

Fundamentos de R

Análisis Exploratorio de Datos, 2009-2010

José María Matías

Dpto. de Estadística e IO

Universidad de Vigo

Índice

1. Introducción
2. Manejo de datos
3. Estadística descriptiva
4. Gráficos

1. Introducción

- R es un lenguaje similar al lenguaje S desarrollado por AT&T Bell Laboratories.
- Otro elemento de la familia es el lenguaje incorporado en el paquete estadístico S-plus.
- Versiones disponibles para: Windows, Linux, Unix y Macintosh OS X

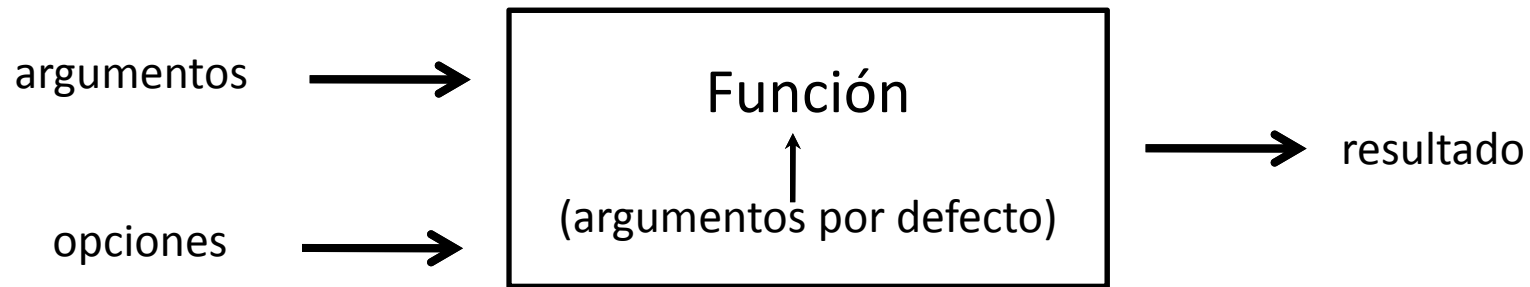
Funcionalidades

- Manejo y almacenamiento de datos.
- Cálculo orientado a matrices y arrays.
- Gran número de funciones para el análisis de datos.
- Facilidades gráficas para el análisis de datos.
- Lenguaje interpretado (no compilado).
- Lenguaje de programación orientado a objetos: variables, datos y funciones se almacenan en forma de objeto.
- Constituye una plataforma estándar en el desarrollo e investigación de nuevas técnicas estadísticas que se testean y ponen a disposición de la comunidad científica.
- Disponibilidad de programas fuentes que admiten ulteriores modificaciones por el usuario.

Directorio de Instalación

- C:\Program Files\R\R-x.x.x
- Carpetas importantes:
 - bin: ejecutables.
 - doc: ayuda.
 - etc: configuración. P. ej. Rconsole, Rprofile.site
 - library: paquetes base y “contributed”
- Extensiones de ficheros:
 - .R: programas.
 - .Rdata: áreas de trabajo.

Una función de R



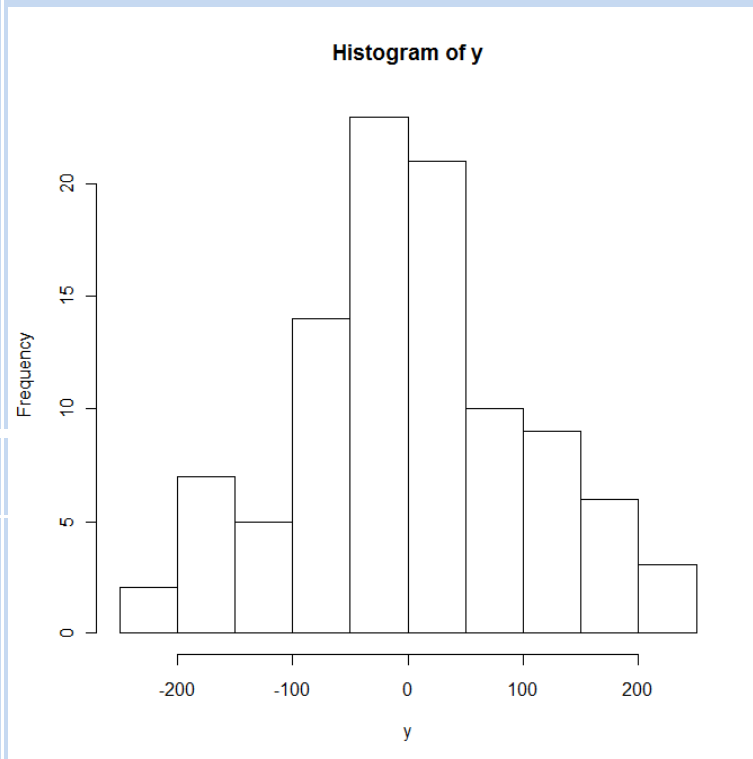
- Argumentos: datos, formulas, expresiones.
- Paquetes de funciones:
 - C:/Program Files/R/R-x.x.x/library
 - C:/Program Files/R/R-2.3.1/library/base

Ejemplo

```
> x = c(1:100)
> y = rnorm(100)*100
> hist(y)
> test.model = lm(y ~ x)
> test.model
> plot(x,y)
> library(help="graphics")
```

On-line help:

```
> ? rnorm
> ? plot
#Information on package 'base'
> library(help="base")
```



2. Manejo de datos

1. Introducción
2. Generación de datos
3. Creación de objetos
4. Importación/exportación de datos.

2.1 Introducción al manejo de datos

- Los objetos poseen dos atributos básicos:
 - Modo (mode) con sus cuatro tipos básicos:
 1. Numérico.
 2. Carácter.
 3. Lógico (TRUE o FALSE).
 4. Complejo.
 - Longitud (length):

```
> x <- c(1:2)
> mode (x)
[1] "numeric"
> length(x)
[1] 2
```

Nombres

- Operador de asignación: “<-” ó “=”.
- El nombre de un objeto debe comenzar con una letra (A-Z o a-z) y puede incluir letras, números y puntos.
- R distingue entre mayúsculas y minúsculas.

```
> A <- "WEPA"; compar <- TRUE; z <- 3+4i  
> mode(A); mode(compar); mode(z)  
[1] "character"  
[1] "logical"  
[1] "complex"
```

Valores especiales

- R representa valores numéricos infinitos:
 - $+\infty = \text{Inf}$
 - $-\infty = -\text{Inf}$
- NaN (Not a Number).

```
> x <- 5/0
```

```
> x
```

```
[1] Inf
```

```
> exp(x)
```

```
[1] Inf
```

```
> exp(-x)
```

```
[1] 0
```

```
> x - x
```

```
[1] NaN
```

Tipos de objeto

Objeto	Modo	Varios modos posibles?
vector	Numérico, carácter, complejo o lógico	No
factor	Numérico o carácter	No
array	Numérico, carácter, complejo o lógico	No
matrix	Numérico, carácter, complejo o lógico	No
data.frame	Numérico, carácter, complejo o lógico	Si
ts	Numérico, carácter, complejo o lógico	Si
List	Numérico, carácter, complejo o lógico, función, expresión,	Si

2.2 Generación de valores

- Valores deterministas:
 - `c()`
 - `seq()`
 - `scan()`
 - `rep()`
 - `sequence()`
 - `gl()`
 - `expand.grid()`
 - Constants
 - Missing values
- Valores aleatorios (más adelante)

Unión de vectores

- `c(...)` : Reúne los valores en un vector.

```
> x = c(2,3,5,2,7,1)
```

```
> x
```

```
[1] 2 3 5 2 7 1
```

```
> y = c(10,15,12)
```

```
> y
```

```
[1] 10 15 12
```

```
> z = c(x, y)
```

```
> z
```

```
[1] 2 3 5 2 7 1 10 15 12
```

Subconjuntos de vectores

```
# Selección de elementos de vectores:
> x <- c(3,11,8,15,12) # Assign to x the values 3, 11, 8, 15, 12
x[c(2,4)] # Extract elements (rows) 2 and 4
[1] 11 15
# Con números negativos se omiten elementos:
> x[-c(2,3)]
[1] 3 15 12
> x>10 # Genera un vector lógico (T o F)
[1] F T F T T
> x[x>10]
[1] 11 15 12
# vectores con elementos nominados:
> c(ALAN=100, SERENA=2000, ANDY=300, ALPHA=400)[c("ALAN", "ANDY")]
ALAN ANDY
100 300
```

Series (1)

- Secuencia de enteros: `seq(from, to, steps)`
- Combinación de valores en un vector o lista: `c()`
- Entrada de datos por consola: `scan()`

```
> x1 = 1:100
> x2 = 100:1
> x3 = seq(1,10, 0.5)
> x4 = seq(length=9, from=1, to=5)
> x5 = c(1,2,2.5,6,10)
> x6 = scan( )
1: 1
2: 2
3: 3
4: 5
5:
Read 4 items
```


Series(2)

- creates a vector with all its elements identical: `rep()`
- creates a series of sequences of integers each ending by the numbers given as arguments: `sequence()`

```
> rep(1,30)
[1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1
> sequence(5)
[1] 1 2 3 4 5
> sequence(c(5,2))
[1] 1 2 3 4 5 1 2
> sequence(c(5,9))
[1] 1 2 3 4 5 1 2 3 4 5 6 7 8 9
```

Generación de niveles (factores)

- `gl()`: genera una serie de niveles de factor.
- `gl(n,k,length=n*k,labels=1:n,ordered = FALSE)`
 - `n`: entero con el número de niveles.
 - `k`: entero con el número de repeticiones.
 - `length`: entero con la longitud del resultado.
 - `labels`: vector opcional de etiquetas para los niveles.
 - `ordered`: si el resultado debe ordenarse o no.

```
> gl(3, 5)
[1] 1 1 1 1 1 2 2 2 2 2 3 3 3 3 3
Levels: 1 2 3
> gl(3, 5, length=30)
[1] 1 1 1 1 1 2 2 2 2 2 3 3 3 3 3 1 1 1 1 1 2 2 2 2 2
    3 3 3 3 3
Levels: 1 2 3
> x = gl(2, 6, label=c("Male", "Female"))
[1] Male Male Male Male Male Male Female Female Female
     Female Female Female
Levels: Male Female
> class(x)
[1] "factor"
```

Generación de data.frames

- `expand.grid(arguments)`: los argumentos pueden ser vectores, factores o listas.

```
> x <- expand.grid(h=c(160, 165, 170), w=c(50, 60),
  sex=c("M", "F"))
> print(x)
h w sex
1 160 50 M
2 165 50 M
3 170 50 M
4 160 60 M
5 165 60 M
6 170 60 M
...
12 170 60 F
> class(x)
[1] "data.frame"
```

Constantes

- LETTERS
- letters
- month.abb
- month.name
- pi

```
> x = LETTERS
> y = x[-c(2:10)]
> length(x)
> length(y)
> z = month.name
> z
```

Valores faltantes

- NA: constante lógica de longitud uno indicando valor faltante

```
> x = c(pi, 1, 2)
> x
[1] 3.141593 1.000000 2.000000
> x[2] = NA
> x
[1] 3.141593 NA 2.000000
> is.na(x[2])
[1] TRUE
> is.na(x[1])
[1] FALSE
# Para reemplazar los NA por ceros:
> x[is.na(x)] = 0
> x
[1] 3.141593 0.000000 2.000000
```

- NULL: objeto nulo en R. Es el resultado de expresiones o funciones cuyo valor no está definido

```
> x = NULL; is.null(x)
[1] TRUE
```

2.3 Creación de objetos

- Listas
- Vectores
- Factores
- Arrays
- Matrices
- Data.frames

Creación de listas

- Función para construir (o convertir a) una lista:
`list()`

```
> data() # list all available data sets
> cars
speed dist
1  4   2
2  4  10
3  7   4
...
49 24 120
50 25  85
> pts = list(x=cars[,1], y=cars[,2])
> plot(pts)
```

Creación de vectores

- Función para crear un vector de longitud y modo dados:

`vector(mode, length)`

– mode: especifica el tipo de objeto:

- numeric (por defecto 0)
- logical (modo por defecto, y por defecto FALSE)
- character (por defecto “ ”)

```
> x <- vector(mode="numeric", length=100)
> is.vector(x)
[1] TRUE
> x <- c("Taiwan", "China", "USA")
> is.vector(x)
[1] TRUE
```


Creación de factores (1)

- Un factor incluye tanto los valores de la variable categórica correspondiente, como los niveles posibles de dichos valores

```
factor (x, levels = sort(unique(x), na.last =  
  TRUE), labels = levels, exclude = NA, ordered =  
  is.ordered(x))
```

- x: vector de datos.
- levels: niveles del factor (por defecto los de x)
- labels: nombres de los niveles
- exclude: vector de niveles a excluir al formar los niveles (por defecto no aplicable)
- ordered: valor lógico indicando si los niveles deben ordenarse (por defecto si lo está x)

Creación de factores (2)

```
> factor(1:3)
[1] 1 2 3
Levels: 1 2 3
> factor(1:3, levels=1:5)
[1] 1 2 3
Levels: 1 2 3 4 5
> factor(1:3, labels=c("A", "B", "C"))
[1] A B C
Levels: A B C
> x = c(1,1,2,1,2,2,1,2)
> factor(x,levels = sort(unique(x), na.last = TRUE),labels =
  c("hombre","mujer"), exclude = NA, ordered = is.ordered(x))
[1] hombre hombre mujer hombre mujer mujer hombre mujer
Levels: hombre mujer
x <- factor(letters[1:6], label="YDU"); x
[1] YDU1 YDU2 YDU3 YDU4 YDU5 YDU6
Levels: YDU1 YDU2 YDU3 YDU4 YDU5 YDU6
class(x)
[1] "factor"
```

Creación de arrays

- Funciones para crear y testear arrays:
 - `array(data = NA, dim = length(data), dimnames = NULL)`
 - `as.array(x)`
 - `is.array(x)`

```
> x <- array(letters)
> class(x)
[1] "array"
> dim(x)
[1] 26
> x <- array(1:3, c(2,4)); x
[,1] [,2] [,3] [,4]
[1,] 1 3 2 1
[2,] 2 1 3 2
> dim(x)
[1] 2 4
> length(x)
[1] 8
> x[1, ] # select row 1
```

Creación de matrices

- Funciones para crear y testear matrices:

- `matrix(data = NA, nrow = 1, ncol = 1, byrow = FALSE, dimnames = NULL)`
- `as.matrix(x)`
- `is.matrix(x)`

```
> matrix.data = matrix(c(1,2,3,4,5,6),nrow=2,  
  ncol=3,byrow=TRUE, dimnames = list(c("row1",  
  "row2"), c("C1", "C2", "C3")))
```

```
> matrix.data
```

```
C1 C2 C3
```

```
row1 1 2 3
```

```
row2 4 5 6
```

Creación de data.frames (1)

- Un data.frame es un objeto utilizado típicamente para almacenar tablas de datos.
- Es una lista de variables de la misma longitud, pero con posibles diferentes tipos (numérico, factor, carácter, lógico, ...)

```
data.frame(..., row.names=NULL, check.rows= FALSE,  
           check.names = TRUE)
```

```
> x = 1:4; n = 10; M = c(10, 35); y = 2:4  
data.frame(x, n)  
  x n  
1 1 10  
2 2 10  
3 3 10  
4 4 10  
> data.frame(x, M)  
  x M  
1 1 10  
2 2 35  
3 3 10  
4 4 35
```

Creación de data.frames (2)

```
> z <- data.frame(var1= rnorm(5), var2=LETTERS[1:5])
      var1 var2
1 -0.50475675    A
2 -0.02497888    B
...
5 -0.60652816    E
> rownames(z)
[1] "1" "2" "3" "4" "5"
rownames(z) = c("uno", "dos", "tres", "cuatro", "cinco"); z
      var1 var2
uno   -0.50475675    A
dos   -0.02497888    B
...
cinco -0.60652816    E
> data(cars)
> help(cars)
> class(cars)
[1] "data.frame"
> cars
  speed dist
1    4    2
2    4   10
...
```

2.4 Importación/Exportación de datos

```
# Conocer el directorio actual de trabajo
> getwd()
# Crear um directorio C:\Midirectorio\data
# Establecer como directorio de trabajo (barras al revés)
> workpath <- "C:/Midirectorio/data"
> setwd(workpath)
# Crear um fichero de datos C:\Midirectorio\data\datos.txt
# Importar fichero de datos
> datos1 = read.table(file="datos.txt", header= TRUE)
> datos1
# Añadir nueva variable x3
x3 <- matrix(c(60,80,65,85,80,90,99), nrow=7, ncol=1, byrow=FALSE,
dimnames = list(c(),c("x3"))); x3
# Unir los dos objetos(datos1 y x3)
datos2 <- data.frame(datos1, x3); datos2
# Exportar el nuevo conjunto de datos
write.table(datos2 , file= "datos.txt", sep = "\t", append=FALSE,
row.names=FALSE, col.names=TRUE, quote= FALSE)
```

datos1		
Id	x1	x2
A1	60	90
A2	70	75
A3	80	85
A4	85	85
A5	75	60
A6	90	80
A7	65	98

3. Estadística Descriptiva

3.1 Operadores

3.2 Funciones matemáticas

3.3 Acceso a datos

3.4 Estadística descriptiva

3.1 Operadores

Aritméticos		De comparación		Lógicos	
+	Suma	<	Menor que	!x	No lógico
-	Resta	>	Mayor que	x & y	Y lógico
/	División	<=	Menor o igual que	x && y	Y lógico
^	Potencia	==	Igual que	x y	O lógico
%%	Módulo	!=	Distinto a	x y	O lógico
%/%	División entera				

Otros operadores:

- \$: campos de una lista o data.frame
- [
- [[
- :
- ?
- = ó <-

3.2. Funciones matemáticas (1)

<code>sum(x)</code>	suma de los elementos de x
<code>prod(x)</code>	producto de los elementos de x
<code>max(x)</code>	máximo de los elementos de x
<code>min(x)</code>	mínimo de los elementos de x
<code>which.max(x)</code>	índice del mayor elemento de x
<code>which.min(x)</code>	índice del menor elemento de x
<code>range(x)</code>	rango de valores de x
<code>length(x)</code>	número de elementos de x
<code>mean(x)</code>	media de los elementos de x
<code>median(x)</code>	mediana de los elementos de x
<code>var(x), cov(x)</code>	cuasivarianza de los elementos de x
<code>cor(x)</code>	matriz de correlación de x si matriz o data.frame (1 si vector)
<code>var(x,y), cov(x,y)</code>	covarianza entre x e y o entre las columnas de x e y si son matrices o dataframes
<code>cor(x,y)</code>	correlación entre x e y, o matriz de correlación si son matrices o dataframes.

Funciones matemáticas (2)

<code>round(x,n)</code>	redondea los elementos de x hasta n decimales
<code>ceiling(x)</code>	vector numérico con los menores enteros no menores que los valores de x
<code>floor(x)</code>	vector numérico con los mayores enteros no mayores que los valores de x
<code>rev(x)</code>	invierte el orden de los elementos de x
<code>sort(x)</code>	ordena los elementos de x en orden creciente. <code>rev(sort(x))</code> en decreciente
<code>rank(x)</code>	vector de rangos de los elementos de x
<code>log(x,base)</code>	logaritmo de x en base “base”
<code>choose(n,k)</code>	combinaciones de n elementos tomados de n en n
<code>sample(x,size)</code>	muestra aleatoria de tamaño “size” sin reemplazamiento. Con reemplazamiento requiere la opción <code>replace = TRUE</code>)

3.3 Acceso a elementos

número de elementos	<code>length(x)</code>
i-ésimo elemento (i=2)	<code>x[2]</code>
todos menos el i-ésimo elemento (i=2)	<code>x[-2]</code>
k primeros elementos (k=5)	<code>x[1:5]</code>
k últimos elementos (k=5)	<code>x[(length(x)-5):length(x)]</code>
elementos específicos (1º, 2º y 3º)	<code>x[c(1,2,3)]</code>
elementos mayores que un valor (=3)	<code>x[x>3]</code>
elementos mayores o menores que dos valores (>2 ó <-2)	<code>x[x<-2 x>2]</code>
índices de los mayores valores	<code>which(x == max(x))</code>

3.4 Estadística descriptiva

```
summary(datos1)
datos1[2]
datos1$x1
x <- datos1$x1
mean(x)
max(x)
min(x)
sd(x)
var(x)
# Desviación típica
sqrt(sum((x - mean(x))^2 / (length(x))))
[1] 10
# otra forma
sdp= function(x) sqrt(sum( (x - mean(x))^2 / (length(x))))
sdp(x)
[1] 10
```

Gráficos

4.1 Dispositivo gráfico (device)

4.2 Plot

4.3 Gráficos de barras

4.4 Gráficos circulares

4.5 Gráfico de caja y patillas (box and whisker)

4.6 Gráfico de tallo y hojas (stem and leaf)

4.1 Dispositivo gráfico

- El resultado de una función gráfica se envía a un dispositivo gráfico: ventana o fichero.
- Dos clases de funciones gráficas:
 - Alto nivel: crean un nuevo gráfico.
 - Bajo nivel: añaden elementos a un gráfico existente.
- Nueva ventana gráfica: `windows ()`
- Lista de los dispositivos disponibles: `dev.list ()`
- Muestra/cambia el dispositivo activo: `dev.cur ()` , `dev.set (n)`
- Cierra el dispositivo activo: `dev.off` , `dev.off (n)`

4.2 plot()

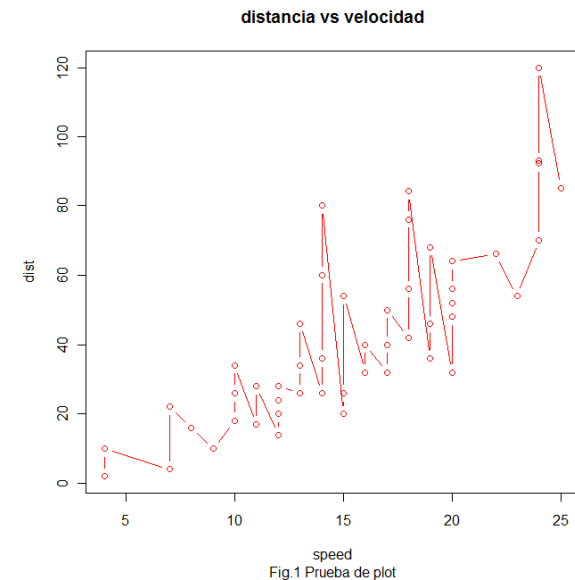
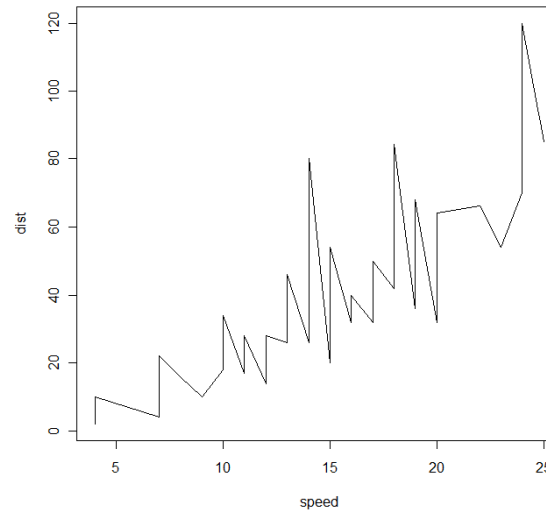
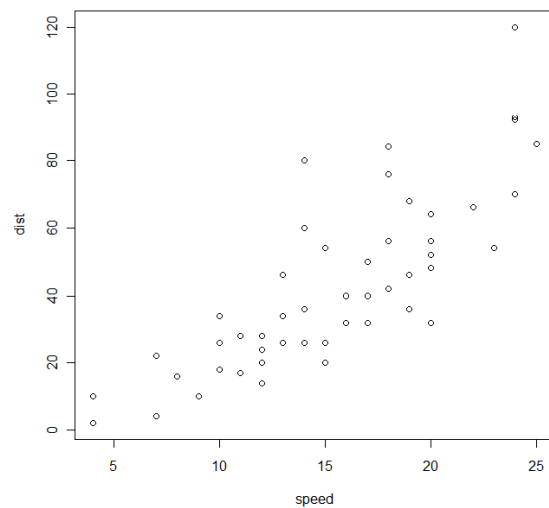
`plot(x,y) # O bien "plot(y ~ x)"`

Argumentos:

- type: p :point, l: line, b: both
- pch: tipo de símbolo: entero entre 1 y 25 o cualquier carácter entre comillas.
- col: color de los símbolos. e.g., "red"
- xlab : nombre del eje x (tipo carácter o entre comillas)
- ylab : nombre del eje y
- main: título del gráfico
- sub: subtítulo del gráfico
- cex: valor que define el tamaño de los textos y símbolos con respecto al valor por defecto.
- lwd: valor que define la anchura de las líneas

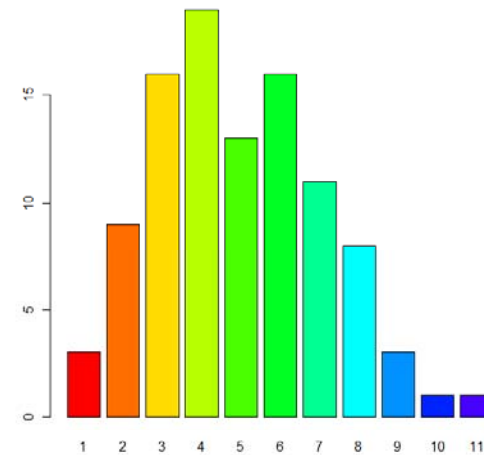
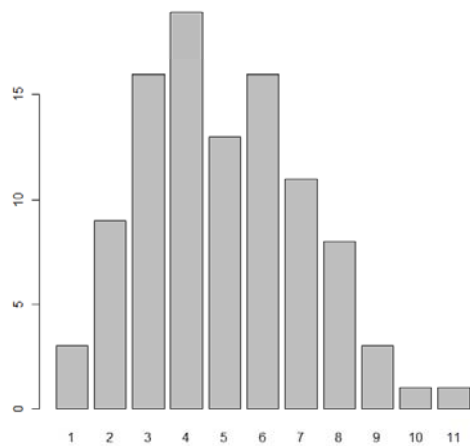
Ejemplo: plot

```
> data()  
> data(cars)  
> plot(cars$dist, cars$speed)  
> plot(cars, type="l")  
> plot(cars, type="b", col = "red", xlab="speed",  
      ylab="dist", main = "distancia vs velocidad", sub="Fig. 1.  
      Prueba de plot")
```



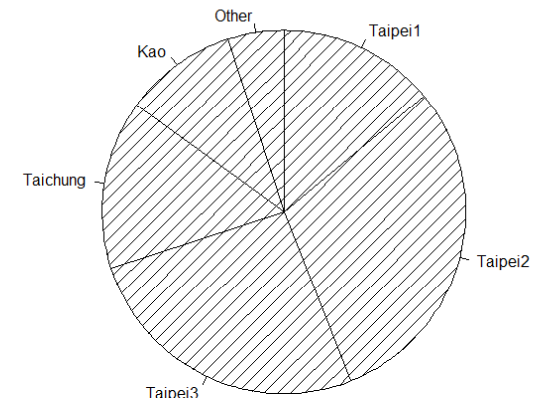
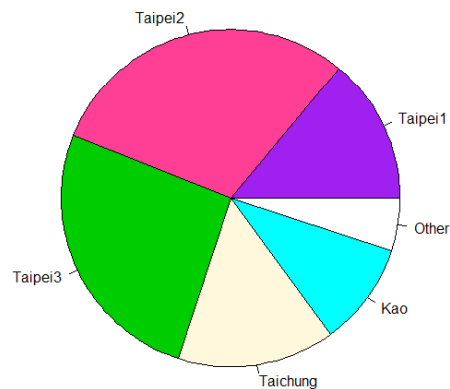
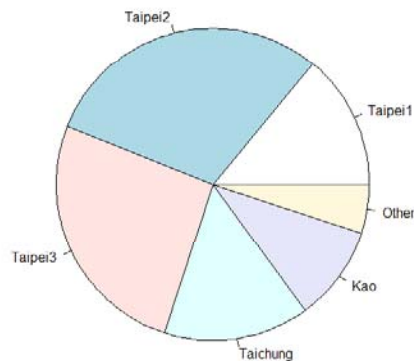
4.3 Diagramas de barra: barplot()

```
> Llegadas = table(NumberOfCar <- rpois(100,lambda=5))  
> Llegadas  
0 1 2 3 4 5 6 7 8 9 10 11 13  
1 4 4 10 19 21 11 16 3 3 5 2 1  
> barplot(Llegadas)  
> barplot(Llegadas, col=rainbow(14))
```



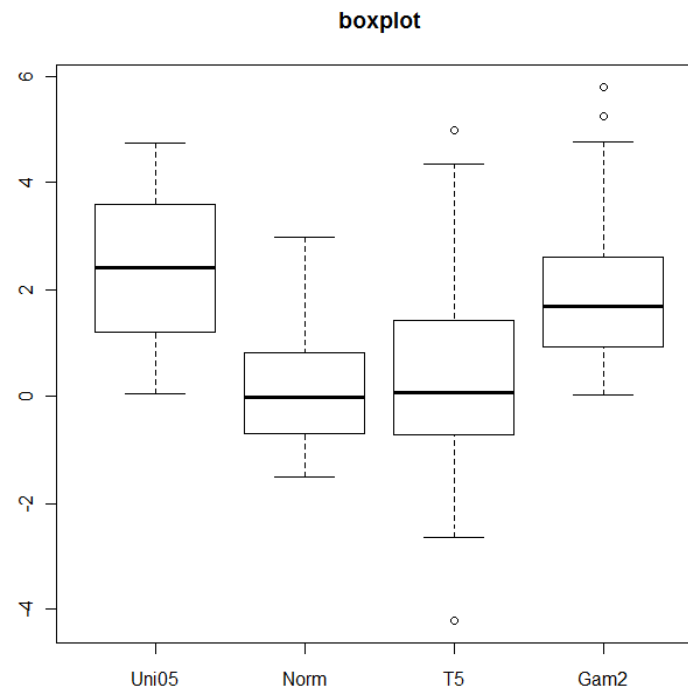
4.4 Gráficos circulares: pie()

```
> pie.sales = c(0.14, 0.30, 0.26, 0.15, 0.10, 0.05) # Ratio ventas
> names(pie.sales) = c("Taipei1", "Taipei2", "Taipei3", "Taichung", "Kao",
  "Other") # Zonas de venta
pie(pie.sales)
pie(pie.sales, col = c("purple", "violetred1", "green3", "cornsilk",
  "cyan", "white"))
pie(pie.sales, density = 10, clockwise=TRUE) # Densidad de las líneas
```



4.5 Gráfico de caja y patillas: boxplot()

```
> mat = cbind(Uni05 = (1:100)/21, Norm = rnorm(100), T5 =  
  rt(100, df = 5), Gam2 = rgamma(100, shape = 2))  
> boxplot(data.frame(mat), main = "boxplot")
```



4.6 Diagrama de Tallo y hojas: stem()

```
> mat = scan()  
1: 2 3 16 23 14 12 4 13 2 0 0 0 6 28 31 14 4 8 2 5  
21:  
Read 20 items  
>stem(mat)
```

The decimal point is 1 digit(s) to the right of the |

```
0 | 000222344568  
1 | 23446  
2 | 38  
3 | 1
```