

Curso de introducción al software estadístico R. Gráficos en R: el comando plot() y librerías específicas

Profesores: Carlos P.G., Roberto D.G. y Marcos C.S.



Fundación General
Universidad de La Laguna



Universidad
de La Laguna

5. Gráficos en R: el comando `plot()` y librerías específicas. Introducción a la interfaz Rstudio.

5.1 Gráficos en R: el comando `plot()` y librerías específicas.

Importación de datos en R

- Desde R podemos importar conjuntos de datos desde varios formatos (txt, SPSS, Excel, ...). Vamos a cargar el conjunto de datos 'ddt.txt':
 - group: Grupo de observaciones (mediciones en dos temporadas)
 - location: Código del lugar de captura (de 1 a 3)
 - location_name: Nombre del lugar (afluente, medio, desembocadura)
 - species: Código de especie (de 1 a 2)
 - species_name: Nombre de la especie del pez (buffalo, catfish)
 - length: Longitud del pez(en cm.)
 - weight: Peso pez (en grs.)
 - DDT_conc: Concentración de DDT (% de peso) medida
- Se puede descargar el archivo en local y luego importarlo

```
# local
ddt<-read.table("ddt.txt",header=TRUE,sep=" ",dec=".")
```

pero también existe la posibilidad de leer directamente desde internet

```
* https://raw.githubusercontent.com/cpgonzal/cursoIntroR2015/master/ddt.txt
* http://dl.dropboxusercontent.com/u/17677514/ddt.txt
```

```
# por https
library(RCurl)
myconn<-getURL("https://raw.githubusercontent.com/cpgonzal/cursoIntroR2015/master/ddt.txt", ssl.verifypeer = FALSE)
ddt<-read.table(textConnection(myconn),header=TRUE,sep=" ",dec=".")
```

- Podemos ejecutar la demo del programa mediante:

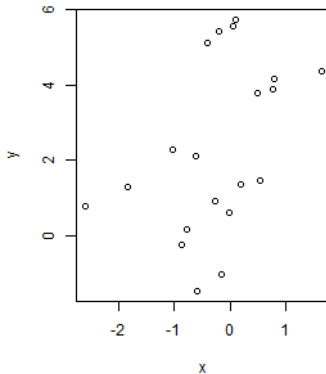
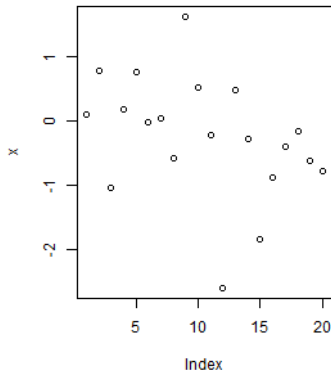
```
demo("graphics")
```

- Las funciones (alto nivel) que permiten crear un nuevo gráfico de un tipo dado (con ejes, etiquetas, títulos, etc..) son:
 - plot()
 - hist()
 - boxplot()
 -
- También existen funciones de bajo nivel que añaden información adicional a los gráficos anteriores:
 - points()
 - lines()
 -

El comando plot()

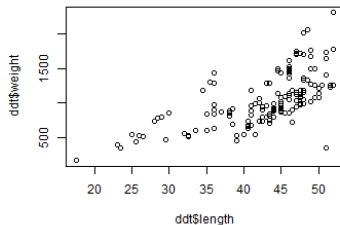
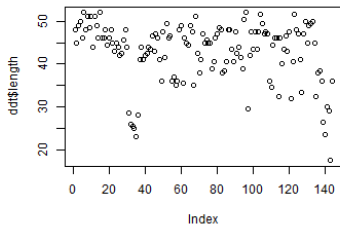
- La función plot() permite representar una o varias variables.

```
# Simulemos 2 vectores numéricos
x<-rnorm(20,0,1)
y<-x+rnorm(10,3,2)
plot(x) # gráfica secuencial
plot(x,y) # gráfica de 2 variables
```



Ejercicio 1

- Sobre el archivo `ddt.txt`, hacer un gráfico secuencial de la variable **length**.
- Representar en un gráfico las variables **length** frente a **weight**.



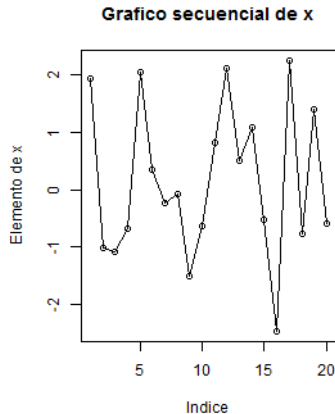
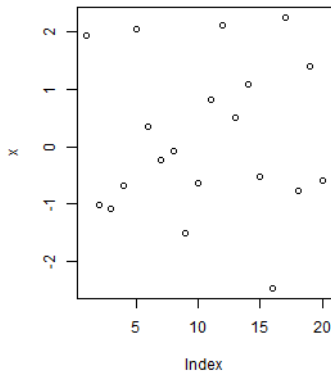
El comando plot(): Argumentos de la función

- Si vemos la ayuda del comando plot (?plot), vemos los siguientes argumentos:
 - **type** type of plot should to be drawn. Possible types are
 - "p" for points, (default)
 - "l" for lines,
 - "b" for both,
 - "c" for the lines part alone of "b",
 - "o" for both 'overplotted',
 - "h" for 'histogram' like (or 'high-density') vertical lines,
 - "s" y "S" for stair steps,
 - "n" for no plotting.
 - **main** an overall title for the plot [=NULL]
 - **sub** a sub title for the plot [=NULL]
 - **xlab** a title for the x axis
 - **ylab** a title for the y axis
 - **asp** the y/x aspect ratio

El comando plot(): Argumentos de la función

- Veamos la diferencia entre un gráfico y otro:

```
plot(x)
plot(x,type="o",main='Grafico secuencial de x',
      xlab='Indice',ylab='Elemento de x')
```

































El comando plot(): Otros argumentos

- Se pueden configurar otras opciones gráficas de plot (véase ?par):
 - **col** The colors for lines and points. Multiple colors can be specified so that each point can be given its own color. If there are fewer colors than points they are recycled in the standard fashion. Lines will all be plotted in the first colour specified.
 - **bg** a vector of background colors for open plot symbols, see points. Note: this is not the same setting as par("bg").
 - **pch** a vector of plotting characters or symbols
 - **cex** a numerical vector giving the amount by which plotting characters and symbols should be scaled relative to the default. This works as a multiple of par("cex"). NULL and NA are equivalent to 1.0. Note that this does not affect annotation
 - **lty** the line type: "blank", "solid", "dashed", "dotted", "dotdash", "longdash", or "twodash" (0, 1, 2, 3, 4, 5 or 6)
 - **cex.main**, **col.lab**, **font.sub**, **etc** settings for main- and sub-title and axis annotation, see title and par.
 - **lwd** the line width

El comando plot(): Otros argumentos

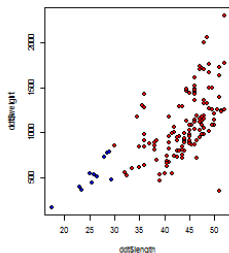
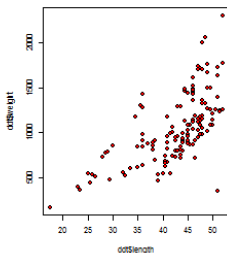
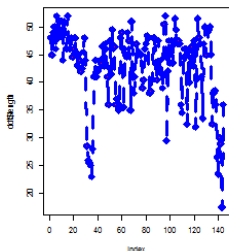
- Los posibles valores de **pch** son:

valores de pch

									
0	1	2	3	4	5	6	7	8	9
									
10	11	12	13	14	15	16	17	18	19
									
20	21	22	23	24	25	0	1	2	3

Ejercicio 2

- Hacer un gráfico secuencial de puntos y líneas del vector **length** utilizando como símbolo el rombo de color rojo, trazo discontinuo de color azul y anchura 3
- Hacer un gráfico de **length** frente a **weight** utilizando como símbolo el punto de color rojo
- En el gráfico anterior, representar los puntos a la derecha de **length=30** de color rojo y a la izquierda de color azul



Añadir otros elementos gráficos a un gráfico (I)

- Para añadir un punto a un gráfico lo hacemos con el comando 'points'

```
mean(ddt$length)
plot(ddt$length,
     type='o',
     bg='red', pch=23,
     col='blue', lty=2, lwd=3)
points(40,mean(ddt$length),pch=18,col="red")
points(80,mean(ddt$length),pch=18,col="red")
points(120,mean(ddt$length),pch=18,col="red")
```

- Si lo que se desea es añadir una línea, lo hacemos con el comando 'lines'

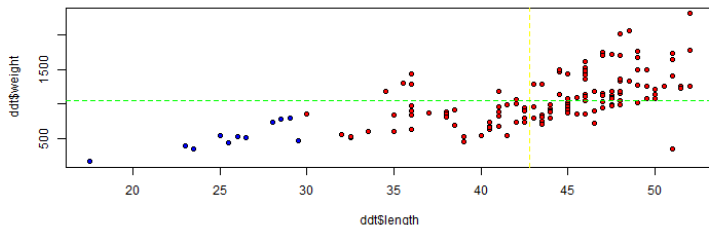
```
plot(ddt$length,
     type='o',
     bg='red', pch=23,
     col='blue', lty=2, lwd=3)
lines(1:144,rep(mean(ddt$length),144),col="red",lty=2)
```

Añadir otros elementos gráficos a un gráfico (II)

- Aunque hay comandos que pueden añadir una línea de forma más sencilla que 'lines'

```
plot(ddt$length,
     type='o',
     bg='red', pch=23,
     col='blue', lty=2, lwd=3)
abline(h=mean(ddt$length), col="red", lty=2)
abline(v=50, col="green", lty=2)
```

- Como ejemplo, realizar el gráfico de **length** frente a **weight** (con los puntos a la derecha de **length=30** de color rojo y a la izquierda de color azul). Trazar 2 líneas con las medias de ambas variables de colores distintos



Nivel ninja: Trazar la gráfica de dispersión de 2 variables (X,Y) y la recta de regresión

- Recordemos que el modelo de regresión de **length** frente a **weight** es

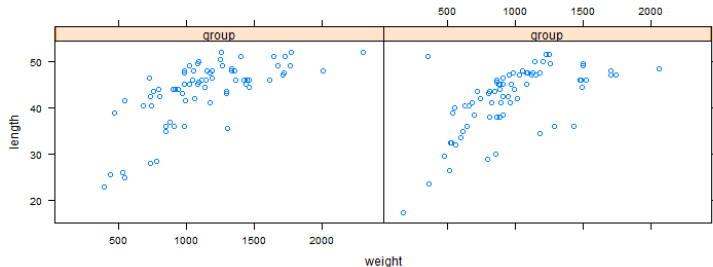
```
reg.lm<-lm(ddt$length~ddt$weight)
```

- Trazar la línea de regresión sobre el gráfico de **length** frente a **weight**

```
reg.lm<-lm(ddt$weight~ddt$length)
plot(ddt$length,ddt$weight,
     type='p',
     bg=colores[1*(ddt$length<30)+2*(ddt$length>=30)], pch=21)
abline(reg.lm,col="green",lty=2)
```

- Si queremos hacer un gráfico donde representemos una o varias variables diferenciando por categorías de un factor podemos usar 'xyplot'

```
library(lattice)
xyplot(ddt$length~ddt$weight | ddt$group)
xyplot(length~weight | group, data=ddt)
xyplot(length~weight | as.factor(group), data=ddt)
```



- Como ejemplo, realizar un gráfico de **DDT_conc** frente a **length** desagrupando por el nombre de la especie.

