UNIDAD 3

GRAFICOS ESTADÍSTICOS

INTRODUCCIÓN

En concreto vamos a describir algunas de las funciones que incluye el núcleo de R para trabajar con gráficos. Las funciones gráficas de R suelen ser genéricas y tienen un gran número de opciones. Por lo tanto, aquí nos limitaremos a comentar sus usos más comunes. Si estás interesado en conocer de manera exhaustiva el funcionamiento de alguna función consulta su ayuda.

HISTOGRAMA

Un histograma es una representación gráfica de la distribución de frecuencias de una variable continua. Consiste en una sucesión de rectángulos levantados sobre un eje que representa los valores de la variable. Cada rectángulo tiene un área proporcional a la frecuencia de valores observada en el intervalo sobre el que se levanta. En esta sección aprenderemos a construir un histograma con R, a superponerle una distribución de probabilidad teórica y otra estimada no paramétricamente, a insertar títulos, etiquetas, etc.

EJERCICIO

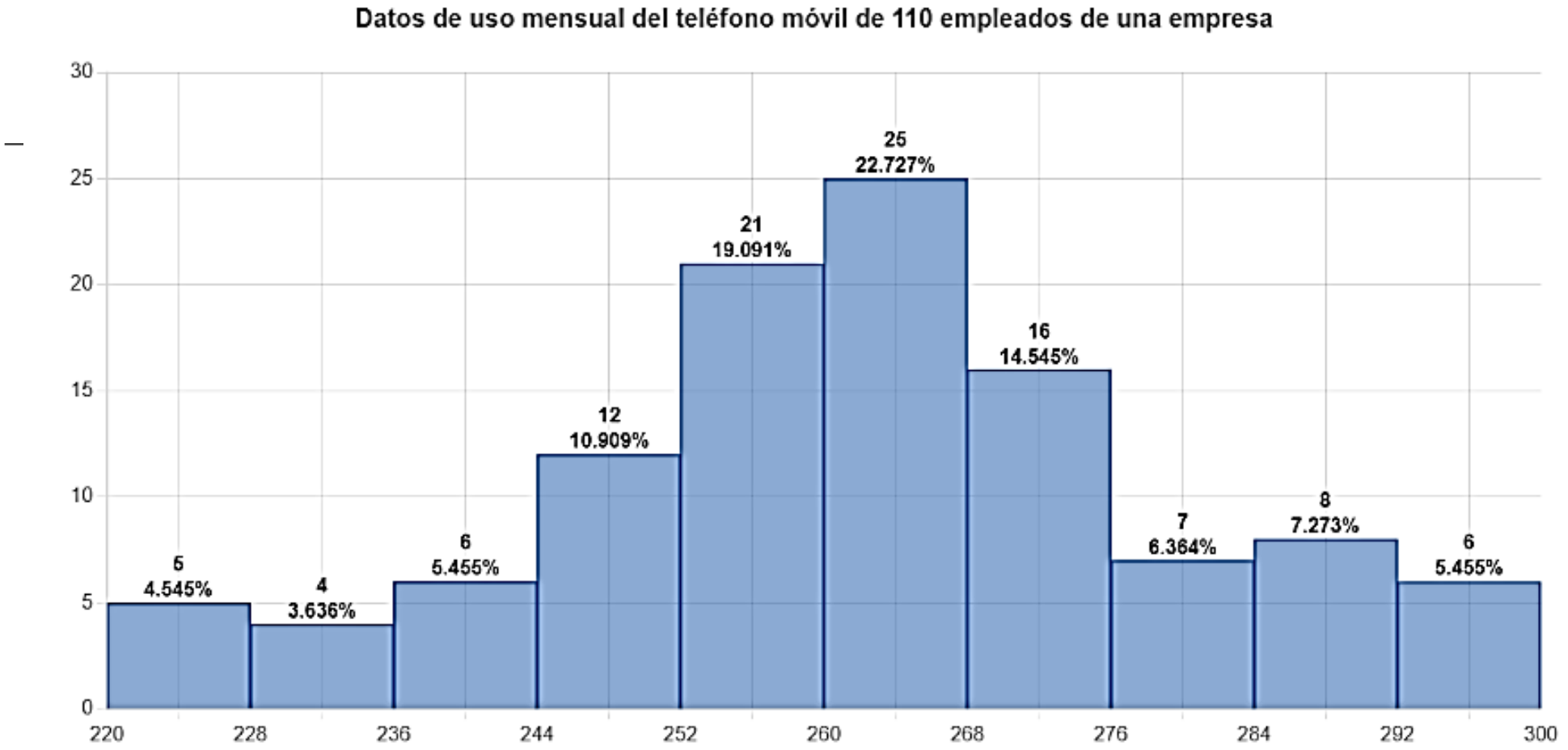
Jenni, directora de marketing de una importante compañía de telefonía móvil, obtuvo los registros de los minutos consumidos por una muestra aleatoria de 110 abonados al plan más barato de la empresa (250 minutos mensuales como máximo en hora punta). La siguiente lista contiene el detalle de los minutos consumidos por cada abonado de la muestra durante un mes. Los datos se encuentran en el fichero de datos Mobile Usage.

271 236 294 252 254 263 266 220 262 278 288
262 237 247 282 224 263 267 254 271 278 263
262 288 247 252 264 263 247 225 281 279 238
252 242 248 263 255 294 268 255 272 271 291
263 242 288 252 226 263 269 227 273 281 267
263 244 249 252 256 263 252 261 245 252 294
288 245 251 269 256 264 252 232 275 284 252
263 274 252 252 256 254 269 234 285 275 263
246 263 294 252 231 265 269 235 275 288 294
263 247 252 269 261 266 269 236 276 248 298

Grafique un histograma.

Elaboramos la tabla de Frecuencias

Clases	Punto Medio	Frecuencia Absoluta	Frecuencia Relativa	Frecuencia Abs. Acumulada	Frecuencia Rel. Acumulada
[220-228)	224	5	4.545%	5	4.55%
[228-236)	232	4	3.636%	9	8.18%
[236-244)	240	6	5.455%	15	13.64%
[244-252)	248	12	10.909%	27	24.55%
[252-260)	256	21	19.091%	48	43.64%
[260-268)	264	25	22.727%	73	66.36%
[268-276)	272	16	14.545%	89	80.91%
[276-284)	280	7	6.364%	96	87.27%
[284-292)	288	8	7.273%	104	94.55%
[292-300)	296	6	5.455%	110	100%


Datos de uso mensual del teléfono móvil de 110 empleados de una empresa
30
25
20
15
10
5
0
220
228
236
244
252
260
268
276
284
292
300
5
4.545%
4
3.636%
6
5.455%
12
10.909%
21
19.091%
25
22.727%
16
14.545%
7
6.364%
8
7.273%
6
5.455%

Histograma en R

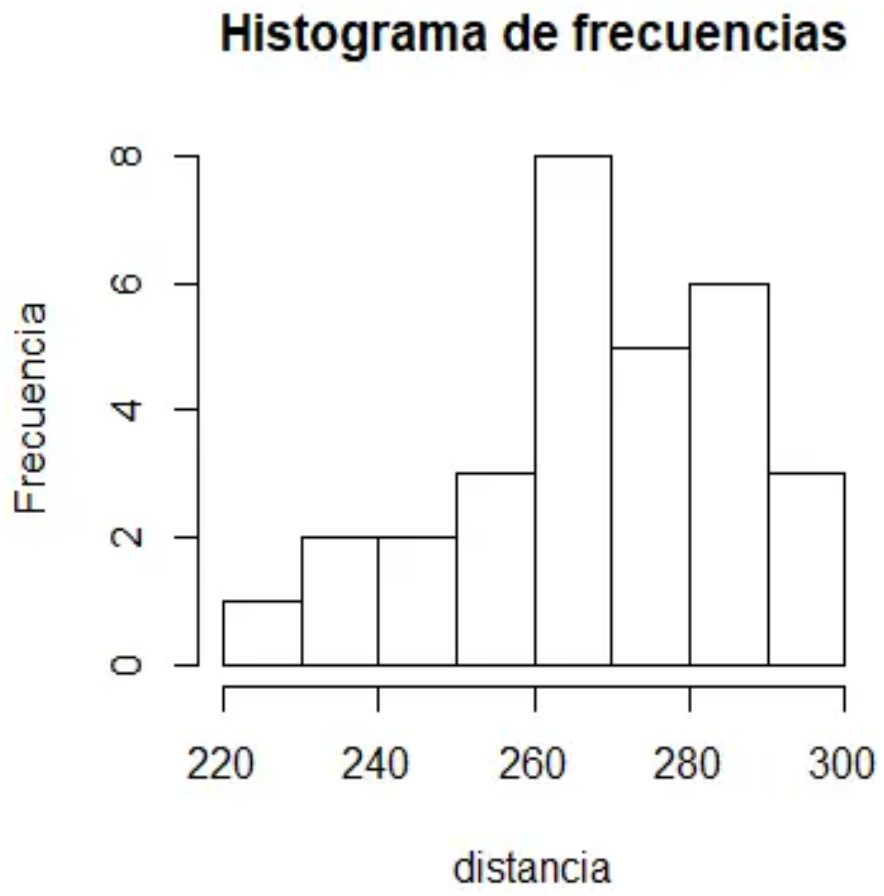
Para explicar los pasos para **crear un histograma en R**, vamos a utilizar los siguientes datos, que representan las distancia (en yardas) que recorre una pelota de golf después de ser golpeada.

```
distancia <- c(241.1, 284.4, 220.2, 272.4, 271.1, 268.3,
               291.6, 241.6, 286.1, 285.9, 259.6, 299.6,
               253.1, 239.6, 277.8, 263.8, 267.2, 272.6,
               283.4, 234.5, 260.4, 264.2, 295.1, 276.4,
               263.1, 251.4, 264.0, 269.2, 281.0, 283.2)
```

Puedes dibujar un histograma en R con la función `hist`. Por defecto, la función creará un **histograma de frecuencias**.

Histograma en R

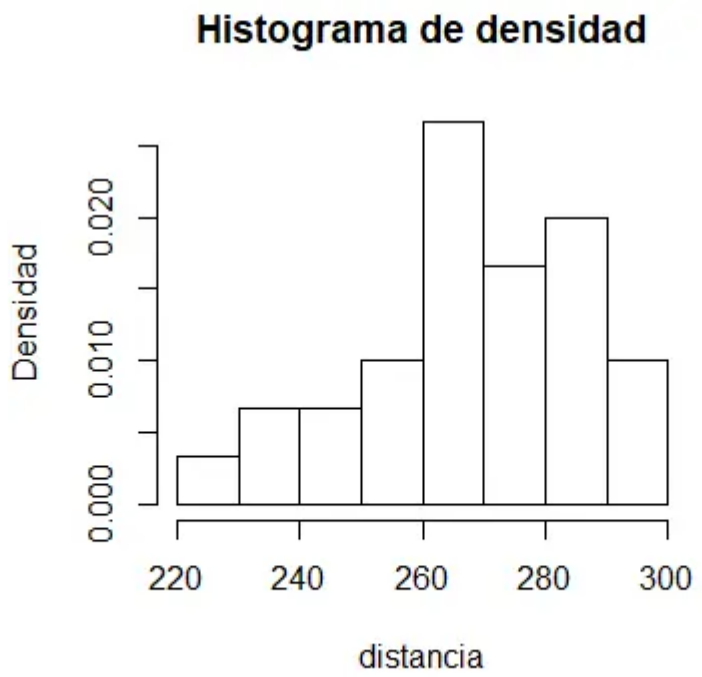
```
hist(distancia, main = "Histograma de frecuencias", # Frecuencia
     ylab = "Frecuencia")
```



Histograma en R

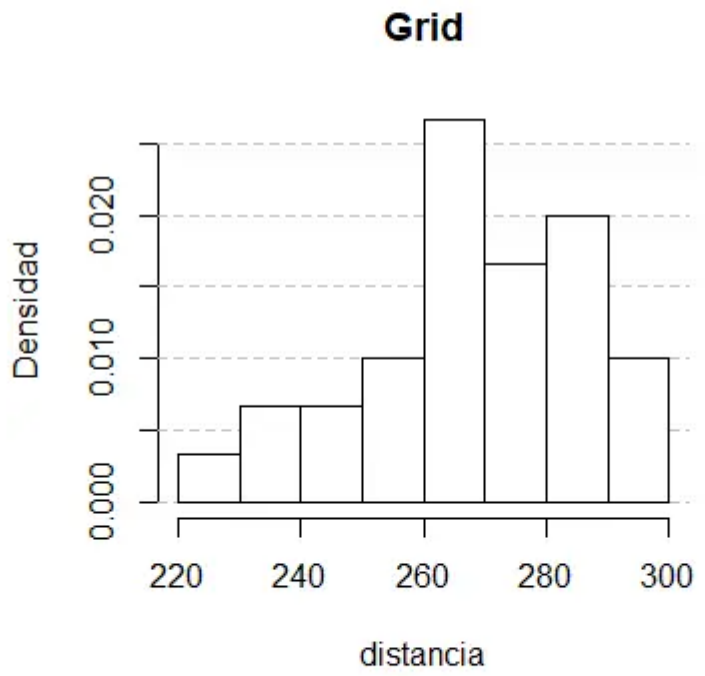
Sin embargo, si estableces el argumento `prob` como `TRUE`, obtendrás un **histograma de densidad**.

```
hist(distancia, prob = TRUE, main = "Histograma de densidad", # Densidad
     ylab = "Densidad")
```



Además, puedes añadir un grid al histograma con la función `grid` de la siguiente manera.

```
hist(distancia, prob = TRUE, ylab = "Densidad", main = "Grid")
grid(nx = NA, ny = NULL, lty = 2, col = "gray", lwd = 1)
hist(distancia, prob = TRUE, add = TRUE, col = "white")
```

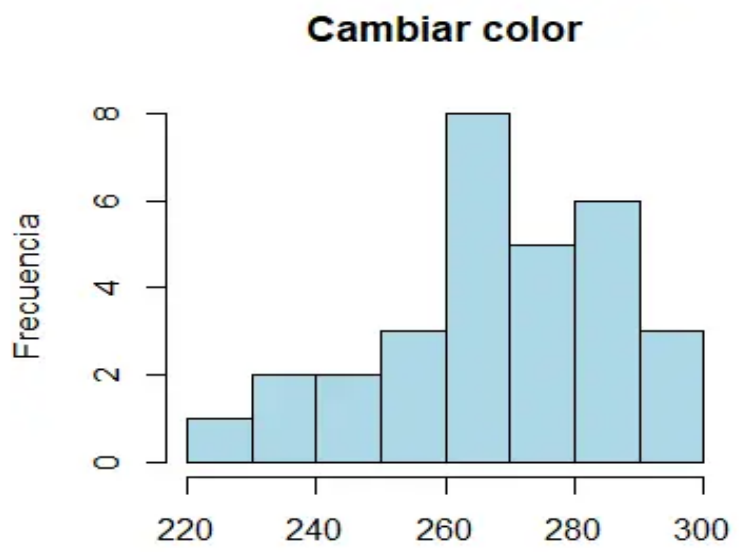


Ten en cuenta que hay que dibujar dos veces el histograma para que el grid salga por debajo de las barras en lugar de por encima.

Cambiar el color del histograma

Ahora que ya sabes crear un histograma en R, también puedes personalizarlo. Si quieres cambiar el color de las barras puede establecer el parámetro `col` al color que prefieras. Al igual que sucede con cualquier otro gráfico en R, puedes personalizar muchas características del gráfico, como el título, los ejes, el tamaño de fuente, la escala de los ejes, ...

```
hist(distancia, main = "Cambiar color", ylab = "Frecuencia", col = "lightblue")
```



El argumento breaks

Los histogramas son muy útiles para **representar la distribución subyacente de los datos** si el número de barras o clases se selecciona correctamente. Sin embargo, la selección del número de barras (o el ancho de las barras) puede ser complicada:

1. Pocas clases agruparán demasiado las observaciones.
2. Con demasiadas clases habrá pocas observaciones en cada una de ellas aumentando la variabilidad del gráfico obtenido.

Hay **varias reglas para determinar el número de barras**. En R, **el método de Sturges se usa por defecto**. Si quieres cambiar el número de barras, pasa al argumento `breaks` el número de clases que quieras.

```r
par(mfrow = c(1, 3))

hist(distancia, breaks = 2, main = "Pocas clases", ylab = "Frecuencia")
hist(distancia, breaks = 50, main = "Demasiadas clases", ylab = "Frecuencia")
hist(distancia, main = "Método de Sturges", ylab = "Frecuencia")

par(mfrow = c(1, 1))
```

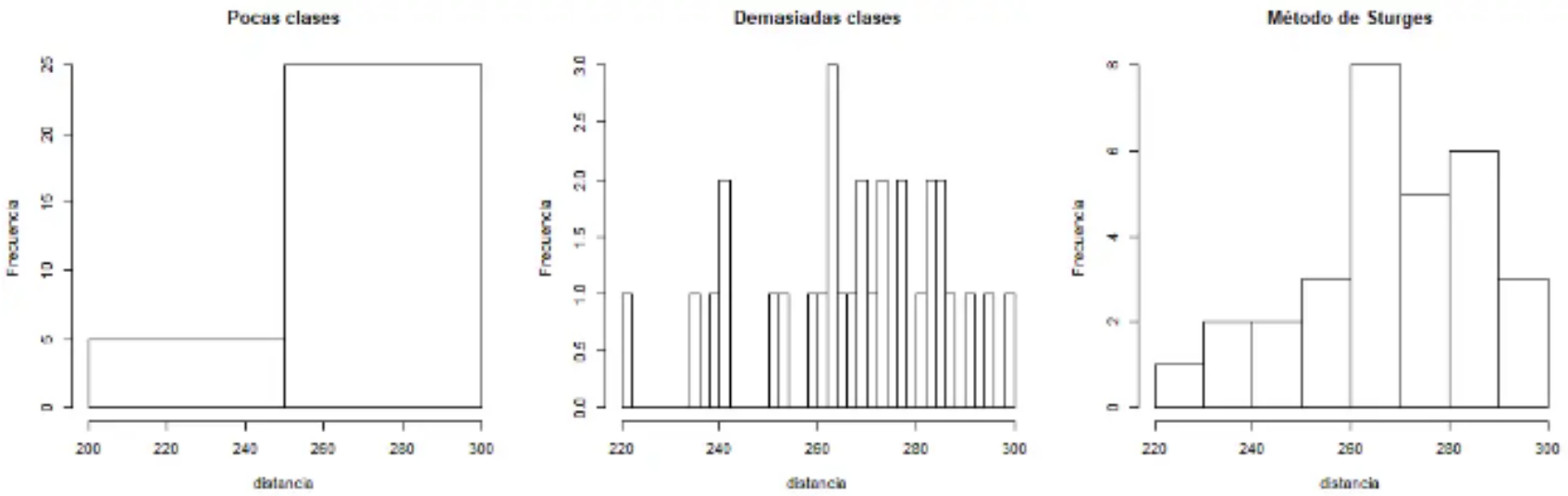


Tabla de frecuencias en R

Este proceso también se lo puede realizar mediante R, para lo cual se requiere conocer algunas funciones adicionales como se determina a continuación.

Min,max (varible): permite determinar el valor mínimo y máximo de un arreglo

Length(variable): determina la longitud de un arreglo.

n.class.Sturges(Variable):permite determinar el numero de intervalos o clases.

Cut():para dividir un vector de datos en intervalos.

function(x,y,b){}:permite crear una función con parámetros.

tabla(variable): permite determinar el numero de frecuencias de los valores

as.vector():transforma el valor en un vector.

cumsum(): suma los valores de una columna.

dataframe(): crea un data frame donde cada columna contiene la información específica de los valores agregados

Diagrama de Barras

Un **diagrama de barras** es un tipo de gráfico estadístico que se utiliza para representar gráficamente variables discretas.

En un diagrama de barras se representa en el eje X cada valor de la variable de estudio, y para cada uno se dibuja una barra rectangular con una altura proporcional a su frecuencia.

Un diagrama de barras puede ser vertical, si tiene las barras representadas verticalmente, u horizontal, cuando tiene las barras dibujadas horizontalmente.

En estadística, el diagrama de barras sirve para comparar la cantidad de veces que se repiten los datos. Principalmente, se usa para analizar la frecuencia de los datos en una muestra de una variable cualitativa.

Los diagramas de barras también se llaman **gráficos de barras** o **gráficos de columnas**.

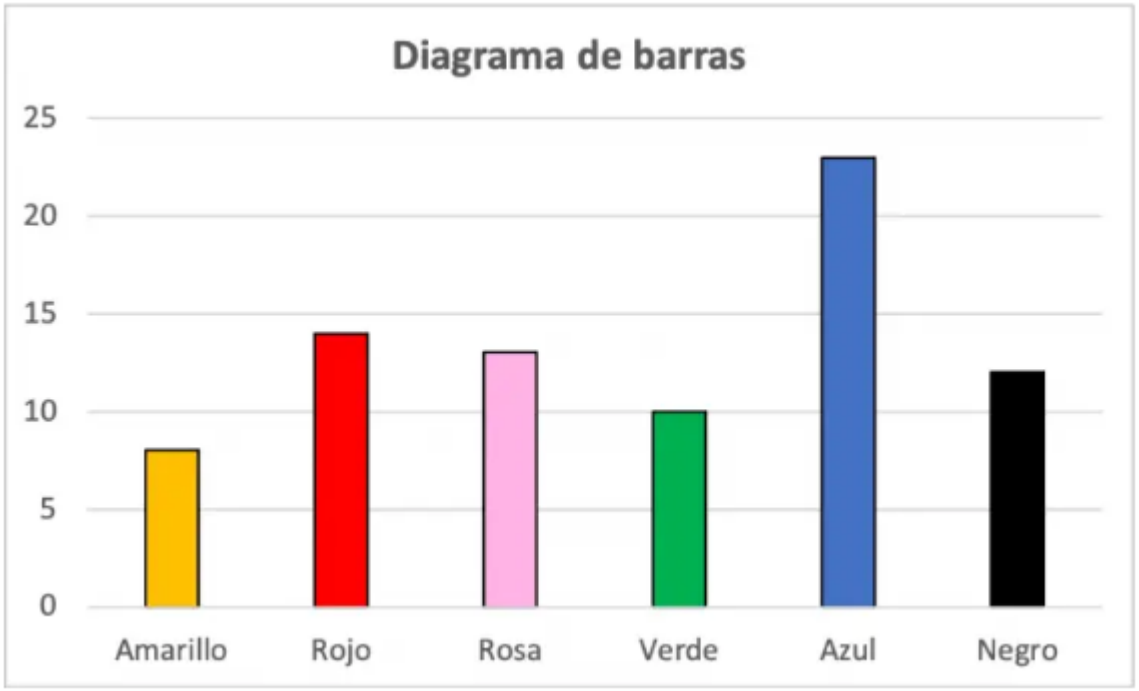


Diagrama de barras
25
20
15
10
5
0
Amarillo
Rojo
Rosa
Verde
Azul
Negro

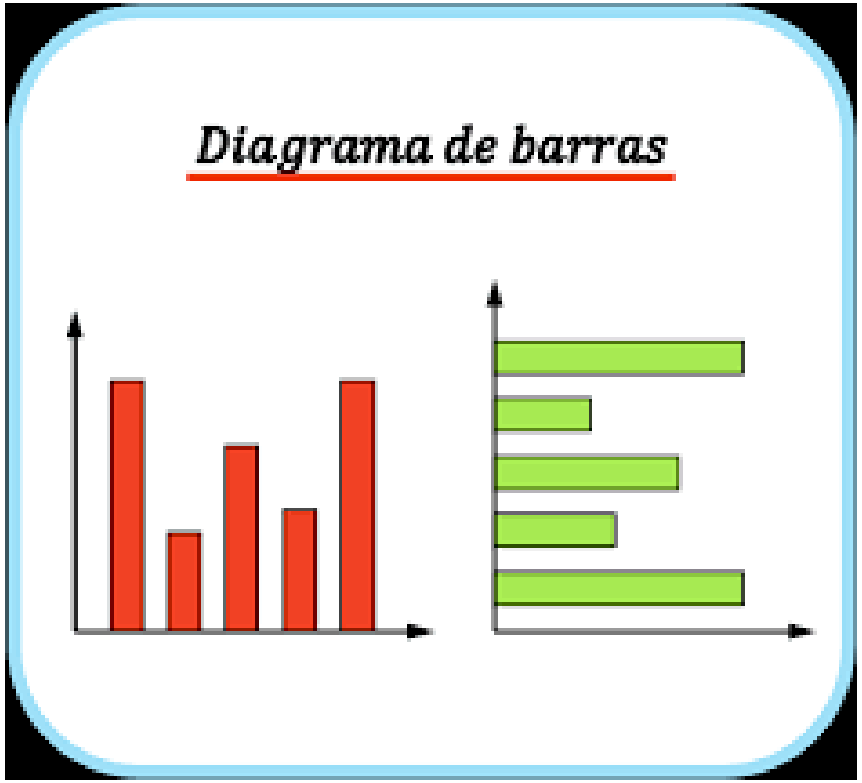


Diagrama de barras

Diagrama de barras en R

Este es quizás el tipo de gráfico mejor conocido de todos. Una gráfica de este tipo nos muestra la frecuencia con la que se han observado los datos de una variable discreta, con una barra para cada categoría de esta variable.

La función plot() puede generar gráficos de barra si damos como argumento x un vector de factor o cadena de texto, sin dar un argumento y

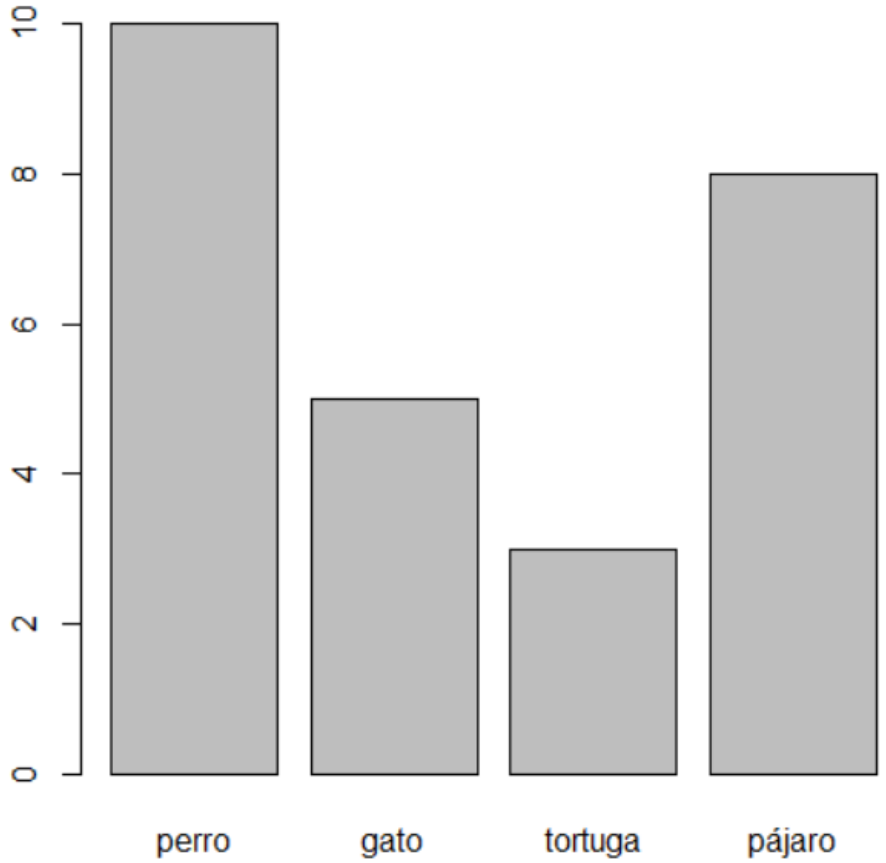
Diagrama de barras utilizando Barplot

```
# Vamos a crear dos variables para la representación
animales <- c('perro', 'gato','tortuga','pájaro')
numero <- c(10,5,3,8)
```

¿Cómo se usa la función BARPLOT de R?

Y ahora vamos a **utilizar la función barplot de R** para hacer la representación de esos datos.

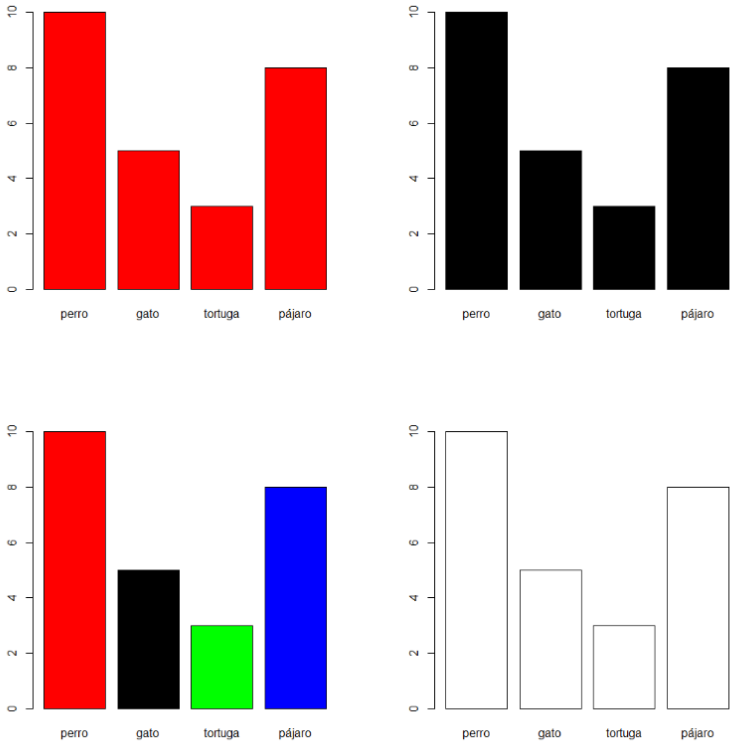
```
barplot(height=numero, names=animales)
```



Por supuesto, este tipo de gráficos también permite personalizarlo para cambiar su aspecto visual. Por ejemplo, podemos **cambiar el color de las barras, añadir nombre a los ejes o ponerle un título al gráfico**.

Cambiar color gráficos en R

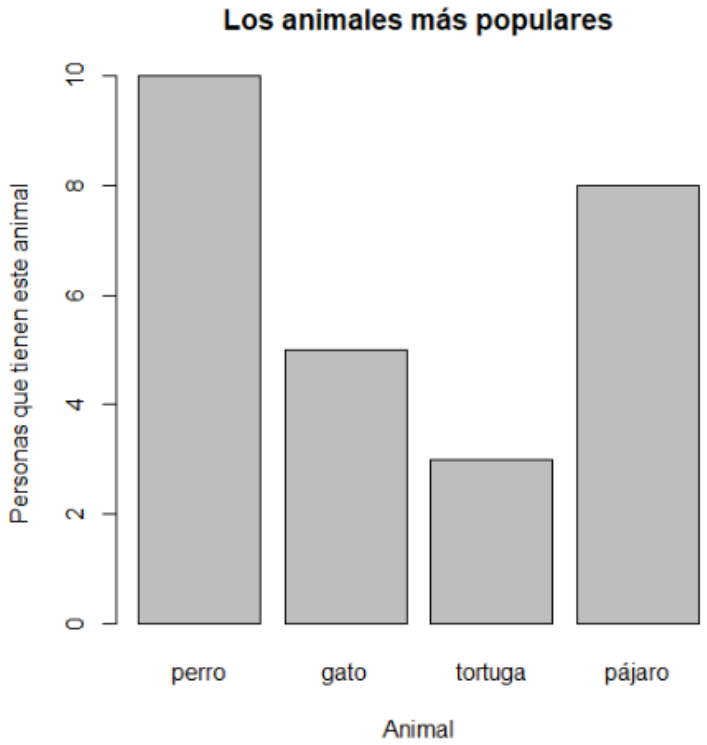
```
# Cambiar color de todas las barras
barplot(height=numero, names=animales, col='red')
barplot(height=numero, names=animales, col='black')

# Un color para cada barra
barplot(height=numero, names=animales,
        col=c('red','black','green','blue'))

# Cambiar color bordes
barplot(height=numero, names=animales,
        border='black', col='white')
```



Otras formas de personalizar gráficos en R

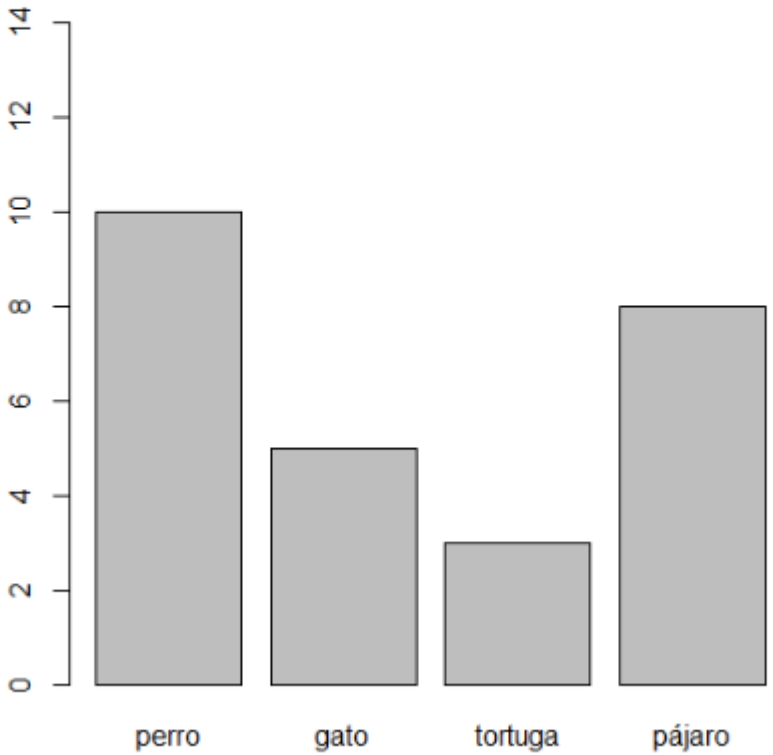
Como os decía antes, podemos personalizar el gráfico todo lo que queramos, **añadiendo un título al gráfico y cambiando el nombre de los ejes**:

```
barplot(height=numero, names=animales,
        xlab="Animal",
        ylab="Personas que tienen este animal",
        main="Los animales más populares")
```



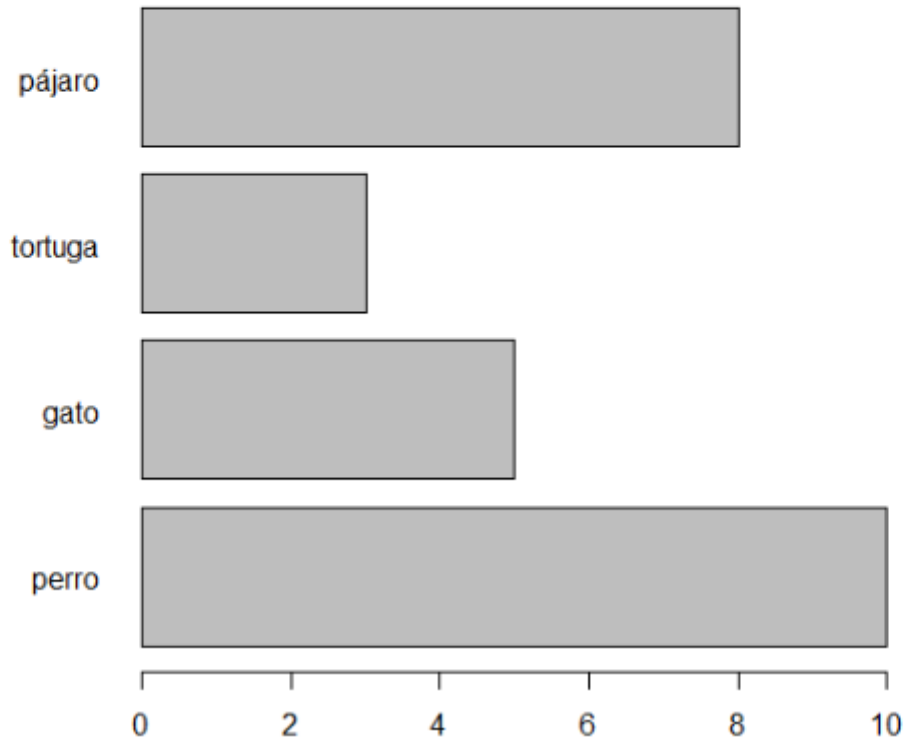
Otra opción que puede resultar muy útil para representar los datos en nuestro gráfico de barras es **cambiar los límites de los ejes de nuestro gráfico**:

```
barplot(height=numero, names=animales, ylim= c(0,14))
```



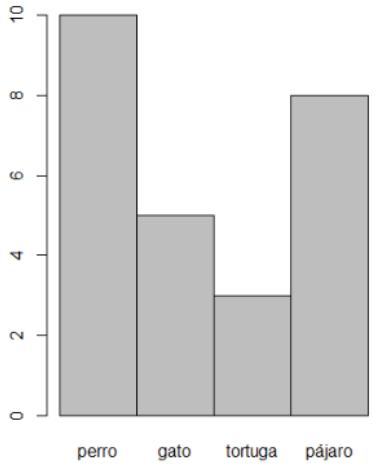
Todos los gráficos anteriores estaban en vertical, sin embargo, es muy sencillo poner un **gráfico barplot en horizontal**, solo tenemos que añadir "horiz=1" (para girar las barras) acompañado de "las=1" (para girar los nombres):

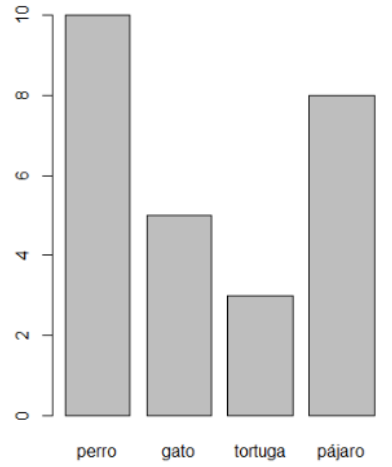
```
barplot(height=numero, names=animales, horiz=1, las=1)
```

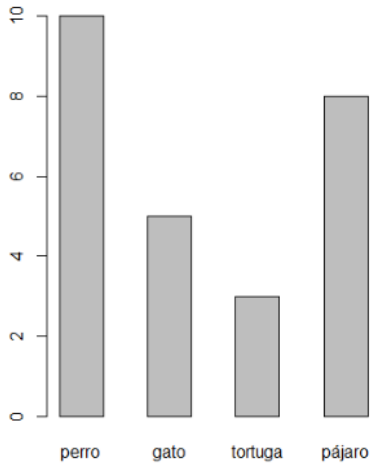


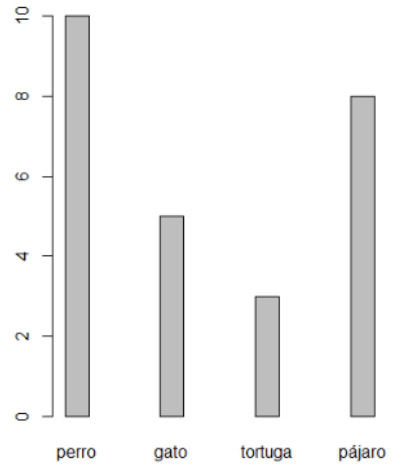
Por último, también podemos **modificar el ancho y la separación de las barras del gráfico en R**. Para **cambiar la separación entre las barras** utilizamos ***space***, seguido de la separación que queremos darle a las columnas:

```
barplot(height=numero, names=animales, space=0)
barplot(height=numero, names=animales, space=0.25)
barplot(height=numero, names=animales, space=1)
barplot(height=numero, names=animales, space=3)
```



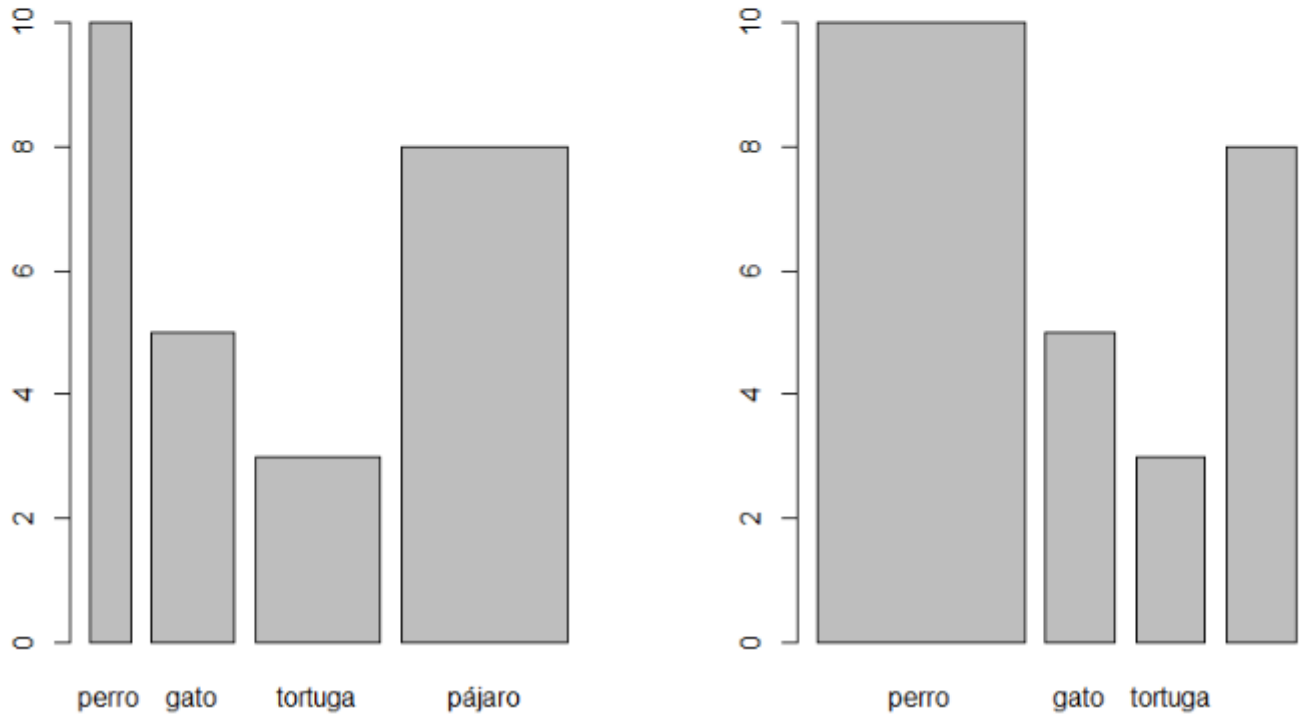






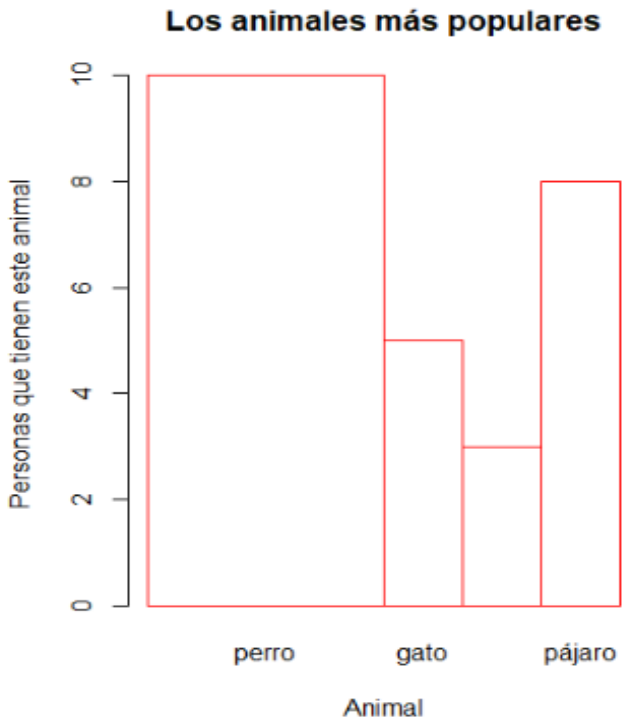
Y para cambiar el ancho de las barras, utilizamos la función **width**, aunque en este caso debes tener en cuenta que para que los anchos sean "visibles", deberán ser distintos entre cada uno de las barras:

```
barplot(height=numero, names=animales,
        width=c(0.1,0.2,0.3,0.4))

barplot(height=numero, names=animales,
        width=c(3,1,1,1))
```



Por supuesto, **podemos utilizar todo lo anterior combinado para conseguir un gráfico personalizado**:

```
barplot(height=numero, names=animales,
        border='red', col='white',
        xlab="Animal",
        ylab="Personas que tienen este animal",
        main="Los animales más populares",
        space=0,
        width=c(3,1,1,1))
```



https://estadisticamente.com/grafico-barras-columnas-barplot-r/

GRAFICO DE PASTEL

Gráfico circular en R

Un gráfico circular se puede crear con la función `pie` de R base. Aunque existen más paquetes para crear gráficos de sectores, como `ggplot2`, en este tutorial revisaremos cómo crear gráficos de tarta con la función `pie`.

Función pie( )

La función `pie` permite crear **gráficos de sectores** en R. Considera, por ejemplo, que quieres crear un gráfico circular de la siguiente variable, que representa el conteo de cierto evento:

```
#Vector de ejemplo
count <- c(12, 37, 6, 33, 3, 54)
pie(count)
```

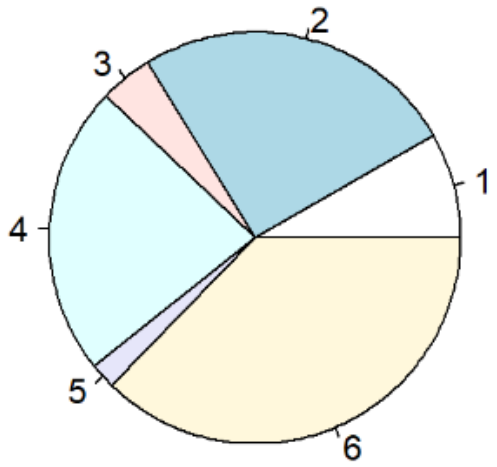
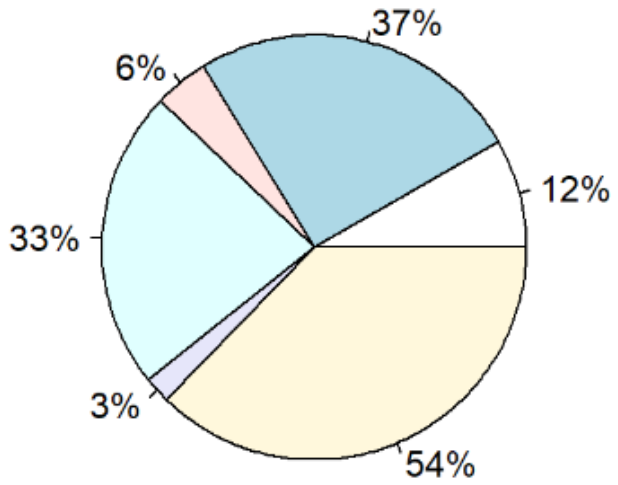


Gráfico circular con porcentajes

Si quieres mostrar *porcentajes* la **función** `pie` no permite crearlos automáticamente, entonces usaremos:

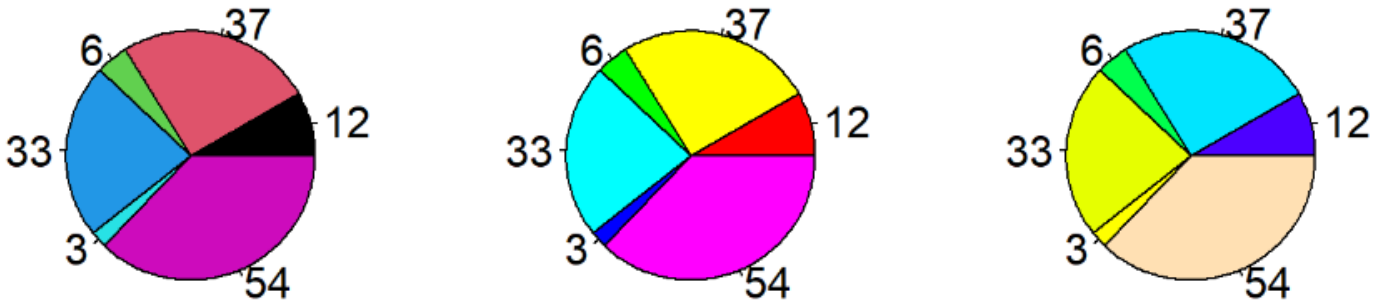
```
pie(count, labels = paste0(count, "%"))
```



Personalización

Además, puedes modificar el color del gráfico con el argumento `col`.

```
par(mfrow = c(1, 3))

pie(count, labels = count, col = 1:6, cex = 2)
pie(count, labels = count, col = rainbow(6), cex = 2)
pie(count, labels = count, col = topo.colors(6), cex = 2)
```



Personalización

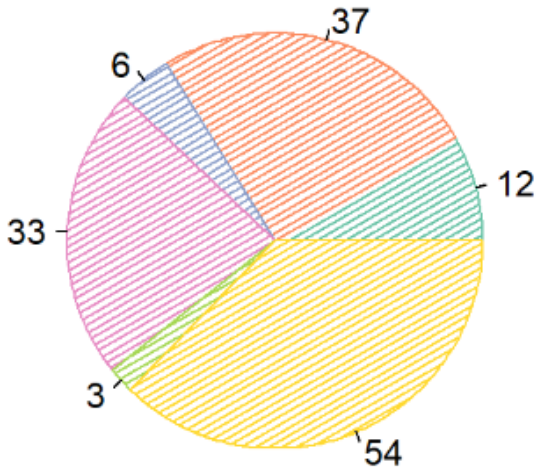
Si quieres modificar los bordes del gráfico, utiliza el argumento `lyt`.

```
library(RColorBrewer)
color<-brewer.pal(length(count),"Set2")
pie(count, labels = count, col = color, lty = 2)
```

Personalización

Si quieres agregar líneas de sombreado utiliza el argumento `density`, y el argumento `angle` para la dirección de las líneas.

```
pie(count, labels = count, col = color, density = 30, angle = 30)
```



https://rpubs.com/AlejandraAraujo/graficocircular

Diagramas 3D con plotrix

La función `pie3D( )` del paquete plotrix proporciona diagramas 3D:

```
library(plotrix)
proporciones <- c(10, 12,4, 16, 8) # creamos un vector con proporciones
etiquetas <- c("Uno", "Dos", "Tres", "Cuatro", "Cinco") # vector con etiquetas
pie3D(proporciones,labels=etiquetas,
      explode=0.1,
      main="Diagrama de tarta")
```