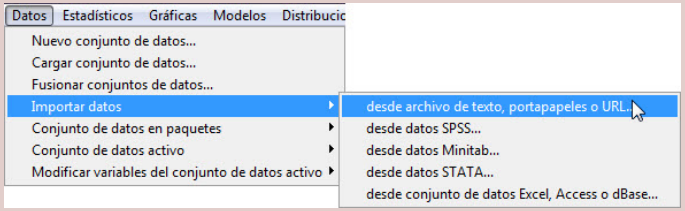


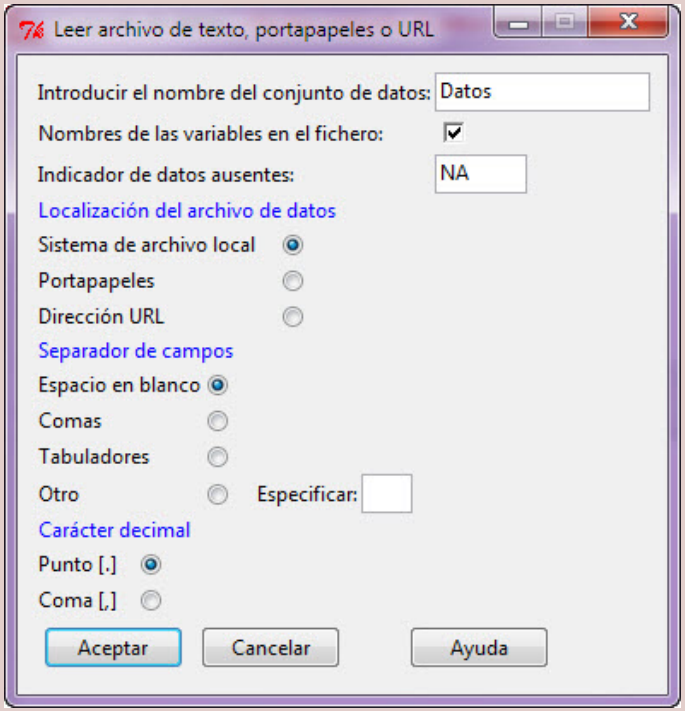
RCOMANDER

```
> library(Rcmdr)
```



RCOMANDER

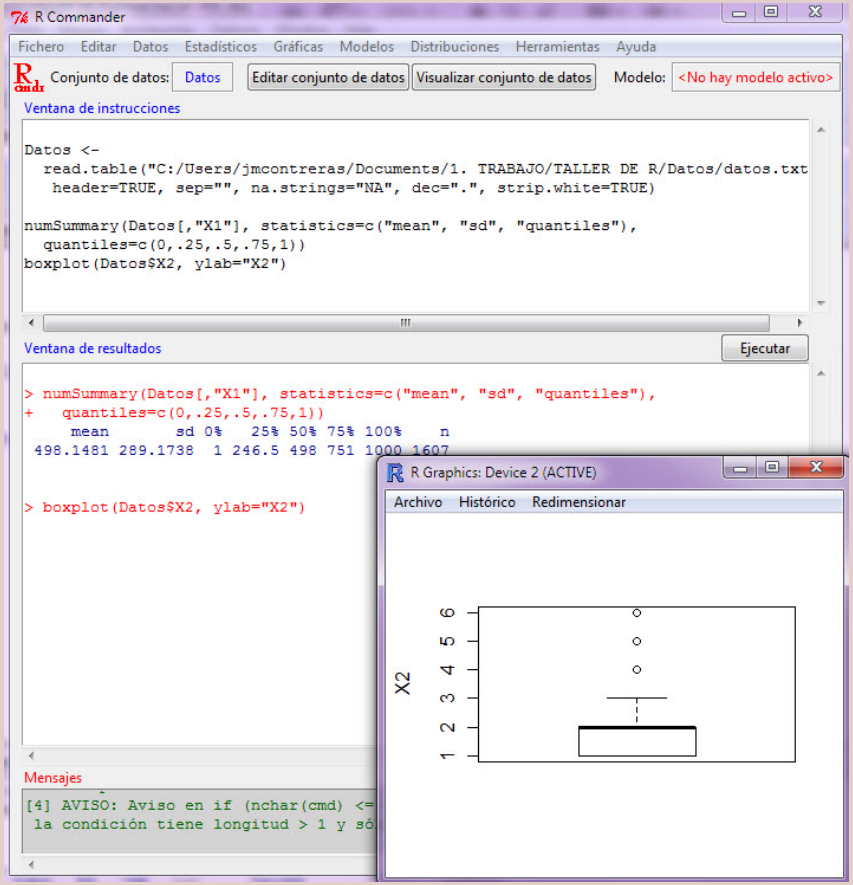

Datos
Estadísticos
Gráficas
Modelos
Distribucic
Nuevo conjunto de datos...
Cargar conjunto de datos...
Fusionar conjuntos de datos...
Importar datos
Conjunto de datos en paquetes
Conjunto de datos activo
Modificar variables del conjunto de datos activo
desde archivo de texto, portapapeles o URL...
desde datos SPSS...
desde datos Minitab...
desde datos STATA...
desde conjunto de datos Excel, Access o dBase...


Leer archivo de texto, portapapeles o URL
Introducir el nombre del conjunto de datos:
Datos
Nombres de las variables en el fichero:
Indicador de datos ausentes:
NA
Localización del archivo de datos
Sistema de archivo local
Portapapeles
Dirección URL
Separador de campos
Espacio en blanco
Comas
Tabuladores
Otro
Especificar:
Carácter decimal
Punto [.]
Coma [,]
Aceptar
Cancelar
Ayuda

RCOMANDER

Estadísticos -> Resúmenes -> Resúmenes numéricos para X1

Gráficas -> Diagrama de caja, para la variable X2



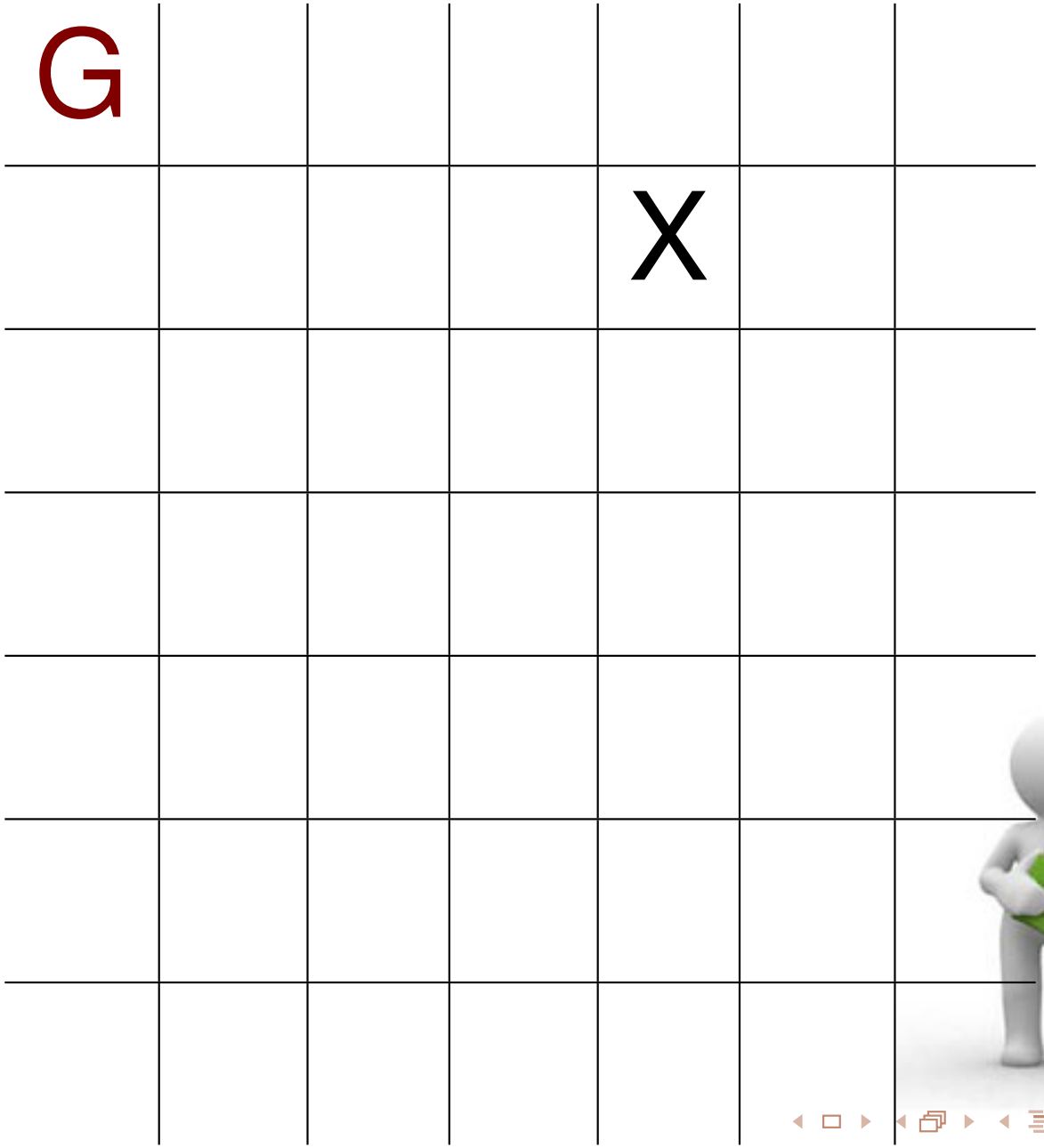
BIBLIOGRAFIA ÚTIL

En Español:

- Introducción a R, R Development Core Team
- Curso básico de R, Carmona F.
- R para Principiantes, Paradis E.
- Estadística con R y Rcomander, U. Cadiz
- Próximamente en mio...

En Inglés:

- A First Course in Statistical Programming with R, Braun J. y Duncan J.M.
- The R Book, Crawley M.J.
- A Handbook of Statistical Analyses Using R, Everitt B.S. y Hothorn T.
- La serie Use R!, Springer.


G
X

G					O	
	R			X		

G					O	
	R			X		
		A				

G					O	
	R			X		
		A				
			C			

G					O	
	R			X		
		A				
			C			
				I		
O						

G					O	
	R			X		
		A				
			C			
				I		
O					A	

G					O	
	R			X		
		A				
			C			
				I		
O					A	
	X					S