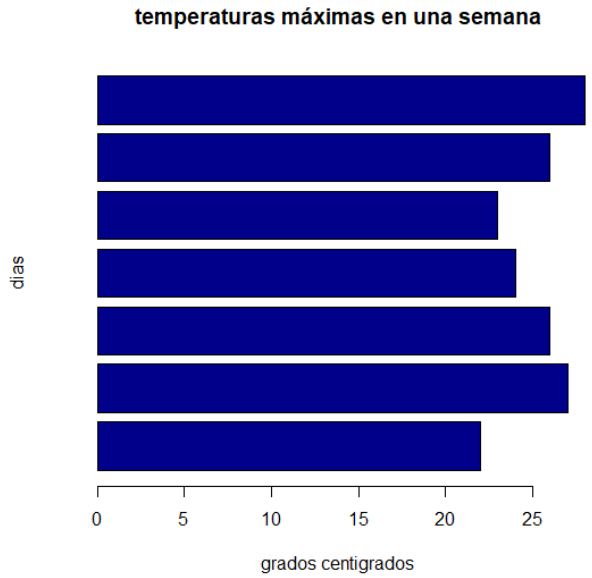
R Commander

23/03/2021

TRATAMIENTO DE DATOS

> max.temp<-c(22,27,26,24,23,26,28)

> barplot(max.temp, main="temperaturas máximas en una semana", xlab="grados centigrados", ylab="dias",col="darkblue",horiz=TRUE) → "horiz" en lugar de "horizontal" sin motivo aparente



> age<-c(17,18,18,17,18,19,18,16,18,18)

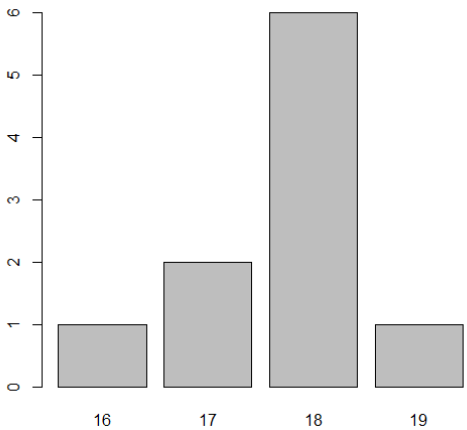
> table(age)

age

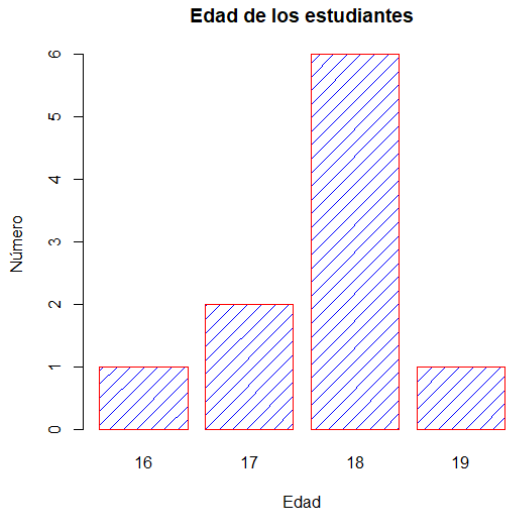
16 17 18 19

1 2 6 1

> barplot(table(age))



> barplot(table(age),main="Edad de los estudiantes", xlab="Edad", ylab="Número", col="blue",border="red",density=10) → "density" = relleno



// > library(survival)

> Titanic

, , Age = Child, Survived = No

Sex

Class Male Female

1st 0 0

2nd 0 0

3rd 35 17

Crew 0 0

…

> Titanic$Male

Error in Titanic$Male : $ operator is invalid for atomic vectors → Porque no está bien definida

> margin.table(Titanic,1)

Class

1st 2nd 3rd Crew

325 285 706 885

> margin.table(Titanic,2)

Sex

Male Female

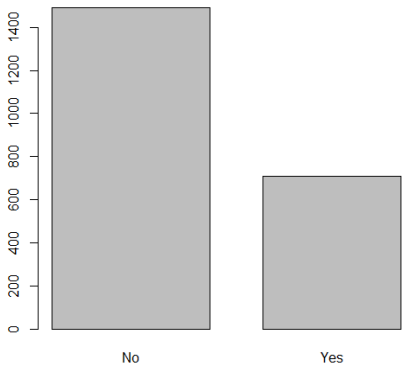
1731 470

```
> margin.table(Titanic,3)
Age
Child Adult
  109  2092
> margin.table(Titanic,4)
Survived
  No  Yes
1490  711
> a<-margin.table(Titanic,4)
> a
Survived
  No  Yes
1490  711
> b<-margin.table(Titanic,1)
> b
Class
 1st  2nd  3rd Crew
 325  285  706  885
> titanic<-c(a,b)
> titanic
  No  Yes  1st  2nd  3rd Crew
1490  711  325  285  706  885
> titanic<-c(b,a)
> titanic
 1st  2nd  3rd Crew   No  Yes
 325  285  706  885 1490  711
```

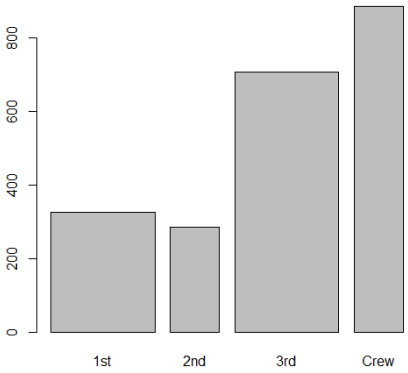
→ Asignando variables

→ Construyendo tablas con vectores. Se colocan en el orden que se llaman.

> barplot(a,b) → Al igual que ocurría con los vectores, los gráficos se colocan en el orden de llamada. En este caso al estar realizando una gráfica de una variable llamando a dos, solo trabaja con la primera.



> barplot(b,a)



// > str(airquality)

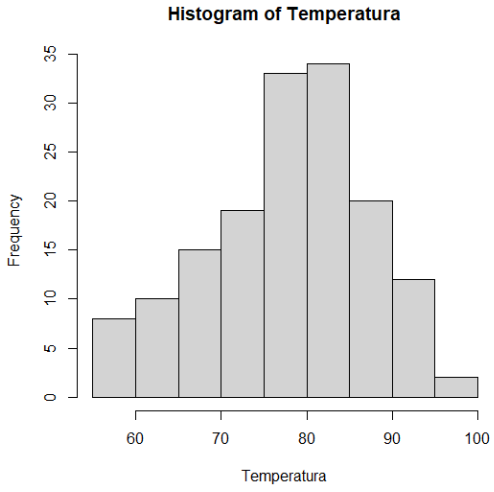
'data.frame': 153 obs. of 6 variables:

$ Ozone : int 41 36 12 18 NA 28 23 19 8 NA ...

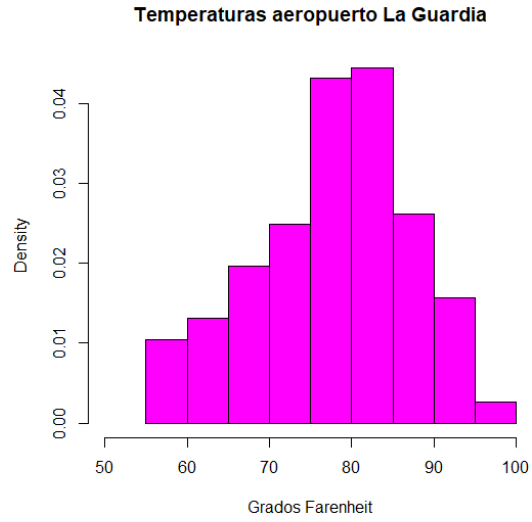
$ Solar.R: int 190 118 149 313 NA NA 299 99 19 194 ...

> Temperatura<-airquality$Temp → Extraer datos construyendo una variable

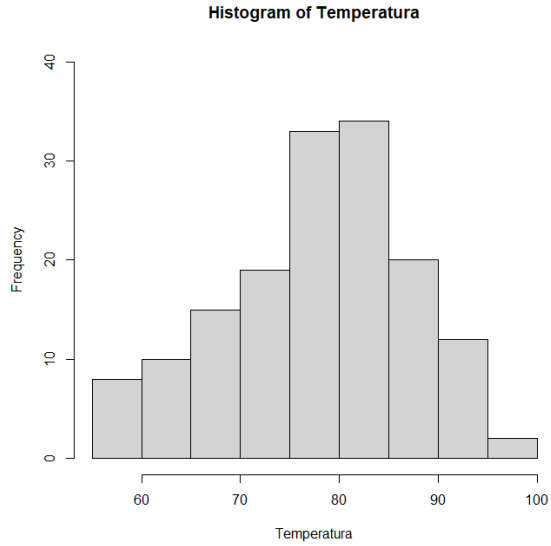
> hist(Temperatura)



> hist(Temperatura,main="Temperaturas aeropuerto La Guardia", xlab="Grados Farenheit",xlim=c(50,100),col="magenta", freq=FALSE) → "xlim" = limites máximo y mínimo de eje x, "freq = FALSE"= cambia la variable y comparativa.



> hist(Temperatura,ylim=c(0,40))



> h<-hist(Temperatura)

> h → > Convirtiendo en vector el histograma podemos sacar la información interna.

$breaks

[1] 55 60 65 70 75 80 85 90 95 100

$counts

[1] 8 10 15 19 33 34 20 12 2

$density

[1] 0.010457516 0.013071895 0.019607843 0.024836601 0.043137255 0.044444444

[7] 0.026143791 0.015686275 0.002614379

$mids

[1] 57.5 62.5 67.5 72.5 77.5 82.5 87.5 92.5 97.5

$xname

[1] "Temperatura"

$equidist

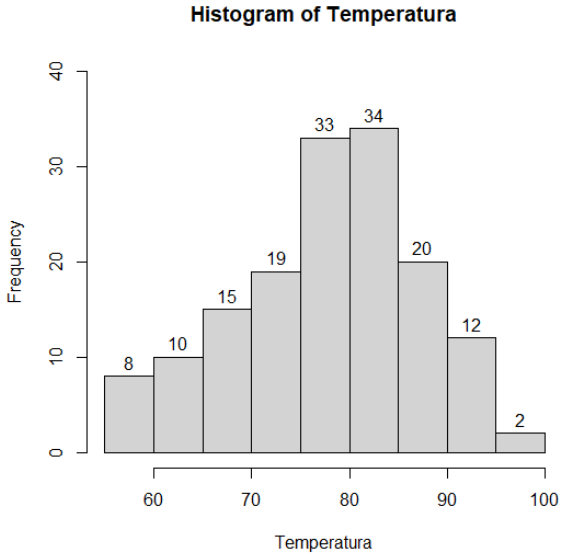
[1] TRUE

attr(,"class")

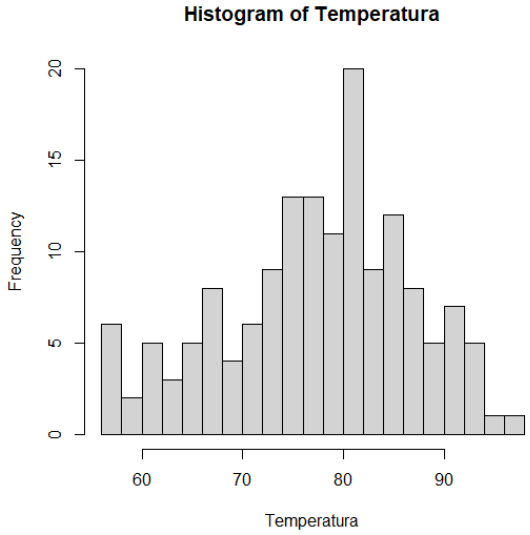
[1] "histogram"

> h<-hist(Temperatura,ylim=c(0,40))

> text(h$mids,h$counts,labels=h$counts,adj=c(0.5,-0.5)) → Modificaciones gráficas (Añadir cantidad de y en la posición de los valores medios).

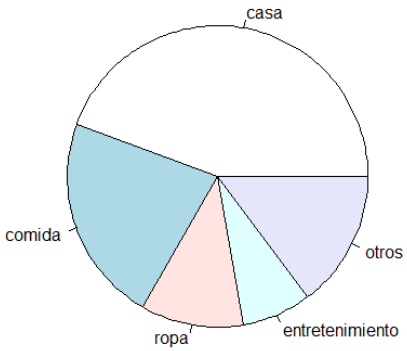


> hist(Temperatura,breaks=20) → "breaks"= Número de divisiones



PIE CHART

```
> gastos<-c("casa","comida","ropa","entretenimiento","otros")
> dinero<-c(600,300,150,100,200)
> pie(dinero,labels=gastos)
```



ESTUDIO DE GRÁFICOS DE CAJAS Y BIGOTES

> #HSAUR2 →Librería con ejemplo muy extenso para trabajar

```
> library(HSAUR2)
> household
  housing food goods service gender
1     820  114   183     154 female
2     184   74     6      20 female
3     921   66  1686     455 female
4     488   80   103     115 female  …
> str(household)
'data.frame':   40 obs. of  5 variables:
 $ housing: int  820 184 921 488 721 614 801 396 864 845 ...
 $ food   : int  114 74 66 80 83 55 56 59 65 64 ...
 $ goods  : int  183 6 1686 103 176 441 357 61 1618 1935 ...
 $ service: int  154 20 455 115 104 193 214 80 352 414 ...
 $ gender : Factor w/ 2 levels "female","male": 1 1 1 1 1 1 1 1 1 1 ...
```

> head(household) → > Semejante a “str” lo saca como tabla pero solo te da los 6 primeros

```
  housing food goods service gender
1     820  114   183     154 female
2     184   74     6      20 female
```

3 921 66 1686 455 female

4 488 80 103 115 female

5 721 83 176 104 female

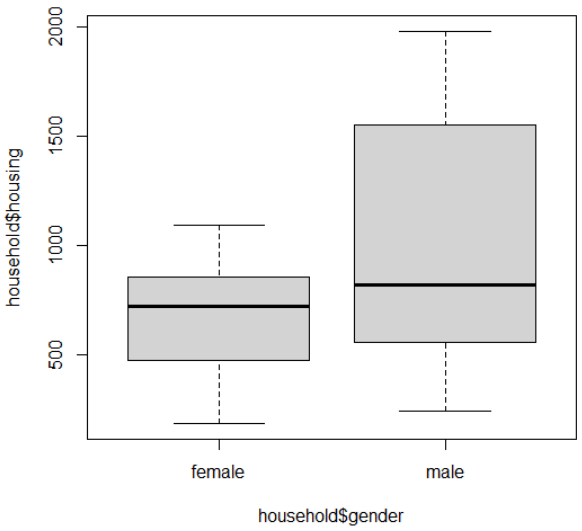
6 614 55 441 193 female

> meanh<-mean(household$housing) → Hacer la media

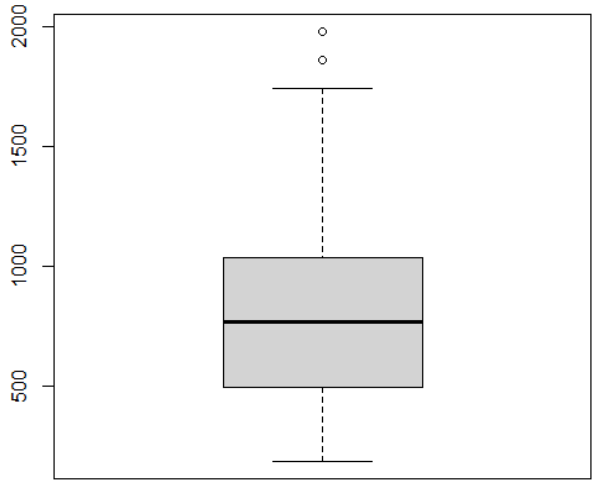
> meanh

[1] 828.375

> plot(household$housing~household$gender) → Enfrentar datos con ~ (En función de) que se hace con Ctrl+Alt+4



> boxplot(household$housing) → Análisis para detectar outliers



> household$housing → Arroja los datos

[1] 820 184 921 488 721 614 801 396 864 845 404 781 457 1029 1047

[16] 552 718 495 382 1090 497 839 798 892 1585 755 388 617 248 1641

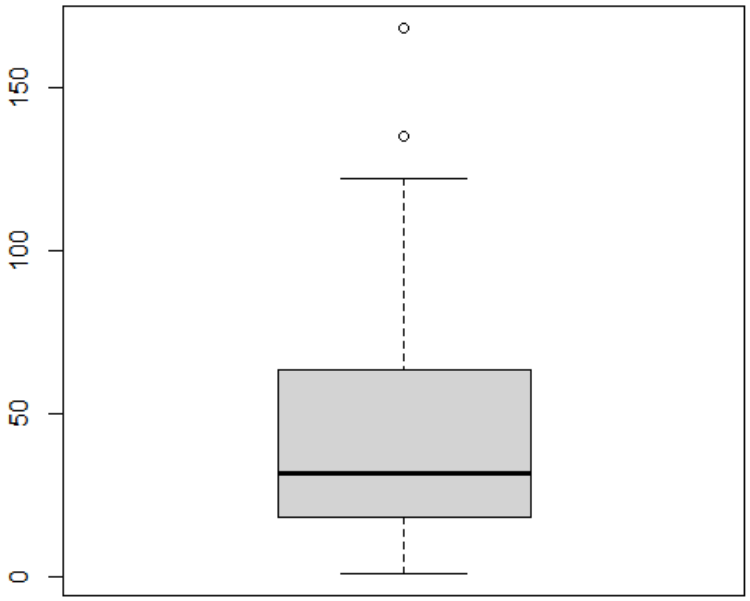
[31] 1180 619 253 661 1981 1746 1865 238 1199 1524

> str(airquality)

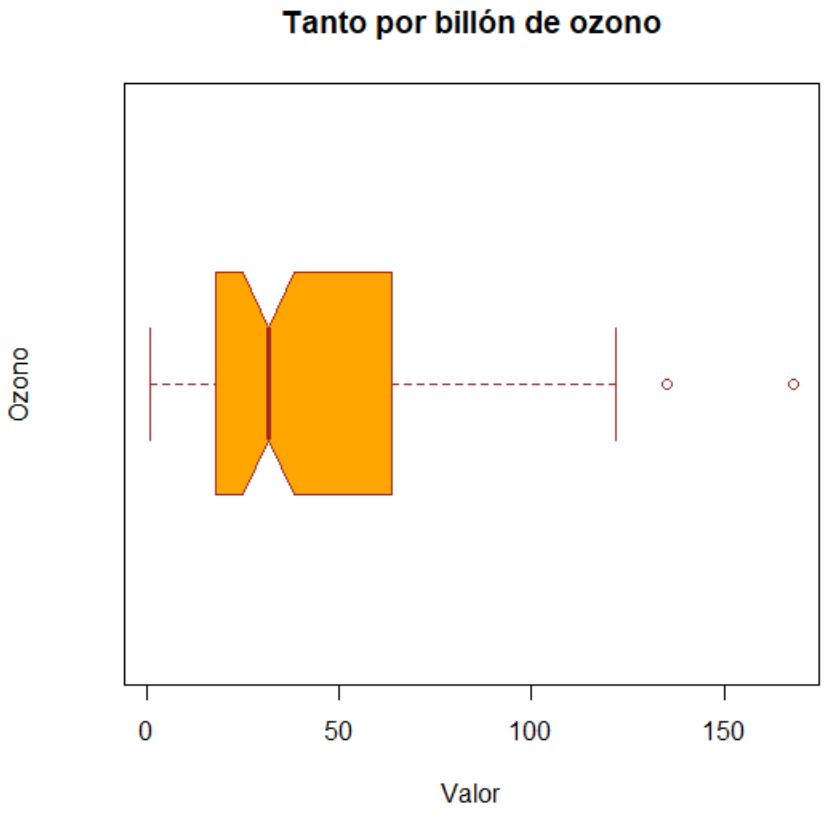
'data.frame': 153 obs. of 6 variables:

$ Ozone : int 41 36 12 18 NA 28 23 19 8 NA ...

> boxplot(airquality$Ozone)



> boxplot(airquality$Ozone,main="Tanto por billón de ozono", xlab="Valor", ylab="Ozono",col="orange",border="brown",horizontal=TRUE,notch=TRUE) → "notch"=entalla



```
> b<-boxplot(airquality$Ozone)
> b
$stats
     [,1]
[1,]   1.0
[2,]  18.0
[3,]  31.5
[4,]  63.5
[5,] 122.0
attr(,"class")
      1
"integer"
$n
[1] 116
```

$conf (Intervalo de confianza)

```
       [,1]
[1,] 24.82518
[2,] 38.17482
```

$out (outliers)

```
[1] 135 168
$group
[1] 1 1
$names
[1] "1"
```

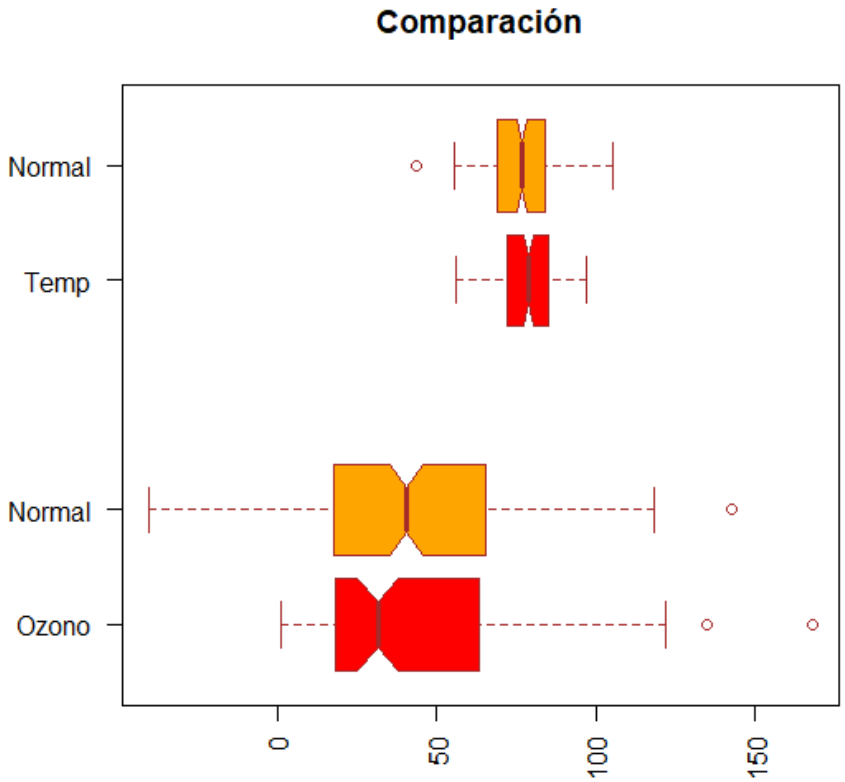
COMPARATIVA POR BIGOTES DE DATOS FRENTE A SU NORMAL

```
> ozono<-airquality$Ozone
> temp<-airquality$Temp
> ozono_norm<-rnorm(200,mean=mean(ozono,na.rm=TRUE),sd=sd(ozono,na.rm=TRUE))
```

- “rnorm” realiza una adaptación a distribución normal.
- 200 medidas hechas con la media y la distribución estándar de ozono.
- “na.rm” se utiliza para que la función no se detenga si no encuentra valores

```
> temp_norm<-rnorm(200,mean=mean(temp,na.rm=TRUE),sd=sd(temp,na.rm=TRUE))
```

>
boxplot(ozono,ozono_norm,temp,temp_norm,main="Comparación",at=c(1,2,4,5),names=c("Ozono","Normal","Temp","Normal"),las=2,col=c("red","orange"),border="brown",horizontal=TRUE,notch=TRUE) →"las" agrupa según como se asigne y "at" indica las posiciones (Colocación)



> boxplot(Temp~Month,data=airquality) →Otra forma de enfrentar datos sin necesidad de "$"

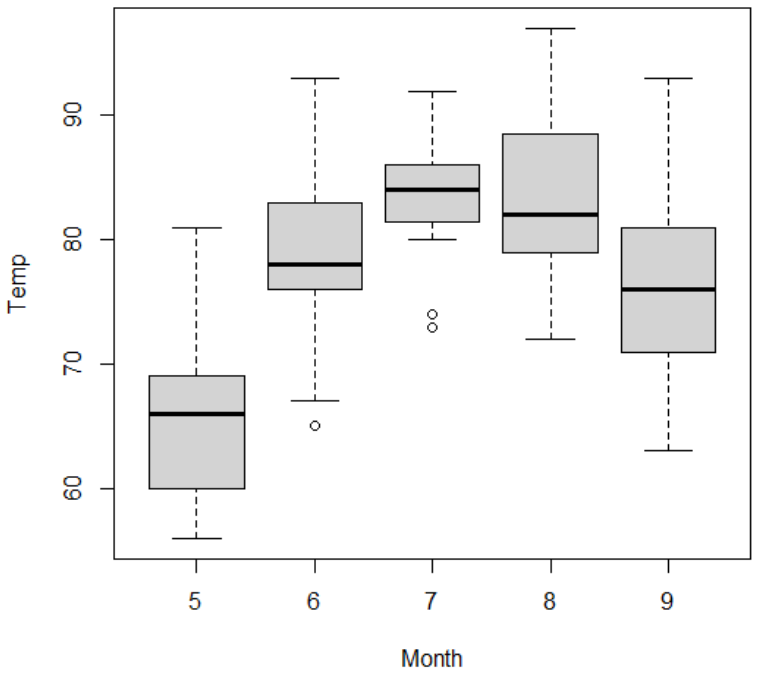
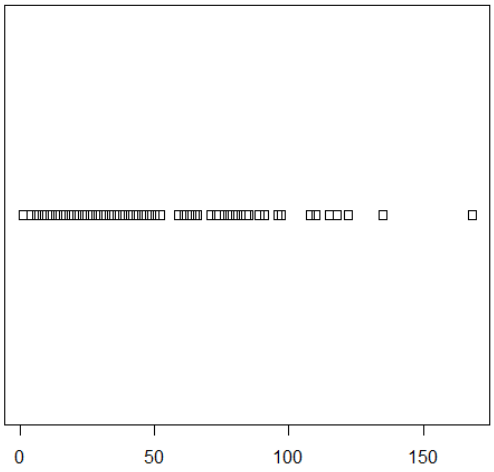


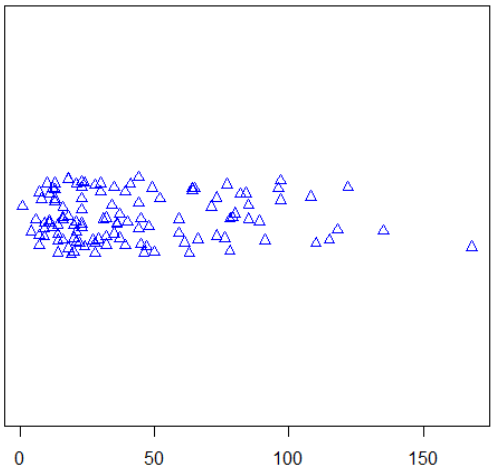
GRÁFICO DE CINTAS

> stripchart(airquality$Ozone) → Permite obtener el número de datos por unidad de tiempo

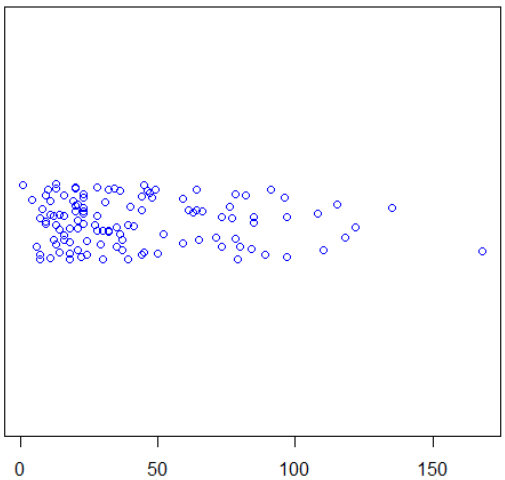


> stripchart(airquality$Ozone, method="jitter",col="blue",pch=2)

- "pch" indica el formato del símbolo.
- el método "jitter" agita los valores (el eje y no significa nada), así se consigue más claridad y que no se solapen.

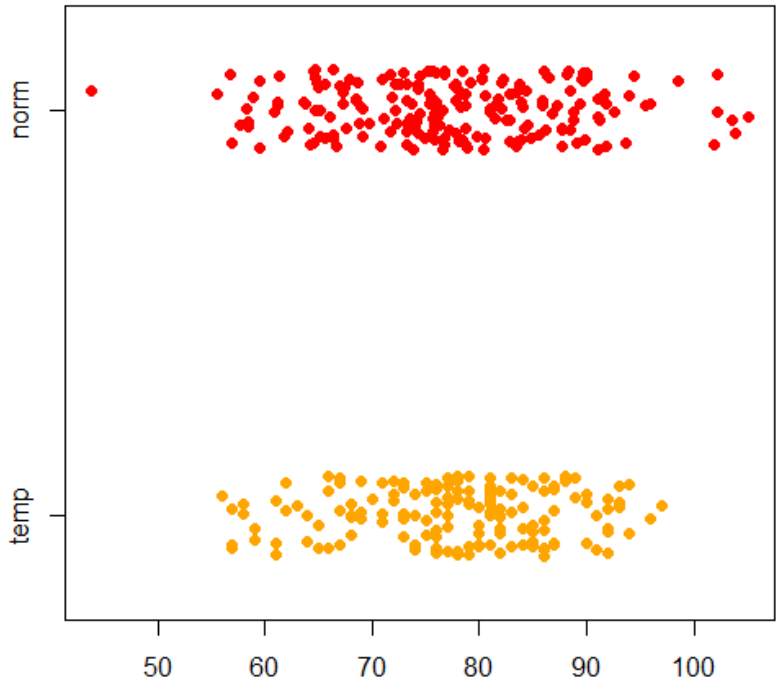


> stripchart(airquality$Ozone, method="jitter",col="blue",pch=1)

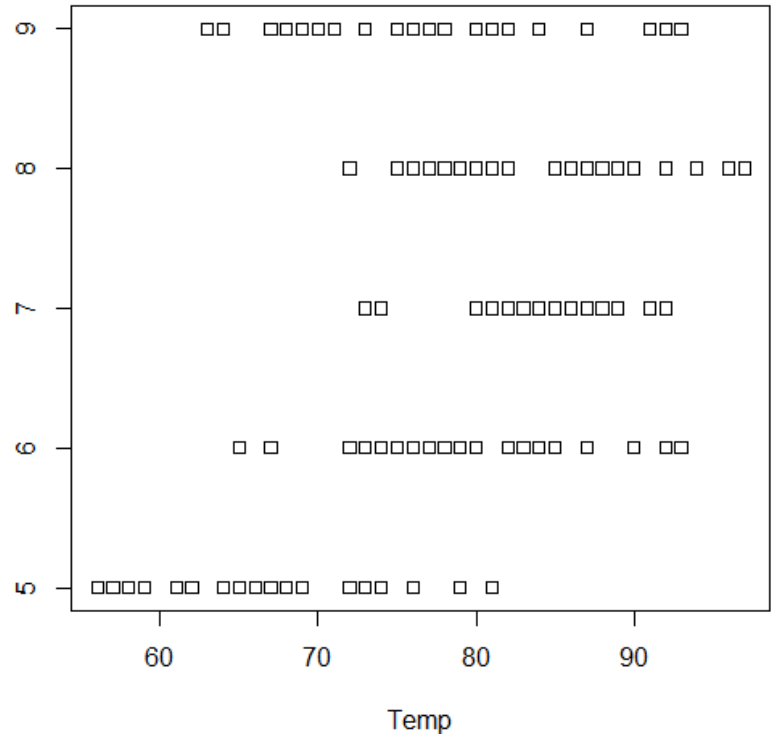


> x<-list("temp"=temp,"norm"=temp_norm)

> stripchart(x, method="jitter",col=c("orange","red"),pch=16)



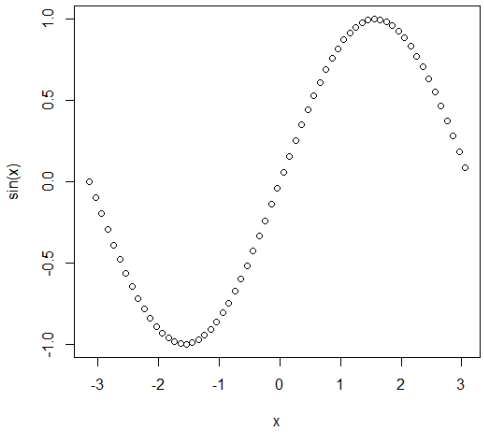
> stripchart(Temp~Month, data=airquality)



TIPOLOGÍA PINTADO DE GRÁFICAS

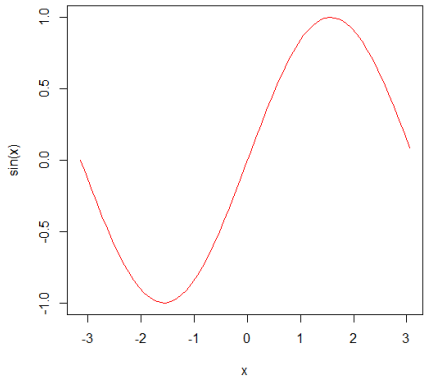
```
> x<-seq(-pi,pi,0.1)
```

```
> plot(x,sin(x))
```

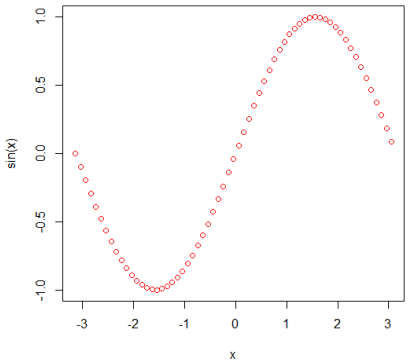


1. “p” es puntos
2. “l” es líneas
3. “b” es ambos
4. “c” puntos se unen con líneas → NO FUNCIONA
5. “o” sobrescriben los puntos y líneas
6. “s” gráfico de sierra
7. “h” ven las líneas como en un histograma
8. “n” no se pone nada

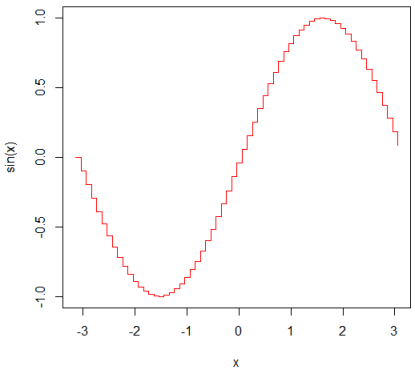
```
> plot(x,sin(x), type="l", col="red")
```




```
> plot(x,sin(x), type="b", col="red")
```

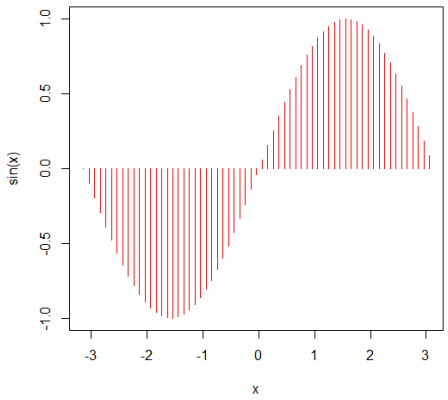


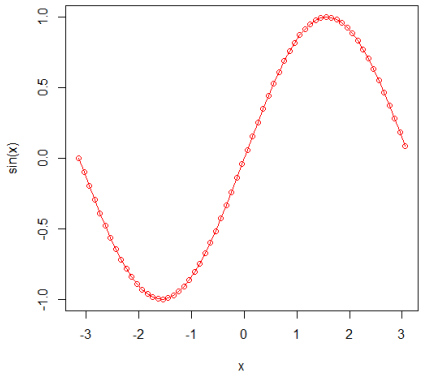
> plot(x,sin(x), type="s", col="red")



> plot(x,sin(x), type="h", col="red")

> plot(x,sin(x), type="o", col="red")

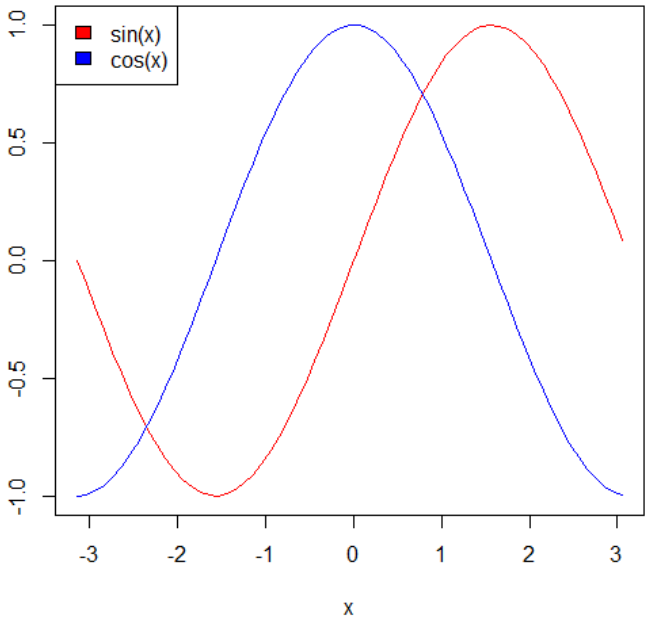




> plot(x,sin(x), ylab="",type="l", col="red")

> lines(x,cos(x),col="blue") → Superponer dos gráficos

> legend("topleft",c("sin(x)","cos(x)"),fill=c("red","blue")) → Diseño de leyenda



MODIFICACIONES PRESENTACIÓN GRÁFICAS

> par() → Función que permite juntar y colocar cosas en un gráfico moverlas y tal. Al ejecutar el código anterior se obtendrá una lista con 72 objetos en la cual se tendrán los valores que cada uno de los parámetros asume inicialmente en una sesión de R. Luego de modificar uno o alguno de los parámetros de la función par, todos los gráficos que se construyan de ahí en adelante estarán afectados por el cambio realizado.

$xlog

[1] FALSE

$ylog

[1] FALSE

$adj

[1] 0.5

…

$mfrow → Número de gráficas dispuestas en el espacio (Matriz)

[1] 1 1

….

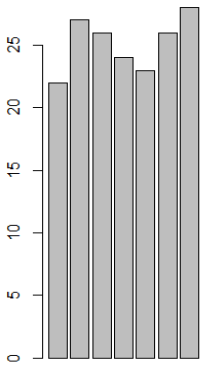
> max.temp (Variable diseñada al principio de la clase (No es una librería))

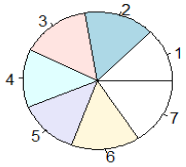
[1] 22 27 26 24 23 26 28

> par(mfrow=c(1,2)) → La función "par" nos permite distribuir la pantalla como queramos. Hemos dicho una fila y dos columnas.

> barplot(max.temp)

> pie(max.temp,radius=1)





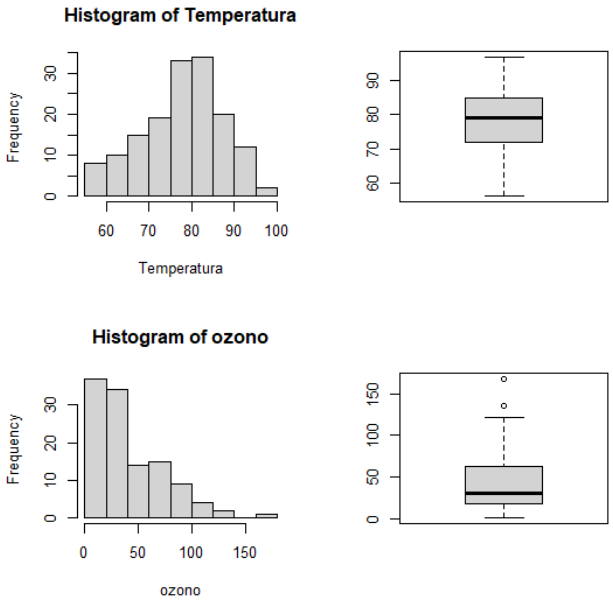
> par(mfrow=c(2,2)) → Hemos dicho dos filas y dos columnas.

> hist(Temperatura)

> boxplot(Temperatura)

> hist(ozono)

> boxplot(ozono)



> colors() → Permite conocer los colores disponibles

```
 [1] "white"           "aliceblue"       "antiquewhite"
 [4] "antiquewhite1"   "antiquewhite2"   "antiquewhite3"
 [7] "antiquewhite4"   "aquamarine"      "aquamarine1"
```

> rgb(0,1,0) → Definir colores hexadecimalmente. R te proporciona el código.

```
[1] "#00FF00"
```

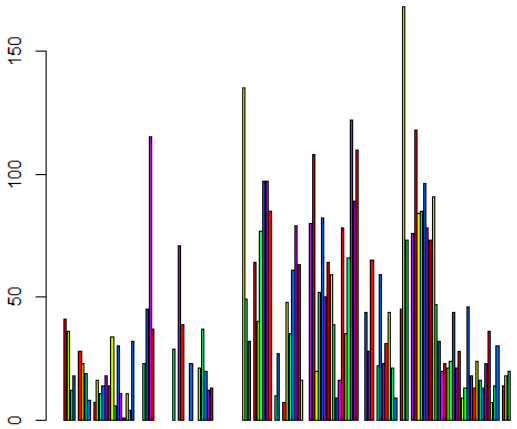
> rgb(0.3,0.7,0.9)

```
[1] "#4DB3E6"
```

> rainbow(5) → Combinaciones de colores implementadas en R. El número indica el número de colores.

```
[1] "#FF0000" "#CCFF00" "#00FF66" "#0066FF" "#CC00FF"
```

> barplot(ozono, col=rainbow(5))

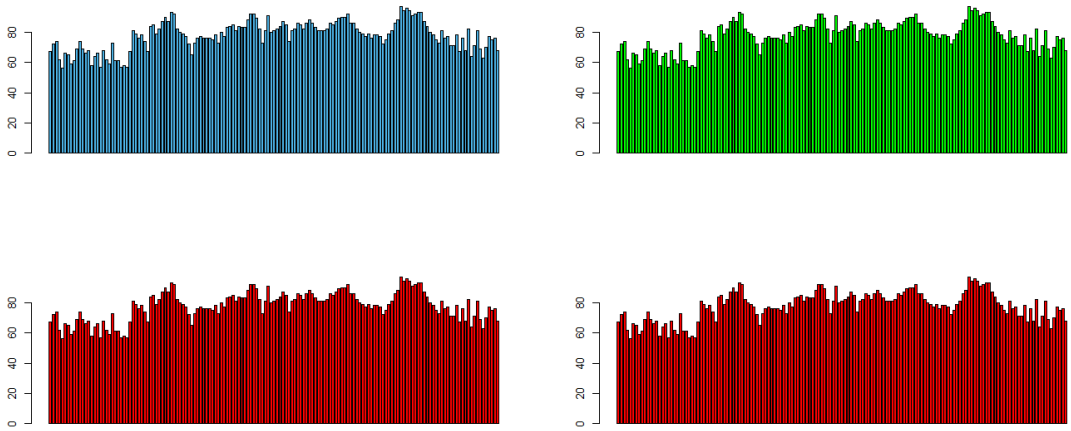


> barplot(temp,col="#4DB3E6") → Implementar colores en código hexagesimal

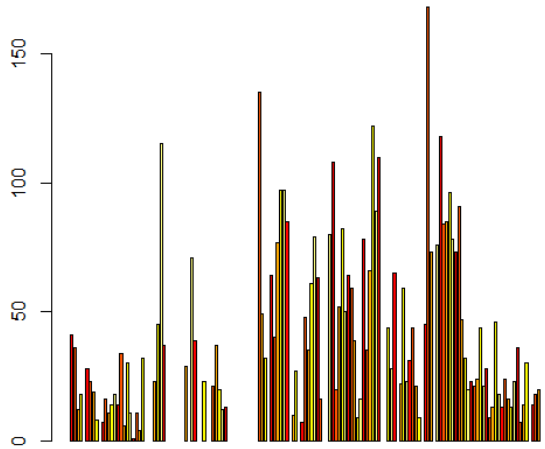
> barplot(temp,col="#00FF00")

> barplot(temp,col="red")

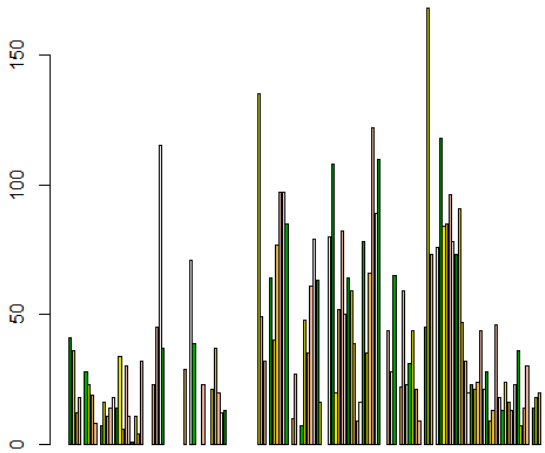
> barplot(Temperatura,col="red")



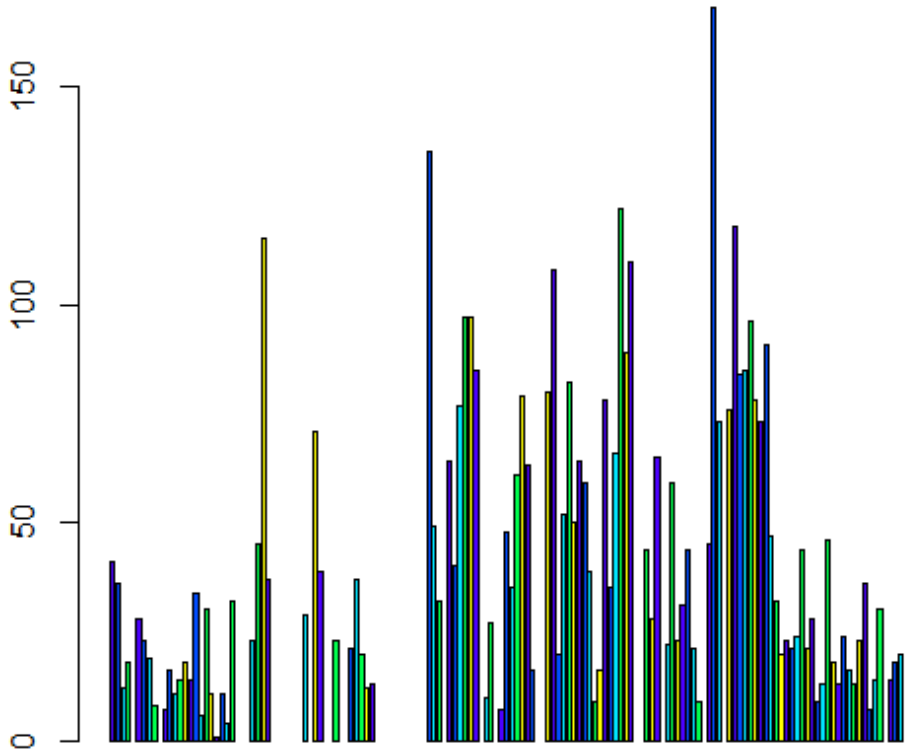
> barplot(ozono, col=heat.colors(5)) → Combinaciones de colores implementadas en R



> barplot(ozono, col=terrain.colors(5)) → Combinaciones de colores implementadas en R.



> barplot(ozono, col=topo.colors(5)) → Combinaciones de colores implementadas en R



FIGURAS 3D

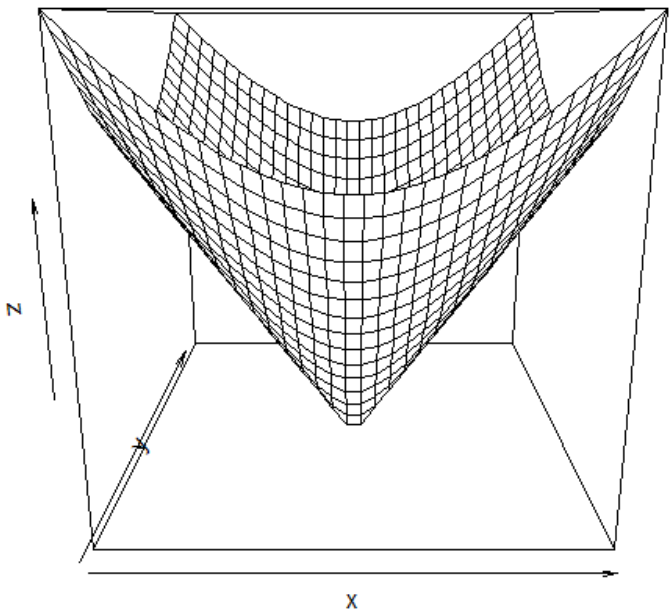
> cono<-function(x,y){

+ sqrt(x^2+y^2) } → Ecuación longitud de generatriz.

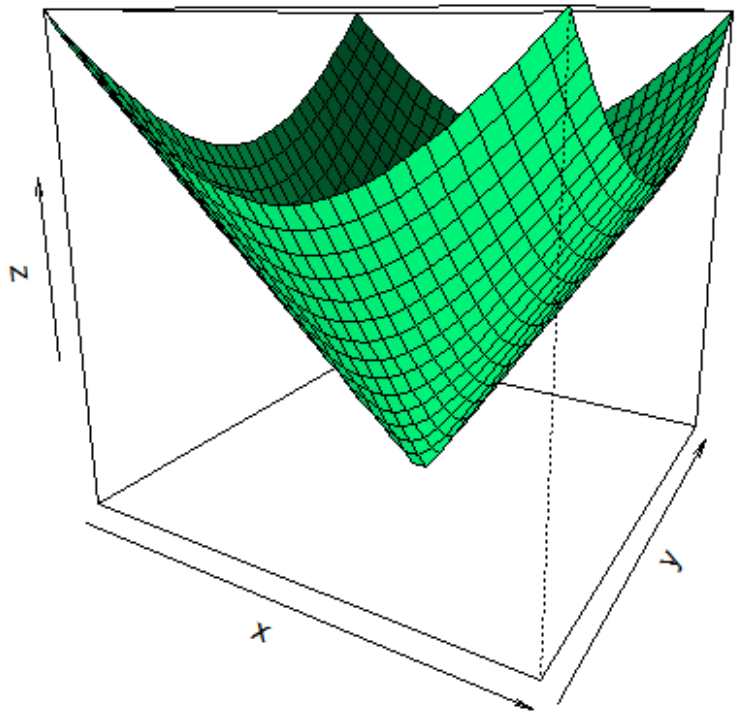
> x<-y<-seq(-1,1,length=30)

> z<-outer(x,y,cono)

> persp(x,y,z) → Graficar modelos 3D.



> persp(x,y,z, theta=30, phi=15, col="springgreen", shade=0.5) → "theta" y "phi" son ángulos de giro, "shade" proporciona sombreado.



COMANDOS INTERNOS Y MANEJO DE CARPETAS

> objects() → Permite conocer los elementos/variables abiertas hasta el momento

```
 [1] "a"         "age"       "b"         "cono"      "dinero"
 [6] "gastos"    "h"         "max.temp"  "meanh"     "ozono"
[11] "ozono_norm" "temp"      "temp_norm" "Temperatura" "titanic"
[16] "x"         "y"         "z"
```

> ls() → Apocope de lo anterior

```
 [1] "a"         "age"       "b"         "cono"      "dinero"
 [6] "gastos"    "h"         "max.temp"  "meanh"     "ozono"
[11] "ozono_norm" "temp"      "temp_norm" "Temperatura" "titanic"
[16] "x"         "y"         "z"
```

> remove() → Aumenta espacio RAM eliminando los objects pero si se vuelve a ejecutar "objects()" se recuperan

> library() →Indica librerías instaladas

> mydata<-read.table("C:/myfolder/abc",header=TRUE, sep="\t", na.strings="-9")

- Leer tablas dentro de documentos.
- 'na.string="-9"' salta una posición cuando encuentra un blanco.
- Poniendo "\t " se detectan elementos separados por comas.

> mydata.stata<-read.dta("http://dss.princeton.edu/training/students.dta")

> mydata.spss<-read.spss("http://dss.princeton.edu/training/students.sav")

> mydata.sas<-sasxport.get("http://dss.princeton.edu/training/students.xpt")

- Obtener archivos vía inernet en función de extensión

> data<-read.csv(file.choose()) → Te envía a la carpeta de documentos para que selecciones el archivo que quieres abrir.

TRATAMIENTO DE DATOS

> summary(Temperatura) → Resumen de un cojunto que hayamos cargado.

> edit(Temperatura) → Permite editar un conjunto de datos.

[1] 67 72 74 62 56 66 65 59 61 69 74 69 66 68 58 64 66 57 68 62 59 73 61 61 57

[26] 58 57 67 81 79 76 78 74 67 84 85 79 82 87 90 87 93 92 82 80 79 77 72 65 73 ….

[126] ….93 93 87 84 80 78 75 73 81 76 77 71 71 78 67 76 68 82 64 71 81 69 63 70 77

[151] 75 76 68 → Escribe resultados tras modificación manual en ventana emergente.

> str(Temperatura) → Estructura interna

int [1:153] 67 72 74 62 56 66 65 59 61 69 ...

> names(airquality) → Nombres encabezamientos de una tabla

[1] "Ozone" "Solar.R" "Wind" "Temp" "Month" "Day"

> head (airquality) → Se explicó anteriormente y solo lista los 6 primeros datos

Ozone Solar.R Wind Temp Month Day

1 41 190 7.4 67 5 1

…

6 28 NA 14.9 66 5 6

> head(airquality, n=15) → Puede modificarse para que proporcione más datos

Ozone Solar.R Wind Temp Month Day

1 41 190 7.4 67 5 1

2 36 118 8.0 72 5 2

…

14 14 274 10.9 68 5 14

15 18 65 13.2 58 5 15

> tail(airquality) → Igual que “head” pero empieza por la cola (6 últimos)

Ozone Solar.R Wind Temp Month Day

148 14 20 16.6 63 9 25

149 30 193 6.9 70 9 26

150 NA 145 13.2 77 9 27

151 14 191 14.3 75 9 28

152 18 131 8.0 76 9 29

153 20 223 11.5 68 9 30

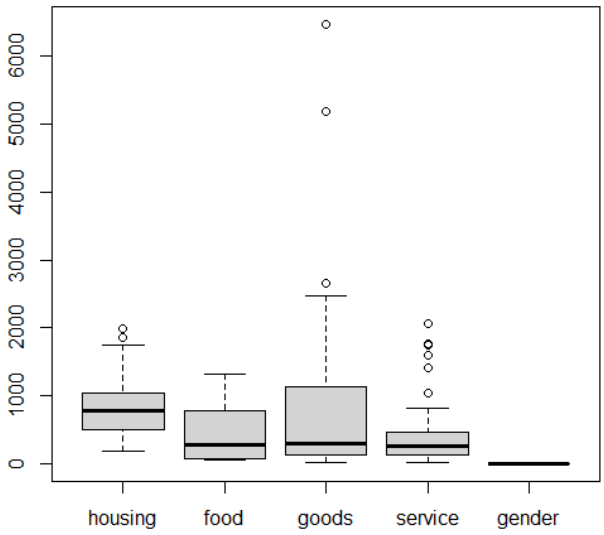
EJERCICIOS CLASE SOLUCIÓN RAFA

```
> summary(household)
```

```
   housing           food           goods          service
 Min.   : 184.0   Min.   :  47.00   Min.   :   6.0   Min.   :  20.0
 1st Qu.: 493.2   1st Qu.:  76.25   1st Qu.: 127.8   1st Qu.: 139.0
 Median : 768.0   Median : 268.00   Median : 294.5   Median : 262.0
 Mean   : 828.4   Mean   : 435.20   Mean   : 873.7   Mean   : 460.6
 3rd Qu.:1033.5   3rd Qu.: 768.25   3rd Qu.: 948.2   3rd Qu.: 452.8
 Max.   :1981.0   Max.   :1308.00   Max.   :6471.0   Max.   :2063.0
   gender
 female:20
 male  :20
```

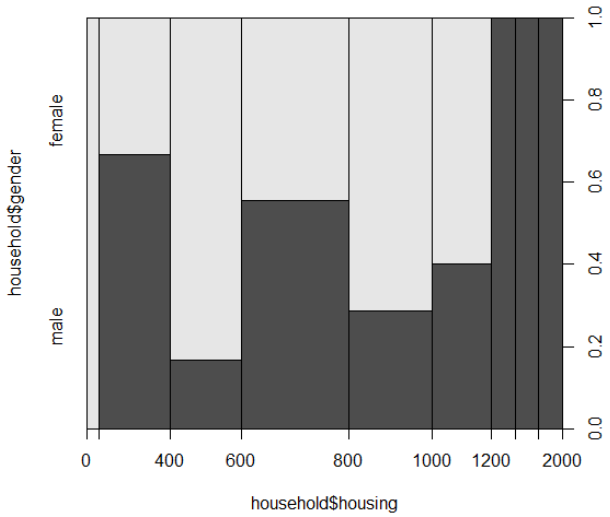
Mean : 828.4 Mean : 435.20 Mean : 873.7 Mean : 460.6 → Solicitado

```
> boxplot(household)
```

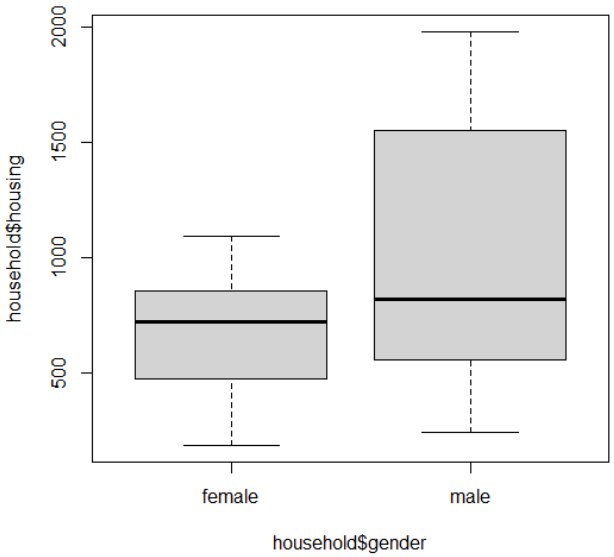



```
> sd(household$housing)
[1] 462.2199
> sd(household$food)
[1] 401.2495
> sd(household$goods)
[1] 1368.529
> sd(household$service)
[1] 531.5572
```

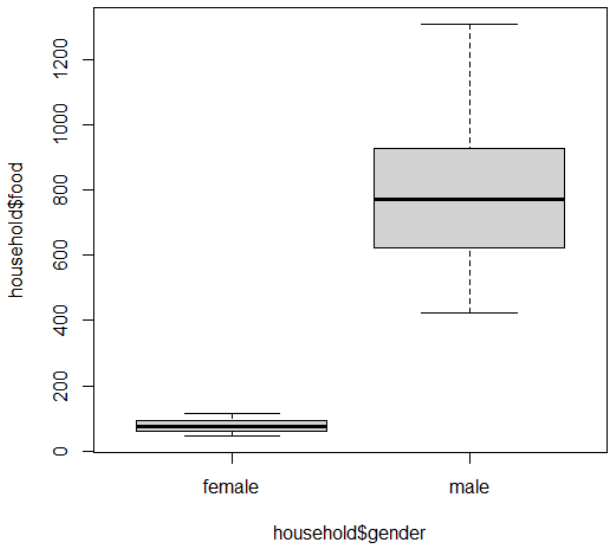
> plot(household$gender~household$housing) → Orden de factores altera el producto



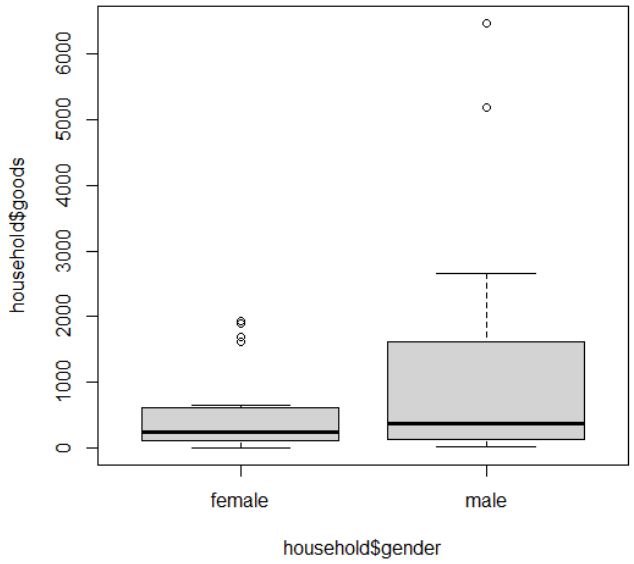
> plot(household$housing~household$gender)



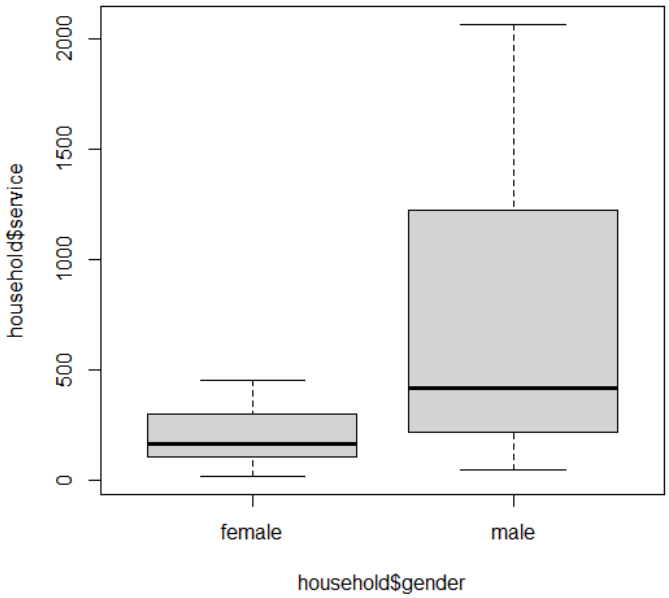
> plot(household$food~household$gender)

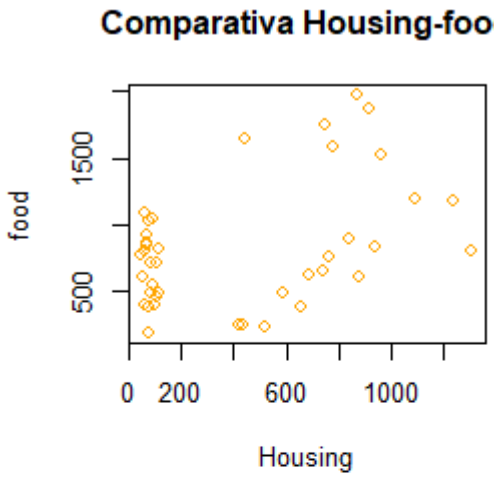


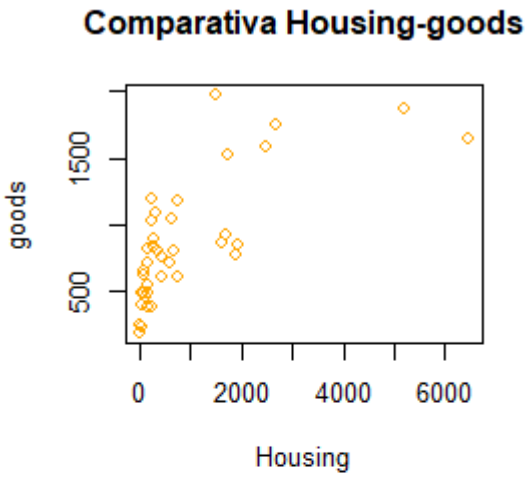
> plot(household$goods~household$gender)

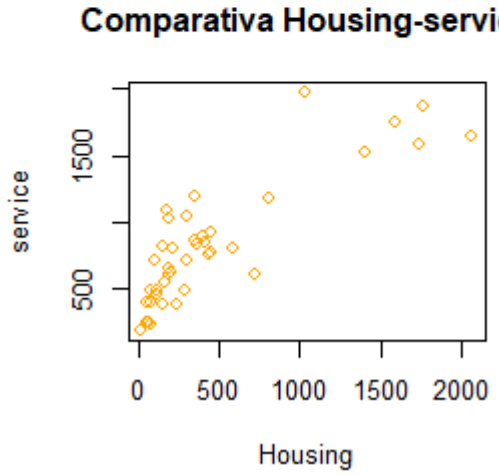


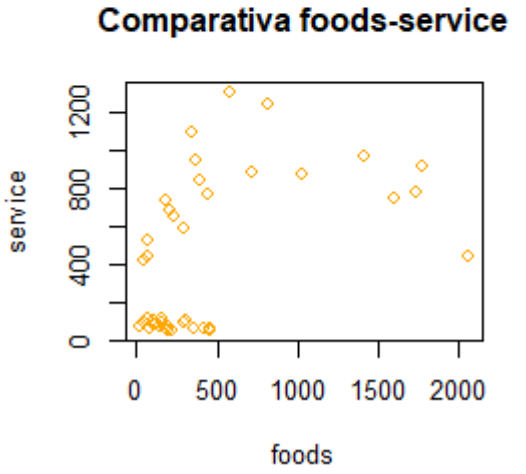
> plot(household$service~household$gender)

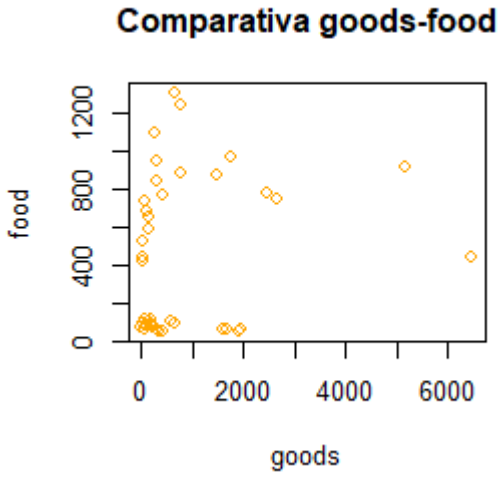


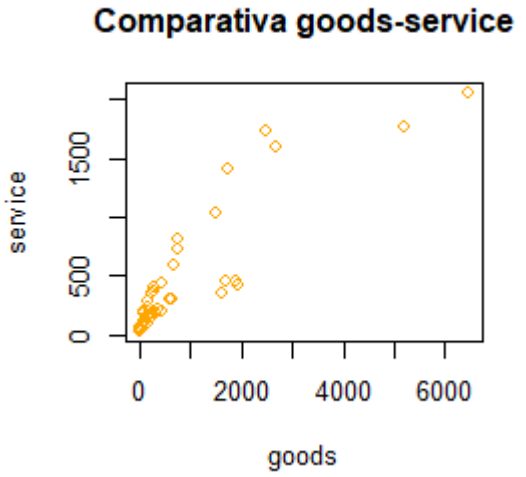

Comparativa Housing-food
food
500
1500
0
200
600
1000
Housing


Comparativa Housing-goods
goods
500
1500
0
2000
4000
6000
Housing


Comparativa Housing-service
service
500
1500
0
500
1000
1500
2000
Housing


Comparativa foods-service
service
0
400
800
1200
0
500
1000
1500
2000
foods


Comparativa goods-food
food
0
400
800
1200
0
2000
4000
6000
goods


Comparativa goods-service
service
0
500
1500
0
2000
4000
6000
goods