LABORATORIO DE ESTADÍSTICA Y PROBABILIDAD CON *R*

Jose Mari Eguzkitza Arrizabalaga

Profesor del Departamento de Matemática Aplicada
U.P.V./E.H.U

Pedidos: BBY Digital 2818 S.L.
Landatxueta kalea, 31
48180 Loiu
94 471 13 52
info@bbydigital.com

ISBN: 978-84-15956-10-5
Depósito Legal: GR 142-2014
Impresión Digital Gami

Marisari

ÍNDICE

INTRODUCCIÓN

La práctica educativa adquirida a lo largo del tiempo en la asignatura de Métodos Estadísticos de la Ingeniería ha puesto de manifiesto, y esto es preocupante, un cierto grado de desinterés hacia esta materia por parte de los estudiantes que aspiran a ser ingenieros, al no percibirla posiblemente como una disciplina esencial en su preparación para el posterior desarrollo profesional. Nada más lejos de la realidad, pues es indiscutible que la estadística es una herramienta trascendental en ese ámbito, ya que en muchas ocasiones el ingeniero ha de manejar gran cantidad de datos y moverse en un ambiente de incertidumbre.

En el contexto descrito, parece ineludible disponer de instrumentos que traten de estimular al alumno en el estudio de la estadística y la probabilidad. Impulsado por esta idea, el objetivo principal que persigue el presente trabajo es que quien acometa por primera vez el estudio de estas materias lo haga de una forma más entusiasta; el título del libro es toda una declaración de intenciones y se ajusta, fundamentalmente, a la materia que se aborda en la segunda parte. La primera parte es así mismo importante, pues constituye en sí misma un manual básico de introducción a una herramienta muy valiosa en el análisis de datos.

Las prácticas con ordenador se han convertido en los últimos años en una de las apuestas más sólidas de la innovación educativa en el área de matemáticas, y en concreto resultan de gran utilidad para facilitar el aprendizaje de la estadística, ya que posibilitan visualizar y resumir grandes conjuntos de datos y permiten experimentar y simular fenómenos aleatorios.

El software presentado aquí es *R,* un potente programa para hacer estadística, de libre distribución en internet. Se trata de un programa versátil, muy apropiado para llevar a cabo trabajos de experimentación. La fortaleza de *R* no radica exclusivamente en su capacidad para efectuar análisis de datos y análisis estadísticos, sino en su enorme potencial para experimentar; el programa es un auténtico laboratorio donde poder efectuar cualquier experiencia en estadística y probabilidad, dado que dispone de un lenguaje de programación muy sencillo.

Los tópicos que se abordan son los que habitualmente se desarrollan en un curso básico de Estadística y Probabilidad: estadística descriptiva, teoría de la probabilidad e inferencia estadística. Con objeto de poder interactuar sobre los temas que se desarrollan en el texto, se ha creado una página web: http://lab-est-prob-r.blogspot.com.es/, en la que estará disponible la fe de erratas, confeccionada con las aportaciones de los lectores. Se presentarán, además, los datos de la mayor parte de los ejercicios y los scripts correspondientes a las experiencias del capítulo 9. Habrá también ampliaciones de los temas con ulteriores experimentos, sugerencias, críticas, etc.

Esperamos que el presente texto contribuya a despertar el interés de los alumnos, desde la cooperación y el trabajo en equipo, por la estadística y el cálculo de probabilidades, propiciando un aprendizaje más activo y comprometido. Así mismo, deseamos que sirva para familiarizarse con una herramienta de uso muy extendido en el mundo de la investigación y en el mundo profesional.

Getxo, enero de 2014

1ª PARTE

ESTADÍSTICA Y PROBABILIDAD CON *R*

Capítulo 1

PRIMEROS PASOS EN *R*

1.1 ¿QUÉ ES *R*?

R es un programa de computación estadística de libre distribución en internet. Es suministrado con una licencia que permite su uso de forma absolutamente gratuita. *R*, además de ser un entorno para manipular datos, efectuar análisis estadísticos y producir gráficos, es un completo lenguaje de programación, lo que hace que sea un programa tremendamente flexible.

R es un clon del programa comercial *S-PLUS*, el cual está escrito en el lenguaje de programación estadística *S*, y del que *R* puede ser considerado como un "dialecto". Para obtener el programa hay que acceder a la página de internet http://cran.r-project.org/ y elegir el *mirror site* más próximo para descargarlo de la forma más rápida. En la misma página web se obtiene la información necesaria para instalar el programa.

Existe otra interesante alternativa que es la utilización online de *R* sin necesidad de instalar el programa en el propio ordenador. Para utilizar este servicio basta acceder a la página web http://pbil.univ-lyon1.fr/Rweb/. Por tanto, con un simple *smartphone* se puede acceder a un potente software y, como consecuencia de ello, a unas completas tablas estadísticas.

1.2 EMPEZANDO A TRABAJAR CON *R*

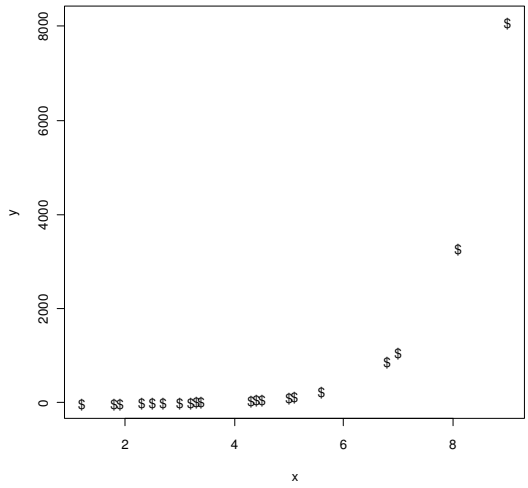
Para hacernos una idea sobre cómo funciona *R* veamos un ejemplo de una sesión. Al abrir el programa aparece lo siguiente:

```
R version 3.0.1 (2013-05-16) -- "Good Sport"
Copyright (C) 2013 The R Foundation for Statistical  Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)
R es un software libre y viene sin GARANTIA ALGUNA.
Usted puede redistribuirlo bajo ciertas circunstancias.
Escriba 'license()' o 'licence()' para detalles de distribución.
R es un proyecto colaborativo con muchos contribuyentes.
Escriba 'contributors()' para obtener más información y 'citation()' para saber
cómo citar R o paquetes de R en publicaciones.
Escriba 'demo()' para demostraciones, 'help()' para el sistema on-line de ayuda,
o 'help.start()' para abrir el sistema de ayuda HTML con su navegador.
Escriba 'q()' para salir de R.
[Previously saved workspace restored]
```

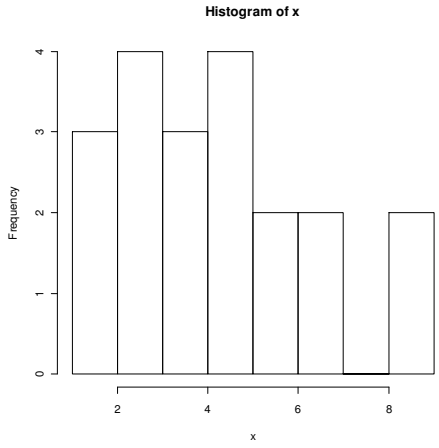
A continuación comenzamos propiamente con la sesión de *R*. Tras teclear el texto en negrita se debe pulsar **Enter**.

```
>x<-c(1.2,2.7,3.2,4.5,3.3,4.4,5.6,7,9,2.3,5,4.3,8.1,3.4,5.1,6.8,1.9,1.8,3,2.5)
#Construimos el vector x formado por 20 valores
> x
[1] 1.2 2.7 3.2 4.5 3.3 4.4 5.6 7.0 9.0 2.3 5.0 4.3
[13] 8.1 3.4 5.1 6.8 1.9 1.8 3.0 2.5
> length(x)
[1] 20

> y<-exp(x)
> y #Valores de una función exponencial
[1]    3.320117   14.879732   24.532530   90.017131   27.112639   81.450869
[7]  270.426407 1096.633158 8103.083928  9.974182  148.413159   73.699794
[13] 3294.468075   29.964100  164.021907  897.847292    6.685894    6.049647
[19]   20.085537   12.182494
```

> plot(x,y) #Gráfico de los pares (x,y,pch=36)



> hist(x) #Histograma de los valores x



En las líneas anteriores se observa que, después de introducir las instrucciones (en negrita) y tras pulsar **Enter**, el programa nos va devolviendo las salidas correspondientes a cada sentencia. Es de resaltar que la primera orden, al igual que ocurre con otras, no proporciona directamente ninguna salida; sencillamente almacena el valor de **x.** Si queremos visualizar ese valor debemos ejecutar lo siguiente:

> x

que nos devuelve los 20 valores de **x** que se habían almacenado en la sentencia anterior.

Para empezar a trabajar con el programa realizaremos una de las tareas más sencillas que pueden ser llevadas a cabo en *R;* se trata de la introducción de una operación matemática simple y la obtención del resultado devuelto por el programa. Por ejemplo,

```
> 5*8
[1] 40
> exp(1)
[1] 2.718282
> #Obtenemos el cociente de la división entera de 15 entre 2
> 15%/%2
[1] 7
> #Obtenemos el resto de la división entera de 15 entre 2
> 15%%2
[1] 1
```

Para asignar un valor a una variable se utilizan los símbolos **<-**. Para insertar comentarios basta anteponerles el símbolo #:

```
> #Definimos la variable x
> x<-5
> x
[1] 5
> x^3
[1] 125
```

Cuando falta algún símbolo para completar una sentencia aparece el símbolo+:

```
> sqrt(3
+
```

Si completamos ahora la expresión, en este caso con el cierre del paréntesis, el programa escribe el resultado:

```
> sqrt(3
+ )
[1] 1.732051
```

1.3 INTRODUCCIÓN DE DATOS

Para construir un vector se utiliza la sentencia **c()**:

```
> pesoenkg<-c(5,2.8,3.7,4.6,8.1,3.2)
> pesoenkg
[1] 5.0 2.8 3.7 4.6 8.1 3.2
```

```
> pesoengr<-pesoenkg*1000
> pesoengr
[1] 5000 2800 3700 4600 8100 3200
```

En lugar de utilizar la construcción anterior se puede emplear la sentencia **scan().** De este modo los datos se introducen de uno en uno mediante la tecla **Enter**. Para indicarle al programa que hemos terminado de introducir datos se debe pulsar dos veces esa tecla:

```
> pesoenkg<-scan()
1: 5
2: 2.8
3: 3.7
4: 4.6
5: 8.1
6: 3.2
7
Read 6 items
> pesoenkg
[1] 5.0 2.8 3.7 4.6 8.1 3.2
```

Para construir series de valores, por ejemplo los números impares comprendidos entre 1 y 65, utilizamos la función **seq()**. Mediante **rep()** es posible repetir un patrón dado, incluso caracteres:

```
> seq(1,65,2)
[1]  1  3  5  7  9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49
[26] 51 53 55 57 59 61 63 65

> rep(98,5)
[1] 98 98 98 98 98
> rep(c("sí","no"),3)
[1] "sí" "no" "sí" "no" "sí" "no"
```

1.4 LECTURA DE DATOS DE UN ARCHIVO

Si se desea cargar datos desde un archivo externo, por ejemplo si se quiere leer el archivo "Tensión de rotura.txt" que tenemos en la unidad C, debemos especificar con total exactitud la ruta y el nombre del archivo, todo ello entre comillas. Se debe poner el símbolo / en lugar de \:

```
> tensión<-read.table("C:/Tensión de rotura.txt")
> tension
Error: Object "tension" not found
```

```
> tensión #Aparece el mensaje anterior porque faltaba el acento
   V1
1  4.05
2  4.58
3  4.42
...
49 3.54
50 4.84
```

También es usual crear un archivo con una hoja de cálculo (*data frame* o marco de datos) y pasarlo a *R*. Por ejemplo, creamos el archivo "DF.txt" y lo guardamos como *Texto (delimitado por tabulaciones)*:

Costo.unit	Costo.mat	Costo.mano.de.obra
13.59	87	80
15.71	78	95
15.97	81	106
20.21	65	115
24.64	51	128

Los nombres de las variables no deben tener espacios, por eso se han colocado puntos, para que la sentencia **read.table()** funcione. Ahora leemos el archivo "DF.txt" y creamos el marco de datos **df**:

```
> df<-read.table("C:/Ej-R/DF.txt",header=T)
> df
  Costo.unit Costo.mat Costo.mano.de.obra
1      13.59        87                80
2      15.71        78                95
3      15.97        81               106
4      20.21        65               115
5      24.64        51               128
```

El argumento **header=T** significa que la primera línea del marco de datos contiene los nombres de las variables.

Una alternativa más cómoda y rápida para leer este archivo, o cualquier archivo de texto, es copiarlo en el portapapeles y después hacer **read.table("clipboard")** añadiendo, si es necesario, el argumento **header=T**.

Con **attach()** las variables son accesibles por su nombre en la sesión de *R* y con **names()** se obtiene una lista de ellas:

```
> attach(df)
> names(df)
[1] "Costo.unit"        "Costo.mat"         "Costo.mano.de.obra"
> Costo.mat
[1] 87 78 81 65 51
```

Otro modo de acceder a una variable, por ejemplo a "Costo.mat" del marco de datos **df**, consiste en seleccionar la correspondiente columna: **df$Costo.mat**. El símbolo **$** se utiliza, en general, para seleccionar elementos de un objeto.

Si queremos conocer de qué tipo es una variable concreta hacemos

```
> class(Costo.mat)
[1] "numeric"
> #Lo que significa que Costo.mat es un vector
```

Mediante la función **data.frame** podemos crear un marco de datos. Por ejemplo:

```
> num<-c(1,2,3,4,5,6,7)
> numcua<-num^2
> numcub<-num^3
> A<-data.frame(num,numcua,numcub)
> A
  num numcua numcub
1   1      1      1
2   2      4      8
3   3      9     27
4   4     16     64
5   5     25    125
6   6     36    216
7   7     49    343
```

1.5 SELECCIÓN DE ELEMENTOS DE UN OBJETO

Existe una función muy interesante en *R,* **str(),** que permite conocer cuál es la estructura de un determinado objeto que haya sido creado en una sesión. Por ejemplo, veamos qué nos dice esta función sobre el objeto **A:**

```
> str(A)
'data.frame':   7 obs. of  3 variables
$ num   : num  1 2 3 4 5 6 7
$ numcua: num  1 4 9 16 25 36 49
$ numcub: num  1 8 27 64 125 216 343
```

La salida anterior nos indica que el objeto **A** es un marco de datos formado por 7 observaciones de 3 variables, denominadas *num, numcua y numcub*. Si queremos escoger la primera de esas variables podemos hacer

```
> A$num
[1] 1 2 3 4 5 6 7
```

De todos modos, como las variables habían sido creadas previamente, ya eran accesibles. Seleccionamos el quinto elemento de la segunda variable:

```
> numcua
[1]  1  4  9 16 25 36 49
> numcua[5]
[1] 25
```

Si deseamos seleccionar, por ejemplo, los cuatro primeros registros del marco de datos **A** del apartado anterior, y generar de este modo un nuevo marco de datos, **A.nuevo**, debemos indicar las filas (1 a 4) y todas las columnas (espacio en blanco tras la coma):

```
> A.nuevo<-A[1:4,]
> A.nuevo
  num numcua numcub
1   1      1      1
2   2      4      8
3   3      9     27
4   4     16     64
```

Podemos así mismo efectuar selecciones condicionales, como por ejemplo:

```
> A
  num numcua numcub
1   1      1      1
2   2      4      8
3   3      9     27
4   4     16     64
5   5     25    125
6   6     36    216
7   7     49    343
> numcua[num<4]
[1] 1 4 9
> numcub[num==7]
[1] 343
```

Para ver qué valores cumplen una cierta condición y conocer los índices correspondientes podemos utilizar la función **which()**:

```
> which(numcua>10)
[1] 4 5 6 7
> which(numcub>0)
[1] 1 2 3 4 5 6 7
> which(numcub>10)
[1] 3 4 5 6 7
```

Para eliminar valores de una variable hacemos:

```
> numcua[-c(1,3,5,7)]
[1]  4 16 36
```

En ocasiones suele ser interesante saber qué elementos de un cierto vector **b** están en otro vector **a**:

```
> a<-1:10
> b<-c(1,3,7,15,20)
> a
 [1]  1  2  3  4  5  6  7  8  9 10
> b
[1]  1  3  7 15 20
> b[b%in%a]
[1] 1 3 7
```

1.6 SALIDA DE RESULTADOS

Cuando se desea guardar los resultados obtenidos en una sentencia de R se puede utilizar la función **write.table(),** indicando la ruta precisa del archivo de salida. Por ejemplo, si queremos guardar en la carpeta "Resultados" de la unidad C, con el nombre "resultadosDF", el marco de datos **df** obtenido anteriormente escribiremos:

```
> write.table(df,file="C:/Resultados/resultadosDF", row.names=F,col.names=F)
```

Se podría decir que **write.table()** es la sentencia inversa de **read.table()**. Los dos últimos argumentos sirven para que no aparezca ningún indicador de fila ni de columna.

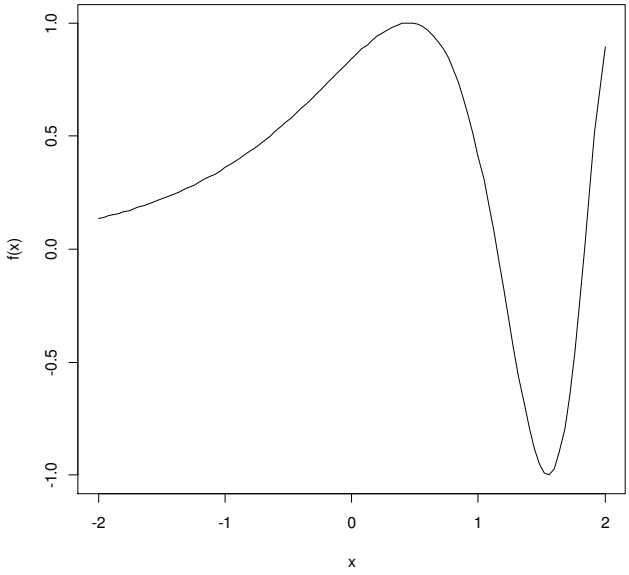
1.7 FUNCIONES

Para definir nuevas funciones en *R*, como por ejemplo f(x) = $2x^2+1$, se hace:

```
> f<-function(x) 2*x^2+1
> f(-5)
[1] 51
```

Ejemplo 1-1 Dada la función f(x) = sen(e^x), se pide: 1º) Obtener f(-1), f(0) y f(1). 2º) Representación gráfica en el intervalo (-2,2). 3º) Calcular el área bajo la curva entre las abscisas x=0 y x=1.

```
> f<-function(x) sin(exp(x))
> f(-1);f(0);f(1)
[1] 0.3596376
[1] 0.841471
[1] 0.4107813

> plot(f,-2,2);
```




```
> integrate(f,0,1)
0.8749572 with absolute error < 9.7e-15
```

También se pueden definir funciones de dos o más variables (argumentos) y aprovechar la forma en que trabaja *R,* con vectores, para realizar todas las operaciones mediante una única evaluación. Supongamos, por ejemplo, que queremos obtener el valor de la función

$z = \frac{x^2 + y^2}{x + y}$ en 8 puntos (a,b). Podemos comenzar creando los vectores que contienen los valores de interés:

```
> a<-1:8
> a
[1] 1 2 3 4 5 6 7 8
> b<-rep(5,8)
> b
[1] 5 5 5 5 5 5 5 5
```

Definimos ahora la función z, la cual será evaluada en los puntos de coordenadas (a,b):

```
> z<-function(x,y) (x^2+y^2)/(x+y)
> z(a,b)
[1] 4.333333 4.142857 4.250000 4.555556 5.000000 5.545455 6.166667 6.846154
```

En el siguiente ejemplo se detalla cómo definir y almacenar una función construida por el propio usuario para que esté disponible en ulteriores usos.

Ejemplo 1-2 Crear y guardar una función, denominada **tresmedias()**, que calcule las medias aritmética, geométrica y armónica de un conjunto de valores almacenados en un vector:

```
> tresmedias<-function(x)
{
+ ma<-mean(x)
+ mg<-exp(mean(log(x)))
+ mh<-1/mean(1/x)
+ cat("Media aritmética:",ma,"\n")
+ cat("Media geométrica:",mg,"\n")
+ cat("Media armónica:",mh,"\n")
}
> x<-c(1,2,3,4,5)
> tresmedias(x)
Media aritmética: 3
Media geométrica: 2.605171
Media armónica: 2.189781
```

La función **cat()** sirve para que se impriman los resultados y el argumento **"\n"** obliga a que cada uno de ellos aparezca en una línea diferente. Para utilizar en sesiones posteriores la función generada podemos pegar el código en un editor de texto y guardarlo. Cuando queramos usarla de nuevo haremos: **Archivo → Interpretar código fuente R...** y así cargaremos la función.

1.8 SCRIPTS

Un script es una porción pequeña de código, en este caso de código de R. Suele ser muy común que determinadas operaciones efectuadas con R se deban repetir en un futuro con datos diferentes. Entonces, lo adecuado es generar un script y almacenarlo para utilizarlo posteriormente cambiando los datos de forma conveniente. Por ejemplo, supongamos que hemos llevado a cabo la sesión siguiente:

```
> x<-c(1,2,3,4,5,6,7,8)
> y<-x^2
> z<-x^3
> sum(x);sum(y);sum(z)
[1] 36
[1] 204
[1] 1296
> history(Inf)
```

La salida de la última instrucción genera el siguiente script:

```
x<-c(1,2,3,4,5,6,7,8)
y<-x^2
z<-x^3
sum(x);sum(y);sum(z)
```

Copiamos el código anterior, vamos a **Archivo → Nuevo script**, lo pegamos y lo guardamos haciendo **Archivo → Guardar como…**, por ejemplo con el nombre "cuadradoscubos.R" (debemos escribir expresamente la extensión R).

Supongamos, ahora, que debemos ejecutar las mismas órdenes del script anterior pero cambiando el vector x. Entonces vamos a **Archivo → Abrir script**, y para facilitar la visualización de las ventanas hacemos, por ejemplo, **Ventanas → Divida Verticalmente.** A continuación, introducimos los nuevos valores de x quedándonos, por ejemplo, el siguiente script:

```
x<-c(11,12,13,14,15,16,17,18)
y<-x^2
z<-x^3
sum(x);sum(y);sum(z)
```

Ahora hacemos **Editar → Ejecutar todo** (también podríamos ejecutar línea a línea) y se obtiene lo siguiente:

```
> x<-c(11,12,13,14,15,16,17,18)
> y<-x^2
> z<-x^3
```

> sum(x);sum(y);sum(z)
[1] 116
[1] 1724
[1] 26216

El manejo de scripts que se ha acaba de describir es una de las formas más habituales de trabajar con R.

1.9 PAQUETES

Algunos paquetes (*packages*) para aplicaciones especiales son parte de la instalación básica de *R*; otros pueden ser obtenidos en la misma página web de donde se descarga el programa.

Para ver qué paquetes están instalados hacemos

> library()

Si el paquete no está en la instalación normal de *R* podemos buscarlo haciendo **Paquetes → Instalar paquete(s)…**, lo que nos lleva a elegir un *mirror* cercano (España, Francia,…) y finalmente elegir el paquete requerido; una vez instalado ha de ser cargado. Por ejemplo, para cargar el paquete **datasets**, hacemos

> library(datasets)

o bien en el menú usamos la herramienta **Paquetes.**

Para ver el contenido de **datasets** hacemos

> library(help=datasets)

Si queremos conocer con detalle, por ejemplo, uno de los objetos pertenecientes al paquete **datasets**, como es **Orange**, hacemos:

> **?Orange**
> Orange #Ahora veríamos el conjunto de datos de este marco de datos

1.10 *R* FRENTE A LAS CALCULADORAS CONVENCIONALES

Una de las aplicaciones más simples de *R*, pero muy práctica, es su uso como calculadora, pues ofrece una ventaja considerable respecto de las calculadoras convencionales.

Supongamos que se dispone de una serie de valores de una variable x_i, y sus correspondientes frecuencias absolutas F_i (número de veces que aparece cada uno de los valores). Para evaluar las fórmulas:

$$f_i = \frac{F_i}{\sum_i F_i};\ \sum_i x_i F_i;\ \sum_i x_i f_i;\ \sum_i x_i^2 f_i$$

podríamos actuar del siguiente modo:

```
> x<-c(20.5,12.6,-23,-6.98,24,32.8,7,-8.6)
> F<-c(3,4,2,6,5,7,4,9)
> sum(F)
[1] 40
> f<-F/sum(F)
> f
[1] 0.075 0.100 0.050 0.150 0.125 0.175 0.100 0.225
> sum(x*F)
[1] 324.22
> sum(x*f)
[1] 8.1055
> sum(x^2*f)
[1] 362.9658
```

Para comprobar que estos cálculos se llevan a cabo mucho más cómodamente en *R* que en una calculadora, pónganse en práctica ambos procedimientos y compárense los tiempos de ejecución.

1.11 CÓMO GESTIONAR UNA SESIÓN DE *R*

Al iniciar una sesión de *R* hay un directorio de trabajo donde el programa busca, por defecto, cualquier archivo que sea solicitado y donde coloca los archivos que se crean durante la sesión. Es conveniente utilizar distintos directorios para distintos proyectos. Para ver en qué directorio se está trabajando se hace

```
> getwd()
[1] "C:/Archivos de programa/R/R-2.3.1"
```

Si queremos cambiar de directorio elegimos en el menú: **Archivo → Cambiar dir…**

Todos los *objetos* (variables) creados en *R* se almacenan en un *área de trabajo* o *workspace* (puede haber varios), que es un espacio de trabajo común. Para ver qué objetos se han definido elegimos en el menú: **Misc → Listar objetos** y para eliminarlos todos: **Misc → Remover todos los objetos.** Si solo se desea eliminar, por ejemplo, los objetos **pesoenkg** y **pesoengr** hacemos

```
> rm(pesoenkg,pesoengr)
```

Estas acciones y otras pueden ser llevadas a cabo también con los botones del menú, igual que hemos hecho para ver o eliminar los objetos del área de trabajo.

Es posible guardar el *workspace* en un archivo en cualquier momento haciendo**: Archivo → Guardar área de trabajo.**

El *workspace* está formado solo por objetos y no por ninguna entrada obtenida durante la sesión. Si se desean guardar todas las entradas y todas las salidas hay que utilizar: **Archivo → Guardar en archivo,** o bien **Copiar y Pegar**.

La historia de los comandos introducidos en una sesión puede ser cargada y guardada mediante: **Archivo → Cargar/Guardar Histórico.**

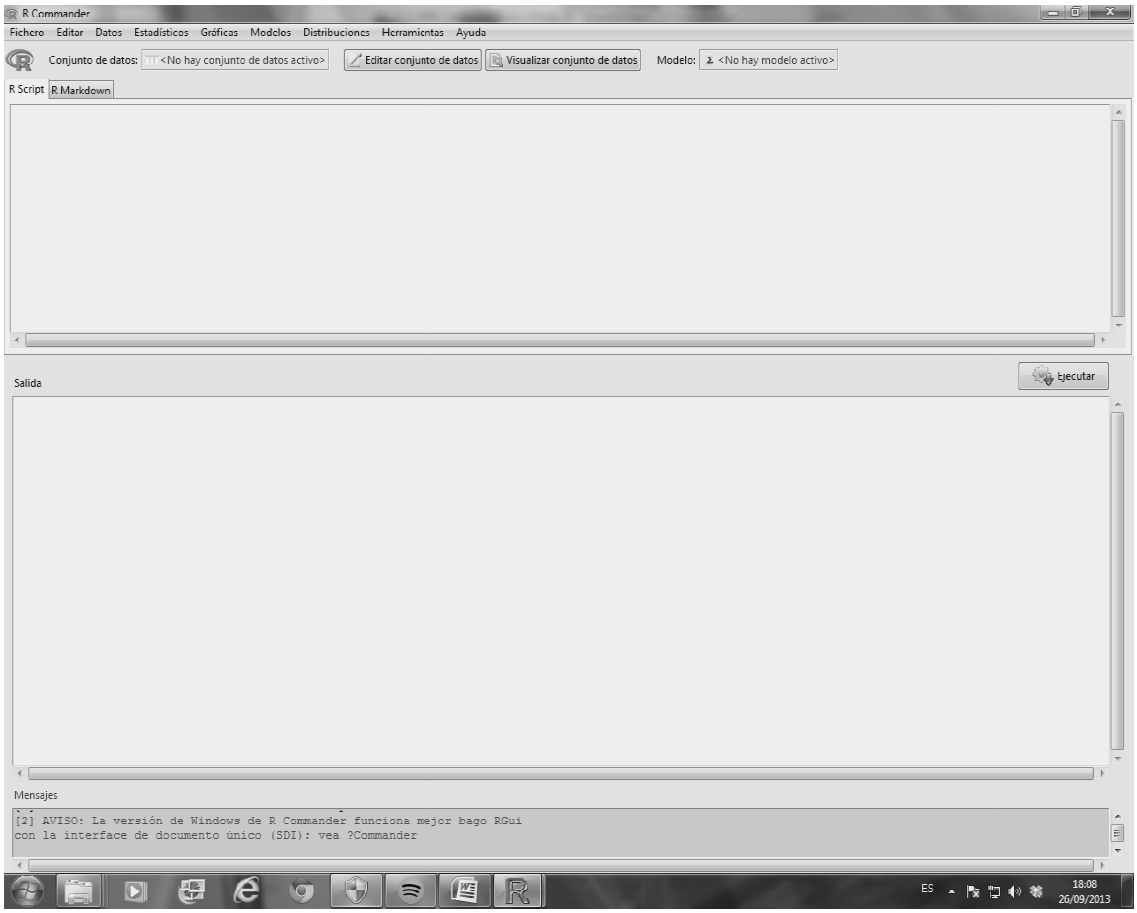
Para ver todos los comandos introducidos durante una sesión hacemos:

```
> history(Inf)
```

1.12 R COMMANDER

Uno de los paquetes más relevantes que pueden ser utilizados en *R* es **Rcmdr.** Al cargar este paquete (después de instalarlo junto con los paquetes complementarios exigidos) lo que hacemos es generar una interfaz visual denominada *R Commander.* Si se cierra esta interfaz, sin cerrar *R*, para volver a cargarla se debe ejecutar la instrucción **Commander().**

El aspecto de *R Commander* es el siguiente:



Simplificando, la forma de utilizar esta herramienta consiste en lo siguiente: en la ventana superior se introducen las instrucciones, se seleccionan, se ejecutan y los resultados se visualizan en la ventana inferior.

Para hacernos una somera idea de las posibilidades que nos brinda esta herramienta, podríamos hacer lo siguiente: **Datos → Nuevo conjunto de datos…** Introducimos, por ejemplo, 10 datos de la variable altura (en cm): 177, 178, 185, 156, 190, 177, 155, 188, 175, 166 y la nueva variable nos aparece, en forma de botón, en la barra de herramientas de *R Commander*. A continuación vamos, por ejemplo, a **Estadísticos → Resúmenes → Conjunto de datos activo** y obtenemos un resumen de seis parámetros importantes, que serán comentados en los capítulos siguientes. De la misma forma, si hacemos: **Gráficas → Histograma**, obtendremos un histograma de la variable, etc.

Esta interfaz, en definitiva, permite trabajar con *R* de una forma más visual, que será la preferida para algunos, frente a la utilización de la consola clásica de *R.*

1.13 OBSERVACIONES IMPORTANTES

- La tecla ⇧ sirve para copiar la última sentencia editada.
- En el menú **Ayuda** se puede conseguir todo tipo de asistencia, incluyendo manuales completos de *R*. Concretamente, haciendo clic en **Ayuda → Funciones R (texto)** se obtiene ayuda sobre las funciones de *R*. Para tratar de resolver una duda concreta hacer: **Ayuda → FAQ en R.**
- Para insertar un gráfico de *R* en un documento de texto copiamos el gráfico como *metafile* y luego lo pegamos. También podemos utilizar las funciones **jpeg()** y **png()** para guardar gráficos.
- Para cualquier búsqueda sobre *R* ir a http://www.rseek.org/.
- En la página web http://cran.r-project.org/doc/contrib/Short-refcard.pdf se puede disponer de una guía muy práctica con las principales funciones de *R*.

1.14 EJERCICIOS RESUELTOS

●**ER 1-1** Considérese la siguiente tabla de valores:

hora	1	3	5	7	9	11	13	15	17	19
nivel	-20,5	23,2	-88	-24,5	22	21	-57	34,8	33	-21,9

Se pide: 1º) Listar todos los objetos que estén en el *workspace* de *R* y eliminarlos. 2º) Con los datos de la tabla construir, mediante un editor de texto, el archivo "entrada.txt" y guardarlo en el disco duro C. 3º) Leer el archivo anterior y generar un marco de datos, denominado "datos". 4º) Obtener el cuadrado de cada uno de los valores de las dos variables y guardar los resultados construyendo el archivo "salida.txt".

```
> ls()
[1] "ref0" "t"    "u"    "x"    "y"
> #La salida anterior depende de lo que haya sido introducido previamente
> rm(list=ls(all=TRUE)) #Eliminamos todos los objetos actuales
> ls()
character(0)

> datos<-read.table ("C:/entrada.txt",header=T)
> datos
  hora nivel
1    1 -20.5
2    3  23.2
3    5 -88.0
4    7 -24.5
5    9  22.0
6   11  21.0
```

```
7    13 -57.0
8    15  34.8
9    17  33.0
10   19 -21.9
```

> hora;nivel

```
Error: objeto "hora" no encontrado
```

> #El mensaje anterior nos indica que las variables no son accesibles por su nombre; para que lo sean debemos usar la función attach()

> attach(datos)
> hora;nivel

```
 [1]  1  3  5  7  9 11 13 15 17 19
 [1] -20.5  23.2 -88.0 -24.5  22.0  21.0 -57.0  34.8  33.0 -21.9
```

> horaalcuadrado<-hora^2
> horaalcuadrado

```
 [1]   1   9  25  49  81 121 169 225 289 361
```

> nivelalcuadrado<-nivel^2
> nivelalcuadrado

```
 [1]  420.25  538.24 7744.00  600.25  484.00  441.00 3249.00 1211.04 1089.00
[10]  479.61
```

> #Por último creamos un archivo de texto denominado salida.txt en el que se guardan los valores anteriores. Previamente debemos construir, mediante la función data.frame(), el correspondiente marco de datos

> #Como alternativa podría haberse usado la función cbind() (véase en ayuda de *R* información sobre esta función)

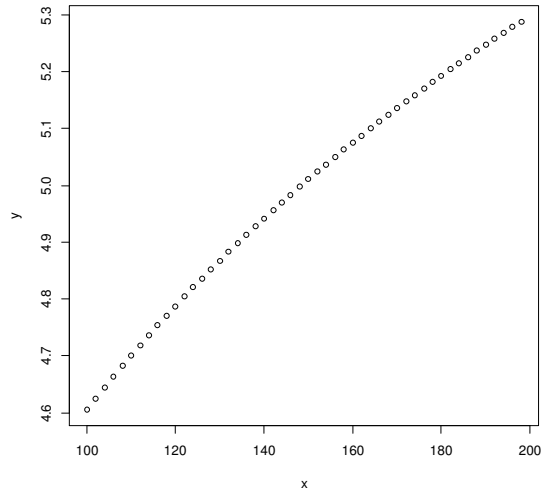
> s<-data.frame(horaalcuadrado,nivelalcuadrado)
> s

```
   horaalcuadrado nivelalcuadrado
1               1         420.25
2               9         538.24
3              25        7744.00
....
9             289        1089.00
10            361         479.61
```

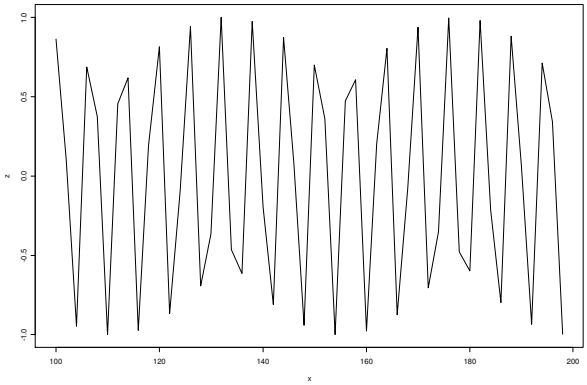
> write.table(s, file = "C:/salida.txt")

●**ER 1-2** Generar todos los números pares comprendidos entre 100 y 199. Realizar un gráfico en el que las abscisas sean estos valores y las ordenadas sus logaritmos naturales, y otro en el que las ordenadas sean sus cosenos.

> x<-seq(100,199,2)
> y<-log(x)
> plot(x,y)



> z<-cos(x)
> x11() #Con este comando abrimos una nueva ventana donde aparecerá el segundo gráfico
> plot(x,z,type="l") #Con la opción "l" creamos un gráfico de tipo línea



> # Con el siguiente comando podemos ver una lista de los dispositivos (*devices)* abiertos
> dev.list()
windows windows
2 3

●**ER 1-3** Cargar el paquete **datasets** y considerar el marco de datos **mtcars**. Construir un nuevo marco de datos solo con los automóviles que tengan cambio manual, otro con los vehículos que hacen más de 16 millas por galón de combustible, y un vector con las potencias de los automóviles que pesan menos de 3000 libras.

```
> library(datasets)
> attach(mtcars)
> mtcars[am==1,]
                mpg cyl  disp  hp drat    wt  qsec vs am gear carb
Mazda RX4      21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag  21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
Datsun 710     22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
Fiat 128       32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
Honda Civic    30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
Toyota Corolla 33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
Fiat X1-9      27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
Porsche 914-2  26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
....
Maserati Bora  15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
Volvo 142E     21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2

> mtcars[mpg>16,]
                  mpg cyl  disp  hp drat    wt  qsec vs am gear carb
Mazda RX4        21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag    21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
Datsun 710       22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive   21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
Valiant          18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
Merc 240D        24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
Merc 230         22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
Merc 280         19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
Merc 280C        17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
Merc 450SE       16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
Merc 450SL       17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
Fiat 128         32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
....
Lotus Europa     30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
Ferrari Dino     19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
Volvo 142E       21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2

> hp[wt<3]
 [1] 110 110  93  66  52  65  97  66  91 113 175 109
```

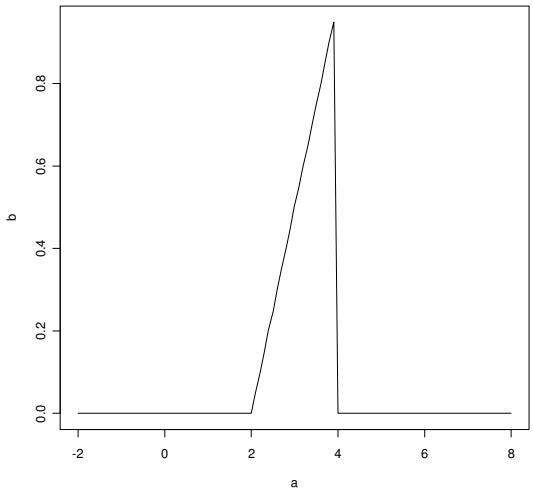
●**ER 1-4** La función **mean()** está integrada en *R* y sirve para calcular la media aritmética de un conjunto de datos. Construir una función que haga lo mismo que la función anterior. Ilustrarlo con un ejemplo.

```
> media<-function(x)
+sum(x)/length(x)

> valores<-c(4,8,5,2,9,6,1,9,3,4)
> media(valores)
[1] 5.1
> mean(valores)
[1] 5.1
```

●**ER 1-5** Definir y representar gráficamente la siguiente función:

$$f(x)=\begin{cases}\dfrac{x}{2}-1 & si\ x\in(2,4)\\ 0 & \text{en otro caso}\end{cases}$$

```
> #Construimos la función utilizando la estructura de control if-else
> f<-function(x) {if(abs(x-3)<1) x/2-1 else 0}

> #Como la función está definida a trozos debemos utilizar un pequeño artificio para dibujarla. Construimos en primer lugar el vector a de abscisas de los puntos que van a ser dibujados
> a<-c(seq(-2,8,0.1))
> length(a)
[1] 101

> #Construimos así mismo el vector de ordenadas de los puntos. Inicialmente está constituido por ceros
> b<-numeric(101)

> #La expresión "-2+(i-1)*0.1" devuelve el valor del elemento de a correspondiente al índice i
> for(i in 1:101) b[i]<-f(-2+(i-1)*0.1) #Aquí utilizamos la estructura de control for
> plot(a,b,type="l")
```



1.15 EJERCICIOS PROPUESTOS

●**EP 1-1** Cargar el paquete **datasets** y explicar qué es la variable **Nile**. Representar gráficamente dicha variable.

●**EP 1-2** Generar un archivo de texto con dos variables formadas cada una de ellas por 10 valores en metros. Leer el archivo con *R* y transformar esas variables en otras dos en centímetros. Obtener un archivo de salida en el *Escritorio* con los valores de las nuevas variables.

●**EP 1-3** Crear la función $y = sen(x^2 + x^3)$ y evaluarla en los valores de x: -3π, -2π, -π, 0, π, 2π, 3π. Dibujar la función en el intervalo (-π,π). Nota: el número π se escribe en *R* como **pi**.

●**EP 1-4** Generar la lista de los primeros 80 números impares. Ordenar los valores obtenidos en orden creciente y decreciente. Utilizar la función **sort()**.

●**EP 1-5** Crear un marco de datos en el que aparezcan las siguientes columnas: *valores, valores al cuadrado, valores al cubo,* siendo *valores* los 20 primeros números pares. Guardar en C este marco de datos.

Capítulo 2

ESTADÍSTICA DESCRIPTIVA DE UNA VARIABLE

2.1 INTRODUCCIÓN

A medida que vayamos introduciendo las funciones de *R* de este capítulo iremos haciendo referencia al ejercicio siguiente:

Ejemplo 2-1 Se ha medido la tensión de rotura en toneladas por cm^2 de 50 pernos de una nueva aleación de aluminio y se han obtenido los valores que aparecen en el archivo "Tensión de rotura.txt". Se pide: 1º) Tabla de frecuencias absolutas y relativas. 2º) Histograma de frecuencias absolutas. 3º) Diagrama boxplot. 4º) Media y desviación típica. 5º) Mediana, primer cuartil, tercer cuartil y percentil 60.

2.2 LECTURA DE DATOS

En primer lugar debemos leer el archivo correspondiente que, en nuestro caso, está en un fichero de texto denominado "Tensión de rotura.txt" y cuya ruta de acceso se ha de detallar con precisión:

```
> datos<-read.table("C:/Tensión de rotura.txt")
> datos
V1
1  4.05
2  4.58
3  4.42
...
49 3.54
50 4.84
> class(datos) #Con esta sentencia confirmamos el tipo de objeto que hemos cargado
[1] "data.frame"
```

Como se ve, el programa ha creado un marco de datos, denominado **datos,** que consta de 50 valores de una variable a la que automáticamente le ha asignado el nombre **V1**. Ahora crearemos un vector con esos 50 valores numéricos, lo que se consigue seleccionando la variable **V1** del *data frame* **datos:**

```
> tenrot<-datos$V1
> tenrot
[1] 4.05 4.58 4.42 4.20 4.41 4.64 4.76 4.58 3.95 4.17 4.56 3.51 3.27 3.80 3.59
[16] 4.70 3.77 3.80 4.27 3.94 3.96 4.86 4.39 4.04 4.36 3.72 4.00 3.46 4.01 4.08
[31] 3.40 3.89 4.46 4.38 4.41 4.33 4.16 4.58 4.03 3.76 4.05 4.17 4.46 3.60 4.76
[46] 3.99 4.43 4.15 3.54 4.84

> length(tenrot)
[1] 50
> #Por tanto, el vector tenrot tiene 50 elementos

> #Ordenamos los valores del vector tenrot
> sort(tenrot)
[1] 3.27 3.40 3.46 3.51 3.54 3.59 3.60 3.72 3.76 3.77 3.80 3.80 3.89 3.94 3.95
[16] 3.96 3.99 4.00 4.01 4.03 4.04 4.05 4.05 4.08 4.15 4.16 4.17 4.17 4.20 4.27
[31] 4.33 4.36 4.38 4.39 4.41 4.41 4.42 4.43 4.46 4.46 4.56 4.58 4.58 4.58 4.64
[46] 4.70 4.76 4.76 4.84 4.86
```

2.3 TABLA DE FRECUENCIAS

En primer lugar vamos a definir el número de intervalos en que agruparemos los datos para construir una tabla de frecuencias:

```
> sqrt(50)
[1] 7.071068
```

Construiremos 7 intervalos con una amplitud que calculamos a continuación:

```
> (max(tenrot)-min(tenrot))/7
[1] 0.2271429
```

Redondeando este valor a 0.3, formaremos 7 intervalos de esa amplitud, empezando en 3 y terminando en 5.1. Para ello construimos el vector formado por los extremos de los intervalos en los que agruparemos los datos:

```
> límites<-seq(3,5.1,0.3)
> límites
[1] 3.0 3.3 3.6 3.9 4.2 4.5 4.8 5.1
```

Veamos ahora a qué intervalo pertenece cada uno de los 50 valores leídos. Por defecto, *R* crea intervalos abiertos por la izquierda y cerrados por la derecha. Si queremos intervalos cerrados por la izquierda y abiertos por la derecha debemos escribir la opción **right=F**:

```
> tenrot.int<-cut(tenrot,límites,right=F)
> tenrot.int
[1] [3.9,4.2) [4.5,4.8) [4.2,4.5) [4.2,4.5) [4.2,4.5)    [4.5,4.8) [4.5,4.8)
[8] [4.5,4.8) [3.9,4.2) [3.9,4.2) [4.5,4.8) [3.3,3.6) [3,3.3)   [3.6,3.9)
[15] [3.3,3.6) [4.5,4.8) [3.6,3.9) [3.6,3.9) [4.2,4.5) [3.9,4.2) [3.9,4.2)
[22] [4.8,5.1) [4.2,4.5) [3.9,4.2) [4.2,4.5) [3.6,3.9) [3.9,4.2) [3.3,3.6)
....
[43] [4.2,4.5) [3.6,3.9) [4.5,4.8) [3.9,4.2) [4.2,4.5) [3.9,4.2) [3.3,3.6)
[50] [4.8,5.1)
7 Levels: [3,3.3) [3.3,3.6) [3.6,3.9) [3.9,4.2) [4.2,4.5) ... [4.8,5.1)
```

Utilizamos ahora la función **table()** para contar el número de veces que aparece cada intervalo, lo que es propiamente una tabla de frecuencias:

```
> table(tenrot.int)
tenrot.int
[3,3.3) [3.3,3.6) [3.6,3.9) [3.9,4.2) [4.2,4.5)
[4.5,4.8) [4.8,5.1)
1       5       7       15      12      8   2
```

Es decir, en el primer intervalo [3,3.3) hay un valor, en el segundo intervalo [3.3,3.6) caen 5 valores, etc.

Si aplicáramos la función anterior al vector **tenrot** se obtendría el resultado siguiente, de poca utilidad en este caso:

```
> table(tenrot)
tenrot
3.27  3.4 3.46 3.51 3.54 3.59  3.6 3.72 3.76 3.77  3.8 3.89 3.94 3.95 3.96 3.99
  1    1    1    1    1    1    1    1    1    1    2    1    1    1    1    1
  4 4.01 4.03 4.04 4.05 4.08 4.15 4.16 4.17  4.2 4.27 4.33 4.36 4.38 4.39 4.41
  1    1    1    1    2    1    1    1    2    1    1    1    1    1    1    2
4.42 4.43 4.46 4.56 4.58 4.64  4.7 4.76 4.84 4.86
  1    1    2    1    3    1    1    2    1    1
```

2.4 DIAGRAMA DE TALLOS Y HOJAS

Un diagrama de tallos y hojas (*stem and leaf)* es una representación sencilla de los datos, similar a un histograma, pero con la ventaja de que se conserva la información numérica de todos y cada uno de los valores.

```
> stem(tenrot)
The decimal point is 1 digit(s) to the left of the |
   32 | 7
   34 | 06149
   36 | 0267
   38 | 0094569
   40 | 01345585677
   42 | 073689
   44 | 1123666888
   46 | 4066
   48 | 46
```

Si queremos aumentar el número de tallos al doble utilizamos la opción **scale=2**.

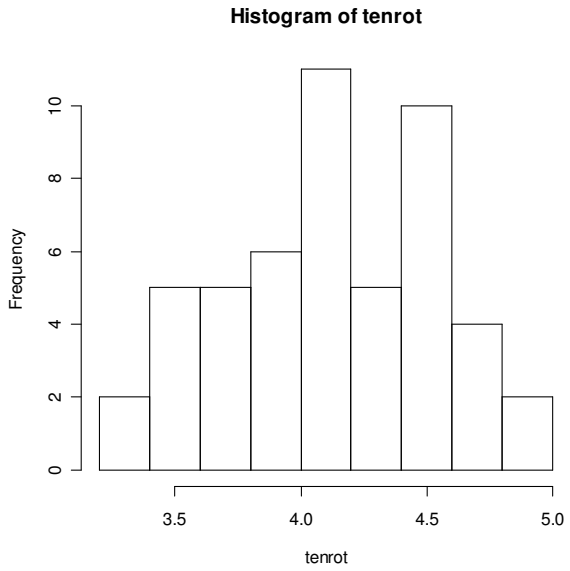
```
> stem(tenrot,scale=2)
The decimal point is 1 digit(s) to the left of the |
   32 | 7
   33 |
   34 | 06
   35 | 149
   36 | 0
   37 | 267
```

38 | 009
39 | 4569
40 | 0134558
41 | 5677
42 | 07
43 | 3689
44 | 112366
45 | 6888
46 | 4
47 | 066
48 | 46

2.5 HISTOGRAMA

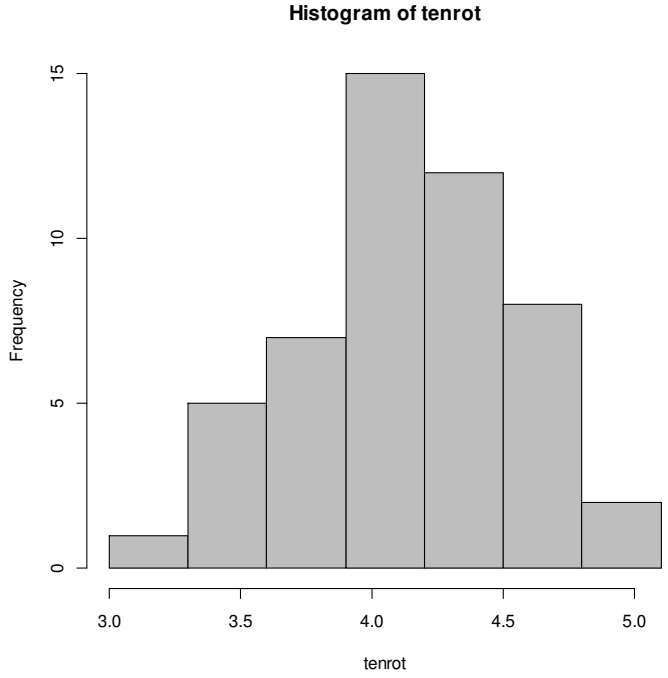
Al igual que el diagrama de tallos y hojas, un histograma da una idea de cómo se distribuyen los datos. La instrucción correspondiente es

> hist(tenrot)



Mediante la instrucción anterior se obtiene un histograma, dividido en 9 intervalos, que el programa genera automáticamente. Con objeto de disponer de un histograma con los siete intervalos definidos por el vector **límites** hacemos lo siguiente:

> hist(tenrot,límites,right=F,col='grey') #Dibujamos así el histograma de frecuencias (con barras de color gris)



2.6 MEDIDAS DE CENTRALIZACIÓN

Obtenemos a continuación la media y la mediana de los valores que forman el vector **tenrot**:

```
> mean(tenrot);median(tenrot)
[1] 4.1448
[1] 4.155
```

Para obtener la media recortada al 10 %; es decir, la media aritmética de todos los valores, exceptuando el 10% de los que están por arriba y el 10% de los que están por abajo, hacemos:

```
> mean(tenrot,trim=0.1)
[1] 4.1535
```

2.7 MEDIDAS DE DISPERSIÓN

La instrucción **var()** nos devuelve un estimador insesgado de la varianza muestral, que se denomina *cuasivarianza*. Para obtener el valor de la varianza de los datos debemos multiplicar el valor anterior por **(length(tenrot)-1)/length(tenrot)).** En nuestro ejemplo el vector tiene 50 datos, por tanto:

```
> varianza.muestra<-((50-1)/50)*var(tenrot)
> varianza.muestra
[1] 0.1583210
> sqrt(varianza.muestra)
[1] 0.3978957
```

El último valor es la desviación típica de los datos. Este valor también puede obtenerse a través de la función **sd()**, que calcula la *cuasidesviación típica.*

```
> desvtípica.muestra<-sqrt(varianza.muestra)
> desvtípica.muestra
[1] 0.3978957
> sqrt((50-1)/50)*sd(tenrot)
[1] 0.3978957
```

El *coeficiente de variación* CV es el cociente entre la desviación típica y el valor absoluto de la media. Como en la instalación normal de R no existe una función específica para obtener este parámetro, podríamos generar la función **cv()**, que lo calcula y lo expresa en tanto por ciento:

```
> cv<-function(x)
+ 100*sd(x)*sqrt(length(x-1)/length(x))/abs(mean(x))
> cv(tenrot)
[1] 9.697339
```

2.8 PERCENTILES

Los percentiles son los valores que dividen el rango de los datos en cien unidades de modo que, por ejemplo, el percentil 20 es el valor que deja por debajo de sí el 20% de las observaciones; el percentil 50 deja por debajo la mitad de las observaciones, y por tanto es la mediana, etc. La función **quantile()** sirve para calcular percentiles.

```
>quantile(tenrot,0.3) #Con esta sentencia calculamos el percentil 30
    30%
    3.957

>quantile(tenrot,0.5) #Volvemos a calcular la mediana de otro modo
    50%
    4.155
```

Mediante la función anterior se puede calcular, por defecto, un resumen de cinco números: mínimo, primer cuartil (Q1), mediana o segundo cuartil (Q2), tercer cuartil (Q3), y máximo.

```
> quantile(tenrot)
 0%   25%   50%   75%  100%
3.2700 3.9025 4.1550 4.4275 4.8600
```

Si además de los cinco valores anteriores queremos conocer la media podemos hacer:

```
> summary(tenrot)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  3.270   3.903   4.155   4.145   4.428   4.860
```

Calculemos el recorrido intercuartílico Q3-Q1:

```
> IQR(tenrot)
[1] 0.525
```

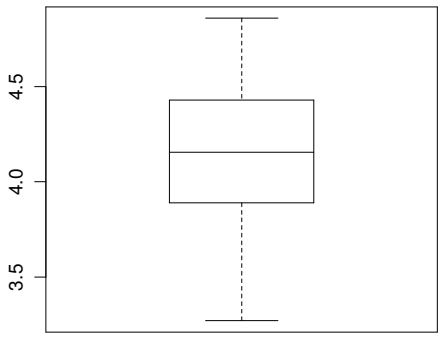
Podemos calcular este valor de otro modo:

```
> quantile(tenrot,0.75)-quantile(tenrot,0.25)
75%
 0.525
```

2.9 DIAGRAMA DE CAJAS (BOX PLOT)

Para dibujar un diagrama de cajas utilizamos la función **boxplot()**, que en su forma más simple (tiene varias versiones) es:

```
> boxplot(tenrot)
```

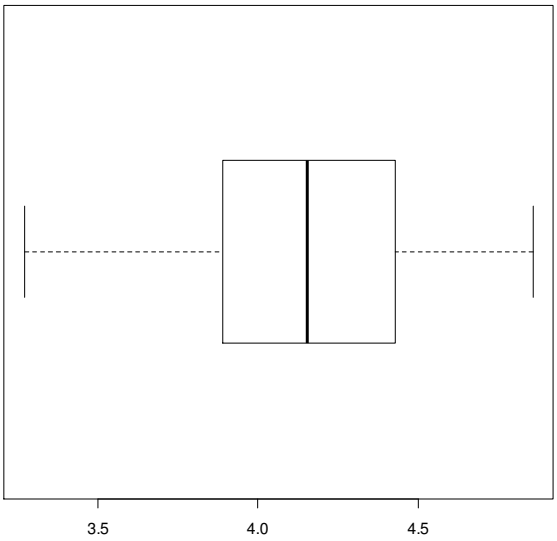


Los extremos del diagrama son el valor máximo y el mínimo. Si hay *outliers* (valores mayores que 3/2*RIQ por encima de Q3 o por debajo de Q1) vienen señalados por un pequeño círculo

y, entonces, los extremos que aparecen son el máximo y el mínimo de los valores que quedan al eliminar los valores atípicos.

También es posible dibujar el diagrama anterior en horizontal:

```
> boxplot(tenrot,horizontal=T)
```



Para calcular los cinco valores que definen el diagrama *boxplot* hacemos:

```
> boxplot.stats(tenrot)
$stats
[1] 3.270 3.890 4.155 4.430 4.860
$n
[1] 50
$conf
[1] 4.034339 4.275661
$out
numeric(0)
```

La última salida se interpreta del siguiente modo: el vector **$stats** muestra, respectivamente, la pata (bigote) inferior, el extremo inferior de la caja, la mediana, el extremo superior de la caja y la pata superior. **$n** es el número de observaciones. **$out** representa el vector de los valores atípicos, en este caso ninguno.

2.10 EJERCICIOS RESUELTOS

●**ER 2-1** Las longitudes en micras de 25 grietas medidas en una pieza de hormigón son: 50, 68, 84, 86, 64, 67, 78, 87, 110, 85, 52, 65, 52, 93, 72, 70, 105, 85, 30, 42, 74, 30, 70, 65, 49. 1) Agrupar los datos en los siguientes intervalos: [30,40), [40,50), [50,60), [60,70), [70,75), [75,85), [85,90), [90,110), [110,∞]. 2) Construir una tabla de frecuencias en la que figuren las columnas: Clases / Frecuencia absoluta / Frecuencia relativa / Frecuencia absoluta acumulada / Frecuencia relativa acumulada.

```
> grietas<-c(50,68,84,86,64,67,78,87,110,85,52,65,52,
93,72,70,105,85,30,42,74,30,70,65,49)
> lim<-c(30,40,50,60,70,75,85,90,110,Inf)
> grietas.int<-cut(grietas,lim,right=F)
> grietas.int
 [1] [50,60)   [60,70)   [75,85)   [85,90)   [60,70)   [60,70)   [75,85)
 [8] [85,90)   [110,Inf) [85,90)   [50,60)   [60,70)   [50,60)   [90,110)
[15] [70,75)   [70,75)   [90,110)  [85,90)   [30,40)   [40,50)   [70,75)
[22] [30,40)   [70,75)   [60,70)   [40,50)
9 Levels: [30,40) [40,50) [50,60) [60,70) [70,75) [75,85) [85,90) ... [110,Inf)
> Intervalos<-levels(grietas.int)
> Intervalos
[1] "[30,40)"   "[40,50)"   "[50,60)"   "[60,70)"   "[70,75)"   "[75,85)"
[7] "[85,90)"   "[90,110)"  "[110,Inf)

> table(grietas.int)
grietas.int
 [30,40)   [40,50)   [50,60)   [60,70)   [70,75)   [75,85)   [85,90)  [90,110)
 2         2         3         5         4         2         4         2
[110,Inf)
 1
> a<-as.data.frame(table(grietas.int))
> a
                grietas.int Freq
1           [30,40)    2
2           [40,50)    2
3           [50,60)    3
4           [60,70)    5
5           [70,75)    4
6           [75,85)    2
7           [85,90)    4
8           [90,110)    2
9           [110,Inf)    1
```

> Frec.abs<-a$Freq #En las instrucciones anteriores hemos transformado la tabla en un marco de datos. Con esta última instrucción construimos el vector Frec.abs; para ello hemos seleccionado la 2ª columna del data.frame *a*.

```
> Frec.abs
[1] 2 2 3 5 4 2 4 2 1
> sum(Frec.abs)
[1] 25
> Frec.rel<-Frec.abs/25
> Frec.abs.ac<-cumsum(Frec.abs)
> Frec.rel.ac<-cumsum(Frec.rel)
> data.frame(Clases,Frec.abs,Frec.rel,Frec.abs.ac,Frec.rel.ac)
  Clases Frec.abs Frec.rel Frec.abs.ac Frec.rel.ac
1 [30,40)       2     0.08          2       0.08
2 [40,50)       2     0.08          4       0.16
3 [50,60)       3     0.12          7       0.28
4 [60,70)       5     0.20         12       0.48
5 [70,75)       4     0.16         16       0.64
6 [75,85)       2     0.08         18       0.72
7 [85,90)       4     0.16         22       0.88
8 [90,110)      2     0.08         24       0.96
9 [110,Inf)     1     0.04         25       1.00
```

●**ER 2-2** Calcular la media, varianza y desviación típica de los siguientes datos, que aparecen agrupados en clases:

Clase	x_i	F_i
[10.5,21.5)	16	6
[21.5,32.5)	27	8
[32.5,43.5)	38	7
[43.5,54.5)	49	17
[54.5,65.5)	60	18
[65.5,76.5)	71	10
[76.5,87.5)	82	3
[87.5,98.5)	93	3

```
> x<-c(16,27,38,49,60,71,82,93)
> F<-c(6,8,7,17,18,10,3,3)
> valores<-c(rep(x,F))
> valores
 [1] 16 16 16 16 16 16 27 27 27 27 27 27 27 27 38 38 38 38 38 38 38 49 49 49 49
[26] 49 49 49 49 49 49 49 49 49 49 49 49 49 60 60 60 60 60 60 60 60 60 60 60 60
[51] 60 60 60 60 60 60 71 71 71 71 71 71 71 71 71 71 82 82 82 93 93 93
> mean(valores)
[1] 51.75
```

> length(valores)
[1] 72
> ((72-1)/72)*var(valores)
[1] 372.2431
> sqrt((72-1)/72)*sd(valores)
[1] 19.2936

●**ER 2-3** Dado el conjunto de datos {1,1,1,3,1,2,3,1,1,1,2,3,2,3,1} calcular la media, mediana, varianza, desviación típica y los cuartiles. Representar este conjunto de datos usando dos gráficos distintas: diagrama de barras y diagrama de sectores.

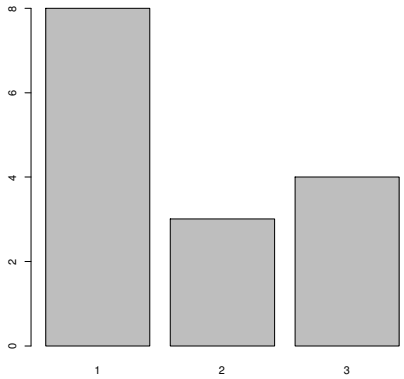
> datos<-c(1,1,1,3,1,2,3,1,1,1,2,3,2,3,1)

> summary(datos)
Min. 1st Qu. Median Mean 3rd Qu. Max.
1.000 1.000 1.000 1.733 2.500 3.000

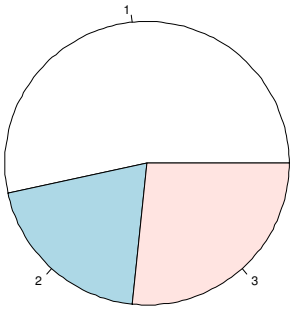
> length(datos)
[1] 15
> var(datos)*14/15 #Obtenemos la varianza
[1] 0.7288889
> sd(datos)*sqrt(14/15) #Obtenemos la desviación típica
[1] 0.8537499

> a<-table(datos)
> a
datos
1 2 3
8 3 4

> barplot(a)



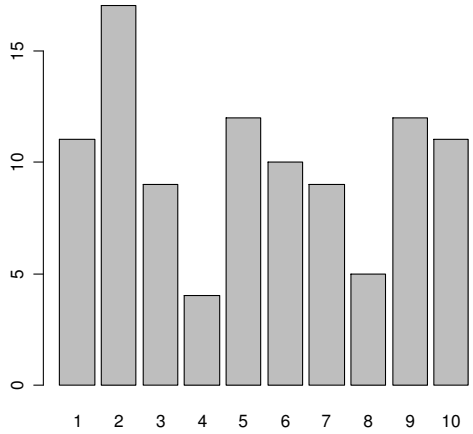
> pie(a)



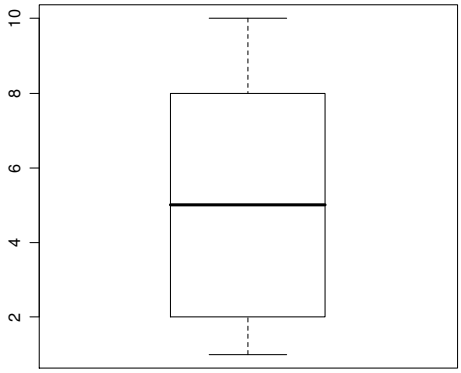
●**ER 2-4** Generar, mediante la función **sample()**, una lista aleatoria de 100 números naturales comprendidos entre 1 y 10. Calcular la media, mediana, varianza, desviación típica y cuartiles de los datos obtenidos. Representar los datos mediante un diagrama de barras y dibujar un diagrama boxplot indicando los valores atípicos, si existen.

> valores<-sample(1:10,100,replace=T) #Las extracciones se hacen, evidentemente, con reemplazamiento
> valores #Como es lógico, al tratarse de una elección aleatoria, cada vez que se ejecute esta sentencia se obtendrán valores diferentes
[1] 2 3 5 9 6 3 6 5 3 9 9 1 9 10 3 9 2 5 5 2 5 8 9 6 1
[26] 2 1 9 10 8 2 6 7 9 10 7 10 7 5 2 5 7 3 9 6 5 1 2 8 7
[51] 1 1 6 3 5 4 6 2 10 10 10 3 10 3 1 5 7 4 8 10 7 10 2 6 2
[76] 3 10 2 2 2 7 4 9 7 2 2 2 9 2 8 1 1 6 1 5 5 4 9 1 6
> summary(valores)
Min. 1st Qu. Median Mean 3rd Qu. Max.
1.00 2.00 5.00 5.29 8.00 10.00
> varianza<-((100-1)/100)*var(valores)
> varianza
[1] 9.1859
> des.típica<-sqrt(varianza)
> des.típica
[1] 3.030825
> a<-table(valores)
> a
valores
1 2 3 4 5 6 7 8 9 10
11 17 9 4 12 10 9 5 12 11

> barplot(a)



> boxplot(valores)



> boxplot.stats(valores)

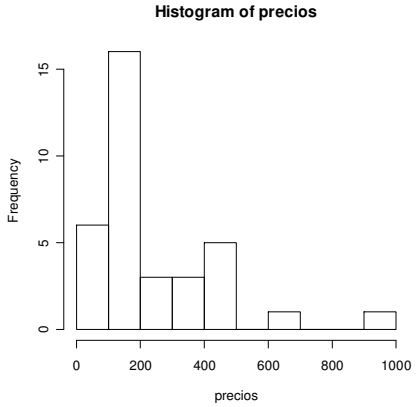
$stats
[1] 1 2 5 8 10
$n
[1] 100
$conf
[1] 4.052 5.948
$out
numeric(0)

> #No se obtienen valores atípicos

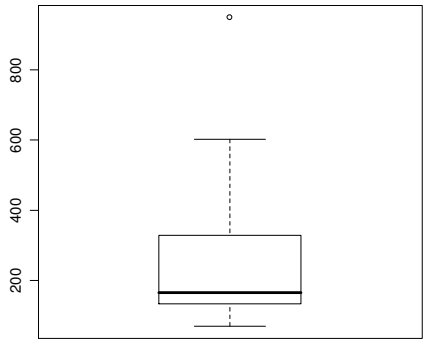
●**ER 2-5** Los valores siguientes son los precios, en miles de euros, de 35 terrenos rurales vendidos en Bizkaia en el año 2013: 115, 232, 181, 161, 155, 137, 165, 171, 139, 130, 406, 69, 171, 135, 135, 132, 88, 410, 87, 90, 123, 157, 345, 323, 411, 334, 80, 87, 235, 198, 450, 223, 602, 415, 950. Se pide: 1º) Histograma. 2º) Diagrama boxplot. 3º) Detectar datos atípicos. 4º) Media, mediana, desviación típica y recorrido intercuartílico antes y después de eliminar los datos atípicos, si existen.

```
> precios<-c(115,232,181,161,155,137,165,171,139,130,406,69,171,
135,135,132,88,410,87,90,123,157,345,323,411,334,80,
87,235,198,450,223,602,415,950)
> precios
 [1] 115 232 181 161 155 137 165 171 139 130 406  69 171 135 135 132  88 410  87
[20]  90 123 157 345 323 411 334  80  87 235 198 450 223 602 415 950
```

```
> hist(precios)
```




```
> boxplot(precios)
```



> boxplot.stats(precios)
$stats
[1] 69.0 131.0 165.0 328.5 602.0
$n
[1] 35
$conf
[1] 112.2539 217.7461
$out
[1] 950

which(precios==950)
[1] 35
> #Con esta sentencia sabemos que el dato atípico, de valor 950, es el que ocupa el lugar 35

> precios.sin.atípico<-precios[-35] #Eliminamos el valor atípico
> precios.sin.atípico
 [1] 115 232 181 161 155 137 165 171 139 130 406 69 171 135 135 132 88 410 87
[20] 90 123 157 345 323 411 334 80 87 235 198 450 223 602 415
> length(precios)
[1] 35
> length(precios.sin.atípico)
[1] 34
>mean(precios);median(precios);sqrt((35-1)/35)*sd(precios)
[1] 235.4857
[1] 165
[1] 177.7582
> IQR(precios)
[1] 197.5

>mean(precios.sin.atípico);median(precios.sin.atípico);sqrt((34-1)/34)*sd(precios.sin.atípico);
[1] 214.4706
[1] 163
[1] 130.6525
> IQR(precios.sin.atípico)
[1] 170.5

2.11 EJERCICIOS PROPUESTOS

●**EP 2-1** En un cierto estudio se han recogido los siguientes datos: 15, 16, 21, 23, 23, 26, 26, 30, 32, 41, 42, 51, 53, 53, 53, 53, 55, 60, 60, 69. Se pide: 1º) Generar una tabla de frecuencias, sin agrupar por intervalos y agrupando en cuatro intervalos. 2º) Diagrama de tallos y hojas. 3º) Histograma por defecto y con los cuatro intervalos anteriores.

●**EP 2-2** Cargar el paquete **datasets** y leer el marco de datos **cars**. 1º) Obtener media, mediana, cuartiles y recorrido intercuartílico de las dos variables que aparecen. 2º) Dibujar un gráfico que relacione ambas variables. 3º) Obtener el diagrama de caja para la variable **speed**.

●**EP 2-3** Cargar el paquete **datasets** y leer el marco de datos **LifeCycleSavings**. Se pide: 1º) Hacer una tabla completa de frecuencias de la variable **pop75** dividiéndola en cinco intervalos. 2º) Buscar los datos atípicos, si existen, de la variable **ddpi**. 3º) Mediana y percentil 35 de la variable **sr**.

●**EP 2-4** Cargar el paquete **vcd**, leer el marco de datos **Baseball** y calcular: 1º) Número de variables del marco de datos. 2º) Diagrama de tallos y hojas, histograma, media y desviación típica de la variable **atbat86**. 3º) Diagrama de barras y diagrama de sectores de la variable **league87**.

●**EP 2-5** Instalar y cargar el paquete **MASS**. Elegir el marco de datos **Forbes2000**, donde aparece el ranking de las 2000 empresas más importantes del mundo en el año 2004. Hacer un análisis descriptivo lo más completo posible de la variable **sales** para las 100 primeras compañías.

Capítulo 3

ESTADÍSTICA DESCRIPTIVA DE DOS VARIABLES

3.1 INTRODUCCIÓN

En el capítulo anterior hemos estudiado cómo describir con *R* el valor que toma una variable en los individuos de una muestra o población. Sin embargo, en ocasiones interesa estudiar cómo se comportan dos variables cuantitativas de manera conjunta, con objeto de estudiar si existe alguna relación entre ellas. Por ejemplo, podemos querer analizar a la vez el PESO y la TALLA de los individuos de una población, para intentar averiguar una posible relación entre ambas variables.

Para ir desarrollando los conceptos de este tema vamos a utilizar el conjunto de datos **rmr,** que está incluido en el paquete **IswR.** Tras instalar y cargar el paquete podemos obtener información sobre él haciendo

```
> library(help=ISwR)
```

Una vez cargado el paquete, leemos el *data frame* **rmr**. Este marco de datos contiene 44 pares de datos referidos a las variables peso en kg (**body.weight**) y tasa de metabolismo basal en kcal/24h (**metabolic.rate**) de 44 mujeres elegidas al azar.

```
> rmr
body.weight metabolic.rate
1        49.9          1079
2        50.8          1146
3        51.8          1115
...
43      125.2          1630
44      143.3          1708
```

Con objeto de que, a partir de ahora, las dos variables sean accesibles debemos hacer

```
> attach(rmr)
```

Ahora sí estamos en condiciones de poder utilizar las variables:

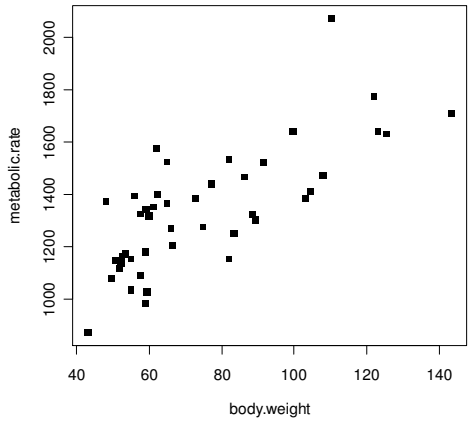
```
> body.weight
 [1]  49.9  50.8  51.8  52.6  57.6  61.4  62.3  64.9  43.1  48.1  52.2  53.5
[13]  55.0  55.0  56.0  57.8  59.0  59.0  59.2  59.5  60.0  62.1  64.9  66.0
[25]  66.4  72.8  74.8  77.1  82.0  82.0  83.4  86.2  88.6  89.3  91.6  99.8
[37] 103.0 104.5 107.7 110.2 122.0 123.1 125.2 143.3

> metabolic.rate
 [1] 1079 1146 1115 1161 1325 1351 1402 1365  870 1372 1132 1172 1034 1155 1392
[16] 1090  982 1178 1342 1027 1316 1574 1526 1268 1205 1382 1273 1439 1536 1151
[31] 1248 1466 1323 1300 1519 1639 1382 1414 1473 2074 1777 1640 1630 1708
```

Es importante señalar que en el capítulo 8 volveremos a abordar este tema desde un punto de vista diferente, el de la estadística de la inferencia. Lo que a continuación se presenta está elaborado exclusivamente desde la perspectiva del análisis descriptivo de datos.

3.2 DIAGRAMA DE DISPERSIÓN

Lo primero que se debe hacer al tratar de estudiar la posible relación entre dos variables estadísticas es visualizar los datos mediante un *diagrama de dispersión* o *scatterplot,* en el que cada par de valores de las variables se corresponde con las coordenadas de un conjunto de puntos del plano (nube de puntos):

> plot(body.weight,metabolic.rate,pch=15) #La opción pch=15 es una de las posibles alternativas existentes para marcar un punto



Una primera visión del gráfico sugiere una cierta relación entre el peso X y la tasa metabólica Y, en el sentido de que para valores grandes de X se encuentran más bien valores grandes de Y, y para valores pequeños de X valores también pequeños de Y.

3.3 COVARIANZA Y COEFICIENTE DE CORRELACIÓN

La figura anterior nos da idea de una cierta "asociación lineal" entre las dos variables. Para cuantificar tal asociación se utiliza la *covarianza* o variación conjunta de ambas variables:

$$s_{xy} = \frac{\sum_{i=1}^{n} (x_i - \overline{x})(y_i - \overline{y})}{n}$$

Para calcular este valor en *R* podemos hacer:

>sum((body.weight-mean(body.weight))*(metabolic.rate-mean(metabolic.rate)))/44
[1] 4185.965

La función **cov()** no nos proporciona exactamente la covarianza muestral, sino un *estimador* (concepto este de la inferencia estadística) de ella. En concreto, mediante esa función se

computa el mismo valor obtenido más arriba, dividido por n-1 en lugar de n, por lo que para determinar la covarianza deberemos multiplicar por n-1 y dividir entre n:

```
> cov(body.weight,metabolic.rate)
[1] 4283.313
> cov(body.weight,metabolic.rate)*(43/44)
[1] 4185.965
```

La covarianza es una medida de la información que proporciona el gráfico de dispersión, en cuanto a la asociación lineal entre dos variables. Sin embargo, este valor depende de las unidades de medida, de modo que no tiene sentido decir qué es una covarianza grande pues cambiará al cambiar las unidades. Para evitar este problema la covarianza se divide por el producto de desviaciones típicas, con lo que se obtiene una medida adimensional, denominada *coeficiente de correlación* o *coeficiente de correlación lineal,* entre las variables X e Y:

$$r(x,y)=\frac{s_{xy}}{s_x s_y}=\frac{\dfrac{\sum_{i=1}^{n}(x_i-\overline{x})(y_i-\overline{y})}{n}}{\sqrt{\dfrac{\sum_{i=1}^{n}(x_i-\overline{x})^2}{n}}\sqrt{\dfrac{\sum_{i=1}^{n}(y_i-\overline{y})^2}{n}}}=\frac{\dfrac{\sum_{i=1}^{n}(x_i-\overline{x})(y_i-\overline{y})}{n-1}}{\sqrt{\dfrac{\sum_{i=1}^{n}(x_i-\overline{x})^2}{n-1}}\sqrt{\dfrac{\sum_{i=1}^{n}(y_i-\overline{y})^2}{n-1}}}$$

Como se ve en esta expresión, para calcular el coeficiente de correlación da lo mismo usar las desviaciones y covarianzas de los datos o sus estimadores correspondientes. El valor anterior se calcula directamente mediante la función **cor()**:

```
> cor(body.weight,metabolic.rate)
[1] 0.7442379
> cor(metabolic.rate,body.weight)
[1] 0.7442379
```

Vemos que el coeficiente de correlación entre X e Y es el mismo que entre Y y X. Un coeficiente de correlación relativamente alto entre dos variables, como en este caso, indica una cierta asociación lineal entre las variables, aunque no necesariamente una relación real entre ellas.

3.4 REGRESIÓN LINEAL

Una vez comprobado que existe una asociación lineal entre las dos variables, trataremos de calcular la recta que "mejor se ajusta" a la nube de puntos mediante el criterio denominado de *mínimos cuadrados*. Obtenemos así la *recta de regresión (RR)*.

Cuando se tienen datos de dos variables X e Y es posible calcular dos rectas de regresión: la de Y sobre X, que sirve para predecir Y conociendo X, y la de X sobre Y que nos da el mejor pronóstico de X conociendo la Y. Calculemos en primer lugar la RR de la tasa de metabolismo basal Y respecto del peso X:

```
> lm(metabolic.rate~body.weight)
        Call:
lm(formula = metabolic.rate ~ body.weight)
Coefficients:
(Intercept)  body.weight
811.23         7.06
```

Este resultado significa que la RR de Y sobre X tiene por ecuación:

$$Y = 811.23 + 7.06X$$

Así, por ejemplo, para un peso de 70 kg la tasa metabólica estimada es

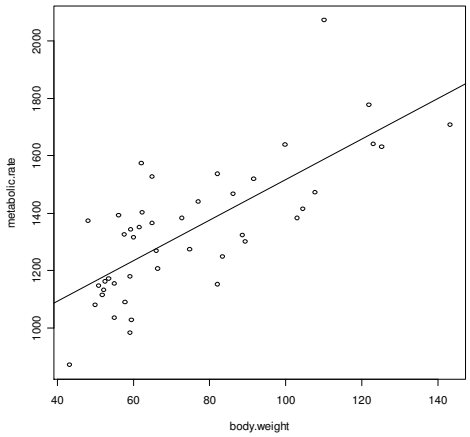
```
> pron70kg<-811.23+7.06*70
> pron70kg
[1] 1305.43
```

La RR de X sobre Y se obtiene así:

```
> lm(body.weight~metabolic.rate)
Call:
lm(formula = body.weight ~ metabolic.rate)
Coefficients:
  (Intercept)  metabolic.rate
    -30.24427         0.07846
```

Dibujemos la RR de Y sobre X superpuesta a la nube de puntos:

```
> plot(body.weight,metabolic.rate) #Al separar por una coma ponemos las variables en el orden X, Y
> abline(lm(metabolic.rate~body.weight)) #Al separar por "~" ponemos las variables en el orden Y, X
```

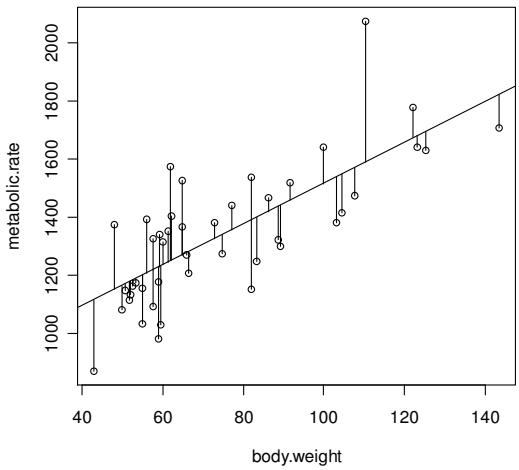


La función **abline()** dibuja rectas cuya ordenada en el origen es *a* (811,23) y cuya pendiente es *b* (7,06).

Una manera efectiva de detectar las posibles deficiencias de un modelo de regresión lineal consiste en llevar a cabo un análisis de los *residuos*, o sea de las diferencias entre las ordenadas de los puntos de la muestra y las correspondientes a la recta de regresión.

Para generar un gráfico que muestre los residuos mediante segmentos que unen las observaciones con los correspondientes puntos de la recta ajustada hacemos lo que sigue a continuación. Previamente es conveniente almacenar el modelo lineal utilizando un nombre, como por ejemplo **lm.tasa**:

```
> lm.tasa<-lm(metabolic.rate~body.weight)
> segments(body.weight,fitted(lm.tasa),body.weight,metabolic.rate)
```



Mediante la función **segments()** se consigue dibujar un segmento que une los puntos de coordenadas (x_1,y_1,x_2,y_2).

Si deseamos calcular los valores de esos 44 residuos (longitudes de los segmentos verticales de la figura anterior) hacemos

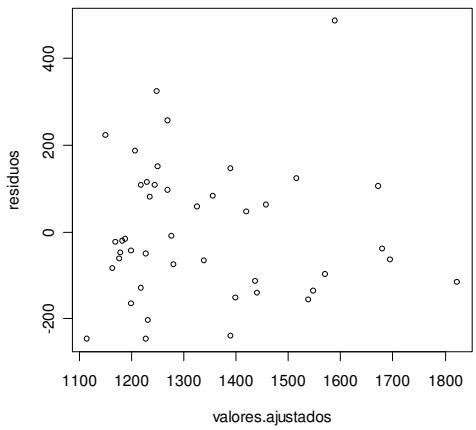
```
> residuos<-resid(lm.tasa)
> residuos
       1        2        3        4        5        6        7
 -84.49711  -23.85069  -61.91022  -21.55784  107.14452  106.31832  150.96474
...
       43        44
 -65.07956 -114.85701
```

Los valores ajustados, es decir, los que toma la recta de regresión para cada valor de la variable observada **body.weight**, son:

```
> valores.ajustados<-fitted(lm.tasa)
> valores.ajustados
     1       2       3       4       5       6       7       8
1163.497 1169.851 1176.910 1182.558 1217.855 1244.682 1251.035 1269.390
...
     41       42       43       44
1672.489 1680.255 1695.080 1822.857
```

Graficando los residuos frente a los valores ajustados podemos detectar deficiencias en el modelo.

```
> plot(valores.ajustados,residuos)
```



Sin entrar en detalles que exceden los objetivos del presente texto, el gráfico anterior sugiere en principio una no adecuación del modelo, pues parece que, según aumentan los valores ajustados, los residuos tienden a disminuir. Da la impresión, en definitiva, de que se viola la hipótesis de homocedasticidad o varianza constante del error (residuo). No obstante, existen técnicas estadísticas para adecuar los datos muestrales a un modelo lineal en una situación como la presente.

3.5 EL CUARTETO DE ANSCOMBE

El cuarteto de Anscombe es un grupo de cuatro conjuntos de datos, creados artificialmente, cuyo objetivo es poner de manifiesto algo crucial a la hora de aplicar un modelo de regresión lineal y que es visualizar los datos previamente. Los datos son los correspondientes a los pares (x1,y1), (x2,y2), (x3,y3) y (x4,y4). Realizaremos la regresión de y1 respecto de x1, de y2 respecto de x2, de y3 respecto dex3 y de y4 respecto de x4.

```
> x1<-c(10,8,13,9,11,14,6,4,12,7,5)
> y1<-c(8.04,6.95,7.58,8.81,8.33,9.96,7.24,4.26,10.84,4.82,5.68)

> x2<-c(10,8,13,9,11,14,6,4,12,7,5)
> y2<-c(9.14,8.14,8.74,8.77,9.26,8.1,6.13,3.1,9.13,7.26,4.74)

> x3<-c(10,8,13,9,11,14,6,4,12,7,5)
> y3<-c(7.46,6.77,12.74,7.11,7.81,8.84,6.08,5.39,8.15,6.42,5.73)

> x4<-c(8,8,8,8,8,8,8,19,8,8,8)
> y4<-c(6.58,5.76,7.71,8.84,8.47,7.04,5.25,12.5,5.56,7.91,6.89)

> modelo1<-lm(y1~x1)
> modelo2<-lm(y2~x2)
> modelo3<-lm(y3~x3)
> modelo4<-lm(y4~x4)

> summary(modelo1)
Call:
lm(formula = y1 ~ x1)
Residuals:
 Min      1Q  Median      3Q     Max
-1.92127 -0.45577 -0.04136  0.70941  1.83882
Coefficients:
  Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.0001     1.1247   2.667  0.02573 *
x1            0.5001     0.1179   4.241  0.00217 **
---
```

```
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
Residual standard error: 1.237 on 9 degrees of freedom
Multiple R-squared:  0.6665,    Adjusted R-squared:  0.6295
F-statistic: 17.99 on 1 and 9 DF,  p-value: 0.00217
```

> summary(modelo2)

```
Call:
lm(formula = y2 ~ x2)
Residuals:
Min      1Q  Median      3Q     Max
-1.9009 -0.7609  0.1291  0.9491  1.2691
Coefficients:
  Estimate Std. Error t value Pr(>|t|)
(Intercept)    3.001      1.125   2.667  0.02576 *
x2             0.500      0.118   4.239  0.00218 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
Residual standard error: 1.237 on 9 degrees of freedom
Multiple R-squared:  0.6662,    Adjusted R-squared:  0.6292
F-statistic: 17.97 on 1 and 9 DF,  p-value: 0.002179
```

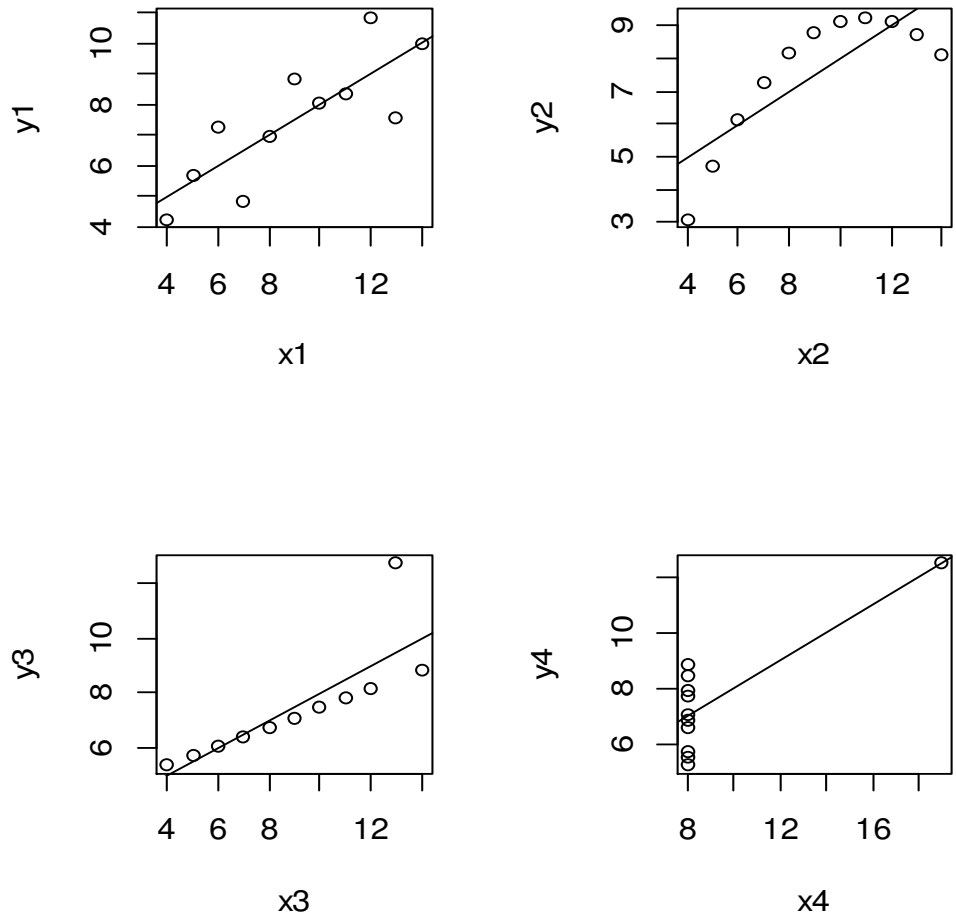
> summary(modelo3)

```
Call:
lm(formula = y3 ~ x3)
Residuals:
   Min      1Q  Median      3Q     Max
-1.1586 -0.6146 -0.2303  0.1540  3.2411
Coefficients:
 Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.0025     1.1245   2.670  0.02562 *
x3            0.4997     0.1179   4.239  0.00218 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
Residual standard error: 1.236 on 9 degrees of freedom
Multiple R-squared:  0.6663,    Adjusted R-squared:  0.6292
F-statistic: 17.97 on 1 and 9 DF,  p-value: 0.002176
```

```
> summary(modelo4)
Call:
lm(formula = y4 ~ x4)
Residuals:
   Min     1Q Median     3Q    Max
-1.751 -0.831  0.000  0.809  1.839
Coefficients:
   Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.0017     1.1239   2.671  0.02559 *
x4            0.4999     0.1178   4.243  0.00216 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 1.236 on 9 degrees of freedom
Multiple R-squared:  0.6667,    Adjusted R-squared:  0.6297
F-statistic:    18 on 1 and 9 DF,  p-value: 0.002165
```

Como se puede observar, los resultados correspondientes a las cuatro regresiones son prácticamente idénticos. Sin embargo, como se aprecia en los gráficos que se obtienen a continuación, el modelo de regresión lineal (*y=3+0.5x*) es adecuado únicamente para el par de variables (x1,y1). Este resultado subraya la importancia de graficar los datos antes de efectuar una regresión.

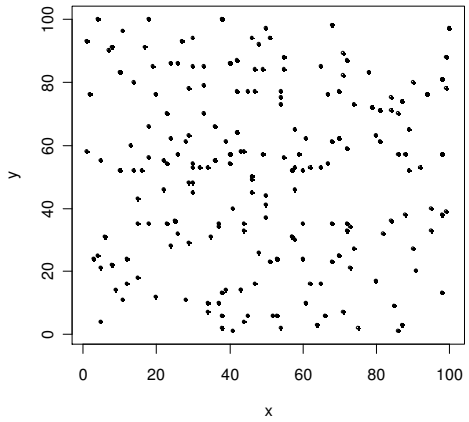
```
> split.screen(c(2,2))
[1] 1 2 3 4
> screen(1)
> plot(x1,y1)
> abline(modelo1)
> screen(2)
> plot(x2,y2)
> abline(modelo2)
> screen(3)
> plot(x3,y3)
> abline(modelo3)
> screen(4)
> plot(x4,y4)
> abline(modelo4)
```



3.6 EJERCICIOS RESUELTOS

●**ER 3-1** Generar 200 pares de números naturales tomados al azar entre 0 y 100 y representar la nube de puntos correspondiente. Calcular el coeficiente de correlación.

```
> valores<-c(1:100)
> x<-c(sample(valores,200,replace=T))
> y<-c(sample(valores,200,replace=T))
> plot(x,y,pch=20)
```



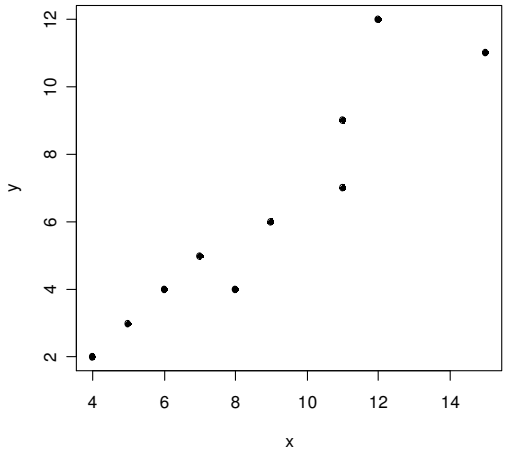
> cor(x,y)
[1] -0.03159351
> #Como era de esperar se obtiene un coeficiente de correlación próximo a cero

●**ER 3-2** La tabla siguiente presenta los valores que las variables X e Y toman en 10 individuos:

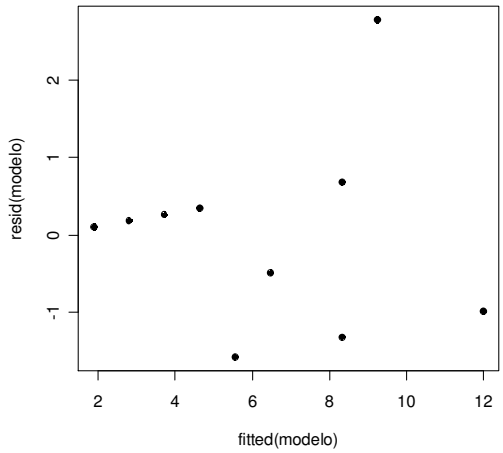
X	4	5	15	11	9	8	12	6	7	11
Y	2	3	11	7	6	4	12	4	5	9

Obtener: 1º) Diagrama de dispersión. 2º) Recta de regresión de Y sobre X. 3º) Diagrama de residuos frente a valores ajustados.

> x<-c(4,5,15,11,9,8,12,6,7,11)
> y<-c(2,3,11,7,6,4,12,4,5,9)
> plot(x,y,pch=16)

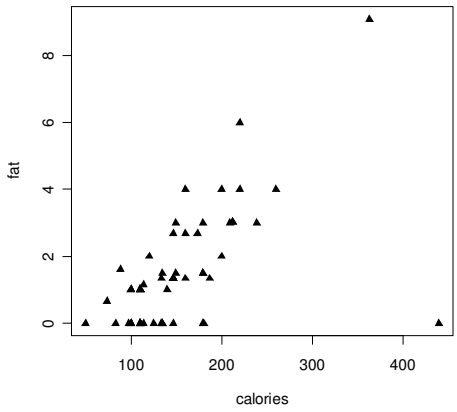


```
> lm(y~x)
Call:
lm(formula = y ~ x)
Coefficients:
(Intercept)          x
   -1.7639       0.9164
> #La recta de regresión de Y sobre X es: Y = -1.7639 + 0.9164*X

> modelo<-lm(y~x)
> plot(fitted(modelo),resid(modelo),pch=16)
```



●**ER 3-3** Utilizar los datos **UScereal** del paquete **MASS** para calcular la recta de regresión de la variable cuantitativa *grasas* (**fat**) sobre la variable cuantitativa *calorías* (**calories**). Repetir la regresión tras eliminar el punto que visualmente se percibe como atípico.

```
> library(MASS)
> data(UScereal)
> attach(UScereal)
> names(UScereal)
 [1] "mfr"       "calories"  "protein"   "fat"       "sodium"    "fibre"
 [7] "carbo"     "sugars"    "shelf"     "potassium" "vitamins"
> plot(calories,fat,pch=17)
```



> #En este gráfico se observa un valor atípico en el ángulo inferior derecho. Para detectarlo utilizamos la instrucción identify(), pulsamos Enter y después accionamos el botón izquierdo del ratón al colocarlo sobre el punto en el gráfico. Así se obtiene el índice (número de orden) que corresponde al punto buscado.

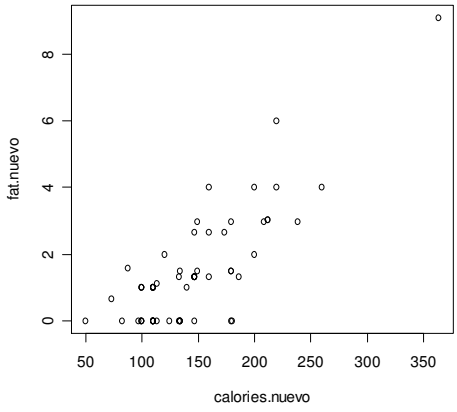
> identify(calories,fat,n=1)

[1] 31

> calories.nuevo<-calories[-31] #Eliminamos el dato atípico

> fat.nuevo<-fat[-31] #Eliminamos el dato atípico

> plot(calories.nuevo,fat.nuevo)



> lm(fat~calories)

Call:

lm(formula = fat ~ calories)

Coefficients:

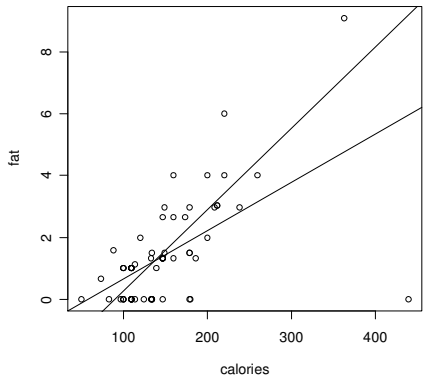
(Intercept) calories

-0.90472 0.01558

> lm(fat.nuevo~calories.nuevo)

```
Call:
lm(formula = fat.nuevo ~ calories.nuevo)
Coefficients:
  (Intercept)  calories.nuevo
     -2.36711         0.02631
```

> plot(calories,fat)
> abline(-0.90472,0.01558)
> abline(-2.36711,0.02631)



> #Como se ve en este último gráfico las rectas de regresión con o sin el punto atípico varían considerablemente

●**ER 3-4** Con los mismos datos del ejercicio anterior (**Uscereal** del paquete **MASS**) dibujar en un mismo gráfico la recta de regresión de **fat** sobre **calories** y la recta de regresión de **calories** sobre **fat**.

> library(MASS)
> data(UScereal)
> attach(UScereal)
> modelo1<-lm(fat~calories)
> modelo2<-lm(calories~fat)
> modelo1

```
Call:
lm(formula = fat ~ calories)
Coefficients:
(Intercept)     calories
   -0.90472      0.01558
```

> modelo2

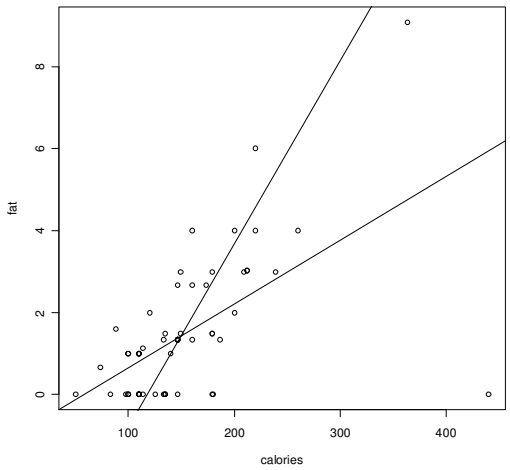
```
Call:
lm(formula = calories ~ fat)
```

Coefficients:
(Intercept) fat
117.60 22.36
> **#Por tanto, la RR del modelo2 es: calories=117.60+22.36*fat. Pero como debemos situar esta recta en el mismo sistema de ejes X(calories), Y(fat) del modelo1, debemos despejar fat en función de calories**

> -117.5988/22.36102;1/22.36102
[1] -5.259098
[1] 0.04472068

> #RR de fat sobre calories: fat=-0.90472+0.01558*calories
> #RR de calories sobre fat: fat=-5.259098+0.04472068*calories

> plot(fat~calories)
> abline(modelo1) #Ahora dibujamos la recta de regresión de fat sobre calories
> abline(-5.259098,0.04472068) #Ahora dibujamos la recta de regresión de calories sobre fat

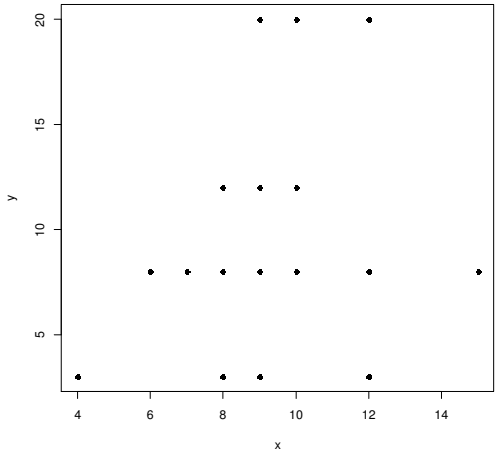


●**ER 3-5** En un estudio sobre la uso de una impresora se midieron en 30 ocasiones los minutos transcurridos entre las sucesivas utilizaciones (X) y el número de páginas impresas (Y), obteniéndose los siguientes resultados:

X	9,9,4,6,8,9,7,6,9,9,9,8,8,9,8,9,9,9,10,9,15,10,12,12,10,10,12,10,10,12
Y	3,8,3,8,3,8,8,8,3,8,12,12,8,8,8,12,12,20,8,20,8,8,20,8,8,12,8,20,20,3

Se pide: a) Distribución de frecuencias conjunta. b) Frecuencias marginales. c) Dibujar el diagrama de dispersión.

```
> x<-   c(9,9,4,6,8,9,7,6,9,9,9,8,8,9,8,9,9,9,10,9,15,10,12,
12,10,10,12,10,10,12)
> y<-   c(3,8,3,8,3,8,8,8,3,8,12,12,8,8,8,12,12,20,8,20,8,8,
20,8,8,12,8,20,20,3)

> dist.conj<-table(x,y)
> dist.conj
          y
x  3 8 12 20
4  1 0  0  0
6  0 2  0  0
7  0 1  0  0
8  1 2  1  0
9  2 4  3  2
10 0 3  1  2
12 1 2  0  1
15 0 1  0  0

> margin.table(dist.conj,1)
x
4  6  7  8  9 10 12 15
1  2  1  4 11  6  4  1
> margin.table(dist.conj,2)
y
3  8 12 20
5 15  5  5

> plot(x,y,pch=16)
```



3.7 EJERCICIOS PROPUESTOS

●**EP 3-1** Determinar la recta de regresión de Y sobre X a partir de las observaciones siguientes:

X	110	93	175	105	245	62	180	205
Y	21	23	19	18	14	24	15	10

●**EP 3-2** En un determinado grupo de familias se han medido las variables P (precio de la vivienda habitual) y G (gasto anual en mejoras de la vivienda) obteniéndose los datos siguientes:

P	300	295	590	450	505	350	365	255
G	1.2	2.3	6.4	5.1	4.4	2.8	0.8	3.1

Dibujar el diagrama de dispersión y calcular el coeficiente de correlación entre las dos variables.

●**EP 3-3** Instalar y cargar el paquete **HSAUR**. Elegir el marco de datos **waves** y realizar un diagrama de dispersión de las dos variables que aparecen. Obtener las dos rectas de regresión correspondientes.

●**EP 3-4** Cargar el paquete **ISwR** y leer el marco de datos **secher**. Se pide: 1º) Diagrama de dispersión entre las variables **bpd** y **ad**. 2º) Covarianza y coeficiente de correlación lineal entre las variables **bwt** y **bpd**. 3º) Recta de regresión de **bwt** respecto de **bpd**. 4º) Diagrama de residuos. 5º) Diagrama de residuos frente a valores ajustados.

●**EP 3-5** Instalar y cargar el paquete **MASS**. Elegir las variables **Wr.Hnd** y **Height**, correspondientes a los varones, del marco de datos **survey** y realizar un gráfico de dispersión. Obtener la recta de regresión de la primera variable sobre la segunda. Generar el correspondiente gráfico de residuos.

Capítulo 4

DISTRIBUCIONES DE PROBABILIDAD DISCRETAS

4.1 DISTRIBUCIONES DISCRETAS

En *R* es posible calcular probabilidades para las principales variables aleatorias discretas. Los nombres reservados a algunas de esas distribuciones son:

- Binomial: **binom**
- Hipergeométrica: **hyper**
- Poisson: **pois**
- Binomial negativa: **nbinom**
- Geométrica: **geom.**

Los nombres anteriores, sin embargo, no son sentencias de *R* que produzcan una salida válida. Es necesario anteponerles los prefijos "**d**", para la función de masa o función de probabilidad,

"**p**" para la función de distribución acumulada, "**r**" para generar valores aleatorios y "**q**" para la función cuantil. Veamos algunos ejemplos.

Calcular la probabilidad de que una variable aleatoria binomial de parámetros **n=10, p=0.3** tome el valor 4:

```
> dbinom(4,size=10,prob=0.3)
 [1] 0.2001209
```

Se puede simplificar la orden anterior:

```
> dbinom(4,10,0.3)
 [1] 0.2001209
```

Este valor también se podría calcular con la fórmula de la función de masa de una variable aleatoria binomial de parámetros (*n*,*p*):

$$\binom{n}{k} p^k (1-p)^{n-k}$$

En *R* la fórmula anterior se expresaría así:

```
> choose(10,4)*0.3^4*(1-0.3)^6
[1] 0.2001209
```

La probabilidad acumulada hasta el valor 4, P(X≤4), de una variable aleatoria B(10,0.3) es

```
> pbinom(4,10,0.3)
[1] 0.8497317
```

La probabilidad de que tome el valor 10 una variable aleatoria de Poisson de parámetro **λ=3.52** y la probabilidad acumulada en ese valor son

```
> dpois(10,lambda=3.52)
[1] 0.002382029
> dpois(10,3.52)
[1] 0.002382029
> ppois(10,3.52)
[1] 0.998933
```

Generar 10 valores aleatorios de una distribución de Poisson de parámetro 3,52:

```
> rpois(10,3.52)
[1] 4 3 3 3 4 5 4 1 2 1
```

4.2 CÓMO SIMULAR EN *R* EL LANZAMIENTO DE UN DADO

En primer lugar vamos a simular el lanzamiento de un dado una vez. Para ello utilizamos la función **sample()**. Mediante esta función se escogen al azar un número de elementos de tamaño especificado entre todos los elementos de un cierto vector. Por ejemplo, podemos usarla para escoger un número al azar entre los naturales del 1 al 6 (lanzamiento de un dado una vez).

```
> dado<-1:6
> sample(dado,1)
[1] 5
```

Para simular el lanzamiento más de una vez, por ejemplo 10, evidentemente debemos indicar la opción con reemplazamiento (por defecto extrae sin reemplazamiento):

```
> sample(dado,10)
Error en sample.int(length(x), size, replace, prob) :
  cannot take a sample larger than the population          when 'replace = FALSE'
> sample(dado,10,replace=T)
      [1] 1 5 3 6 6 4 2 3 2 2
```

Si quisiéramos simular el lanzamiento de un dado cargado, en el que, por ejemplo, las probabilidades de los valores 1 a 5 son 0.1 y la de 6 es 0.5, hacemos:

```
> prdadocarg<-c(0.1,0.1,0.1,0.1,0.1,0.5)
> sample(dado,10,replace=T,prob=prdadocarg)
 [1] 5 2 6 4 6 6 5 3 6 2
```

4.3 FUNCIÓN DE MASA O PROBABILIDAD

Ejemplo 4-1 Simular 1000 veces el experimento consistente en lanzar 5 dados. Considérese la variable aleatoria *número de unos y doses obtenidos entre los 5 dados.* Hacer un gráfico de frecuencias relativas y compararlo con la función de probabilidad de una variable aleatoria B(5,2/6).

Para simular el lanzamiento de cinco dados 1000 veces, lo que hacemos es generar una matriz de 1000 filas por 5 columnas (5000 elementos) del siguiente modo:

```
>lanzamientos<-matrix(
+sample(dado,5000,replace=T),nrow=1000,ncol=5)

> #Por ejemplo, los resultados correspondientes a las diez primeras filas son los
que se muestran a continuación
```

```
> lanzamientos[1:10,]
      [,1] [,2] [,3] [,4] [,5]
 [1,]    3    1    3    3    2
 [2,]    5    6    2    5    6
 [3,]    3    6    3    4    2
 ....
[10,]    1    1    1    4    5
```

Seguidamente debemos distinguir los resultados en los que se ha obtenido uno o dos (1) de aquellos en los que se ha obtenido tres, cuatro, cinco o seis (0). Para ello creamos la siguiente función:

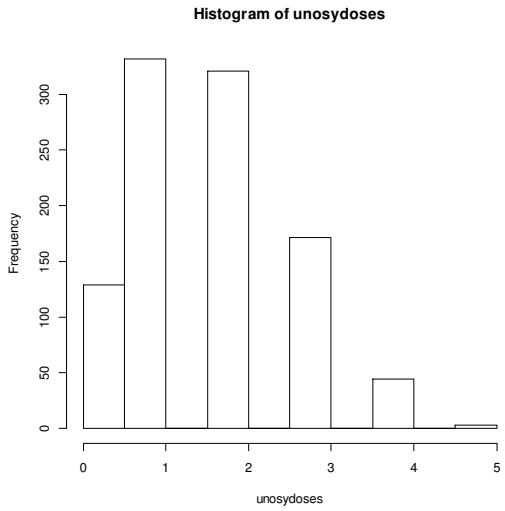
```
> f<-function(x) if(x==1|x==2) 1 else 0
```

Ahora utilizaremos una orden muy práctica, **apply()**, que evita la realización de bucles, pues permite aplicar directamente una función a los elementos de una matriz. Funciones similares son **lapply()**, **sapply()** y **tapply()**. Estudiaremos con más detalle esta familia de funciones en el capítulo 9.

```
> LANZ<-apply(lanzamientos,c(1,2),f)
> #Escribimos la opción c(1,2) para indicarle al programa que la función f
debe ser aplicada a filas (1) y columnas (2)

> LANZ[1:10,]
      [,1] [,2] [,3] [,4] [,5]
 [1,]    0    1    0    0    1
 [2,]    0    0    1    0    0
 [3,]    0    0    0    0    1
....
 [10,]    1    1    1    0    0

> unosydoses<-rowSums(LANZ) #Con la función rowSums() se obtiene un vector
cuyos elementos son la suma de cada una de las filas de la matriz LANZ
> unosydoses[1:10]
 [1] 2 1 1 2 0 3 2 2 1 3
```

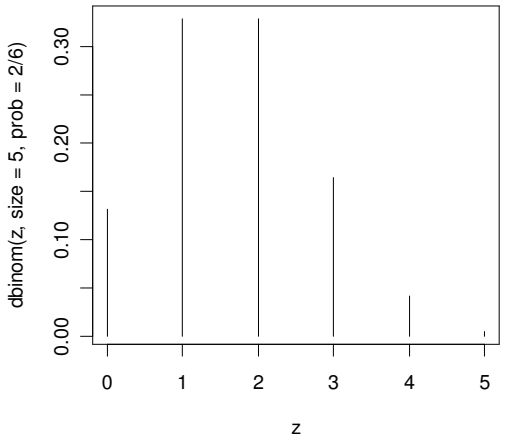
Construimos un histograma con los valores anteriores:

```
> hist(unosydoses)
```



Vamos a dibujar ahora la función de masa de una distribución B(5,2/6), que es la distribución de la variable aleatoria en estudio, y a compararla con el histograma obtenido. Si queremos conservar el gráfico anterior para que esté disponible más adelante podemos abrir una nueva ventana gráfica con la instrucción **windows()**:

```
> windows()
> z<-0:5 #Estos son los valores que puede tomar una variable aleatoria B(5,2/6)
> plot(z,dbinom(z,size=5,prob=1/3),type="h")
```

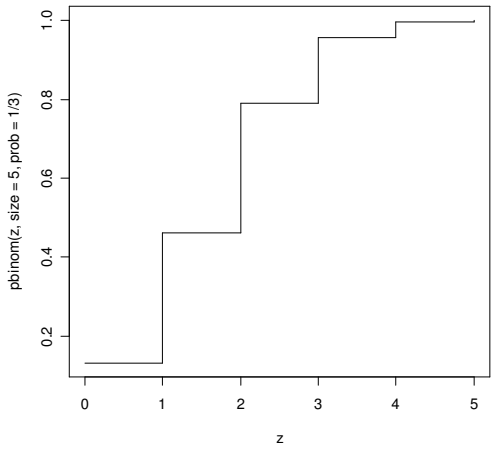


Los dos gráficos son parecidos, aunque no iguales, como ocurre cuando se simula una variable aleatoria un determinado número de veces, en este caso 1000.

4.4 FUNCIÓN DE DISTRIBUCIÓN

Utilizando la función **pbinom()** podemos dibujar la función de distribución acumulada de la variable aleatoria que representa el número de unos y doses que aparecen al lanzar cinco dados. Utilizamos la opción **type="s"** para construir la función en forma de escalera.

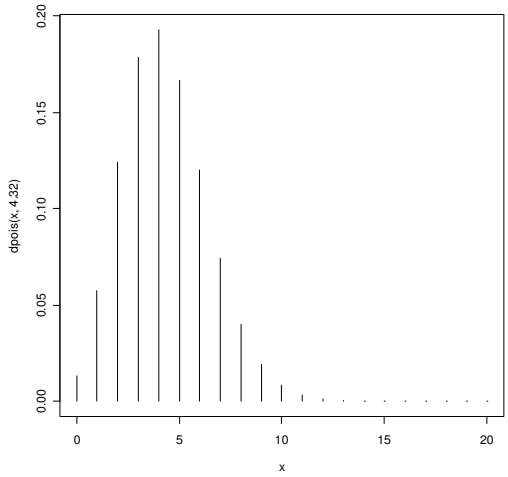
```
> plot(z,pbinom(z,5,1/3),type="s")
```



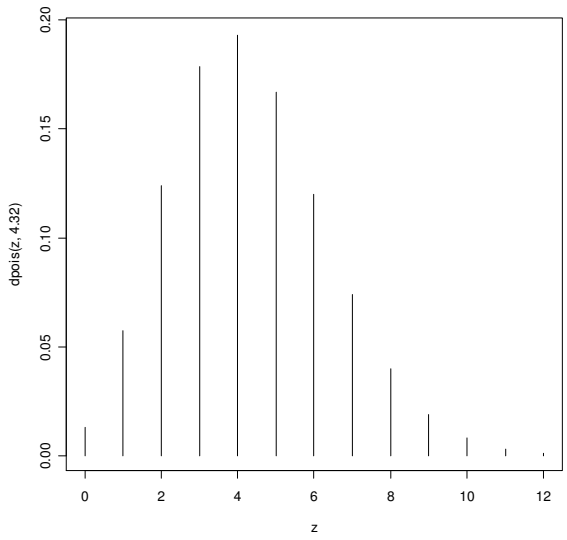
4.5 EJERCICIOS RESUELTOS

●**ER 4-1** Representar gráficamente la función de masa y la función de distribución de una variable aleatoria que sigue una distribución de Poisson de parámetro 4,32.

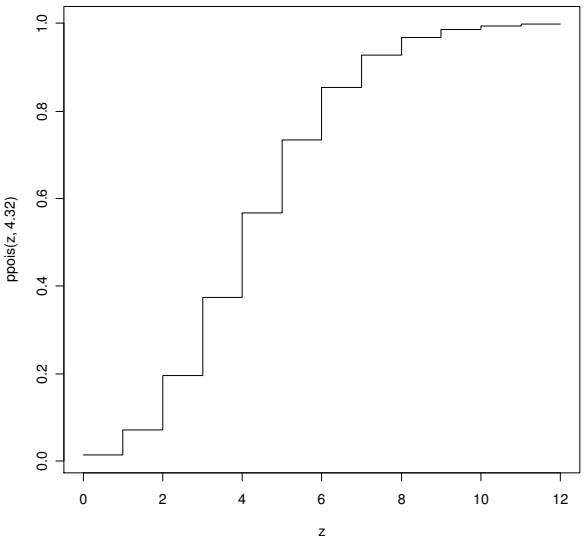
```
> x<-0:20
> plot(x,dpois(x,4.32),type="h")
```



> #Vemos que las probabilidades, a partir de 12 aproximadamente, son muy pequeñas por lo que hacemos un nuevo gráfico
> x<-0:12
> plot(x,dpois(x,4.32),type="h")

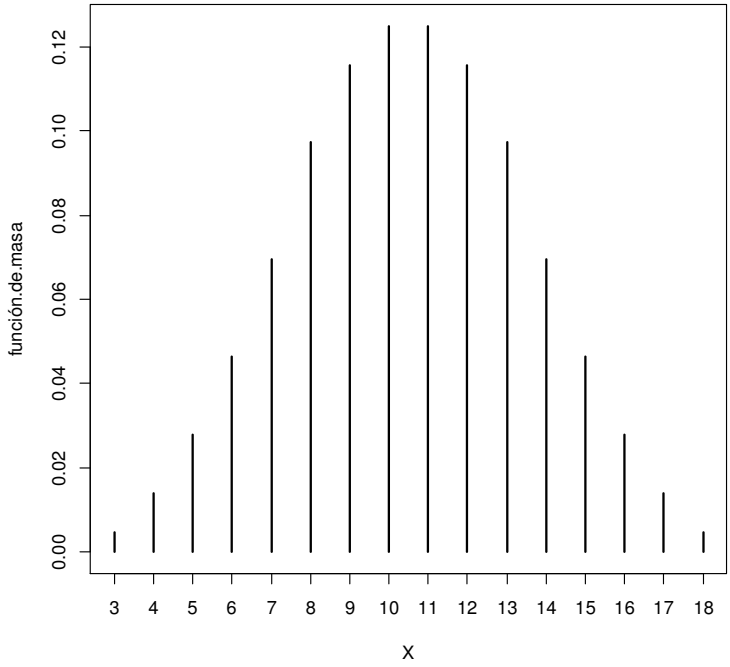


> plot(x,ppois(x,4.32),type="s")



●**ER 4-2** En el experimento aleatorio consistente en lanzar un dado tres veces se define la variable aleatoria X, *suma de los puntos obtenidos en los tres lanzamientos*. Obtener la función de probabilidad de X. Hallar la media, varianza y desviación típica de esta variable aleatoria.

```
> #Vamos a crear, en primer lugar, un marco de datos con todos los resultados posibles del experimento
> dado<-1:6
> a<-expand.grid(dado,dado,dado)
> a
   Var1 Var2 Var3
1     1    1    1
2     2    1    1
3     3    1    1
...
214   4    6    6
215   5    6    6
216   6    6    6

> X<-rowSums(a) #Sumamos las filas y a continuación calculamos la frecuencia de cada una de las sumas (valores entre 3 y 18)
> table(X)
          X
 3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18
 1  3  6 10 15 21 25 27 27 25 21 15 10  6  3  1

> función.de.masa<-table(X)/length(X)
> función.de.masa
X
        3         4         5         6         7         8         9
0.00462963 0.01388889 0.02777778 0.04629630 0.06944444 0.09722222 0.11574074
       10        11        12        13        14        15        16
0.12500000 0.12500000 0.11574074 0.09722222 0.06944444 0.04629630 0.02777778
       17        18
0.01388889 0.00462963

> #Se comprueba a continuación que, efectivamente, se trata de una función de masa
> sum(función.de.masa)
[1] 1
> plot(función.de.masa)
```

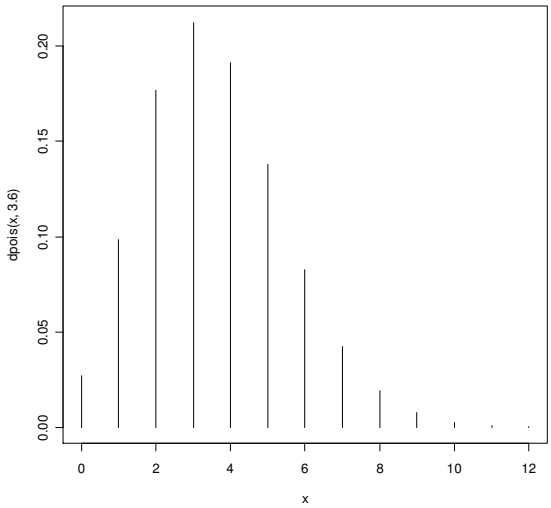


> #Calculamos la media, varianza y desviación típica de la variable aleatoria X

> mean(X);(215/216)*var(X);sqrt((215/216)*var(X))
[1] 10.5
[1] 8.75
[1] 2.95804

●**ER 4-3** En la inspección de una tubería se detectaron 3,6 defectos por metro. Obtener y representar gráficamente la función de masa de la variable aleatoria que cuenta el número de defectos por metro. Calcular la probabilidad de que en un tramo de 1 m de longitud se encuentren más de dos defectos.

> #La variable aleatoria X = "nº de defectos en un tramo de 1 m de longitud" sigue una distribución de Poisson de parámetro 3.6
> x<-0:12
> plot(x,dpois(x,3.6),type="h")



> #Calculamos P(X>2)=1-P(X≤2)
> 1-ppois(2,3.6)
[1] 0. 6972532

●**ER 4-4** De un lote de 100 piezas, se sabe que 5 de ellas no cumplen con los requisitos de aceptación. Si se toman al azar 10, ¿cuál es la probabilidad de encontrar uno o menos elementos defectuosos en la muestra? Resolverlo para dos casos: a) el muestreo se hace sin reemplazamiento, b) el muestreo se hace con reemplazamiento.

> #Calcularemos las probabilidades, en ambos casos, mediante la función de masa y mediante la función de distribución

> #Sin reemplazamiento: X-->H(100,10,5/100), siendo los parámetros de la distribución hipergeométrica, respectivamente: *N* (nº total de piezas), *n* (nº de extracciones) y *p* (probabilidad inicial de pieza defectuosa)
>choose(5,0)*choose(95,10)/choose(100,10)+choose(5,1)*choose(95,9)/choose(100,10) #Aplicando directamente la fórmula correspondiente
[1] 0.9231433
> dhyper(0,5,95,10)+dhyper(1,5,95,10) #Los parámetros de la distribución hipergeométrica que hay que introducir, siguiendo la sintaxis de *R* son: nº de piezas defectuosas (5), nº de piezas buenas (95) y nº de extracciones (10)
[1] 0.9231433
> phyper(1,5,95,10)
[1] 0.9231433

> #Con reemplazamiento: Y --> B(10,5/100)
> choose(10,0)*0.05^0*0.95^10+
choose(10,1)*0.05^1*0.95^9 #Aplicando directamente la fórmula correspondiente
[1] 0.9138616
> dbinom(0,10,0.05)+dbinom(1,10,0.05)
[1] 0.9138616
> pbinom(1,10,5/100)
[1] 0.9138616

●**ER 4-5** La central telefónica de una empresa dispone de 5 líneas, siendo 3 las que están ocupadas de media en hora punta. Se pide: a) ¿Cuál es la probabilidad de que en una hora punta estén todas las líneas ocupadas? b) ¿Qué número de líneas sería el adecuado para garantizar que la probabilidad de que todas las líneas estén ocupadas en la hora punta sea menor del 1%?

> #Sea la variable aleatoria X="nº de líneas ocupadas en hora punta entre 5 líneas". Como np=3, entonces 5p=3 y p=3/5=0.6 Por tanto, la v.a. X sigue una distribución B(5,0.6)

> #Lo que se pide en el apartado a) es P(X=5) en una B(5,0.6)
> dbinom(5,5,0.6)
[1] 0.07776

> #En el apartado b) se trata de determinar el valor de n tal que P(X=n)<=0.01 para una distribución B(n,3/n). Lo resolvemos por tanteo.
> dbinom(6,6,3/6)
[1] 0.015625
> dbinom(7,7,3/7)
[1] 0.002655599

> #La solución es n=7 líneas

4.6 EJERCICIOS PROPUESTOS

●**EP 4-1** Simular el lanzamiento de una moneda no cargada, de dos maneras diferentes: usando **rbinom()** y usando **sample()**.

●**EP 4-2** Considérese una moneda cargada cuya probabilidad de cara es p=0.7. Simular el lanzamiento de la moneda 1000 veces y obtener un histograma de las 1000 realizaciones de la variable aleatoria "*número de caras obtenidas al lanzar la moneda una vez*". Comparar ese resultado con la correspondiente función de masa. (Distribución binomial).

●**EP 4-3** El número medio de automóviles que llegan a una gasolinera es 210 por hora. Si la instalación puede atender como máximo diez automóviles por minuto, determinar la probabilidad de que, en un minuto dado, lleguen más automóviles de los que pueden ser atendidos. (Distribución de Poisson).

●**EP 4-4** El temario de una oposición consta de 50 temas. Un opositor estudia en profundidad 30 de ellos. Calcular la probabilidad de que conozca, al menos, tres temas entre 5 elegidos al azar. (Distribución hipergeométrica).

●**EP 4-5** Cada ítem de un test tiene 5 posibles respuestas, de las que únicamente una es correcta. Suponiendo que se contesta completamente al azar, calcular la probabilidad de que en la quinta pregunta contestada se consiga el primer acierto. (Distribución geométrica).

Capítulo 5

DISTRIBUCIONES DE PROBABILIDAD CONTINUAS

5.1 DISTRIBUCIONES CONTINUAS

En *R* es posible calcular probabilidades para las principales variables aleatorias continuas. Los nombres reservados a las distribuciones continuas más importantes son:

- Uniforme: **unif**
- Exponencial: **exp**
- Normal: **norm**
- χ^2: **chisq**
- t de Student: **t**
- F de Snedecor: **f**

Las denominaciones anteriores, igual que para las distribuciones discretas, no son sentencias de *R* que produzcan una salida válida. Es necesario anteponerles los prefijos "**d**" para la función de densidad, "**p**" para la función de distribución acumulada, "**r**" para generar valores aleatorios y "**q**" para la función cuantil (inversa de la función de distribución).

Por ejemplo, si queremos conocer la ordenada de la función de densidad de una variable aleatoria N(-2,4) en una determinada abscisa, valor que por otro lado no tiene ninguna utilidad práctica desde el punto de vista de las probabilidades, haremos:

```
> dnorm(3,-2,4)
[1] 0.04566227
```

Este resultado lo podemos obtener así mismo mediante la función de densidad de la variable aleatoria normal:

```
> exp(-(1/2)*((3+2)/4)^2)/(4*sqrt(2*pi))
[1] 0.04566227
```

La probabilidad acumulada hasta el valor 3 en la distribución anterior es

```
> pnorm(3,-2,4)
[1] 0.8943502
```

Para calcular la probabilidad de obtener un valor entre 3 y 5 hacemos

```
> pnorm(5,-2,4)- pnorm(3,-2,4)
[1] 0.06559062
```

Generemos 4 valores al azar de la distribución U(-5,3)*:*

```
> runif(4,-5,3)
[1] -2.48284134 -2.43764603 -0.18214369 -0.02041526
```

La probabilidad de que X sea menor que 2 en la distribución anterior es

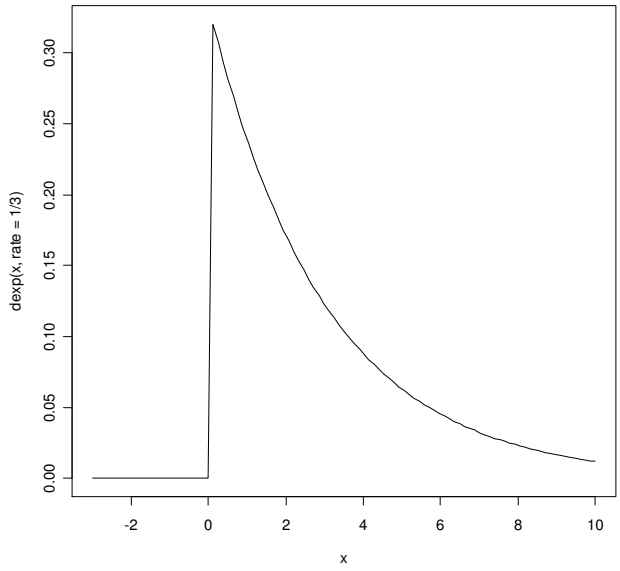
```
> punif(2,-5,3)
[1] 0.875
```

5.2 FUNCIÓN DE DENSIDAD

En el primer ejemplo vamos a dibujar la función de densidad de una variable aleatoria exponencial de parámetro 3. Posteriormente, veremos cómo se aproxima una distribución binomial por una normal.

Ejemplo 5-1 Dibujar la función de densidad de una variable aleatoria exponencial de media 3 (parámetro 1/3 en la sintaxis de *R*).

```
> #Una forma muy simple de dibujar la función de densidad es usando la función
curve()
> curve(dexp(x,rate=1/3),from=-3,to=10)
```

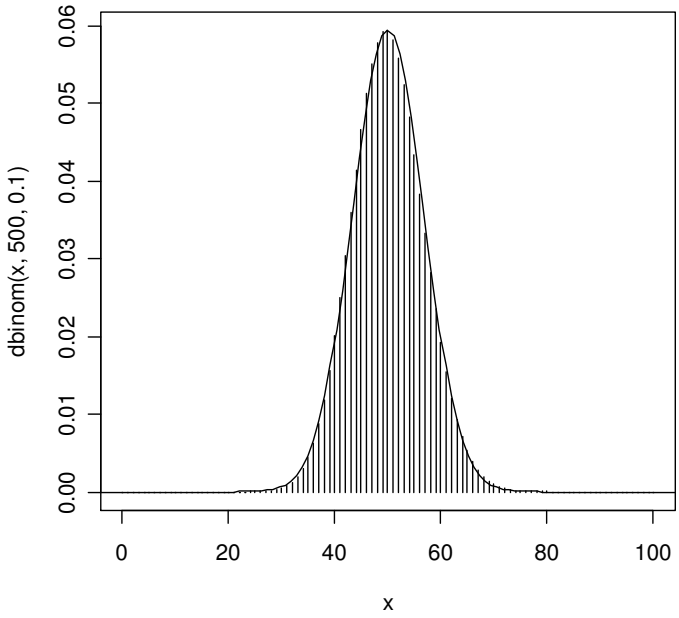


Ejemplo 5-2 La probabilidad de que un cierto componente electrónico falle a lo largo de un mes es 0,1. Si hay 500 componentes en funcionamiento, calcular la probabilidad de que en un mes: 1º) fallen como mucho 50 componentes, 2º) fallen entre 30 y 60, 3º) fallen más de 65. Resolver, en primer lugar, el problema utilizando la distribución binomial y después mediante la aproximación de la binomial a la normal. Justificar gráficamente tal aproximación.

```
> #Calculamos las probabilidades exactas mediante la distribución binomial
> pbinom(50,500,0.1)
[1] 0.5375688
> pbinom(60,500,0.1)-pbinom(29,500,0.1)
[1] 0.9376227
> 1-pbinom(65,500,0.1)
[1] 0.01269429
```

La distribución $B(n, p)$ se aproxima a la $N(np, \sqrt{np(1-p)})$, y la aproximación es buena si np>5 y n(1-p)>5. Aplicando este resultado, con las correcciones por continuidad correspondientes, obtendremos las probabilidades anteriores de manera aproximada.

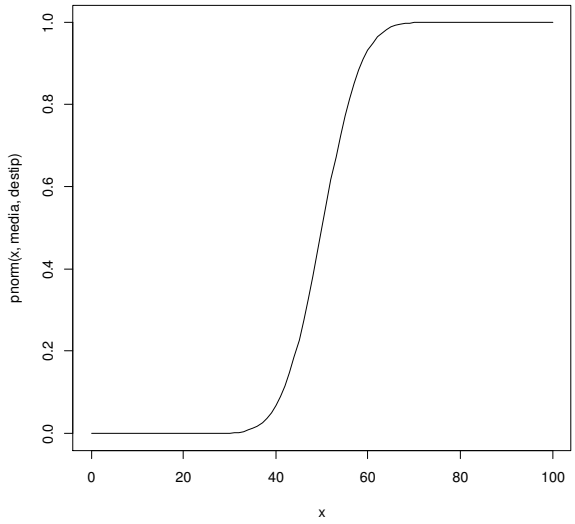
```
> n<-500;p<-0.1
> n*p;n*(1-p)
[1] 50
[1] 450
> media<-n*p;destip<-sqrt(n*p*(1-p))
> pnorm(50.5,media,destip)
[1] 0.5297079
> pnorm(60.5,media,destip)-pnorm(29.5,media,destip)
[1] 0.9401159
> 1-pnorm(65.5,media,destip)
[1] 0.01042738

> #Dibujamos la función de masa de la distribución B(500,0.1) y le superponemos la función de densidad de la N(media,destip)
> z<-0:100
> plot(z,dbinom(z,500,0.1),type="h")
> curve(dnorm(x,media,destip),add=T)
> #Notemos que en la sentencia anterior hemos utilizado la opción add=T (TRUE) para indicarle al programa que superponga este gráfico al anterior
```



5.3 FUNCIÓN DE DISTRIBUCIÓN

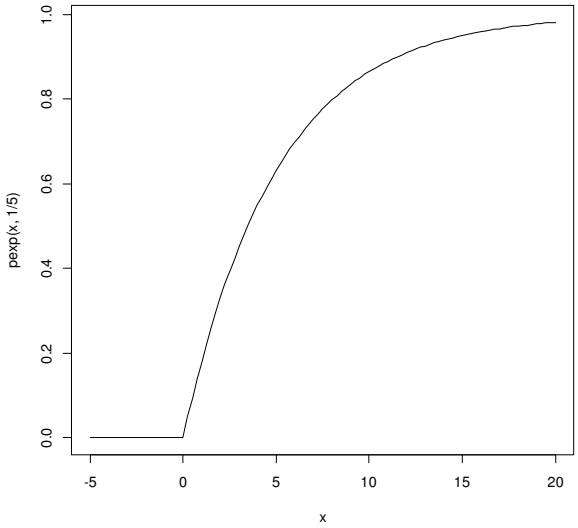
Vamos a dibujar ahora la función de distribución de la variable aleatoria normal del ejemplo anterior:

> curve(pnorm(x,media,destip),from=0,to=100)



La función de distribución de una variable aleatoria exponencial de media 5 sería, por ejemplo,

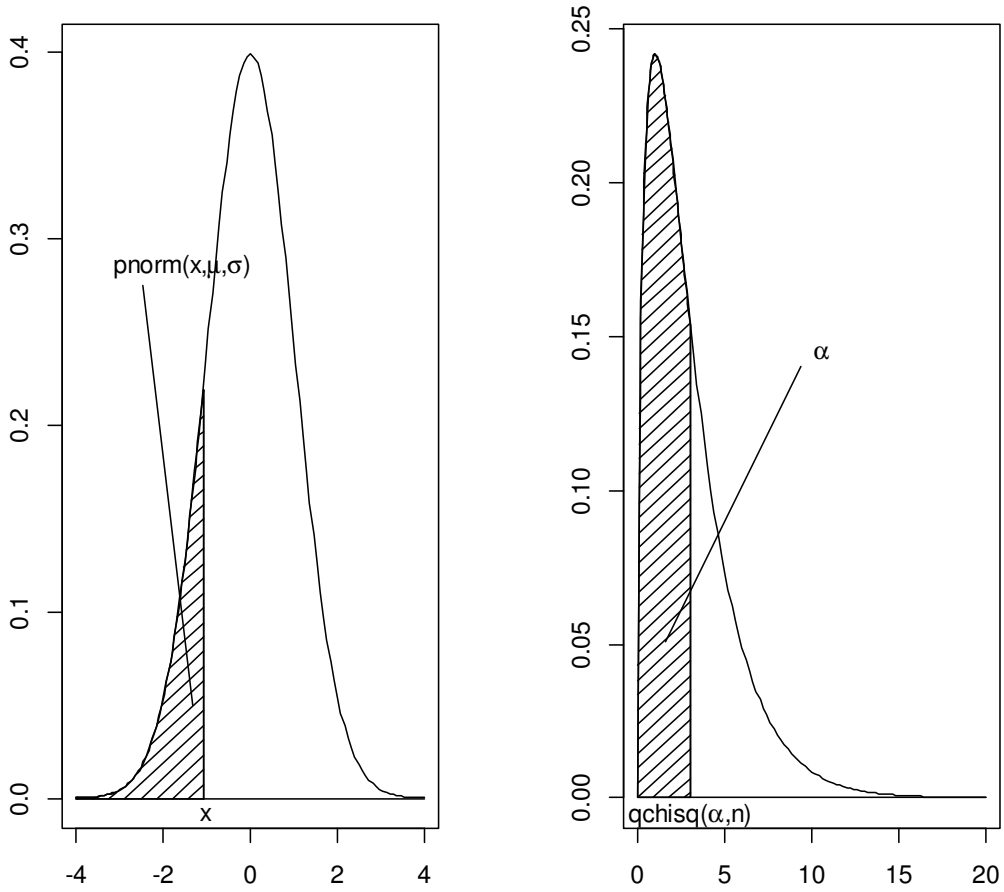
> curve(pexp(x,1/5),from=-5,to=20)



5.4 CÓMO UTILIZAR *R* COMO ALTERNATIVA A LAS TABLAS ESTADÍSTICAS

Como podemos deducir de lo visto en este capítulo y en el precedente, el programa *R* encierra en sí mismo unas completas tablas estadísticas que, además, resultan de muy fácil manejo.

En los gráficos siguientes se recuerda el significado geométrico que tienen la *función de distribución* (**pnombredeladistribución)** y la *función cuantil* (**qnombredeladistribución):**



La forma en que el programa *R* puede ser utilizado como alternativa a las tablas estadísticas clásicas se resume en el siguiente cuadro (solo se indican los escenarios de uso más corriente):

Distribución	***Parámetros***	***Función de masa o probabilidad***	***Función de distribución***	***Función cuantil***
Binomial	n, p	dbinom(x,n,p)	pbinom(x,n,p)	
Poisson	λ	dpois(x,λ)	ppois(x,λ)	
Normal	μ,σ		pnorm(x,μ,σ)	qnorm(α,μ,σ)
χ^2	n		pchisq(x,n)	qchisq(α,n)
t de Student	n		pt(x,n)	qt(α,n)
F de Snedecor	n, m		pf(x,n,m)	qf(α,n,m)

Veamos algunos ejemplos de aplicación:

> #Calcular P(X=2) si X --> Binomial(5,0.2)
> dbinom(2,5,0.2)
[1] 0.2048

> #Calcular P(X≤3) si X --> Poisson(2)
> ppois(3,2)
[1] 0.8571235

> #Calcular x sabiendo que P(X≤x)=0.238 y X --> N(-4,1)
> qnorm(0.238,-4,1)
[1] –4.712751

> #Calcular P(X≤34.2) si X --> χ^2(20)
> pchisq(34.2,20)
[1] 0.9751968

> #Calcular P(X≤2.14) si X --> t_{14}
> pt(2.14,14)
[1] 0.9747763

> #Calcular x si P(X≤x)=0.9 y X --> $F_{4;8}$
> qf(0.9,4,8)
[1] 2.806426

El valor **qf(0.9,4,8)** se puede calcular igualmente escribiendo **qf(0.1,4,8,lower.tail=F)**, indicándole al programa la probabilidad que deja a la derecha (0.1) el valor buscado en lugar de la que deja a su izquierda (0.9).

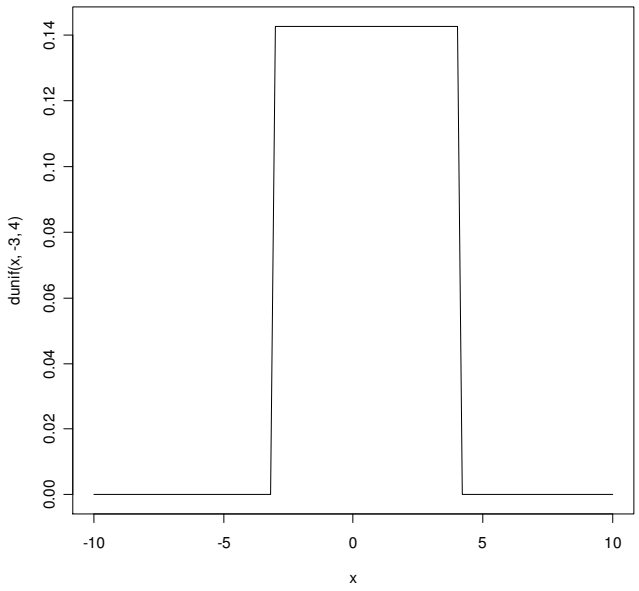
5.5 EJERCICIOS RESUELTOS

●**ER 5-1** Calcular: 1º) P(X<3) si X --> N(2,4). 2º) P(X>5) si X --> $\chi^2(8)$. 3º) $F_{3,5;0.05}$. 4º) P(X≤3) si X --> U[-1,7]. 5º) P(X>3) si X --> exp(5). 6º) $t_{5;0.01}$.

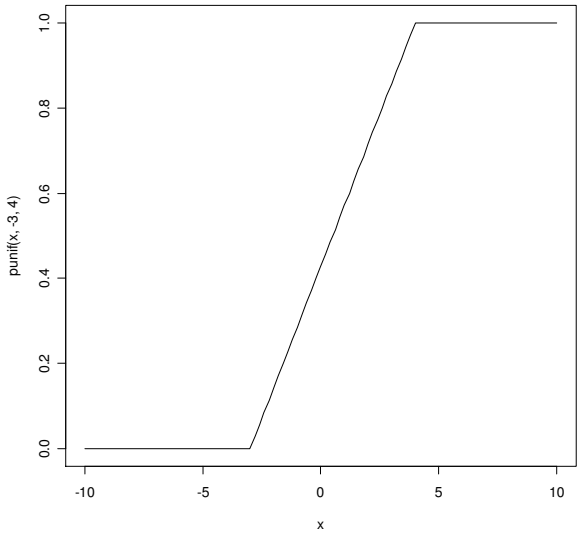
```
> pnorm(3,2,4)
[1] 0.5987063
> 1-pchisq(5,8)
[1] 0.7575761
> qf(0.95,3,5)
[1] 5.409451
> punif(3,-1,7)
[1] 0.5
> 1-pexp(3,1/5)
[1] 0.5488116
> qt(0.01,5,lower.tail=F)
[1] 3.36493
```

●**ER 5-2** Dibujar la función de densidad, y la función de distribución de una variable aleatoria uniforme continua de parámetros (-3,4).

```
> #Función de densidad de la variable aleatoria UC[-3,4]
> curve(dunif(x,-3,4),from=-10,to=10)
```



> #Función de distribución de la variable aleatoria UC[-3,4]
> curve(punif(x,-3,4),from=-10,to=10)

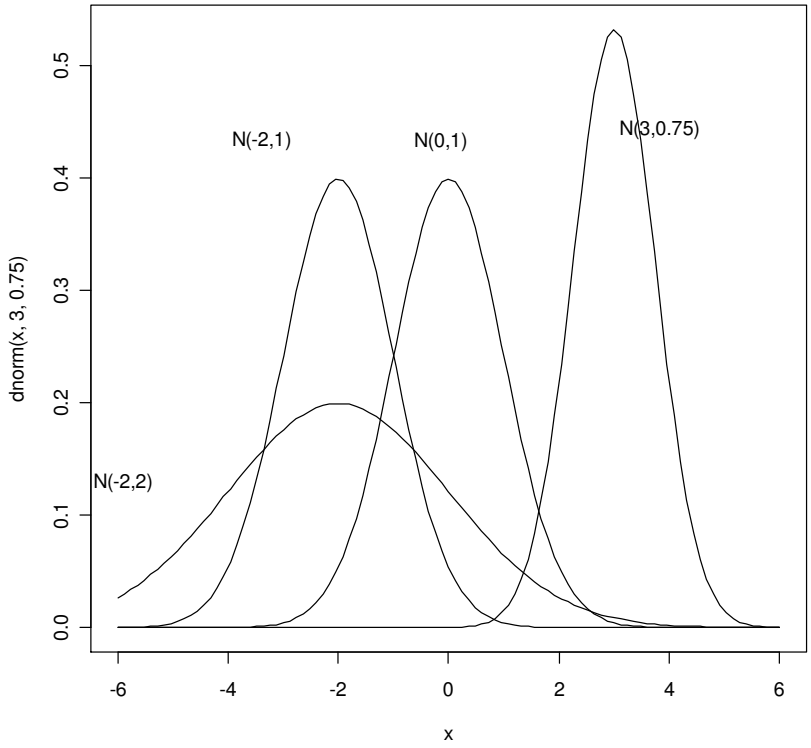


●**ER 5-3** Dibujar, superpuestas en un mismo gráfico, las funciones de densidad de las siguientes variables aleatorias normales: N(3,0.75), N(0,1), N(-2,1), y N(-2,2).

> curve(dnorm(x,3,0.75),from=-6,to=6)
> curve(dnorm(x,0,1),add=T)
> curve(dnorm(x,-2,1),add=T)
> curve(dnorm(x,-2,2),add=T)

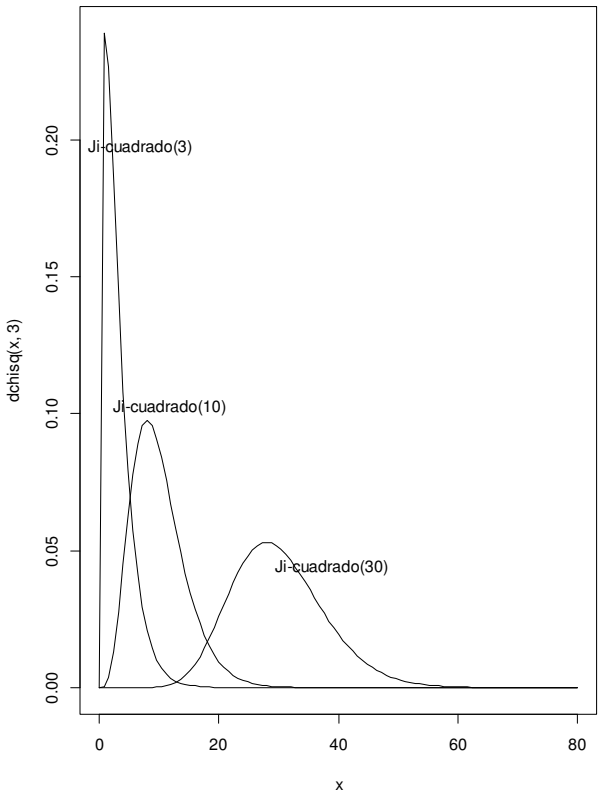
> #Ahora vamos a obtener las coordenadas de 4 puntos elegidos en el gráfico donde posteriormente situaremos los rótulos de las curvas. Para ello, una vez ejecutada la sentencia siguiente, nos colocamos con el ratón sobre los puntos elegidos y pulsamos el botón izquierdo
> a<-locator(n=4)
$x
[1] -4.4329751 -1.6721848 0.4601328 4.7023227
$y
[1] 0.1241017 0.4210175 0.4199178 0.5045938

> #Colocamos los cuatro rótulos sobre el gráfico en los puntos cuyas coordenadas (x,y) acabamos de obtener
>text(a$x[1],a$y[1],"N(-2,2)")
>text(a$x[2],a$y[2],"N(-2,1)")
>text(a$x[3],a$y[3],"N(0,1)")
>text(a$x[4],a$y[4],"N(3,0.75)")



●**ER 5-4** Dibujar, superpuestas en un mismo gráfico, las funciones de densidad de las siguientes variables aleatorias: $\chi^2(3)$, $\chi^2(10)$ y $\chi^2(30)$.

```
> curve(dchisq(x,3),from=0,to=80)
> curve(dchisq(x,10),add=T)
> curve(dchisq(x,30),add=T)

> a<-locator(n=3)
> text(a$x[1],a$y[1],"Ji-cuadrado(3)")
> text(a$x[2],a$y[2],"Ji-cuadrado(10)")
> text(a$x[3],a$y[3],"Ji-cuadrado(30)")
```

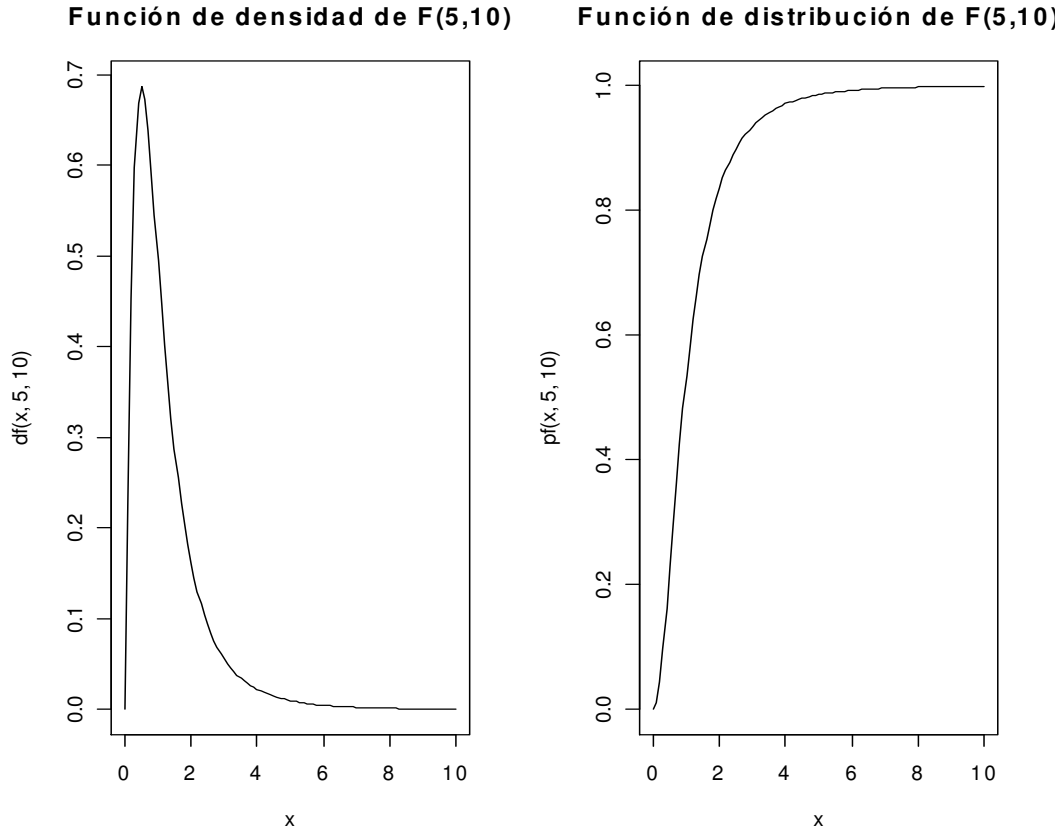


●**ER 5-5** Dibujar la función de densidad y la función de distribución de una variable aleatoria F(5,10).

> split.screen(c(1,2)) #Con esta sentencia dividimos la pantalla gráfica en dos zonas (dos zonas en una fila y dos columnas)
[1] 1 2

> screen(1) #Elegimos la zona izquierda para situar la función de densidad
> curve(df(x,5,10),from=0,to=10,main="Función de densidad de F(5,10)")

> screen(2) #Elegimos la zona derecha para situar la función de distribución
> curve(pf(x,5,10),from=0,to=10,main="Función de distribución de F(5,10)")



5.6 EJERCICIOS PROPUESTOS

●**EP 5-1** El peso medio de un grupo de jóvenes es 60 kg y la desviación típica 6 kg. Suponiendo que los pesos se distribuyen normalmente, se pide: a) Probabilidad de que un joven pese menos de 65 kg. b) Probabilidad de que un joven pese más de 57 kg.

●**EP 5-2** Dibujar, superpuestas en un mismo gráfico, las funciones de densidad de las variables aleatorias t_2 y N(0,1). Colocar los rótulos correspondientes.

●**EP 5-3** Sabiendo que X es una variable aleatoria uniforme continua de parámetros (-2,8), se pide: 1º) Dibujar la función de densidad. 2º) Dibujar la función de distribución. 3º) Generar muestras aleatorias de X, de tamaño arbitrariamente creciente, y obtener sucesivamente la media y la varianza correspondientes. Graficar esos valores.

●**EP 5-4** Se supone que el acceso de los usuarios a una red de ordenadores se puede modelar como un proceso de Poisson de media 25 accesos por hora. Calcular la probabilidad de que no haya accesos en 6 minutos. ¿Cuál es la probabilidad de que el tiempo que transcurre entre dos accesos consecutivos esté entre 2 y 3 minutos. (Recuérdese la relación entre las distribuciones de Poisson y exponencial en un proceso de Poisson).

●**EP 5-5** Considérese una variable aleatoria continua T con función de densidad

$$f(t)=\begin{cases} k(1+t^2)\ si\ 0<t\le 3 \\ 0 \qquad en\ otro\ caso \end{cases}$$

1º) Calcular la constante *k* mediante cálculos aproximados en *R* de la integral correspondiente. 2º) P(T>2). 3º) P(1<T<2).

Capítulo 6

ESTIMACIÓN POR PUNTO Y POR INTERVALO

6.1 INTRODUCCIÓN

Los conceptos que vamos a ir desarrollando a lo largo de este capítulo, en el que se abordarán las técnicas más básicas de la inferencia estadística, hacen referencia a los dos ejercicios siguientes. En ambos casos se supone que los datos provienen de poblaciones normales.

Ejemplo 6-1 Se midieron en 10 días tomados al azar los niveles de cloro del agua que sale de una planta de tratamiento, obteniéndose los valores siguientes: 2,2 – 1,9 – 1,7 – 1,6 – 1,7 – 1,8 – 1,7 – 1,9 – 2,0 – 2,0. Se pide: a) Dar una estimación puntual de la media y de la varianza de la variable aleatoria que representa el nivel de cloro. b) Intervalo de confianza al 95% para la media y la varianza. c) Ídem al 99%.

Ejemplo 6-2 Se quiere saber si existen diferencias significativas en la facturación de dos tiendas de joyería de una misma cadena. Para ello se eligieron al azar 11 días en los que se contabilizaron las ventas en la joyería A y otros 10 días en la joyería B. Los datos obtenidos fueron:

Ventas A	1320	1495	990	1250	12900	1900	1500	1100	1250	1100	1930
Ventas B	1110	1405	985	1290	1300	1705	1200	1105	1150	1210	

Obtener las estimaciones por punto y por intervalo de las medias, varianzas, diferencia de medias y cociente de varianzas.

6.2 ESTIMACIÓN DE LA MEDIA

Vamos a obtener en primer lugar la estimación puntual de la media, o sea la media muestral, para el primer ejemplo.

```
> #Ejemplo 6-1
> nivclor<-c(2.2,1.9,1.7,1.6,1.7,1.8,1.7,1.9,2,2)
> nivclor
 [1] 2.2 1.9 1.7 1.6 1.7 1.8 1.7 1.9 2.0 2.2
> mean(nivclor)
[1] 1.85
```

Para el segundo ejemplo estimaremos las medias de las variables **VentasA** y **VentasB**. Se supone que los datos de este ejercicio están en el archivo "Ejemplo 6-2.txt", cuya ruta de acceso es: "C:/Ejemplo 6-2.txt".

```
> #Ejemplo 6-2. Leemos el archivo de datos
> datos<-read.table("C:/Ejemplo 6-2.txt",header=T)
Error en scan(file, what, nmax, sep, dec, quote, skip, nlines, na.strings,  :
la línea 11 no tiene 2 elementos
```

Como se observa aparece un mensaje de error porque **"VentasB"** tiene un dato menos que **"VentasA"**. En estos casos lo mejor es utilizar la función **read.delim()**, que asigna el valor NA (*not available*) a los datos que faltan:

```
> datos<- read.delim("C:/Ejemplo 6-2.txt")
> #A diferencia de read.table(), header=T es la opción por defecto en
read.delim()
> datos
 VentasA VentasB
1   1320 1110
2   1495 1405
```

....
10 1100 1210
11 1930   NA
> attach(datos)

> VentasA
[1] 1320 1495   990 1250 12900 1900 1500 1100 1250 1100 1930
> VentasB
 [1] 1110 1405 985 1290 1300 1705 1200 1105 1150 1210   NA
> mean(VentasA,na.rm=T);mean(VentasB,na.rm=T) #Con la opción na.rm=T no tenemos en cuenta los valores perdidos (NA) que pudiera haber
[1] 2430.455
[1] 1246

> #Otra alternativa consiste en eliminar esos valores, tras haberlos detectado previamente
> which(is.na(VentasB))
[1] 11
> #El elemento que ocupa el lugar 11 en VentasB es NA; lo eliminamos
> VentasB.nuevo<-VentasB[-11]
> mean(VentasA);mean(VentasB.nuevo)
[1] 2430.455
[1] 1246

6.3 ESTIMACIÓN DE LA VARIANZA Y DE LA DESVIACIÓN TÍPICA

Los estimadores centrados de la varianza y de la desviación típica son, respectivamente, la cuasivarianza y la cuasidesviación típica muestrales. Calculemos esos valores para los ejemplos 6-1 y 6-2.

> var(nivclor)
[1] 0.03388889
> sd(nivclor)
[1] 0.1840894
> var(VentasA);var(VentasB,na.rm=T)
[1] 12151922
[1] 39993.33
> sd(VentasA);sd(VentasB,na.rm=T)
[1] 3485.961
[1] 199.9833

Como ya se ha indicado en el capítulo 2, dedicado a la estadística descriptiva de una variable, mediante la función **var()** *(variance)* se obtiene la cuasivarianza muestral y no la varianza muestral. Así mismo, la función **sd()** (*standard deviation*) proporciona la cuasidesviación típica

muestral en lugar de la desviación típica muestral. Como es lógico, se obtienen los mismos valores anteriores si hacemos

```
> sqrt(var(nivclor))
[1] 0.1840894
> sqrt(var(VentasA));sqrt(var(VentasB,na.rm=T))
[1] 3485.961
[1] 199.9833
```

6.4 INTERVALO DE CONFIANZA PARA LA MEDIA

Para construir un intervalo de confianza para la media del ejemplo 6-1, de cuya población no se conoce la varianza, usamos la función **t.test()**. En el capítulo siguiente serán ampliadas las aplicaciones de esta función.

Veamos cómo se obtiene el intervalo de confianza al nivel del 99% (por defecto se genera al 95%) para la media de la variable **nivclor**:

```
> t.test(nivclor,conf.level=0.99)$conf
[1] 1.660814 2.039186
attr(,"conf.level")
[1] 0.99
```

A continuación comprobaremos que se puede obtener el mismo resultado mediante la aplicación de la fórmula correspondiente (S es la *cuasidesviación típica muestral*):

$$(\overline{x} \pm t_{n-1;\alpha/2} S/\sqrt{n})$$

```
> xraya<-mean(nivclor)
> S<-sd(nivclor)
> 0.01/2
[1] 0.005
>ICal99<-c(xraya-qt(0.005,9,lower.tail=F)*S/sqrt(10),
+xraya+qt(0.005,9,lower.tail=F)*S/sqrt(10))
> ICal99
[1] 1.660814 2.039186
```

6.5 INTERVALO DE CONFIANZA PARA LA VARIANZA

Para construir un intervalo de confianza para la varianza de **nivclor** (ejemplo 6-1), al nivel del 99%, utilizamos la fórmula correspondiente al caso de una población normal de media desconocida:

$$\left[\frac{(n-1)S^2}{\chi^2_{n-1;\alpha/2}}, \frac{(n-1)S^2}{\chi^2_{n-1;1-\alpha/2}}\right]$$

```
> #Redondeamos los valores a 4 decimales
>round(c(9*var(nivclor)/qchisq(0.995,9),9*var(nivclor)/qchisq(0.005,9)),4)
[1] 0.0129 0.1758
```

Si queremos disponer de una función que nos sirva en aplicaciones posteriores para evaluar intervalos de confianza para otros conjuntos de datos, podemos crear directamente en *R* un *objeto* (función) que, evidentemente, puede ser almacenado.

```
> IC.para.la.varianza<-function(x,alfa)
+round(c((length(x)-1)*var(x)/qchisq(1-alfa/2,length(x)-1),
+(length(x)-1)*var(x)/qchisq(alfa/2,length(x)-1)),4)

> IC.para.la.varianza(x=nivclor,alfa=0.01)
[1] 0.0129 0.1758

> IC.para.la.varianza(nivclor,0.05)
[1] 0.0160 0.1129
```

6.6 INTERVALO DE CONFIANZA PARA EL COCIENTE DE VARIANZAS

Para obtener un intervalo de confianza para el cociente de varianzas de una forma directa, sin usar la fórmula correspondiente, debemos utilizar la función **var.test()** sobre la que profundizaremos en el capítulo siguiente.

Veamos cómo podemos obtener un intervalo de confianza para el cociente de las varianzas correspondientes a las variables **VentasA** y **VentasB** del ejemplo 6-2.

```
> var.test(VentasA,VentasB)$conf
[1]   76.65465 1148.23288
attr(,"conf.level")
[1] 0.95
> #Como el valor 1 no está incluido en el intervalo, deducimos que las varianzas
de las dos poblaciones no pueden ser consideradas iguales
```

6.7 INTERVALO DE CONFIANZA PARA LA DIFERENCIA DE MEDIAS

Al igual que en casos anteriores, obtendremos el intervalo de confianza para la diferencia de medias de las variables **VentasA** y **VentasB** utilizando una función que proporciona un resultado más amplio y que, como ya se ha indicado, conoceremos en profundidad en el capítulo siguiente.

```
> #De acuerdo al resultado obtenido en el apartado 6.6 debemos elegir la opción
var.equal=F para indicarle al programa que las varianzas no pueden ser
consideradas iguales

> t.test(VentasA,VentasB,var.equal=F)$conf
[1] -1159.397  3528.307
attr(,"conf.level")
[1] 0.95
```

6.8 INTERVALO DE CONFIANZA PARA UNA PROPORCIÓN

Para estimar una proporción, es decir, para estimar el parámetro p de una distribución binomial, utilizamos la función **prop.test()**:

```
> #Se trata de estimar mediante un intervalo de confianza la proporción de
piezas defectuosas de un lote. Para ello se escogen al azar 150 piezas,
observándose que 12 de ellas son defectuosas.

> 12/150 #Estimación puntual de p
[1] 0.08
> prop.test(12,150)$conf #Estimación de p por intervalo de confianza
[1] 0.04388586 0.13863413
attr(,"conf.level")
[1] 0.95
```

6.9 ESTUDIO DE LA NORMALIDAD DE LOS DATOS

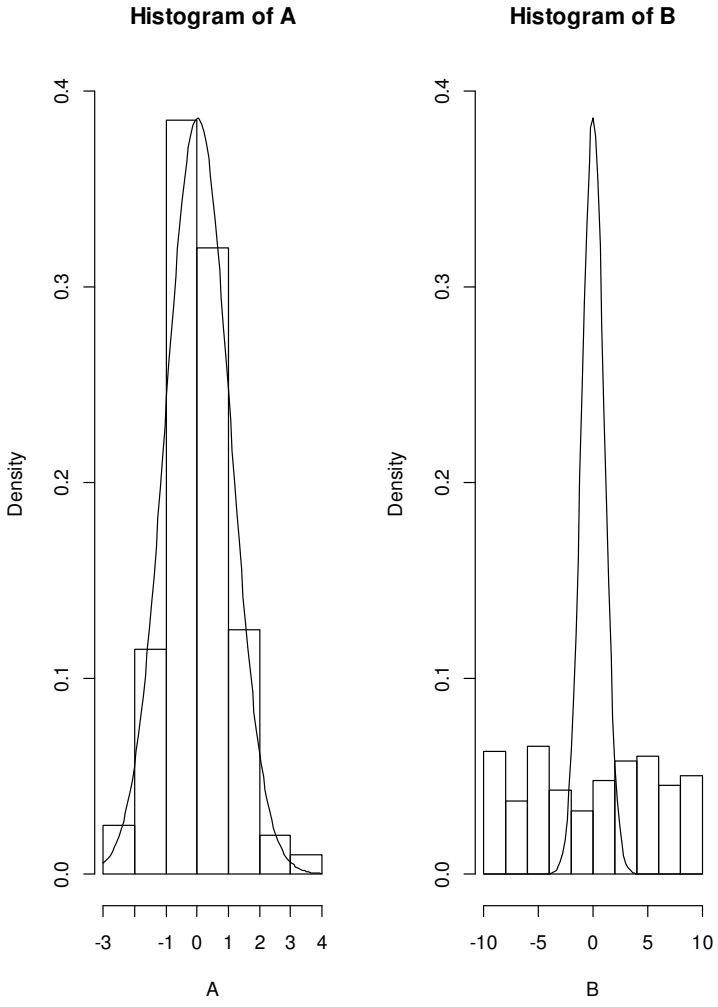
En muchas ocasiones los datos se suelen ajustar a una distribución normal. De hecho, en todos los intervalos de confianza que se han estudiado en este capítulo se supone, aunque no se haya mencionado explícitamente, que los datos provienen de una población normal. Una primera tarea consiste, por tanto, en estudiar la normalidad de los datos. Para ello, y sin profundizar en aspectos que excederían el objeto de este texto, se puede hacer lo siguiente:

1. Analizar si el histograma se ajusta a una distribución normal, para lo que podemos superponerle la curva normal.

2. Estudiar si en el gráfico **qqnorm()** los puntos se ajustan a la recta **qqline()** (normalidad) o se separan de ella.
3. Comprobar si se obtienen p-valores altos (mayores que 0.1) en el test de Shapiro-Wilk, función **shapiro.test()**, o en las pruebas de normalidad disponibles en el paquete **nortest**: test de Anderson-Darling, de Cramer von Mises, de Kolmogorov-Smirnof, de Pearson (Ji-cuadrado) y de Shapiro-Francia. Para muestras pequeñas ($n \leq 50$) se recomienda utilizar el test de Shapiro-Wilk y para muestras grandes el de Kolmogorov-Smirnov. Si los datos están agrupados por frecuencias el test adecuado es el test ji-cuadrado de Pearson.

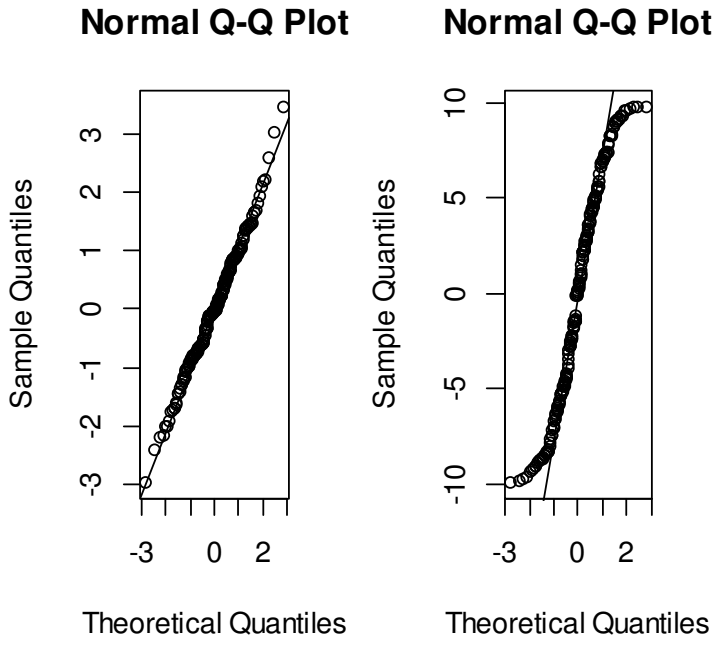
A continuación vamos a comprobar, mediante un ejemplo concreto, qué ocurre cuando las acciones anteriores se aplican a una distribución normal estándar y a una distribución no normal como la uniforme continua de parámetros [-10,10]. Ambas serán estudiadas a través de sendas muestras de tamaño 200.

```
> A<-rnorm(200,0,1)

> B<-runif(200,-10,10)

> split.screen(c(1,2))
[1] 1 2

> screen(1)
> hA<-hist(A,plot=F) #Se genera el histograma de los datos A pero no se dibuja, en principio
> #La sentencia siguiente tiene por objeto no cortar la parte superior de la función de densidad de la distribución normal
> ylimA<-range(0,hA$density,dnorm(mean(A)))
> hist(A,freq=F,ylim=ylimA)
> curve(dnorm(x,mean(A),sd(A)),add=T) #Superponemos una normal con media y desviación típicas las de los valores A

> screen(2)
> #Procederemos de forma análoga para los datos B
> hB<-hist(B,plot=F)
> ylimB<-range(0,hB$density,dnorm(mean(B)))
> hist(B,freq=F,ylim=ylimB)
> curve(dnorm(x,mean(B),sd(A)),add=T)
```



> dev.off() #Eliminamos el gráfico actual
null device
1
> split.screen(c(1,2))
[1] 1 2
> screen(1)
> qqnorm(A)
> qqline(A)

> screen(2)
> qqnorm(B)
> qqline(B)



> #A continuación efectuamos los test de normalidad
> shapiro.test(A)
Shapiro-Wilk normality test
data: A
W = 0.9944, p-value = 0.6543
> shapiro.test(B)
Shapiro-Wilk normality test
data: B
W = 0.9473, p-value = 1.041e-06
> #Cargamos el paquete nortest
> library(nortest)
> ad.test(A)
Anderson-Darling normality test
data: A
A = 0.3428, p-value = 0.4877
> ad.test(B)
Anderson-Darling normality test
data: B
A = 2.8045, p-value = 4.425e-07

```
> cvm.test(A)
  Cramer-von Mises normality test
data:  A
W = 0.0575, p-value = 0.4069
> cvm.test(B)
    Cramer-von Mises normality test
data:  B
W = 0.4168, p-value = 1.747e-05

> lillie.test(A)
     Lilliefors (Kolmogorov-Smirnov) normality test
data:  A
D = 0.0449, p-value = 0.4172
> lillie.test(B)
 Lilliefors (Kolmogorov-Smirnov) normality test
data:  B
D = 0.0912, p-value = 0.0003571

> pearson.test(A)
 Pearson chi-square normality test
data:  A
P = 23.21, p-value = 0.05694
> pearson.test(B)
 Pearson chi-square normality test
data:  B
P = 59.76, p-value = 1.293e-07

> sf.test(A)
     Shapiro-Francia normality test
data:  A
W = 0.9929, p-value = 0.3872
> sf.test(B)
     Shapiro-Francia normality test
data:  B
W = 0.9525, p-value = 1.171e-05
```

Como se aprecia claramente en los gráficos y a través de los p-valores de los test, los datos A se ajustan muy bien a una distribución normal, mientras que los datos B no superan ninguna de las pruebas de normalidad.

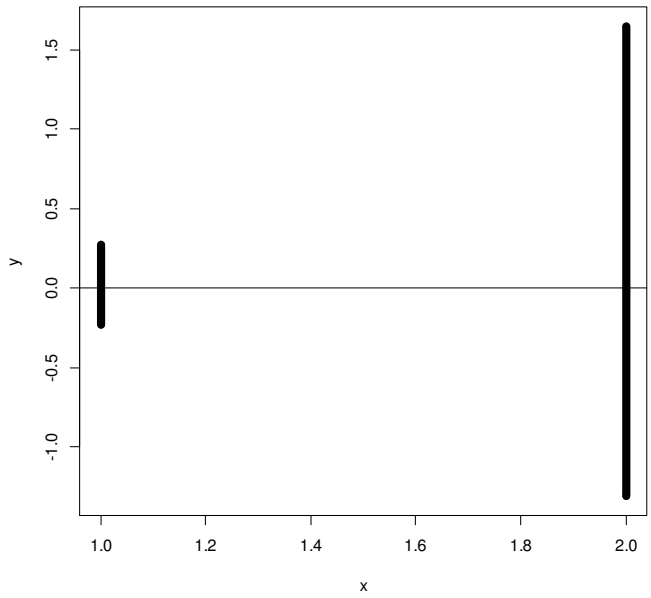
6.10 EJERCICIOS RESUELTOS

●**ER 6-1** Generar al azar una lista de 20 valores provenientes de una población normal de media 10 y varianza 4. Utilizando estos datos calcular un intervalo de confianza al 95% para la media de la población y otro al 99% para la varianza de la población.

```
> valores<-rnorm(20,10,2)
> t.test(valores)$conf
[1]  9.052416 10.825978
attr(,"conf.level")
[1] 0.95
> IC.varianza<-c(9*var(valores)/qchisq(0.975,9),9*var(valores)/qchisq(0.025,9))
> IC.varianza
[1]  1.698568 11.965486
```

●**ER 6-2** Dibujar los intervalos de confianza al 95% para las medias correspondientes a dos muestras de tamaño 50 de una población N(0,1) y de una N(0,5).

```
> muestra1<-rnorm(50,0,1)
> muestra2<-rnorm(50,0,5)

> a<-t.test(muestra1)$conf
> b<-t.test(muestra2)$conf

> a
[1] -0.2306505  0.2692323
attr(,"conf.level")
[1] 0.95
> b
[1] -1.311466  1.648882
attr(,"conf.level")
[1] 0.95

> x<-c(1,1,2,2)
> y<-c(a[1],a[2],b[1],b[2])
> plot(x,y)
> abline(0,0)
> segments(x[1],y[1],x[2],y[2],lwd=8) #Con el argumento lwd establecemos el grosor de las líneas
> segments(x[3],y[3],x[4],y[4],lwd=8)
```



●**ER 6-3** Se quiere estimar el precio del metro cuadrado de vivienda nueva en el municipio de Getxo. Para ello se han tomado 12 viviendas al azar, obteniéndose los valores siguientes, en miles de euros por metro cuadrado (se supone normalidad): 4.01, 3.87, 4.68, 2.83, 3.88, 4.92, 4.46, 5.64, 4.91, 2.35, 4.12, 1.11.

```
> datos<-c(4.01,3.87,4.68,2.83,3.88,4.92,4.46,5.64,4.91,2.35,4.12,1.11)
> mean(datos)
[1] 3.898333
> t.test(datos)$conf
[1] 3.099149 4.697517
attr(,"conf.level")
[1] 0.95
> t.test(datos,conf.level=0.99)$conf
[1] 2.770606 5.026061
attr(,"conf.level")
[1] 0.99
```

●**ER6-4** Para el conjunto de datos **vitcap,** perteneciente al paquete **IswR,** estimar la diferencia entre las medias de la variable **vital.capacity** cuando la variable **group** toma el valor 1 y cuando toma el valor 3. Se supone normalidad.

```
> library(ISwR)
LMensajes de aviso perdidos
package 'ISwR' was built under R version 3.0.2
```

```
> data(vitcap)
> attach(vitcap)
> vitcap
 group age vital.capacity
1     1  39          4.62
2     1  40          5.29
3     1  41          5.52
....
23    3  33          4.44
24    3  27          5.52

> vc1<-vital.capacity[group==1]
> vc3<-vital.capacity[group==3]
> vc1
 [1] 4.62 5.29 5.52 3.71 4.02 5.09 2.70 4.31 2.70 3.03 2.73 3.67
> vc3
 [1] 5.29 3.67 5.82 4.77 5.71 4.47 4.55 4.61 5.86 5.20 4.44 5.52

> mean(vc1)-mean(vc3) #Estimación puntual
[1] -1.043333

> var.test(vc1,vc3)$conf
[1] 0.6651437 8.0260128
attr(,"conf.level")
[1] 0.95
> #Como el valor 1 está contenido en el anterior intervalo de confianza podemos considerar iguales las varianzas
> t.test(vc1,vc3,var.equal=T)$conf
[1] -1.783637 -0.303030
attr(,"conf.level")
[1] 0.95
```

●**ER 6-5** En un estudio sobre hábitos de fumadores una muestra de 400 zurdos reveló que 190 de ellos fumaban y una muestra de 800 diestros que lo hacían 300. Construir un intervalo de confianza al 98% para la diferencia entre las proporciones de fumadores zurdos y diestros.

```
>  prop.test(c(190,300),c(400,800),conf.level=0.98)$conf
[1] 0.02770133 0.17229867
attr(,"conf.level")
[1] 0.98
```

6.11 EJERCICIOS PROPUESTOS

●**EP 6-1** Generar una muestra aleatoria de tamaño 100 de una variable aleatoria N(-2,5). Calcular los correspondientes intervalos de confianza para la media de la población a los niveles de confianza del 95% y del 98%.

●**EP 6-2** Generar una muestra aleatoria de tamaño 200 de una variable aleatoria N(20,10) y calcular un intervalo de confianza para la varianza de la población al nivel del 99%.

●**EP 6-3** Estudiar la normalidad de los datos **rivers** incluidos en el paquete **datasets**.

●**EP 6-4** Obtener un intervalo de confianza para la diferencia de proporciones de pasajeras y pasajeros supervivientes en naufragios de barcos de pasaje, tomando como muestra (no representativa, muy probablemente) los datos del hundimiento del Titanic, que aparecen en la tabla **Titanic** del paquete **datasets**. Transformar, en primer lugar, esa tabla en un marco de datos mediante la función **as.data.frame()**.

●**EP 6-5** Generar 100 muestras aleatorias de tamaño 100 de una variable aleatoria N(10,2). Calcular los correspondientes intervalos de confianza para la media al nivel del 95% y el número de ellos que no la contienen.

Capítulo 7

CONTRASTES DE HIPÓTESIS

7.1 INTRODUCCIÓN

Este capítulo está íntimamente relacionado con el anterior, pues en muchos casos se utilizan las mismas funciones para hacer contrastes de hipótesis y para obtener intervalos de confianza. Volveremos, por tanto, a hacer referencia a los ejemplos 6-1 y 6-2. Además, veremos otros ejemplos referidos a contrastes χ^2.

Sin ánimo de ser exhaustivos se presentan a continuación únicamente algunos contrastes, entre los muchos existentes, con objeto de poner de manifiesto el procedimiento general para la implementación de esta técnica inferencial en *R*.

7.2 CONTRASTES SOBRE LA MEDIA Y LA VARIANZA DE UNA POBLACIÓN NORMAL

En relación a los datos del ejemplo 6-1 vamos a contrastar si se puede aceptar la hipótesis de que provienen de una población normal de media 1.9. Como la varianza de la población es desconocida debemos realizar un contraste t de Student:

```
> #Ejemplo 6-1 (cont.)
> nivclor<-c(2.2,1.9,1.7,1.6,1.7,1.8,1.7,1.9,2,2)
> nivclor
 [1] 2.2 1.9 1.7 1.6 1.7 1.8 1.7 1.9 2.0 2.0
> t.test(nivclor,mu=1.9)
  One Sample t-test
data:  nivclor
t = -0.8589, df = 9, p-value = 0.4127
alternative hypothesis: true mean is not equal to 1.9
95 percent confidence interval:
 1.718310 1.981690
sample estimates:
mean of x
   1.85
```

La interpretación de esta salida de resultados es la siguiente: se ha efectuado un test bilateral (véase la hipótesis alternativa) de la t de Student con 9 grados de libertad (**df**). El estadístico de contraste es -0.8589. El intervalo de confianza para la media, al nivel 95%, es (1,72;1,98). El p-valor es 0.4127. Dado que este p-valor es muy alto, no se puede rechazar la hipótesis nula de que la media de la población es 1,9. La media muestral (estimador muestral) es 1.85.

A continuación se desea contrastar si la varianza de la población, cuyos datos son los del ejemplo 6-1, es 0,05. Para ello construiremos una función que no está directamente implementada en *R*. Lo haremos de una forma simplificada, obteniendo la región de aceptación y calculando posteriormente el p-valor.

```
> #Vamos a definir una función para efectuar el test sobre la varianza de una
población normal de media desconocida
> var.test.una.población.normal<-function(x,conf.level=0.95) {
+ n=length(x)
+ alfa=1-conf.level
+ valcrit1=qchisq(1-alfa/2,n-1)
+ valcrit2=qchisq(alfa/2,n-1)
+ c((n-1)*var(x)/valcrit1,(n-1)*var(x)/valcrit2)}
[1] 0.01603342 0.11294667
> #Como el valor 0.05 pertenece al intervalo anterior no podemos rechazar la
hipótesis de que la varianza de la población es 0.05
```

```
> #A continuación obtenemos el p-valor del test
> var(nivclor)
[1] 0.03388889
> 2*pchisq(9*var(nivclor)/0.05,9)
[1] 0.5402607
> #Valor muy grande que da gran seguridad en aceptar la hipótesis nula
```

7.3 CONTRASTE SOBRE LA IGUALDAD DE VARIANZAS DE DOS POBLACIONES NORMALES

El objetivo final que se persigue en el ejemplo 6-2 es comparar las ventas en las joyerías A y B, para lo cual debemos confrontar las ventas medias en ambos establecimientos. Sin embargo, previamente debemos efectuar un contraste sobre la igualdad de varianzas, cuyo resultado será utilizado posteriormente en el contraste sobre las medias.

```
> datos<- read.delim("C:/Ejemplo 6-2.txt",header=T)
> datos
  VentasA VentasB
1    1320    1110
....
11   1930      NA
> attach(datos)
> var.test(VentasA,VentasB)
F test to compare two variances
data:  VentasA and VentasB.nuevo
F = 303.8487, num df = 10, denom df = 9, p-value = 7.775e-10
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
 76.65465 1148.23288
sample estimates:
ratio of variances
303.8487
```

Interpretación del resultado anterior: en primer lugar se observa que se ha efectuado un test mediante la distribución F de Snedecor sobre la igualdad de varianzas de dos poblaciones (normales), cuyo estadístico de contraste F es 303.8487. Los grados de libertad del numerador y denominador son, respectivamente, 10 y 9. El p-valor resultante es muy pequeño, por lo que se rechaza la hipótesis nula de igualdad de varianzas. La hipótesis alternativa expresa que el cociente de varianzas no es igual a 1 (varianzas distintas). Se obtiene, así mismo, el intervalo de confianza al 95% (valor por defecto) para el cociente de varianzas, siendo la correspondiente estimación puntual de este parámetro 303.8487.

7.4 CONTRASTE DE IGUALDAD DE MEDIAS DE DOS POBLACIONES NORMALES

Ahora ya estamos en condiciones de contrastar la igualdad de medias de las ventas en las joyerías A y B del ejemplo 6-2. Como se ha deducido del contraste anterior que las varianzas poblacionales de ambos establecimientos se pueden considerar diferentes, introduciremos el argumento **var.equal=F** (*false*).

```
> t.test(VentasA,VentasB,var.equal=F)
  Welch Two Sample t-test
data:  ventasA and ventasB
t = 1.1249, df = 10.072, p-value = 0.2867
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -1159.397  3528.307
sample estimates:
mean of x mean of y
 2430.455  1246.000
```

Si deseamos incrementar el nivel de confianza hasta el 99% podemos hacer:

```
> t.test(VentasA,VentasB,var.equal=F,conf.level=0.99)
  Welch Two Sample t-test
data:  VentasA and VentasB
t = 1.1249, df = 10.072, p-value = 0.2867
alternative hypothesis: true difference in means is not equal to 0
99 percent confidence interval:
 -2147.289  4516.198
sample estimates:
mean of x mean of y
-30.455 1246.000
```

La salida de resultados se interpreta del siguiente modo: el programa ha efectuado una aproximación (Welch) al no poder considerarse iguales las varianzas poblacionales. El valor del estadístico de contraste es t=1.1249, los grados de libertad son 10.072 (se trata de una aproximación) y el p-valor es 0.2867. La hipótesis alternativa se refiere a que la diferencia de medias no es igual a cero o, equivalentemente, que las medias no son iguales. Adicionalmente se obtienen los correspondientes intervalos de confianza al 95% (por defecto) y al 99%.

Como el p-valor es muy grande no podemos rechazar la igualdad de medias y concluimos que no hay diferencias significativas en las ventas de los establecimientos A y B.

7.5 CONTRASTE SOBRE UNA PROPORCIÓN

Ejemplo 7-1 Se ha encuestado a 110 personas sobre si están de acuerdo con la construcción del tren de alta velocidad, habiendo contestado 48 de ellas afirmativamente. ¿Respalda este resultado la hipótesis de que la proporción de opiniones afirmativas en la población es el 50%?

```
> prop.test(48,110,p=0.5)
  1-sample proportions test with continuity correction
data:  48 out of 110, null probability 0.5
X-squared = 1.5364, df = 1, p-value = 0.2152
alternative hypothesis: true p is not equal to 0.5
95 percent confidence interval:
0.3431002 0.5341288
sample estimates:
   p
0.4363636
```

Se ha efectuado un test para la proporción de una población con corrección por continuidad. En lugar de utilizar el habitual test, que implica la aproximación de la distribución binomial a la normal, se emplea aquí otro basado en la distribución χ^2. El alto p-valor obtenido respalda la hipótesis nula de que la proporción de opiniones afirmativas en la población es del 50%.

7.6 CONTRASTES χ^2

En este apartado vamos a ver dos ejemplos de aplicación de contrastes χ^2: el primero de bondad de ajuste y el segundo de independencia.

Ejemplo 7-2 Las letras que más frecuentemente aparecen en el idioma inglés son E, N, T, R y O. Cuando alguna de ellas se presenta en un texto, la probabilidad de que aparezca cada una viene dada en la tabla siguiente:

E	N	T	R	O
0.29	0.17	0.21	0.17	0.16

Esta información es útil en criptografía. Supongamos que en un cierto texto se han contabilizado estas cinco letras, apareciendo cada una de ellas el número de veces que se indica en la tabla:

E	N	T	R	O
100	80	110	55	14

Efectuar un contrate χ^2 de bondad de ajuste de los datos muestrales a la distribución teórica.

```
> x<-c(100,80,110,55,14)
> prob.teóricas<-c(0.29,0.17,0.21,0.17,0.16)
> chisq.test(x,p=prob.teóricas)
  Chi-squared test for given probabilities
data:  x
X-squared = 55.3955, df = 4, p-value = 2.685e-11
```

Como el valor-p es muy pequeño se rechaza la hipótesis de que los datos muestrales se ajustan a la distribución teórica y se concluye, por tanto, que es muy improbable que el texto esté escrito en inglés.

Ejemplo 7-3 Una compañía evalúa una propuesta para fusionarse con una corporación. Una muestra aleatoria simple de 250 accionistas proporciona la siguiente información:

Núm. de acciones por accionista	A favor	En contra	Indecisos
Menos de 200	38	29	9
Entre 200 y 1000	30	42	7
Más de 1000	32	59	4

¿Existe alguna razón para dudar de que la opinión con respecto a la propuesta es independiente del número de acciones que posee cada accionista?

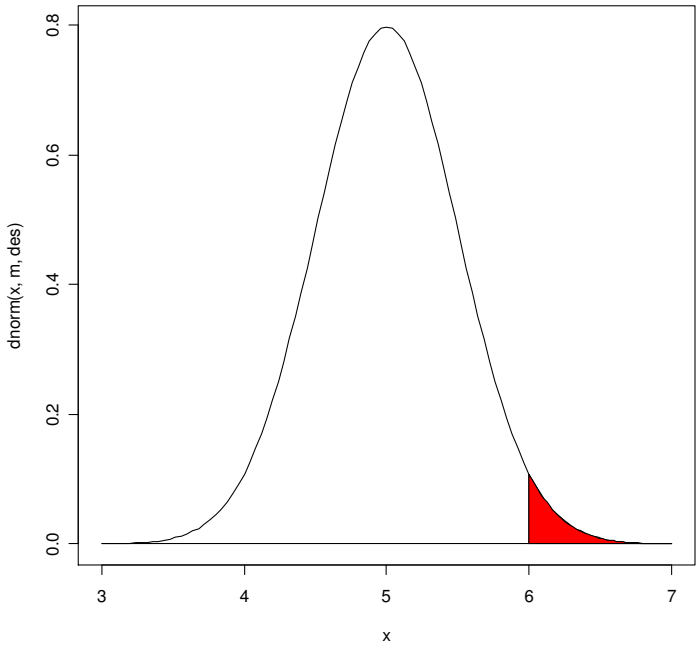
```
> pequeño<-c(38,29,9)
> mediano<-c(30,42,7)
> grande<-c(32,59,4)

> chisq.test(data.frame(pequeño,mediano,grande))
  Pearson's Chi-squared test
data:  data.frame(pequeño, mediano, grande)
X-squared = 10.7957, df = 4, p-value = 0.02896
```

El p-valor obtenido en este test de independencia no es ni pequeño (<0.01) ni grande (>0.1), por lo que el resultado es algo ambiguo y lo más conveniente sería repetir la prueba aumentando, si es posible, el tamaño muestral.

7.7 EJERCICIOS RESUELTOS

●**ER 7-1** Indicar la región crítica o región de rechazo al realizar el siguiente contraste de hipótesis, con nivel de significación α=0.05, tomando una muestra de 16 datos que ha sido extraída de una población normal con desviación típica σ=2: $H_0 : \mu = 5 \; / \; H_a : \mu > 5$. Se pide, así mismo: a) Hallar el p-valor del contraste para $\overline{x} = 6$. b) Calcular la probabilidad de cometer error de tipo II con este contraste cuando μ=6. c) ¿Cuál es la probabilidad de que la cuasivarianza muestral sobreestime la varianza poblacional?

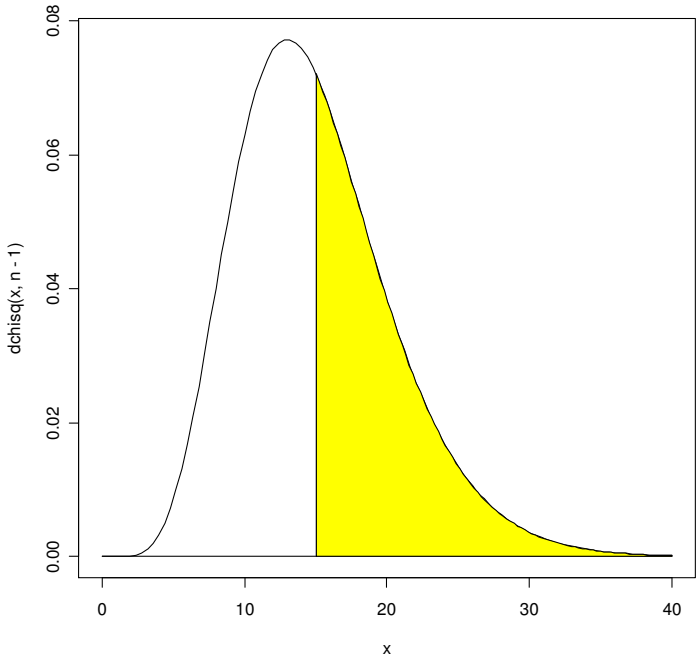
```
> mu=5;alfa=0.05;n=16;sigma=2
> Región.Crítica<-c(mu+qnorm(alfa,lower.tail=F)*sigma/sqrt(n),Inf)
> Región.Crítica
[1] 5.822427      Inf

> pvalor<-1-pnorm(6,mu,sigma/sqrt(n))
> pvalor
[1] 0.02275013

> curve(dnorm(x,mu,sigma/sqrt(n)),from=3,to=7) #Dibujamos la distribución de la media muestral
> segments(3,0,7,0) #En el gráfico anterior colocamos el eje de abscisas
> segments(6,0,6,dnorm(6,mu,sigma/sqrt(n))) #Línea vertical hasta la curva que deja a su derecha el p-valor
> xvals <- seq(6,7,length=50) #Abscisas del recinto a sombrear
> yvals <- dnorm(xvals,mu,sigma/sqrt(n)) #Ordenadas del recinto a sombrear
> polygon(c(xvals,rev(xvals)),
+ c(rep(0,50),rev(yvals)),col='red') #Sombreamos en rojo el p-valor
```




```
> #Calculamos la probabilidad de cometer error de tipo II, o sea de aceptar que la media de la población es 5 cuando en realidad es 6: P(error de tipo II) = P(aceptar H0|H0 es falsa) = P(xraya<5.822427|μ=6)
> pnorm(5.822427,6,sigma/sqrt(n))
[1] 0.3612401
```

> #Calculamos ahora la probabilidad de que la cuasivarianza muestral sobreestime la varianza poblacional: $P(S^2>\sigma^2)=P((n-1)S^2/\sigma^2>n-1)$ y la distribución de $(n-1)S^2/\sigma^2$ es $\chi^2(n-1)$
> 1-pchisq(n-1,n-1)
[1] 0.4514172

> #Vamos a dibujar en amarillo la probabilidad anterior
> curve(dchisq(x,n-1),from=0,to=40)
> segments(0,0,40,0)
> segments(n-1,0,n-1,dchisq(n-1,n-1))
> xx<-seq(n-1,40,length=100)
> yy<-dchisq(xx,n-1)
> polygon(c(xx,rev(xx)),c(rep(0,100),rev(yy)),
+ col="yellow")



●**ER 7-2** Con objeto de determinar el contenido calórico de las barras de pan, una panificadora efectuó un estudio antes y después de la puesta en práctica de un nuevo proceso, obteniéndose los siguientes resultados:

Proceso antiguo: $n_1 = 30$; $\bar{x}_1 = 1330$ calorías; $S_1 = 238$ calorías

Proceso nuevo: $n_2 = 50$; $\bar{x}_2 = 1255$ calorías; $S_2 = 215$ calorías

Se desea saber si los resultados proporcionan evidencia suficiente para concluir que el número medio de calorías por barra ha disminuido con el nuevo proceso. Nivel de significación α=0.05. Calcular el nivel crítico correspondiente a los datos muestrales.

```
> n1<-30;xraya1<-1330;S1<-238
> n2<-50;xraya2<-1255;S2<-215
> alfa<-0.05
>Región.crítica<-c(qnorm(1-alfa)
+ *sqrt((S1^2/n1)+(S2^2/n2)),Inf)
> Región.crítica
[1] 87.2336     Inf
> xraya1-xraya2
[1] 75
> #Como 75 no pertenece a la región crítica no podemos rechazar la hipótesis nula y, por tanto, no hay evidencia suficiente para concluir que el nuevo proceso produce barras de pan con menos calorías

> #Calculemos ahora el p-valor
> 1-pnorm(75,0,sqrt((S1^2/n1)+(S2^2/n2)))
[1] 0.07865452
```

●**ER 7-3** Con objeto de comparar las varianzas de dos poblaciones normales se han tomado dos muestras de tamaños 6 y 10, obteniéndose para la primera los valores 6, 8, 5, 4, 9, 5 y para la segunda 6, 7, 6, 9, 6, 2, 9, 4, 6, 4. Contrastar al nivel α=0.05 si puede admitirse la igualdad de varianzas poblacionales.

```
> muestra1<-c(6,8,5,4,9,5)
> muestra2<-c(6,7,6,9,6,2,9,4,6,4)

> var.test(muestra1,muestra2)
 F test to compare two variances
data:  muestra1 and muestra2
F = 0.7902, num df = 5, denom df = 9, p-value = 0.8354
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
 0.1762126 5.2794346
sample estimates:
ratio of variances
   0.7902098

> #Como se ha obtenido un p-valor muy grande no se puede rechazar la hipóteis nula de igualdad de varianzas
```

●**ER 7-4** En una ciudad se implantó un plan para incentivar a los automóviles con dos o más ocupantes. Para ello se observaron 2000 vehículos antes del plan y 1500 después, obteniéndose 655 y 576 automóviles, respectivamente, con dos o más pasajeros. ¿Indican los datos que el plan consiguió su propósito? Nivel de significación α=0.05. Hallar el p-valor correspondiente a los datos muestrales del problema.

```
> n1<-2000;x1<-655
> n2<-1500;x2<-576

> x1/n1;x2/n2
[1] 0.3275
[1] 0.384

> alfa<-0.05

>#Como se trata de ver si el plan mejora la ocupación de los automóviles, vamos
a ver si la diferencia entre la proporción después y la proporción antes es
significativa. Como, por defecto, el test de proporciones es bilateral, en este
caso debemos indicar a R que realice un test unilateral: alternative=c("greater")

> prop.test(c(x2,x1),c(n2,n1),
+ alternative=c("greater"))
 2-sample test for equality of proportions with continuity correction
data:  c(x2, x1) out of c(n2, n1)
X-squared = 11.7538, df = 1, p-value = 0.0003036
alternative hypothesis: greater
95 percent confidence interval:
 0.02899840 1.00000000
sample estimates:
prop 1 prop 2
0.3840 0.3275

>#R ha efectuado un test de proporciones con corrección por continuidad cuyo
p-valor es muy pequeño, por lo que podemos concluir que hay evidencia
suficiente de que la proporción de automóviles con dos o más pasajeros aumentó
después de la puesta en marcha del plan
```

●**ER 7-5** Se quiere estudiar si existen o no diferencias significativas entre tres institutos en relación a las calificaciones obtenidas por sus alumnos en la asignatura de matemáticas. Para ello se seleccionaron al azar 50 alumnos en cada uno de los tres centros, obteniéndose los siguientes resultados:

	Calificaciones		
	0-4	5-7	8-10
Instituto A	17	20	13
Instituto B	20	15	15
Instituto C	25	16	9

```
> datos<-matrix(c(17,20,13,20,15,15,25,16,9),
+ nrow=3,byrow=T)
> colnames(datos)<-c("0-4","5-7","8-10")
> rownames(datos)<-c("Instituto A",
+ "Instituto B","Instituto C")
> datos
             0-4 5-7 8-10
Instituto A  17  20   13
Instituto B  20  15   15
Instituto C  25  16    9

> chisq.test(datos)
 Pearson's Chi-squared test
data:  datos
X-squared = 3.9177, df = 4, p-value = 0.4173
> #El alto p-valor obtenido respalda con fuerza la hipótesis de homogeneidad de
las clasificaciones de matemáticas en los tres institutos
```

7.8 EJERCICIOS PROPUESTOS

●**EP 7-1** Generar dos muestras aleatorias de tamaño 150 de una distribución N(10,2). Efectuar un test sobre la igualdad de varianzas y otro sobre la igualdad de medias. Repetir lo anterior para una muestra de una población N(10,2) y otra muestra de una población N(15,2).

●**EP 7-2** Un fabricante de vigas de acero asegura que el 95% de las vigas que construye no sufren corrosión al cabo de 5 años. Con objeto de probar tal aseveración se han sometido a análisis 60 vigas, encontrándose que, después de 5 años de funcionamiento, 52 de ellas no presentaban signos de deterioro. ¿Hay pruebas de que el fabricante estaba en lo cierto? (α=0,05)

●**EP 7-3** La demanda de un cierto producto durante los cuatro trimestres del año 2013 fue:

Trimestre	Nº de unidades
1º	100
2º	95
3º	110
4º	98

Contrastar al nivel 1% si la distribución de la demanda es uniforme.

●**EP 7-4** Realizar 50 test de la t de Student sobre 50 muestras simuladas de tamaño 25 de una distribución normal estándar. Comentar los resultados. NOTA: Para obtener el p-valor de un test sobre la muestra x hacer: **t.test(x)$p.value**.

●**EP 7-5** Para contrastar si mediante el proceso B se consigue disminuir, respecto del proceso A, el tiempo de ejecución de ciertos trabajos se ejecutaron seis tareas con ambos procesos, obteniéndose los siguientes tiempos, medidos en horas:

	Tareas					
	1	2	3	4	5	6
Proceso A	2,5	7,1	5	8,5	7	8,1
Proceso B	2,3	7,1	4	8	6,6	5

Admitiendo normalidad y con un nivel de significación α=0,05, ¿qué conclusiones pueden obtenerse? NOTA: Se debe suponer que los datos son apareados, pues los tiempos del proceso A no pueden considerarse como independientes de los del proceso B al haber sido computados en las mismas tareas.

Capítulo 8

REGRESIÓN LINEAL Y ANÁLISIS DE LA VARIANZA

8.1 INTRODUCCIÓN

En este capítulo examinaremos, a un nivel meramente introductorio, dos técnicas muy útiles en estadística que pueden ser puestas en práctica fácilmente mediante el programa *R*.

La regresión lineal simple ya ha sido tratada desde el punto de vista del análisis de datos en el capítulo 3, dedicado a la estadística descriptiva de dos variables. Ahora añadiremos la perspectiva de la inferencia estadística para decidir si los coeficientes de regresión cumplen determinados test estadísticos y para estimarlos mediante intervalos de confianza.

En cuanto al análisis de la varianza se realizará únicamente una breve introducción, restringiendo nuestro estudio exclusivamente al análisis de la varianza con un factor.

Se presentarán dos ejercicios de aplicación con objeto de analizar el modo en que se utiliza *R* en la resolución de estos dos tipos de problemas. En el primero de ellos se trata de ver la relación entre dos variables cuantitativas, por lo que el modelo adecuado es la regresión lineal. En el segundo se desea estudiar la relación entre una variable cuantitativa y una variable cualitativa, por lo que procede efectuar un análisis de la varianza.

8.2 ANÁLISIS DE LOS DATOS EN LA REGRESIÓN LINEAL

Lo primero que se debe hacer en una regresión lineal es un análisis de los datos. Lo más adecuado, como se explicó en el capítulo 3, es empezar por visualizarlos mediante un diagrama de dispersión, con la variable regresora o predictora en el eje horizontal y la variable respuesta en el eje vertical. Veamos el ejemplo siguiente.

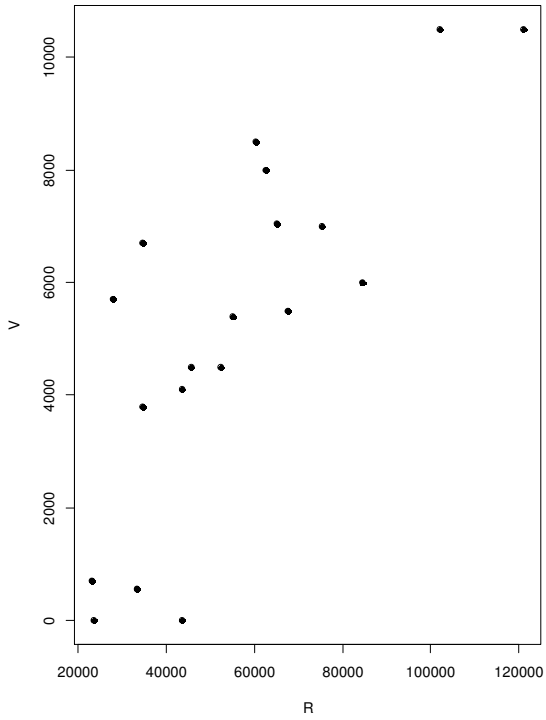
Ejemplo 8-1 Se han estudiado en 19 familias las variables R (nivel de renta) y V (gastos en vacaciones) obteniéndose los resultados que aparecen en el archivo "Renta-gastos.txt". Se pide: 1º) Representar la nube de puntos correspondiente a los 19 pares de valores. 2º) Recta de regresión de V sobre R. 3º) Coeficiente de correlación. 4º) Contrastar si existe una relación lineal entre V y R.

```
> #Ejemplo 8-1
> #Leemos los datos contenidos en el archivo correspondiente
> datos<-read.table("C:/Renta-gastos.txt",header=T)
> datos
       R     V
1   60205  8500
2   55000  5400
3   43575  4105
4   84567  6000
5   33454   550
6   23146   700
7  102000 10500
8   67500  5500
9   52300  4500
10  34567  6700
11  65005  7050
12  75200  7000
13  45650  4500
14 121070 10500
15  62580  8000
16  34521  3800
17  28000  5700
18  43500     0
19  23560     0
```

> attach(datos)

Ahora efectuamos el diagrama de dispersión:

> plot(V~R,pch=19) #V~R significa que V es función de R; es decir, que V es la variable dependiente o respuesta y R la variable independiente o predictora o regresora. Obtendríamos el mismo resultado haciendo plot(R,V,pch=19)



En la figura se adivina una cierta relación lineal entre las variables R y V: a medida que aumenta la renta R da la impresión de que aumenta el nivel de gastos en vacaciones V, lo cual parece absolutamente lógico. En consecuencia, resulta plausible utilizar un modelo de regresión lineal, como haremos a continuación, con el objetivo último de pronosticar V conociendo R.

8.3 ANÁLISIS DE REGRESIÓN

Vamos a estimar los coeficientes de la recta de regresión mediante las fórmulas que se deducen de las ecuaciones normales:

$$b = r_{xy}\frac{S_y}{S_x}; a = \overline{y} - b\overline{x}$$

donde r_{xy} es el coeficiente de correlación y S_x y S_y las desviaciones típicas respectivas de x e y.

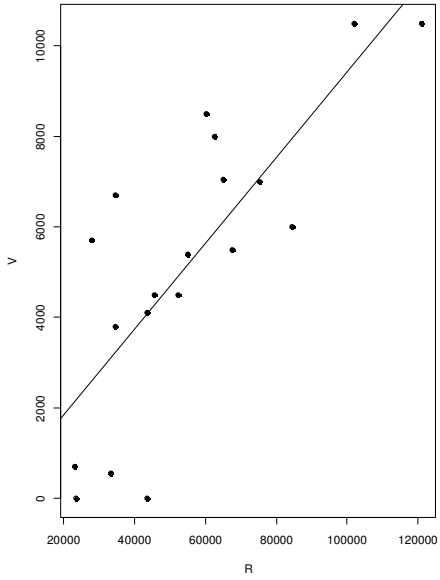
```
> #Seguimos trabajando en el ejemplo 8-1
> mean(R);mean(V)
[1] 55547.37
[1] 5210.789

> sd(R)*sqrt(18/19);sd(V)*sqrt(18/19)
[1] 25733.93
[1] 3119.816
> cor(R,V)
[1] 0.781503

> b<-0.781503*(3119.816/25733.93)
> a<-5210.789-b*55547.37
> a;b
[1] -52.01282
[1] 0.0947444
```

La recta de regresión estimada es, por tanto, $V = -52.01282 + 0.0947444 \cdot R$. Obtengamos esta ecuación de forma directa:

```
> modelo<-lm(V~R)

> summary(modelo)
Call:
lm(formula = V ~ R)
Residuals:
   Min      1Q  Median      3Q     Max
-4069.37 -1179.82    28.53   915.62  3476.98
Coefficients:
      Estimate Std. Error t value Pr(>|t|)
(Intercept)  -52.01267 1123.05085  -0.046    0.964
R             0.09474    0.01834   5.165 7.78e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 2058 on 17 degrees of freedom
Multiple R-Squared: 0.6107,     Adjusted R-squared: 0.5878
F-statistic: 26.67 on 1 and 17 DF,  p-value: 7.776e-05
```

Podemos dibujar, superpuesto al diagrama de dispersión obtenido anteriormente, la recta de regresión:

> abline(modelo)



Como se ve, la ecuación de la recta de regresión es la misma que hemos obtenido anteriormente: ordenada en el origen = -52,01267 y pendiente de la recta = 0,09474. Sin embargo, ahora lo hemos hecho de una forma mucho más directa y además con mucha información adjunta que pasamos a comentar.

En primer lugar aparece un breve análisis descriptivo de los residuales, proporcionando los 5 valores principales (mínimo, máximo, mediana, primer y tercer cuartil), lo cual tiene importancia a la hora de decidir si es pertinente aplicar un modelo de regresión lineal o no.

Posteriormente se obtienen los estimadores por mínimos cuadrados de la recta de regresión, con sus correspondientes errores estándar y p-valores. Se completa el informe con el error o varianza residual, el coeficiente de determinación R^2 y el coeficiente de determinación ajustado, así como el estadístico F para comprobar si el modelo sirve para pronosticar V en función de R y el p-valor o nivel crítico correspondiente.

Mediante la sentencia **plot(lm(V~R))** podríamos obtener diferentes gráficos para contrastar la validez del modelo. No obstante, y sin entrar a profundizar en esta cuestión, podemos decir que en el ejemplo 8-1 el modelo lineal se adecua bastante bien a los datos muestrales, pues el valor-p de la pendiente es muy pequeño, lo que lleva a rechazar la hipótesis nula de pendiente cero, y el coeficiente de determinación es alrededor del 60%, lo que indica que el modelo explica el 60% de la variabilidad de los datos.

A continuación vamos a ver cómo se pueden extraer, de un modo independiente, algunos de los resultados ya obtenidos. Si queremos acceder a los residuales utilizamos la función **resid**:

```
> resid(modelo)
1          2          3          4          5          6          7
2847.92601   241.07062    28.52540 -1960.23708 -2567.56652 -1440.94124   888.08378
 8          9         10         11         12         13         14
-843.23439  -403.11950  3476.98296   943.15289
-72.76628   226.93077  -918.69194
15         16         17         18         19
 2122.90806   581.34120  3099.16944 -4069.36877 -2180.16542
```

Si queremos acceder exclusivamente a los coeficientes hacemos:

```
> coef(modelo)
(Intercept)           R
-52.0126660   0.0947444
```

Vamos a obtener ahora los valores ajustados al modelo; es decir los valores $\hat{V} = -52.01282 + 0.0947444 \cdot R$ para cada uno de los valores de R de la muestra.

```
> fitted(modelo)
  1        2        3        4        5        6        7        8
 5652.074  5158.929  4076.475  7960.237  3117.567  2140.941  9611.916  6343.234
    9       10       11       12       13       14       15       16
4903.119  3223.017  6106.847  7072.766  4273.069 11418.692  5877.092  3218.659
   17       18       19
 2600.831  4069.369  2180.165
```

Calculemos las predicciones cuando R=70000 y R=90000:

```
> predict(modelo,data.frame(R=c(70000,90000)))
     1        2
6580.095 8474.983
```

Veamos, por último, cómo se pueden obtener intervalos de confianza para la media y para la predicción, respectivamente, cuando R toma los valores anteriores :

```
> predict(modelo,data.frame(R=c(70000,90000)),
+level=0.9,interval="confidence")
   fit     lwr      upr
1 6580.095 5638.20 7521.991
2 8474.983 7102.65 9847.317
```

```
> predict(modelo,
+ data.frame(R=c(70000,90000)),
+ level=0.9,interval="prediction")
      fit      lwr      upr
1 6580.095 2878.538 10281.65
2 8474.983 4641.231 12308.74
```

Se observa que los intervalos de confianza de las predicciones tienen mayor amplitud que los de las medias.

8.4 ANÁLISIS DE LA VARIANZA CON UN FACTOR

El test t de Student que hemos visto en el capítulo 7 se emplea para comparar las medias de dos muestras independientes provenientes de dos poblaciones normales. El análisis de la varianza permite hacer ese mismo contraste para más de dos muestras.

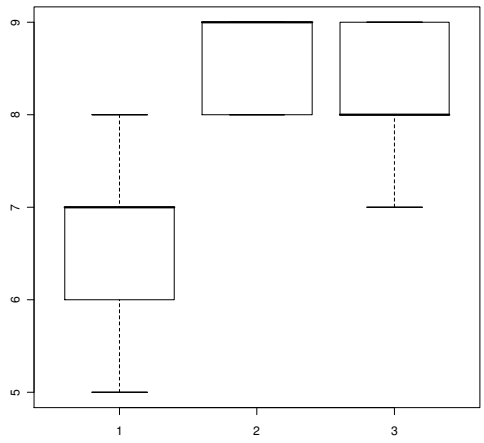
Mediante la resolución del ejemplo 8-2 veremos, a un nivel introductorio, cómo se puede llevar a cabo un análisis de la varianza de un factor.

Ejemplo 8-2 Una industria de fabricación de calzado está considerando tres métodos alternativos para el adiestramiento de sus operarios en una determinada técnica. Para ello ha organizado aleatoriamente tres grupos, de 5 operarios cada uno, a los que ha preparado en cada uno de los métodos. Finalizado el período de adiestramiento los operarios han sido sometidos a una prueba común, obteniéndose las siguientes puntuaciones:

Método A	6	7	8	7	5
Método B	9	9	8	9	8
Método C	7	9	8	8	9

¿Existen diferencias significativas entre los tres métodos?

```
> #Ejemplo 8-2
> #En primer lugar vamos a introducir los datos

> A<-c(6,7,8,7,5)
> B<-c(9,9,8,9,8)
> C<-c(7,9,8,8,9)

> #A continuación obtenemos los diagramas de cajas para comparar las tres distribuciones

> boxplot(A,B,C)
```



> #Ahora debemos crear el marco adecuado de datos para efectuar un análisis de la varianza.

> x<-data.frame(A,B,C)
> x

	A	B	C
1	6	9	7
2	7	9	9
3	8	8	8
4	7	9	8
5	5	8	9

> datos<-stack(x)
> datos

	values	ind
1	6	A
2	7	A
3	8	A
4	7	A
5	5	A
6	9	B
7	9	B
8	8	B
9	9	B
10	8	B
11	7	C
12	9	C
13	8	C
14	8	C
15	9	C

El marco de datos generado es el sistema de organización de los datos más adecuado para llevar a cabo un análisis de la varianza, pues los valores de la misma variable están en la misma columna. El análisis de la varianza se lleva a cabo mediante la función **aov()**.

```
> an.var.datos<-aov(values~ind,data=datos)
> an.var.datos
Call:
  aov(formula = values ~ ind, data = valores)
Terms:
                ind Residuals
Sum of Squares  11.2       9.2
Deg. of Freedom   2        12
Residual standard error: 0.875595
Estimated effects may be unbalanced

> summary(an.var.datos)
            Df Sum Sq Mean Sq F value  Pr(>F)
ind          2   11.2   5.600   7.304 0.00841 **
Residuals   12    9.2   0.767
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

De acuerdo a los resultados obtenidos, y como el p-valor es muy pequeño, podemos rechazar la hipótesis nula de igualdad de los métodos de adiestramiento. Sin entrar en más consideraciones, esta conclusión también parece deducirse de la observación directa de los diagramas de cajas.

8.5 EJERCICIOS RESUELTOS

●**ER 8-1** Habiéndose medido la proporción de cloruro potásico Y a diversas profundidades X en un determinado pozo, se obtuvieron los siguientes valores:

X	20	25	30	35	40	45	50	55	60
Y	12	11,1	10,5	8,4	9	6,5	6	6,2	5,8

Determinar la recta de regresión de Y sobre X y contrastar si la profundidad influye en la proporción de cloruro potásico.

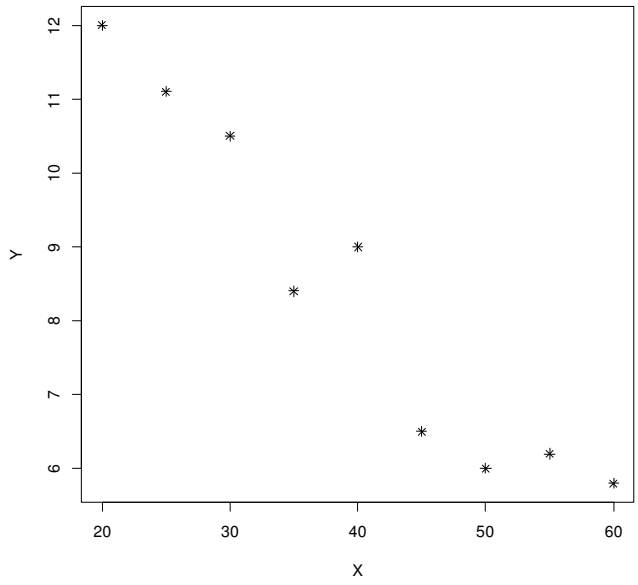
```
> #Los datos están en el archivo “ER8-1.txt”
> datos<-read.table("C:/ER8-1.txt",header=T)
```

> datos

```
  X    Y
1 20 12.0
....
9 60  5.8
```

> attach(datos)

> plot(datos,pch=8)



> modelo<-lm(Y~X)

> summary(modelo)

```
Call:
lm(formula = Y ~ X)
Residuals:
  Min      1Q  Median      3Q     Max
-1.0489 -0.7089  0.2511  0.4311  0.7711
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 15.10889    0.78397  19.272 2.52e-07 ***
X           -0.16800    0.01865  -9.007 4.24e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 0.7224 on 7 degrees of freedom
Multiple R-Squared: 0.9206,     Adjusted R-squared: 0.9092
F-statistic: 81.13 on 1 and 7 DF,  p-value: 4.244e-05
```

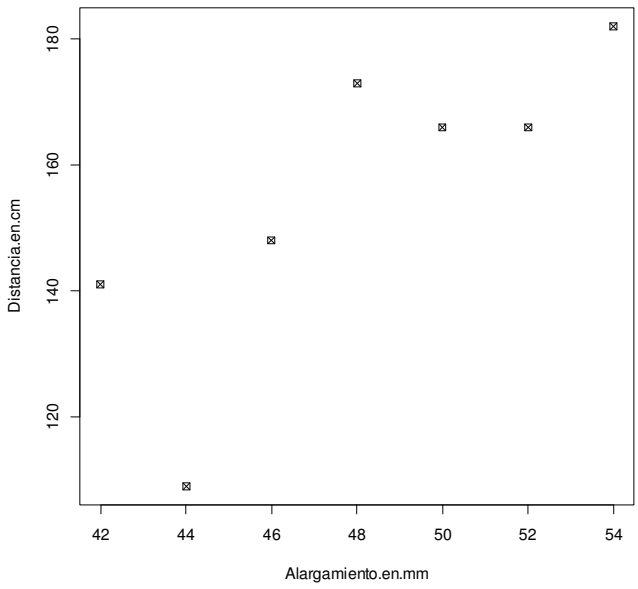
> #De la salida de resultados se deduce: 1º) Recta de regresión: Y=15.10889-0.168*X. 2º) Como el p-valor para el contraste de la pendiente de regresión, 4.24e-05, es muy pequeño se acepta la hipótesis de que la profundidad influye en la concentración de cloruro potásico

●**ER 8-2** Los datos de la tabla son 7 pares de observaciones del alargamiento de una banda elástica sometida a tracción y de la correspondiente distancia que recorre la banda cuando es liberada:

Alargamiento en mm	46	54	48	50	44	42	52
Distancia en cm	148	182	173	166	109	141	166

Se pide: 1º) Diagrama de dispersión. 2º) Recta de regresión de la distancia sobre el alargamiento. 3º) Intervalo de confianza al nivel del 90% para la predicción de la distancia cuando el alargamiento es 47 cm.

> datos<-read.table("C:/ER8-2.txt",header=T)
> datos
Alargamiento.en.mm Distancia.en.cm
1 46 148
2 54 182
....
7 52 166
A<-datos$Alargamiento.en.mm
D<-datos$Distancia.en.cm
> plot(A,D,pch=7)

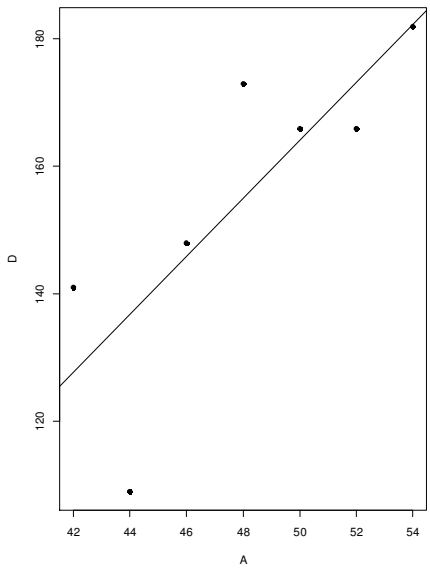


> summary(lm(D~A))

```
Call:
lm(formula = D ~ A)
Residuals:
     1       2       3       4       5       6       7
 2.1071  -0.3214  18.0000   1.8929 -27.7857  13.3214  -7.2143
Coefficients:
   Estimate Std. Error t value Pr(>|t|)
(Intercept)  -63.571     74.332  -0.855   0.4315
A              4.554      1.543   2.951   0.0319 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 16.33 on 5 degrees of freedom
Multiple R-Squared: 0.6352,     Adjusted R-squared: 0.5622
F-statistic: 8.706 on 1 and 5 DF,  p-value: 0.03186
```

> #La recta de regresión de la distancia D sobre el alargamiento A es:
D=-63.571+4.554*A

> abline(lm(D~A))



> predict(lm(D~A),data.frame(A=47),level=0.9,interval="prediction")

```
    fit     lwr     upr
[1,] 150.4464 115.1271 185.7657
```

●**ER 8-3** Las puntuaciones obtenidas por 10 individuos en una prueba de rendimiento Y, según el número de horas de práctica X, fueron:

X	5	5	6	6	6	7	7	11	11	16
Y	25	30	30	35	45	40	45	55	60	65

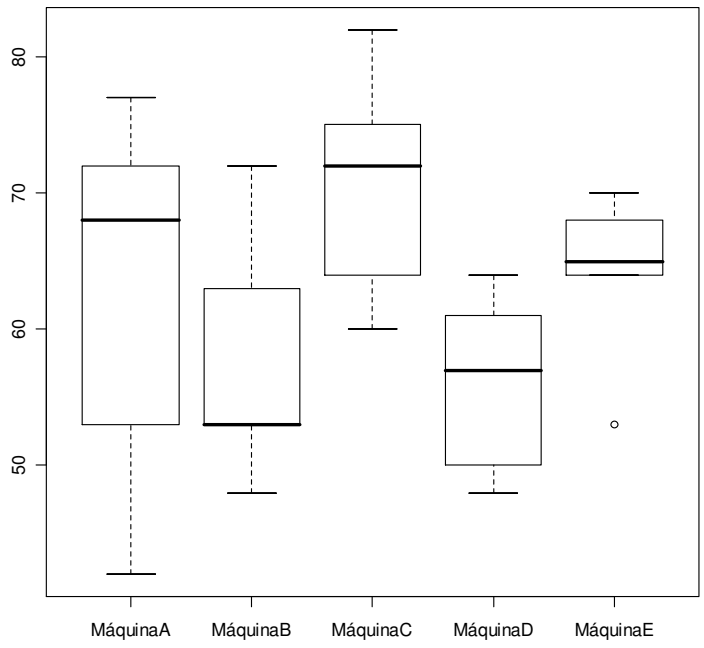
Estimar el rendimiento mediante un intervalo de confianza al 95% para 50 horas de práctica. Ídem al 99%.

```
> datos<-read.table("C:/ER8-3.txt",header=T)
> datos
  X  Y
1   5 25
2   5 30
3   6 30
4   6 35
....
9  11 60
10 16 65
> attach(datos)
> predict(lm(Y~X),data.frame(X=50),
+ level=0.95,interval="prediction")
     fit      lwr      upr
[1,] 188.5263 131.8712 245.1815

> predict(lm(Y~X),data.frame(X=50),
+ level=0.99,interval="prediction")
      fit      lwr      upr
[1,] 188.5263 106.0894 270.9633

> #Como es obvio, el intervalo de confianza al nivel 99% es de mayor amplitud
que el obtenido al 95%
```

●**ER 8-4** Una empresa desea elegir una máquina entre 5 diferentes. Cada una de ellas ha sido manejada por un operario experto distinto durante tiempos iguales. La tabla muestra los números de unidades producidas por las máquinas. Contrastar la hipótesis de que no hay diferencias entre ellas.

Máquina A	Máquina B	Máquina C	Máquina D	Máquina E
68	72	60	48	64
72	53	82	61	65
77	63	64	57	70
42	53	75	64	68
53	48	72	50	53

```
> datos<-read.table("C:/ER8-4.txt",header=T)
> datos
  MáquinaA MáquinaB MáquinaC MáquinaD MáquinaE
1       68       72       60       48       64
2       72       53       82       61       65
3       77       63       64       57       70
4       42       53       75       64       68
5       53       48       72       50       53
> boxplot(datos)
```




```
> df<-stack(datos)
> df
   values      ind
1      68 MáquinaA
2      72 MáquinaA
3      77 MáquinaA
4      42 MáquinaA
5      53 MáquinaA
6      72 MáquinaB
7      53 MáquinaB
8      63 MáquinaB
9      53 MáquinaB
10     48 MáquinaB
```

```
11    60 MáquinaC
12    82 MáquinaC
13    64 MáquinaC
14    75 MáquinaC
15    72 MáquinaC
16    48 MáquinaD
17    61 MáquinaD
18    57 MáquinaD
19    64 MáquinaD
20    50 MáquinaD
21    64 MáquinaE
22    65 MáquinaE
23    70 MáquinaE
24    68 MáquinaE
25    53 MáquinaE
```

```
> análisis.varianza<-aov(values~ind,data=df)
> análisis.varianza
Call:
  aov(formula = values ~ ind, data = df)
Terms:
            ind Residuals
Sum of Squares   658.16   1883.20
Deg. of Freedom       4        20
Residual standard error: 9.703608
Estimated effects may be unbalanced
```

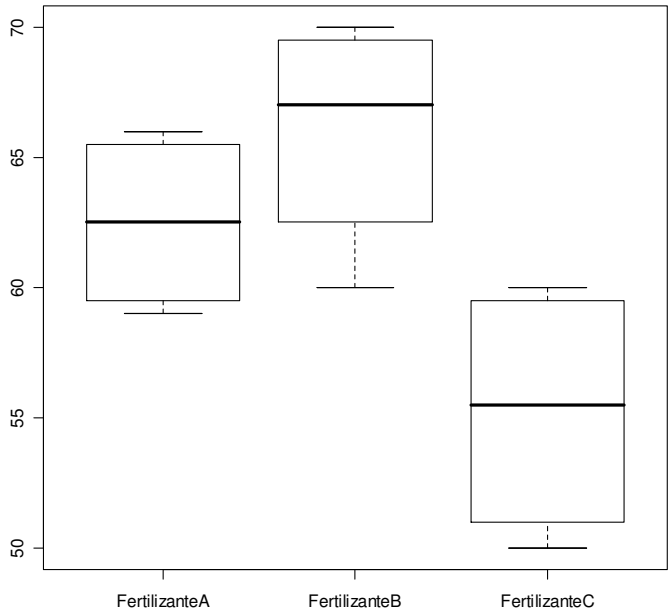
```
> summary(análisis.varianza)
            Df  Sum Sq Mean Sq F value Pr(>F)
ind          4  658.16  164.54  1.7475 0.1792
Residuals   20 1883.20   94.16
```

```
> #Como el p-valor es alto no hay evidencia suficiente para afirmar que hay diferencias entre las máquinas
```

●**ER 8-5** Efectuar un análisis de la varianza de los datos siguientes sobre rendimientos de una cosecha de maíz, en relación al tipo de fertilizante empleado:

Fertilizante A	Fertilizante B	Fertilizante C
60	70	50
65	69	59
59	65	60
66	60	52

```
> ej8.5<-read.table("C:/ER8-5.txt",header=T)
> ej8.5
 FertilizanteA FertilizanteB FertilizanteC
1            60            70            50
2            65            69            59
3            59            65            60
4            66            60            52
> boxplot(ej8.5)
```




```
> ej8.5<-stack(ej8.5)
> ej8.5
   values          ind
1      60 FertilizanteA
2      65 FertilizanteA
3      59 FertilizanteA
4      66 FertilizanteA
5      70 FertilizanteB
6      69 FertilizanteB
7      65 FertilizanteB
8      60 FertilizanteB
9      50 FertilizanteC
10     59 FertilizanteC
11     60 FertilizanteC
12     52 FertilizanteC
```

```
> an.var.ej8.5<-aov(values~ind,data=ej8.5)

> an.var.ej8.5
Call:
 aov(formula = values ~ ind, data = ej8.5)
Terms:
                ind Residuals
Sum of Squares  240.50    173.75
Deg. of Freedom      2         9

Residual standard error: 4.393809
Estimated effects may be unbalanced

> summary(an.var.ej8.5)
      Df  Sum Sq Mean Sq F value  Pr(>F)
ind           2 240.500 120.250  6.2288 0.02004 *
Residuals     9 173.750  19.306
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

> #Como el p-valor está comprendido entre 0.01 y 0.05 la hipótesis alternativa de diferencias entre los fertilizantes no tiene un respaldo suficiente. Sería conveniente, si es posible, repetir el experimento con muestras de mayor tamaño.

8.6 EJERCICIOS PROPUESTOS

●**EP 8-1**: Cargar el paquete **HSAUR** y explicar qué es el marco de datos **water**. Obtener un diagrama de dispersión de las variables **hardness** y **mortality**. Repetir lo anterior separando en dos grupos de acuerdo a la variable **location**.

●**EP 8-2** Calcular las tres rectas de regresión correspondientes al ejercicio anterior.

●**EP 8-3** Cargar el paquete **ISwR** y considerar el conjunto de datos **thuesen**. Se pide: 1º) Diagrama de dispersión. 2º) Recta de regresión de **short.velocity** sobre **blood.glucose**. 3º) Contrastar si la segunda variable influye en la primera. 4º) Hacer un gráfico de los residuos frente a los valores ajustados.

●**EP 8-4** Una cadena de librerías tiene tres establecimientos en una determinada ciudad. Para averiguar si las ventas presentan diferencias significativas se anotaron las ventas de cuatro lunes consecutivos en cada uno de los establecimientos, obteniéndose los resultados siguientes:

Establecimiento 1	25	23	25	27
Establecimiento 2	23	25	26	27
Establecimiento 3	27	29	25	28

Contrastar al nivel α=0.05 la hipótesis de igualdad de medias en los tres establecimientos.

●**EP 8-5** Efectuar un análisis de la varianza para el siguiente conjunto de valores simulado: vector A generado por la sentencia **rnorm(10,20,2)**, vector B formado por el doble de cada elemento de A y vector C formado por el cuádruple de cada elemento de A. Obtener, así mismo, el correspondiente diagrama boxplot. Interpretar los resultados.

2ª PARTE

EXPERIMENTACIÓN CON *R*

Capítulo 9

EXPERIMENTACIÓN CON *R*

9.1 INTRODUCCIÓN

Podría asegurarse, sin lugar a dudas, que este es el capítulo central del texto. En él dirigiremos nuestra atención a *R* desde el punto de vista de la experimentación, como si se tratara de un laboratorio donde poder realizar ensayos libremente.

La importancia de *R* no radica exclusivamente en su capacidad para efectuar análisis de datos y análisis estadísticos, sino en su enorme potencial para experimentar. El programa es un auténtico laboratorio en el que podemos efectuar cualquier experiencia en estadística y probabilidad, dado que dispone de un lenguaje de programación sencillo. En esta característica reside la gran potencia de *R*, en su flexibilidad para simular con sencillez y rapidez.

Con la llegada de los ordenadores creció exponencialmente la utilización de los números aleatorios, algo imprescindible para el desarrollo de experimentos de simulación, necesarios en muchas situaciones en las que no se pueden determinar con exactitud las ecuaciones que describen el sistema. Como la mayor parte de las experiencias que se van a presentar en este capítulo están basadas en el manejo de grandes cantidades de números aleatorios, utilizaremos con profusión funciones tales como **runif()**, **rnorm()**, **sample()**, etc.

- La familia de funciones "apply"

Antes de empezar con las experiencias de simulación vamos a describir una familia de funciones muy útiles que sustituyen, de forma eficaz en muchos casos, a los bucles de programación **for()**, **while()**, etc.

La función **apply()** devuelve un vector o lista de valores obtenida al aplicar una función a las filas o columnas de una matriz (o a los índices de un *array* multidimensional, que es la generalización de una matriz a más de dos dimensiones).

```
> #Creamos una matriz de 5 filas y 3 columnas
> m<-matrix(c(1:15),nrow=5,ncol=3)
> m
     [,1] [,2] [,3]
[1,]    1    6   11
[2,]    2    7   12
[3,]    3    8   13
[4,]    4    9   14
[5,]    5   10   15
> #Obtenemos la media de las columnas (índice 2)
> apply(m,2,mean)
[1]  3  8 13
```

Para imputar esta orden a todos los elementos de la matriz debemos indicar el vector de índices **c(1,2)**, es decir, las filas y columnas:

```
> f<-function(x) 3*x
> #Aplicamos la función f a todos los elementos de la matriz m y obtenemos el
triple de cada  valor
> apply(m,c(1,2),f)
     [,1] [,2] [,3]
[1,]    3   18   33
[2,]    6   21   36
[3,]    9   24   39
[4,]   12   27   42
[5,]   15   30   45
```

Para aplicar una función a cada elemento de un marco de datos o a un vector, podemos utilizar las funciones **lapply()** y **sapply()**. La primera de ellas devuelve una lista y la otra trata de simplificar la salida a un vector o una matriz, si es posible. Supongamos, por ejemplo, que se desea obtener las medias de cada una de las variables del marco de datos **cars**, perteneciente al paquete **datasets**:

```
> library(datasets)
> data(cars)
> cars
speed dist
1     4    2
2     4   10
3     7    4
...
50    25   85
> lapply(cars,mean)
$speed
[1] 15.4
$dist
[1] 42.98
> sapply(cars,mean)
speed  dist
15.40 42.98

> #A continuación vamos a utilizar sapply() para obtener los cuadrados de los elementos de un vector
> f<-function(x) x^2
> a<-c(1,2,3,4,5)
> sapply(a,f)
[1]  1  4  9 16 25
```

Una función muy útil en la simulación de sucesos que conllevan la generación de números aleatorios es **replicate()**. Se suele utilizar cuando se trata de evaluar repetidamente una expresión. Por ejemplo, si queremos calcular la media de 200 valores elegidos al azar de una distribución N(10,2), y esa operación repetirla 5 veces, hacemos:

```
> replicate(5,mean(rnorm(200,10,2)))
 [1]  9.933356 10.005959  9.922064 10.055265 10.190314
> #Ídem para una distribución N(10,20)
> replicate(5,mean(rnorm(200,10,20)))
[1]  9.633658 10.535978  9.284357  8.052079  7.034240
```

La función **tapply()** permite crear tablas con el valor de una función sobre subgrupos definidos por un segundo argumento (**INDEX**), que puede ser un factor o una lista de factores. Por

ejemplo, supongamos que se desea calcular la circunferencia media, según el tipo de árbol (1, 2, 3, 4 o 5), de los valores de la variable **circumference** que aparecen en el marco de datos **Orange**:

```
> library(datasets)
> data(Orange)
> Orange
   Tree  age circumference
1     1  118            30
2     1  484            58
3     1  664            87
....
35    5 1582           177
> tapply(Orange$circumference,INDEX=Orange$Tree,FUN=mean)
      3       1       5       2       4
 94.00000  99.57143 111.14286 135.28571 139.28571
```

Para poner de manifiesto la ventaja que supone emplear estas funciones, en lugar de bucles de programación, veamos un ejemplo muy sencillo. Se trata de simular el lanzamiento 10 veces de un par de dados y después calcular el valor máximo en cada lanzamiento:

```
> dado<-1:6
> A<-matrix(sample(dado,20,replace=T),nrow=10,ncol=2)
> A
      [,1] [,2]
 [1,]    4    1
 [2,]    3    5
 [3,]    3    5
 [4,]    2    3
 [5,]    1    6
 [6,]    1    1
 [7,]    5    4
 [8,]    5    1
 [9,]    2    6
[10,]    2    1

> #Usamos un bucle for()
> val.max.BU<-numeric(10)
> for (i in 1:10)
+ val.max.BU[i]<-max(A[i,])
> val.max.BU
 [1] 4 5 5 3 6 1 5 5 6 2
```

```
> #Usamos la función apply()
> val.max.AP<-apply(A,1,max)
> val.max.AP
 [1] 4 5 5 3 6 1 5 5 6 2
```

Funciones adicionales pertenecientes a esta familia son **by()**, **eapply()**, **mapply()**, **vapply()**, **rapply()**, y otras incluidas en el paquete **plyr**.

- La función set.seed()

En ocasiones es necesario que los resultados de una simulación sean reproducibles. Teniendo en cuenta que los números aleatorios que se generan por ordenador son, en realidad, pseudo-aleatorios, introduciendo una "semilla" (un entero positivo) mediante la función **set.seed()**, lograremos obtener los mismos valores cada vez que efectuemos un ensayo. Veamos un ejemplo:

```
> rnorm(5)
[1]  1.11576202 -0.30224775 -0.07175354 -0.77251049  1.20946719
> rnorm(5)
[1]  0.35532061  0.04166434  0.13922733 -0.16869070  0.81364495
> #Se observa que en cada simulación se obtienen valores diferentes

> set.seed(1)
> rnorm(5)
[1] -0.6264538  0.1836433 -0.8356286  1.5952808  0.3295078
> set.seed(1)
> rnorm(5)
[1] -0.6264538  0.1836433 -0.8356286  1.5952808  0.3295078
> set.seed(12)
> rnorm(5)
[1] -1.4805676  1.5771695 -0.9567445 -0.9200052 -1.9976421
> set.seed(12)
> rnorm(5)
[1] -1.4805676  1.5771695 -0.9567445 -0.9200052 -1.9976421
> #Ahora, cada vez que simulamos, obtenemos los mismos valores para la misma semilla
```

A continuación se presentan 20 experiencias sobre estadística y probabilidad llevadas a cabo con el programa *R.* Esperamos que sirvan para animar al lector a desarrollar sus propios experimentos, con el objetivo último de mejorar la comprensión de los fenómenos aleatorios.

9.2 INFLUENCIA DE LOS DATOS ATÍPICOS EN LAS MEDIDAS DE CENTRALIZACIÓN Y DE DISPERSIÓN

En esta experiencia se observa como la existencia de datos atípicos influye enormemente en parámetros como la media y la varianza, que son parámetros no robustos, y muy poco en otros más robustos como la mediana, la media recortada y el recorrido intercuartílico.

```
> datos1<-c(6,7,11,9,13,12,12,6,13,13,10,11,11,11,13,6,12,10,8,13)
> mean(datos1);median(datos1);mean(datos1,trim=0.1)
[1] 10.35
[1] 11
[1] 10.5625
> var(datos1);sd(datos1);IQR(datos1)
[1] 6.344737
[1] 2.518876
[1] 3.5

> #Introducimos el dato atípico 1000 y la media y varianza cambian sensiblemente, pero no la mediana, ni la media recortada, ni el recorrido intercuartílico
>datos2<-c(6,7,11,9,13,12,12,6,13,13,10,11,11,11,13,6,12,10,8,13,1000)
> mean(datos2);median(datos2);mean(datos2,trim=0.1)
[1] 57.47619
[1] 11
[1] 10.70588
> var(datos2);sd(datos2);IQR(datos2)
[1] 46644.46
[1] 215.9733
[1] 4
```

9.3 LEY EMPÍRICA DE ESTABILIDAD DE LAS FRECUENCIAS

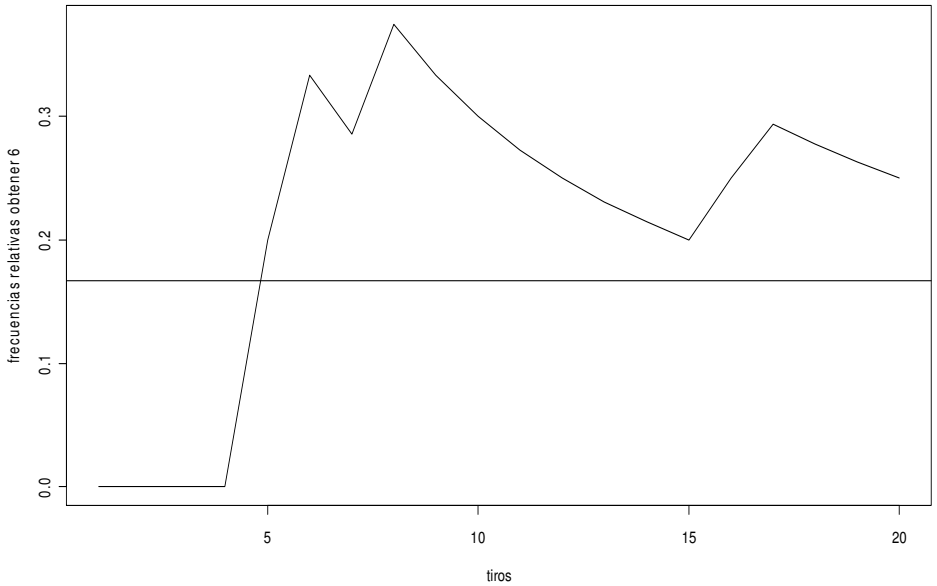
En todo fenómeno aleatorio, al considerar un gran número de pruebas, se observa una *regularidad* en la frecuencia de un suceso, en el sentido de que esta tiende a aproximarse a un número fijo al aumentar el número de experiencias. Este hecho se denomina *ley empírica de estabilidad de las frecuencias*. Se trata de una ley empírica que tiene su contrapartida, dentro de la teoría de la probabilidad, en un conjunto de teoremas denominados *Leyes de los grandes números.*

Casi todo el mundo tiene interiorizada en su mente esta ley empírica, pero suele confundirse con una especie de ley de compensación que lleva a pensar que los resultados de sucesivas pruebas de un experimento aleatorio deben compensar resultados anteriores. Ello equivaldría, por ejemplo, a que un dado "recordara" los resultados de tiradas precedentes, algo absurdo

evidentemente. Lo que ocurre en realidad es que, a corto plazo, se produce una variabilidad y a largo plazo una regularidad de las frecuencias relativas; las frecuencias absolutas pueden seguir un comportamiento completamente impredecible.

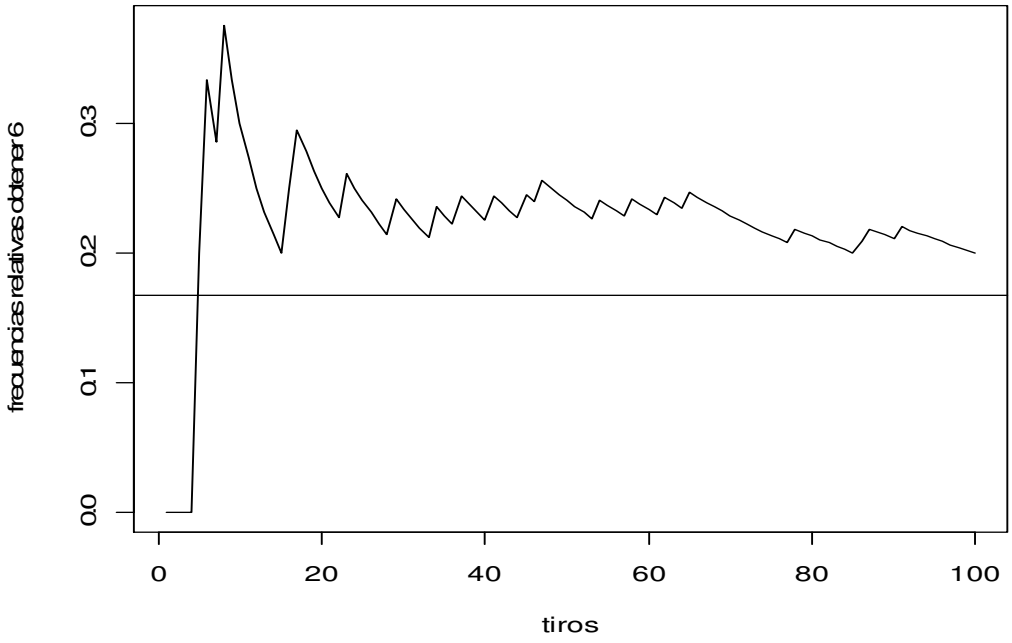
La experiencia que vamos a realizar consiste en lanzar un dado un gran número de veces y calcular la frecuencia relativa del suceso "obtener un 6". Veremos que esta frecuencia se va estabilizando alrededor del valor 1/6.

```
> dado<-1:6
> lanzamientos<-function(tiros)
{
+ resultados<-sample(dado,tiros,replace=T)
+ seises<-resultados==6
+ #Calculamos la frecuencia relativa del suceso "obtener 6"
+ frec.rel<-sum(seises)/tiros
+ #Calculamos las frecuencias relativas acumuladas
+ #y hacemos el gráfico correspondiente
+ frec.rel.ac<-cumsum(seises)/(1:tiros)
+ plot(frec.rel.ac,xlab="tiros",ylab="frecuencias relativas obtener 6",type="l")
+ abline(h=1/6)
}

> set.seed(2) #Para obtener las mismas simulaciones
> lanzamientos(20)
```

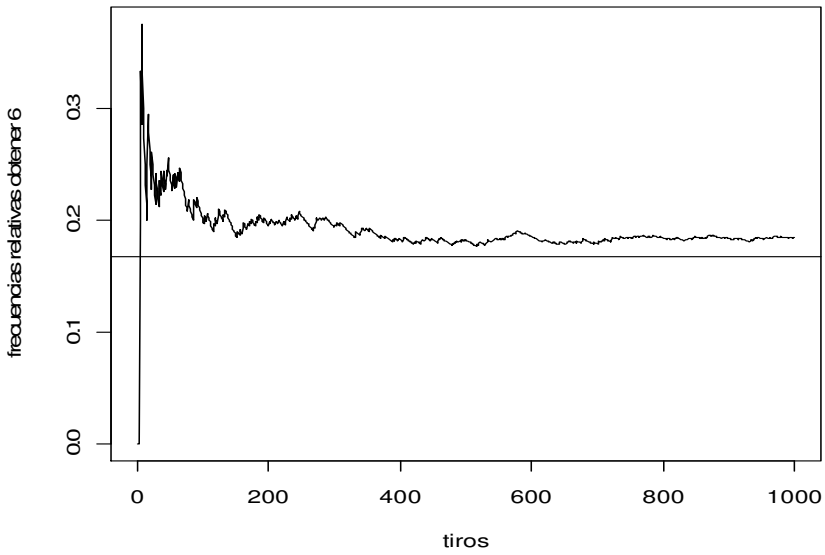


> #Con pocos lanzamientos se obtiene un gráfico con una alta variabilidad. De hecho, podría haber salido cualquier otra curva, aunque si antes de simular establecemos la misma semilla (*seed*) obtenemos exactamente los mismos resultados

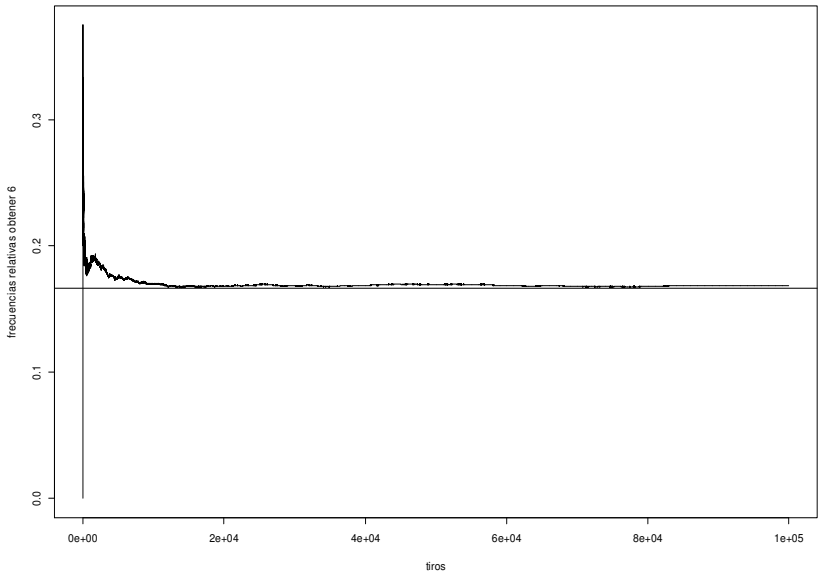
> set.seed(2)
> lanzamientos(100) #Ya se empieza a observar una cierta tendencia a la regularidad



> set.seed(2)
> lanzamientos(1000) #La regularidad es más evidente



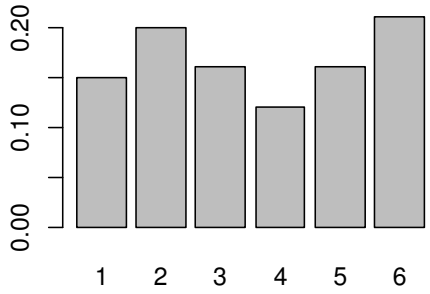
> set.seed(2)
> lanzamientos(100000) #La regularidad a la larga es prácticamente total



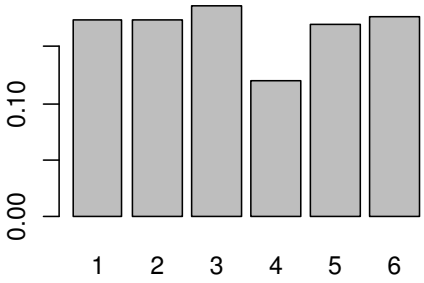
9.4 FRECUENCIAS RELATIVAS EN EL LANZAMIENTO DE UN DADO EQUILIBRADO

En este experimento se vuelve a poner de manifiesto la *ley empírica de estabilidad de las frecuencias.* Aquí observamos que las frecuencias relativas de los seis sucesos elementales del experimento consistente en lanzar un dado equilibrado se van estabilizando, a medida que aumentamos el número de pruebas, hacia el valor 1/6 (sucesos equiprobables).

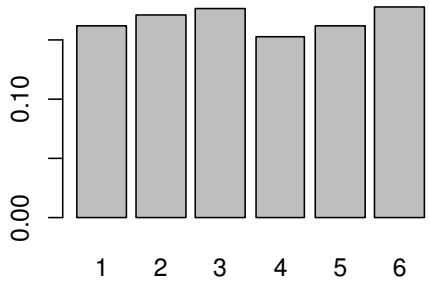
```
> dado<-1:6
> set.seed(5)
> x100<-sample(dado,100,rep=T) #Lanzamos el dado 100 veces
> #Para obtener las frecuencias relativas hacemos lo siguiente
> x100fr<-table(x100)/100

> set.seed(5)
> x500<-sample(dado,500,rep=T) #Lanzamos el dado 500 veces
> x500fr<-table(x500)/500

> set.seed(5)
> x1000<-sample(dado,1000,rep=T) #Lanzamos el dado 1000 veces
> x1000fr<-table(x1000)/1000

> set.seed(5)
> x10000<-sample(dado,10000,rep=T) #Lanzamos el dado 10000    veces
> x10000fr<-table(x10000)/10000

> split.screen(c(2,2))
[1] 1 2 3 4

> screen(1); barplot(x100fr,sub="100 lanzamientos")
> screen(2); barplot(x500fr,sub="500 lanzamientos")
> screen(3); barplot(x1000fr,sub="1000 lanzamientos")
> screen(4); barplot(x10000fr,sub="10000 lanzamientos")
```



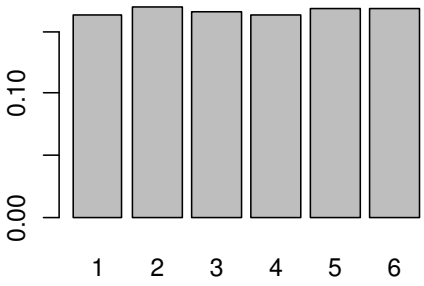
100 lanzamientos



500 lanzamientos



1000 lanzamientos



10000 lanzamientos

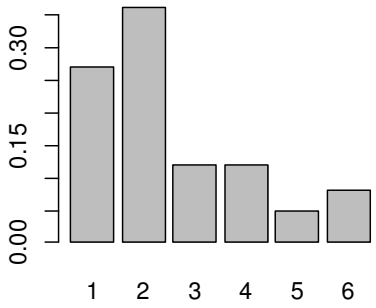
En los sucesivos gráficos se ve claramente que las frecuencias relativas de los distintos sucesos se van aproximado a 1/6.

9.5 FRECUENCIAS RELATIVAS EN EL LANZAMIENTO DE UN DADO NO EQUILIBRADO

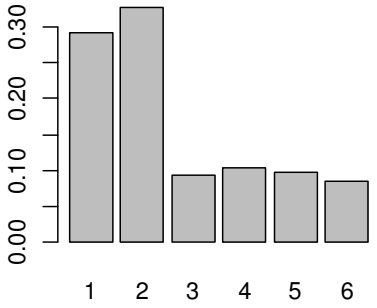
Repetimos la experiencia anterior pero ahora con un dado cargado en el que las probabilidades de los sucesos 1, 2, 3, 4, 5, 6 son, respectivamente, 0.3, 0.3, 0.1, 0.1, 0.1 y 0.1.

```
> dado.ne<-1:6; pr<-c(0.3,0.3,0.1,0.1,0.1,0.1)

> set.seed(13); x100<-sample(dado.ne,100,rep=T,prob=pr) #Lanzamos el dado no equilibrado 100 veces
> x100fr<-table(x100)/100
```

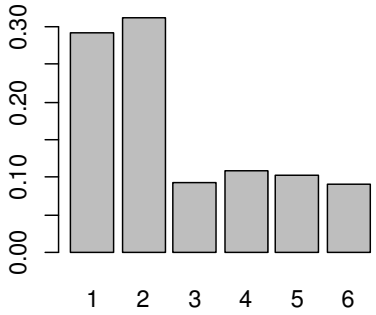
```
> set.seed(13); x500<-sample(dado.ne,500,rep=T,prob=pr) #Lanzamos el dado no equilibrado 500 veces
> x500fr<-table(x500)/500
> set.seed(13); x1000<-sample(dado.ne,1000,rep=T,prob=pr) #Lanzamos el dado no equilibrado 1000 veces
> x1000fr<-table(x1000)/1000
> set.seed(13); x10000<-sample(dado.ne,10000,rep=T,prob=pr) #Lanzamos el dado no equilibrado 10000 veces
> x10000fr<-table(x10000)/10000
> split.screen(c(2,2))
[1] 1 2 3 4
> screen(1); barplot(x100fr,sub="100 lanzamientos dado no equilibrado")
> screen(2); barplot(x500fr,sub="500 lanzamientos dado no equilibrado")
> screen(3); barplot(x1000fr,sub="1000 lanzamientos dado no equilibrado")
> screen(4); barplot(x10000fr,sub="10000 lanzamientos dado no equilibrado")
```



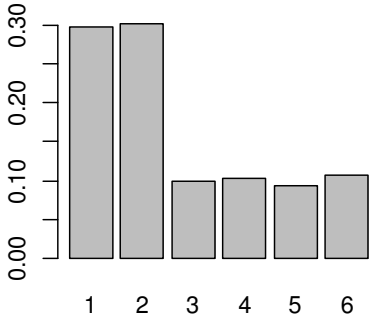
100 lanzamientos dado no equilibrado



500 lanzamientos dado no equilibrado



1000 lanzamientos dado no equilibrado



10000 lanzamientos dado no equilibrado

9.6 ESTIMACIÓN DE UNA PROBABILIDAD MEDIANTE UNA FRECUENCIA A LA LARGA

Ahora vamos a utilizar la *ley empírica de estabilidad de las frecuencias* para estimar la probabilidad de un suceso mediante la evaluación de su frecuencia relativa a la larga. Previamente calcularemos, mediante la regla de *Laplace*, la probabilidad de que al extraer dos cartas de una baraja española de 40 elementos, la suma de los números correspondientes sea menor que 13. Veamos las sumas posibles en el cuadro siguiente:

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

Para calcular la probabilidad pedida habrá que sumar la probabilidad de obtener dos cifras iguales (6 casos) más la probabilidad de obtener dos cifras diferentes (46 casos):

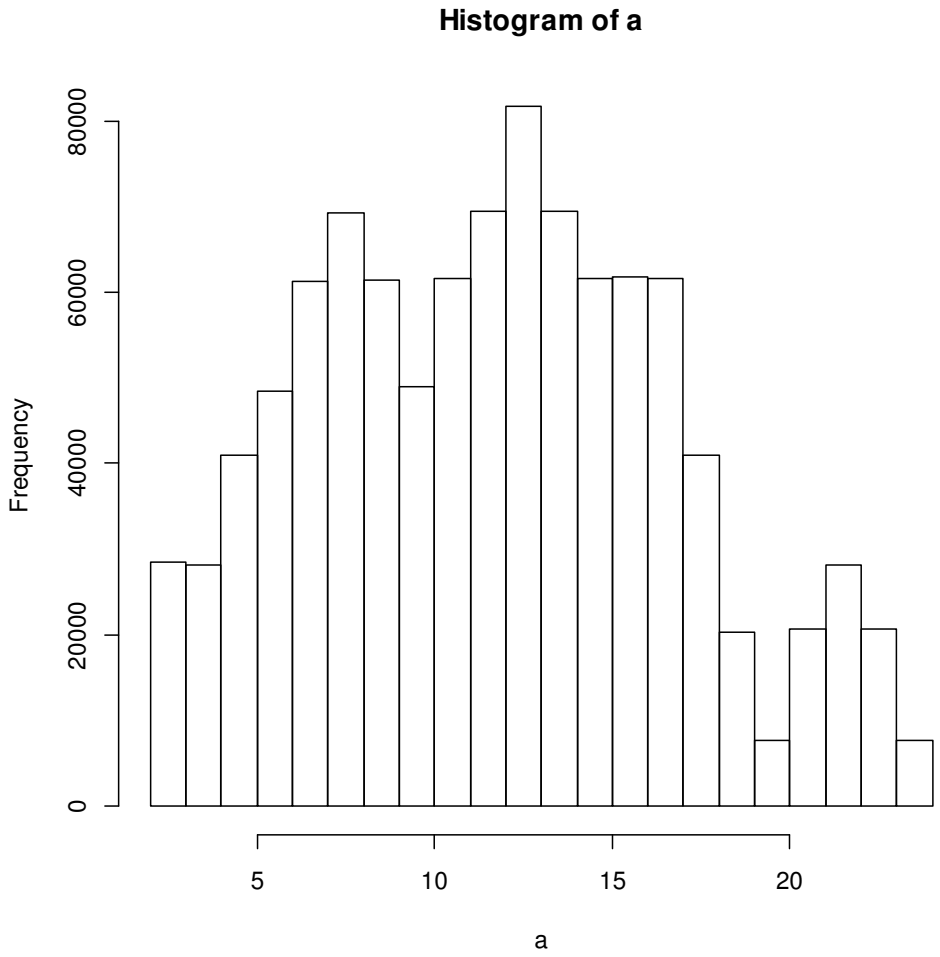
$$\frac{4}{40}\cdot\frac{3}{39}\cdot 6+\frac{4}{40}\cdot\frac{4}{39}\cdot 46=0,518$$

Veamos cómo se puede obtener aproximadamente el valor anterior simulando con *R*:

```
> #No tenemos en cuenta el palo: hay 4 cartas con el 1, 4 cartas con el 2..., hasta
4 cartas con el 12
> cartas<-c(rep(1,4),rep(2,4),rep(3,4),rep(4,4),rep(5,4),
+ rep(6,4),rep(7,4),rep(10,4),rep(11,4),rep(12,4))
> a<-replicate(1000000,sum(sample(cartas,2)))
> sum(a<13)/1000000
[1] 0.518035
```

El histograma de los valores de **a**, que aquí se han obtenido a través de un millón de simulaciones (recuérdese que al no establecer la *semilla* con la función **set.seed()** cada conjunto de ensayos resultará diferente), es

> hist(a)

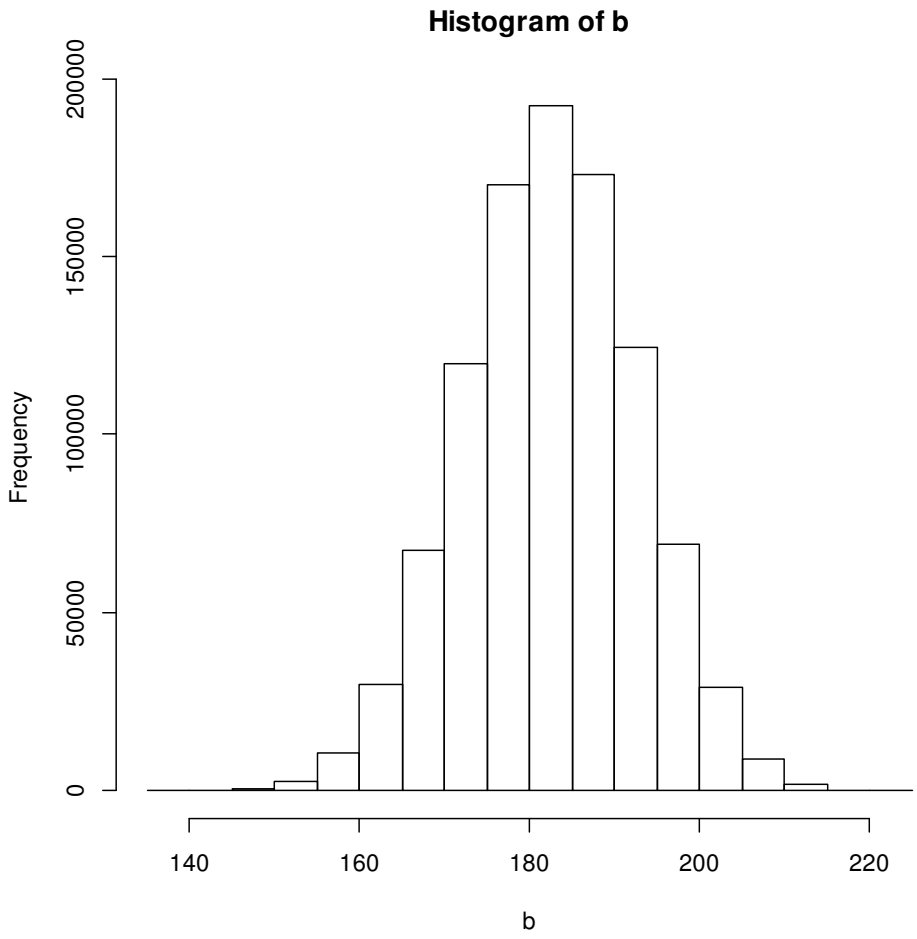


Supongamos ahora que queremos resolver el mismo problema anterior con números más grandes. Supongamos, por ejemplo, que estamos interesados en obtener la probabilidad de que al extraer treinta cartas de la baraja la suma de los números correspondientes sea menor que 200. Evidentemente, calcular esta probabilidad de manera exacta, utilizando la regla de Laplace, es algo prácticamente inabordable, pero en *R*, adaptando las sentencias anteriores de forma adecuada, se obtiene una aproximación de esta probabilidad muy fácilmente:

```
b<-replicate(1000000,sum(sample(cartas,30)))
sum(b<200)/1000000
[1] 0.950133
```

El histograma de los valores de **b**, que ahora resulta más parecido a una normal pues se extraen 30 cartas en lugar de 2, es el siguiente:

```
> hist(b)
```



9.7 PROBLEMA DEL CUMPLEAÑOS

Este clásico problema se plantea del siguiente modo: calcular la probabilidad de que en un grupo de *n* personas haya por lo menos dos que cumplan años el mismo día. Para calcular esta probabilidad es mejor obtener la probabilidad del suceso contrario (ninguna coincidencia) y luego restar a 1 ese valor:

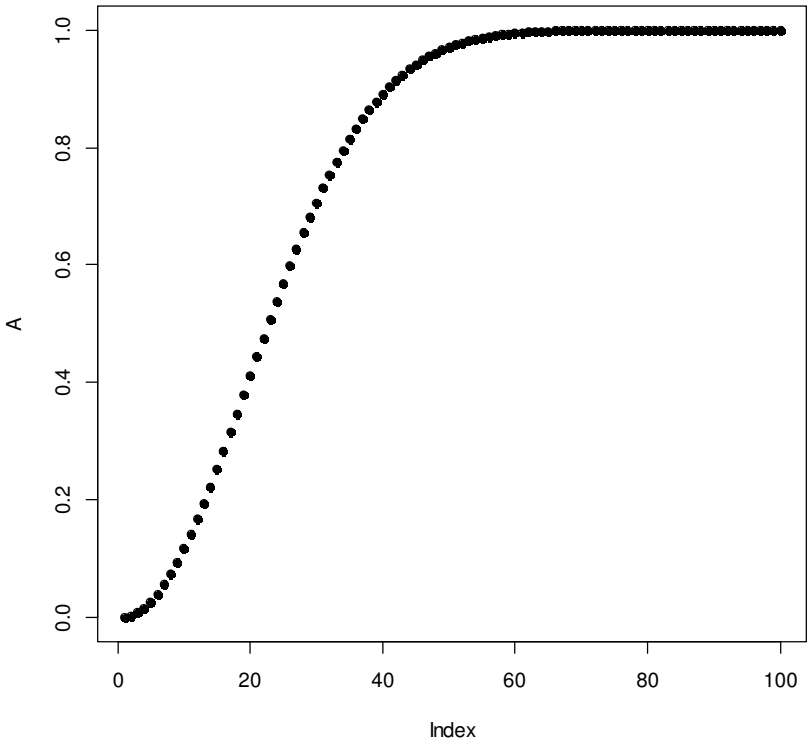
$$1-\frac{365\cdot 364\cdot \ldots \cdot (365-n+1)}{365^n}=1-\frac{365!}{365^n(365-n)!}=1-\frac{n!\binom{365}{n}}{365^n}$$

Calculemos en *R* la probabilidad anterior para un grupo de 20 personas:

> 1-factorial(20)*choose(365,20)/365^20
[1] 0.4114384

Para obtener un gráfico que muestre estas probabilidades, en función del número de individuos hacemos:

> prob<-function(n) 1-factorial(n)*choose(365,n)/365^n
> prob(20)
[1] 0.4114384
> A<-sapply(1:100,prob)
> plot(A,type='p',pch=19)



> #Como se observa, la probabilidad se aproxima rápidamente a 1, a partir de un grupo de unas 40 personas

> #Vamos a efectuar una simulación con un grupo de 20 individuos
> ENE<-paste(1:31,"ene")
> FEB<-paste(1:28,"feb")
> MAR<-paste(1:31,"mar")
> ABR<-paste(1:30,"abr")

```
> MAY<-paste(1:31,"may")
> JUN<-paste(1:30,"jun")
> JUL<-paste(1:31,"jul")
> AGO<-paste(1:31,"ago")
> SEP<-paste(1:30,"sep")
> OCT<-paste(1:31,"oct")
> NOV<-paste(1:30,"nov")
> DIC<-paste(1:31,"dic")
> fechas<-c(ENE,FEB,MAR,ABR,MAY,JUN,
+ JUL,AGO,SEP,OCT,NOV,DIC)

> #Fechas de nacimiento de un grupo de 20 personas
> m<-sample(fechas,20,replace=T)
> m
 [1] "26 ago" "8 mar"  "8 feb"  "22 mar" "11 jun" "2 feb"  "14 may" "29 jul"
[9] "6 jun"  "8 nov"  "7 jul"  "20 ago" "4 sep"  "2 abr"  "19 oct" "30 mar"
[17] "4 mar"  "6 oct"  "30 dic" "29 ene"
> table(m)
n
11 jun 14 may 19 oct  2 abr  2 feb 20 ago 22 mar 26 ago 29 ene 29 jul 30 dic
   1      1      1      1      1      1      1      1      1      1      1
30 mar  4 mar  4 sep  6 jun  6 oct  7 jul  8 feb  8 mar  8 nov
  1      1      1      1      1      1      1      1      1
> #No ha habido ninguna coincidencia

> #Hagamos otra prueba
> #Fechas de nacimiento de otro grupo de 20 personas
> m<-sample(fechas,20,replace=T)
> m
 [1] "19 ago" "10 mar" "11 jul" "11 mar" "23 abr" "19 nov" "29 may" "30 ene"
 [9] "1 mar"  "13 oct" "4 feb"  "11 mar" "10 feb" "30 abr" "22 dic" "11 may"
[17] "13 jun" "18 jul" "19 ago" "7 jun"
> table(m)
n
1 mar 10 feb 10 mar 11 jul 11 mar 11 may 13 jun 13 oct 19 ago 18 jul 19 nov
  1      1      1      1      2      1      1      1      2      1      1
22 dic 23 abr 29 may 30 abr 30 ene  4 feb  7 jun
  1      1      1      1      1      1      1

> #Ha habido una coincidencia el 11 de marzo y otra el 19 de agosto
```

9.8 LA DISTRIBUCIÓN DE CAUCHY CARECE DE MEDIA

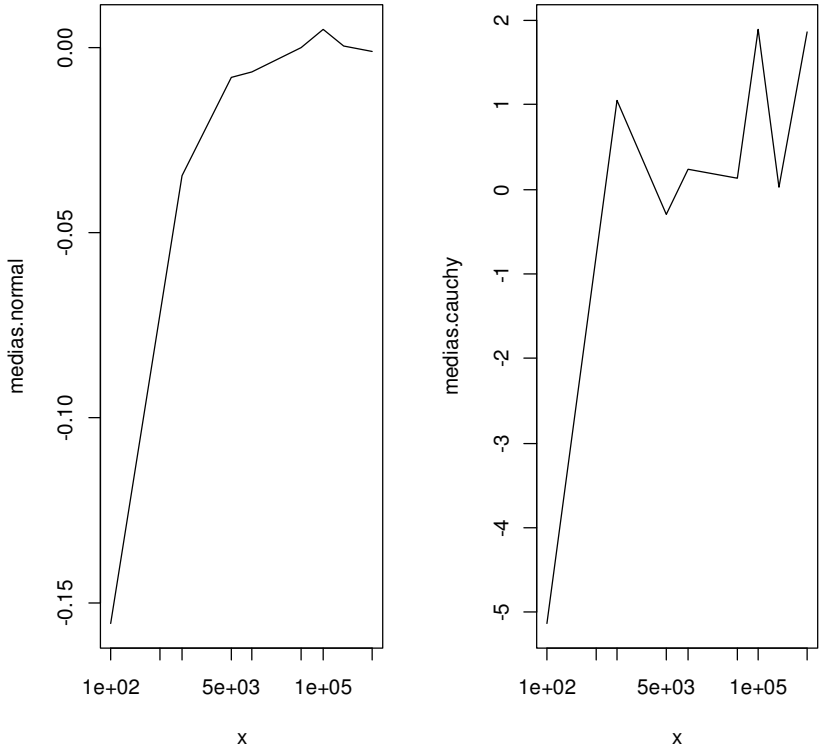
No todas las distribuciones de probabilidad tienen media. Para que exista la media la esperanza matemática de la variable aleatoria debe ser un valor real, lo que implica la convergencia de una suma (para variables aleatorias discretas) o de una integral (para variables aleatorias continuas). La mayor parte de las distribuciones de probabilidad usuales tienen media; una excepción es la distribución de Cauchy, ya que la siguiente integral impropia (media de la distribución de Cauchy estándar) no es convergente:

$$\int_{-\infty}^{\infty} \frac{x}{\pi(1+x^2)} dx$$

Efectuaremos, en principio, una simulación con valores de una distribución normal, y comprobaremos que las medias muestrales tienden a acercarse a la media de la distribución. Esto no va a ocurrir con la distribución de Cauchy, como veremos en la segunda parte de la prueba. Usaremos, en ambos casos, las correspondientes distribuciones estándar (parámetros 0 y 1).

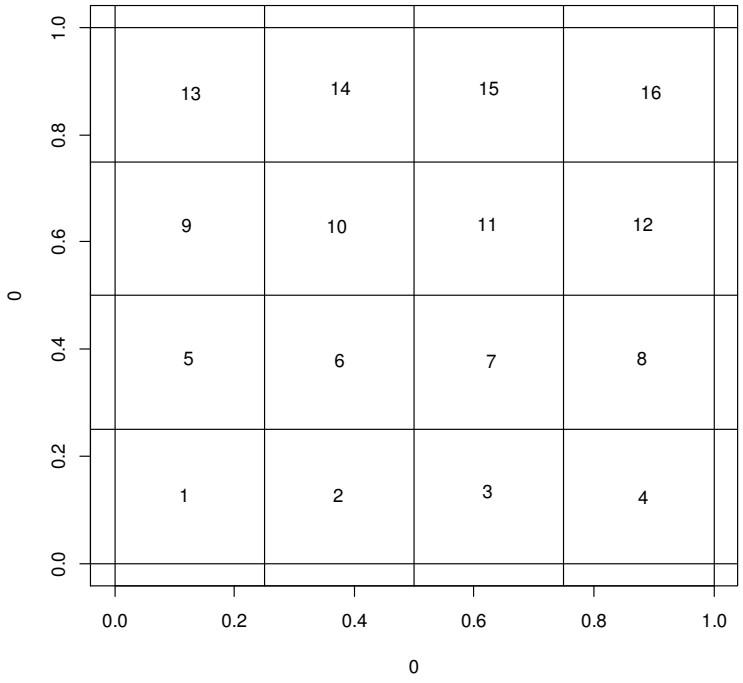
```
> x1<-rnorm(100) #Simulamos 100 valores de la distribución N(0,1)
> x2<-rnorm(1000)
> x3<-rnorm(5000)
> x4<-rnorm(10000)
> x5<-rnorm(50000)
> x6<-rnorm(100000)
> x7<-rnorm(200000)
> x8<-rnorm(500000)

> medias.normal<-c(mean(x1),mean(x2),mean(x3),mean(x4),mean(x5),
+ mean(x6),mean(x7),mean(x8))
> medias.normal
[1] -0.0836126701 -0.0525360663
[3]  0.0128999253   0.0060681572
[5]  0.0057327697  -0.0006194139
[7] -0.0010792445   0.0002100266

> y1<-rcauchy(100) #Simulamos 100 valores de la distribución Cauchy(0,1)
> y2<-rcauchy(1000)
> y3<-rcauchy(5000)
> y4<-rcauchy(10000)
> y5<-rcauchy(50000)
> y6<-rcauchy(100000)
> y7<-rcauchy(200000)
> y8<-rcauchy(500000)
```

```
> medias.cauchy<-c(mean(y1),mean(y2),mean(y3),mean(y4),mean(y5),
+ mean(y6),mean(y7),mean(y8))
> medias.cauchy
[1]  0.83794015  1.48411763
[3] -1.42971287  0.01554158
[5] -0.46778834 -0.19142550
[7] -2.36953980  1.66364389

> x<-c(100,1000,5000,10000,50000,100000,200000,500000)

> split.screen(c(1,2))
[1] 1 2
> screen(1)
> plot(x,medias.normal,type='l',log="x") #Usamos escala logarítmica en el eje X
> screen(2)
> plot(x,medias.cauchy,type='l',log="x") #Usamos escala logarítmica en el eje X

> #En la distribución normal las medias se estabilizan alrededor del valor 0
> #En la distribución de Cauchy las medias no se estabilizan alrededor de ningún
valor
```



9.9 GOTAS DE LLUVIA SOBRE EL SUELO

Las gotas de lluvia que caen sobre un suelo siguen una distribución de Poisson de parámetro λ, de modo que si dividimos el suelo en una retícula de n baldosas (cuadrados), el número de gotas por cada baldosa es una variable aleatoria de Poisson de parámetro λ/n. Se demuestra, además, que las coordenadas de cada gota son variables aleatorias uniformes e independientes. A continuación veremos cómo podrían caer alrededor de cien gotas de lluvia sobre un suelo cuadrado de lado unidad en un determinado intervalo de tiempo. En el gráfico hemos dividido el cuadrado unidad en 16 baldosas.



> lambda<-100

> M<-expand.grid(c(0,0.25,0.5,0.75),c(0,0.25,0.5,0.75))
> M #Coordenadas (x,y) de los vértices inferiores izquierdos de cada uno de los cuadrados

```
   Var1 Var2
1  0.00 0.00
2  0.25 0.00
3  0.50 0.00
4  0.75 0.00
5  0.00 0.25
6  0.25 0.25
```

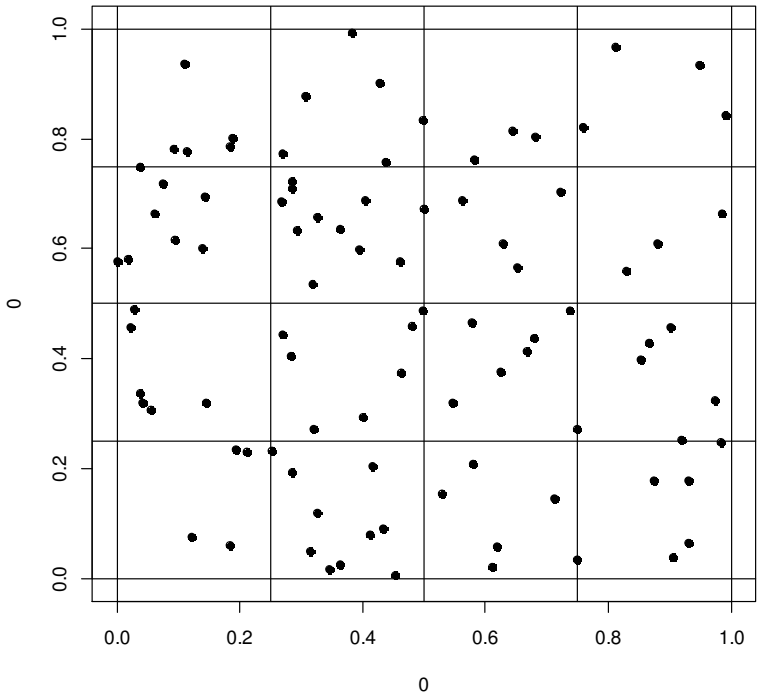
7 0.50 0.25
8 0.75 0.25
9 0.00 0.50
10 0.25 0.50
11 0.50 0.50
12 0.75 0.50
13 0.00 0.75
14 0.25 0.75
15 0.50 0.75
16 0.75 0.75

> a1<-M$Var1;b1<-M$Var2
> a2<-a1+0.25;b2<-b1
> a3<-a1+0.25;b3<-b1+0.25
> a4<-a1;b4<-b1+0.25

> S<-data.frame(a1,b1,a2,b2,a3,b3,a4,b4)
> S #Coordenadas (a,b), en sentido antihorario, de los cuatro vértices de cada baldosa
a1   b1   a2   b2   a3   b3   a4   b4
1 0.00 0.00 0.25 0.00 0.25 0.25 0.00 0.25
2 0.25 0.00 0.50 0.00 0.50 0.25 0.25 0.25
3 0.50 0.00 0.75 0.00 0.75 0.25 0.50 0.25
4 0.75 0.00 1.00 0.00 1.00 0.25 0.75 0.25
5 0.00 0.25 0.25 0.25 0.25 0.50 0.00 0.50
6 0.25 0.25 0.50 0.25 0.50 0.50 0.25 0.50
7 0.50 0.25 0.75 0.25 0.75 0.50 0.50 0.50
8 0.75 0.25 1.00 0.25 1.00 0.50 0.75 0.50
9 0.00 0.50 0.25 0.50 0.25 0.75 0.00 0.75
10 0.25 0.50 0.50 0.50 0.50 0.75 0.25 0.75
11 0.50 0.50 0.75 0.50 0.75 0.75 0.50 0.75
12 0.75 0.50 1.00 0.50 1.00 0.75 0.75 0.75
13 0.00 0.75 0.25 0.75 0.25 1.00 0.00 1.00
14 0.25 0.75 0.50 0.75 0.50 1.00 0.25 1.00
15 0.50 0.75 0.75 0.75 0.75 1.00 0.50 1.00
16 0.75 0.75 1.00 0.75 1.00 1.00 0.75 1.00

> g<-rpois(16,lambda/16);g #g es el número aleatorio de gotas que caen en cada una de las 16 baldosas
 [1] 4 10 6 5 6 7 7 5 8 11 4 3 5
[14] 6 3 5
> sum(g)
[1] 95
> #En esta simulación han caído 95 gotas en el cuadrado unidad

```
> plot(0,0,type="n",xlim=c(0,1),ylim=c(0,1)) #Preparamos el gráfico

> for (i in 1:16)
+ {
+ #Caen g[i] gotas en la baldosa i
+ for (k in 1:g[i])
+ {
+ x<-numeric(g[i]);y<-numeric(g[i])
+ x[k]<-runif(1,a1[i],a3[i]);y[k]<-runif(1,b1[i],b3[i])
+  points(x[-which(x==0)],y[-which(y==0)],pch=19) #Dibujamos cada uno de los puntos, excepto el (0,0)
+ }
+ }

> #Dibujamos la retícula
> abline(h=0); abline(h=0.25); abline(h=0.5); abline(h=0.75); abline(h=1)
> abline(v=0); abline(v=0.25); abline(v=0.5); abline(v=0.75); abline(v=1)
```



En este gráfico se aprecia con claridad, contrariamente a lo que pudiera pensar el inexperto, que la aleatoriedad no aparece como regularidad o tendencia al agrupamiento.

9.10 DISTRIBUCIÓN DE LAS CONGRUENCIAS, MÓDULO 4, DEL NÚMERO TOTAL DE LETRAS DE LOS APELLIDOS DE UN GRUPO DE PERSONAS

Un profesor propone a sus alumnos cuatro trabajos para que sean realizados en grupo. Para ello decide repartir al azar las tareas utilizando el siguiente criterio: cada grupo, formado por 5 alumnos, hará el trabajo (numerado del 0 al 3) que coincida con el resto de la división por 4 de la suma total de las letras de los apellidos del grupo. El profesor se pregunta, a continuación, si cada uno de los trabajos tendrá la misma probabilidad de ser asignado a un grupo concreto o, lo que es lo mismo, si la distribución de la variable aleatoria que representa el número del trabajo asignado es uniforme discreta. Intuye que la distribución de la cantidad total de letras de cada grupo no es uniforme pero cree que la congruencia de ese número, módulo 4, sí lo es. Como comprende que el problema es complicado desde un punto de vista teórico, decide llevar a cabo la siguiente simulación:

```
> #Cargamos el paquete "randomNames"
> library(randomNames)

> #Vamos a generar, por ejemplo, 10 nombres y apellidos
> randomNames(10)
 [1] "Lindsey, Isaac"
 [2] "Nelson, Justin"
 [3] "Williams Taylor, Shariya"
 [4] "Lopez, Cheyenne"
 [5] "Liu, Mario"
 [6] "Grantham, Marlo"
 [7] "Davis, Alexander"
 [8] "Perez, Brandon"
 [9] "Clemons, Ayrika"
 [10] "Thach, Charanaka"

> #Escogemos solo los apellidos de 10000 personas imaginarias
> apellidos<-randomNames(10000,which.names='last')
> #Formamos 2000 grupos de 5 personas (apellidos) cada uno; en total tendremos 10000 apellidos, que agruparemos en una matriz de 2000 filas y 5 columnas
> a<-matrix(sample(apellidos,10000),nrow=2000,ncol=5)

> #Por ejemplo, los cuatro primeros grupos son los siguientes
> a[1:4,]
  [,1]            [,2]             [,3]      [,4]         [,5]
[1,] "Ponce-Campa"    "Nibler"         "Acosta"  "Brannigan"  "Grayson"
[2,] "Hazard"         "Cleveland Leaks" "Thesz"  "Le"         "Vasquez-Lara"
[3,] "Sanchez-Ensaldo" "Garcia"        "Thao"    "Munoz Saenz" "Villezca-Perez"
[4,] "Saiz Carbajal"  "Saines"         "Williams" "Nguyen"    "Lucero"
```

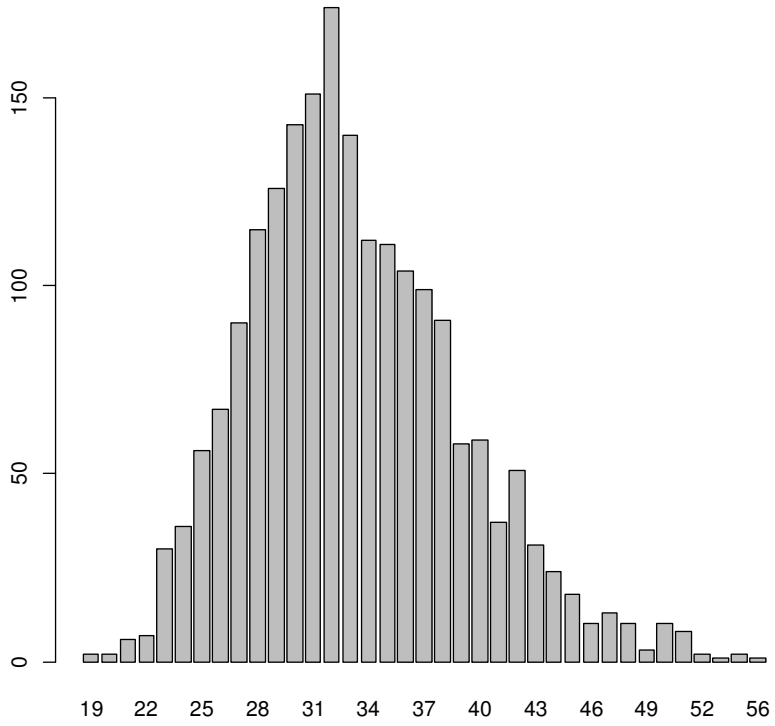
> #Transformamos ahora la matriz a en otra matriz A formada por el número de letras de cada apellido
> A<-apply(a,c(1,2),nchar) #La función nchar() cuenta el número de caracteres de una cadena de caracteres
> A[1:4,] #Las cuatro primeras filas de la matriz A

```
 [,1] [,2] [,3] [,4] [,5]
[1,]   11    6    6    9    7
[2,]    6   15    5    2   12
[3,]   15    6    4   11   14
[4,]   13    6    8    6    6
```

> #Sumamos las letras de cada grupo
> X<-rowSums(A)
> X[1:4]

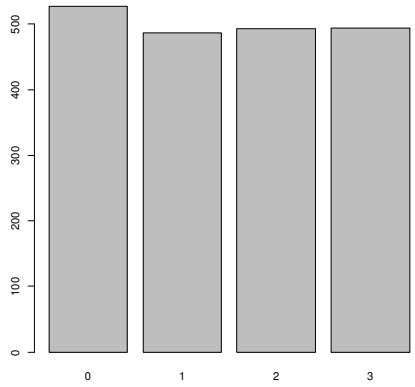
```
[1] 39 40 50 39
```

> #Veamos cómo se distribuye la variable aleatoria X
> barplot(table(X))



> #Obtenemos los valores correspondientes al vector X, módulo 4
> m<-X%%4
> m[1:4] #Congruencias, módulo 4, de los cuatro primeros grupos
[1] 3 0 2 3

> barplot(table(m))



Da la impresión de que, como había intuido el profesor, se obtiene una distribución uniforme discreta.

9.11 TEOREMA CENTRAL DEL LÍMITE

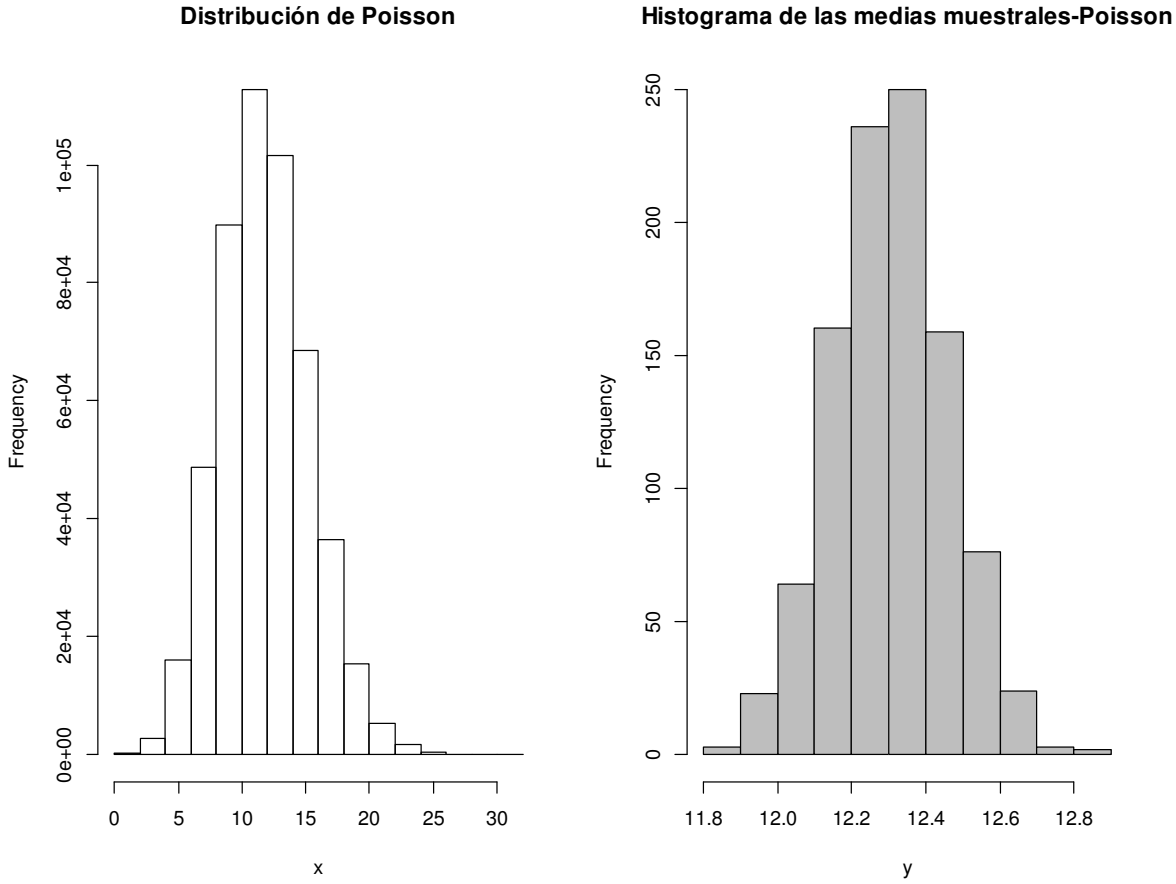
En esta experiencia vamos a constatar que el importante resultado de la teoría de la probabilidad, conocido como *Teorema Central del Límite,* se verifica para cualquier distribución que tenga media y varianza finitas. Probaremos con una distribución discreta: Poisson, y con tres continuas: uniforme, exponencial y Cauchy. Veremos que este resultado no se cumple para una variable aleatoria de Cauchy pues, como se ha comentado en el apartado 9.8, esta distribución carece de media.

#1)Prueba con la distribución Poisson(12.3)

#En primer lugar formamos una matriz de 1.000 filas por 500 columnas (en total 500.000 valores), cuyos elementos son valores tomados al azar de la distribución de Poisson(12.3)
x<-matrix(rpois(500000,12.3),nrow=1000,byrow=T)

split.screen(c(1,2))
screen(1)
hist(x,main="Distribución de Poisson") #Forma aproximada de la distribución de Poisson

```
#Construimos el vector formado por las medias muestrales de cada fila
y<-c(margin.table(x,1)/500)

#Construimos un histograma de las medias muestrales
screen(2)
hist(y,col="grey",main="Histograma de las medias muestrales-Poisson")
```

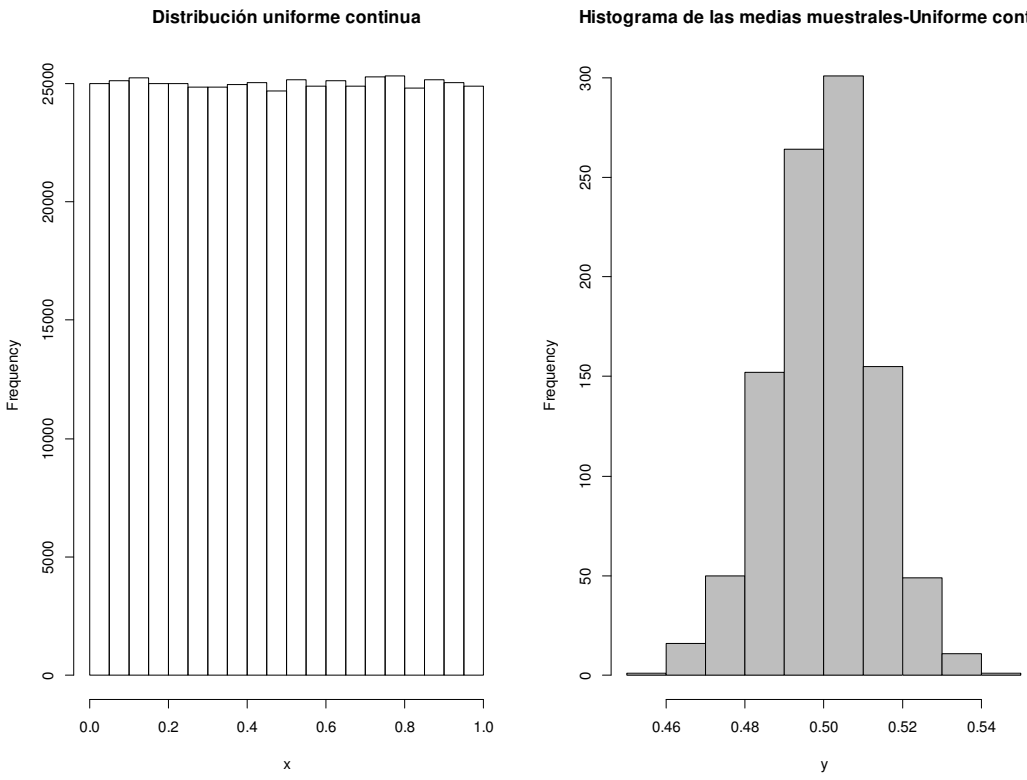



```
dev.off()
rm(list=ls(all=TRUE))

#2)Prueba con la distribución UC[0,1]

#En primer lugar formamos una matriz de 1.000 filas por 500 columnas (en total
500.000 valores), cuyos elementos son valores tomados al azar de la distribución
UC[0,1]
x<-matrix(runif(500000,min=0,max=1),nrow=1000,byrow=T)
```

split.screen(c(1,2))
screen(1)
hist(x,main="Distribución uniforme continua") #Vemos la forma aproximada de la distribución uniforme

#Construimos el vector formado por las medias muestrales de cada fila
y<-c(margin.table(x,1)/500)

#Construimos un histograma de las medias muestrales
screen(2)
hist(y,col="grey",main="Histograma de las medias muestrales-Uniforme continua")

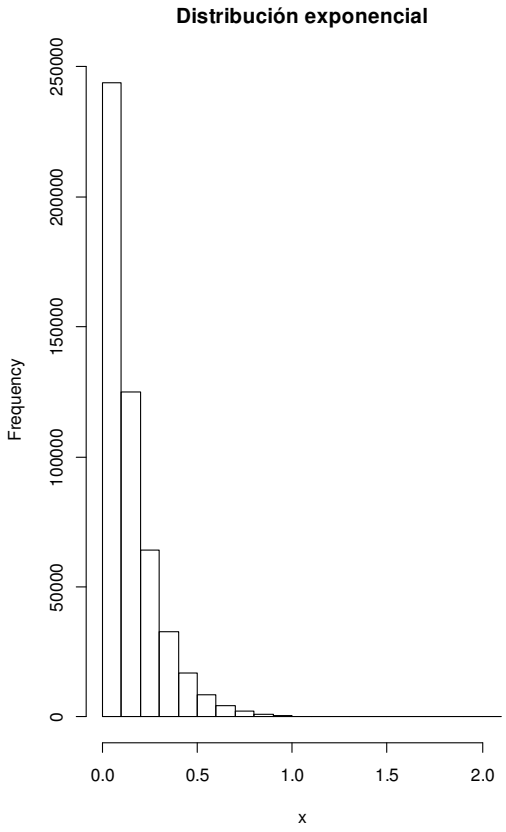


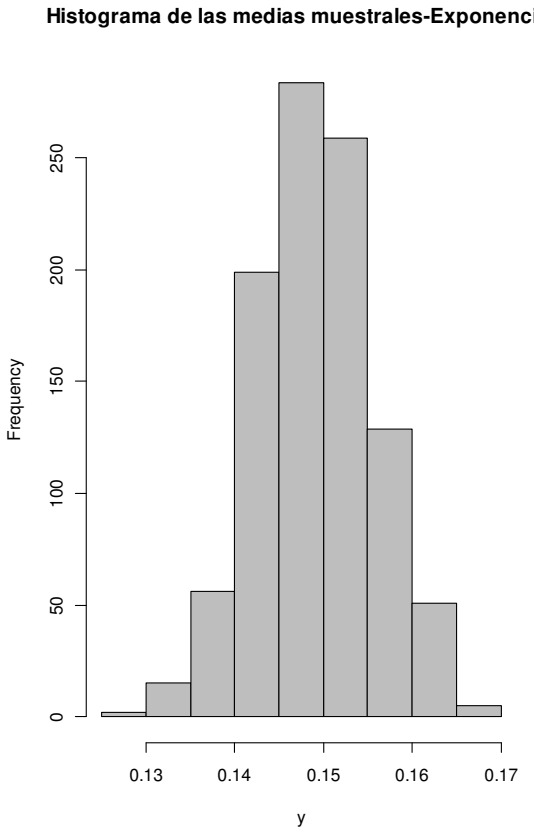
dev.off()
rm(list=ls(all=TRUE))

#3)Prueba con la distribución Exponencial(6.7)

#En primer lugar formamos una matriz de 1.000 filas por 500 columnas (en total 500.000 valores), cuyos elementos son valores tomados al azar de la distribución Exponencial(5)
x<-matrix(rexp(500000,6.7),nrow=1000,byrow=T)

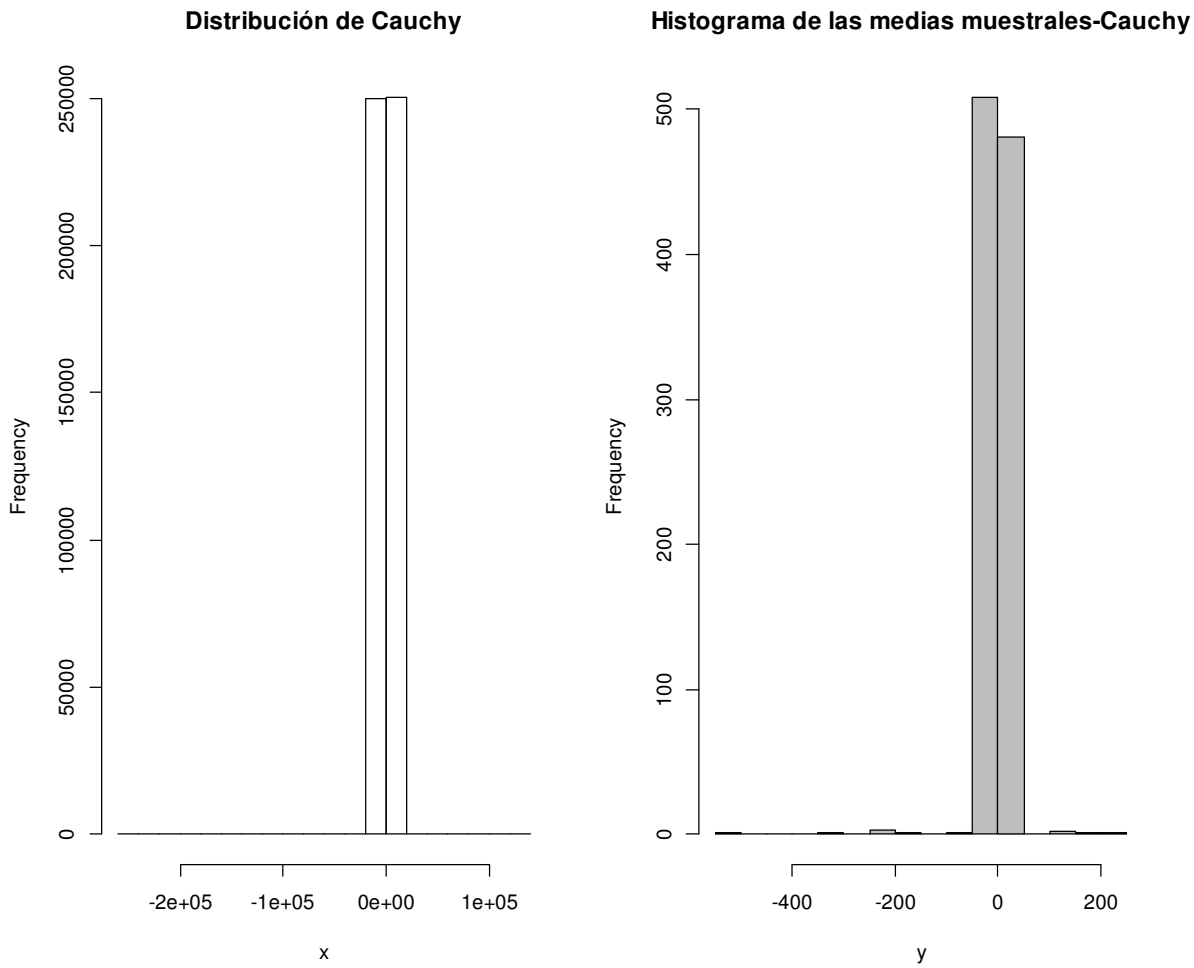
```
split.screen(c(1,2))
screen(1)
hist(x,main="Distribución exponencial") #Vemos la forma aproximada de la distribución exponencial

#Construimos el vector formado por las medias muestrales de cada fila
y<-c(margin.table(x,1)/500)

#Construimos un histograma de las medias muestrales
screen(2)
hist(y,col="grey",main="Histograma de las medias muestrales-Exponencial")
```






```
dev.off()
rm(list=ls(all=TRUE))

#4)Prueba con la distribución de Cauchy (0,1)

#En primer lugar formamos una matriz de 1.000 filas por 500 columnas (en total 500.000 valores), cuyos elementos son valores tomados al azar de la distribución de Cauchy(0,1)
x<-matrix(rcauchy(500000,0,1),nrow=1000,byrow=T)
```

```
split.screen(c(1,2))
screen(1)
hist(x,main="Distribución de Cauchy") #Vemos la forma aproximada de la distribución uniforme

#Construimos el vector formado por las medias muestrales de cada fila
y<-c(margin.table(x,1)/500)

#Construimos un histograma de las medias muestrales
screen(2)
hist(y,col="grey",main="Histograma de las medias muestrales-Cauchy")
```

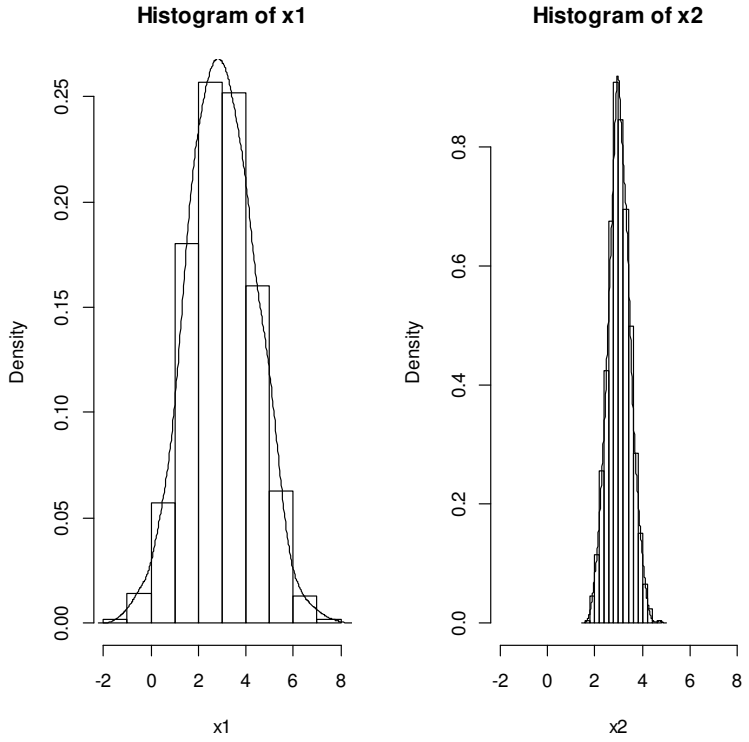


En todos los casos, excepto para la distribución de Cauchy, se aprecia claramente que el histograma de las medias muestrales se aproxima al correspondiente a una distribución normal.

9.12 ERROR DE MUESTREO

Se denomina error de muestreo o error de estimación a la desviación típica de un determinado estimador. Por ejemplo, la media muestral de una muestra proveniente de una distribución normal $N(\mu, \sigma)$, que es el estimador de la media poblacional, se distribuye según una normal $N(\mu, \frac{\sigma}{\sqrt{n}})$, luego el error de muestreo es $\frac{\sigma}{\sqrt{n}}$. Vamos a comprobar a continuación que, en este caso efectivamente, al aumentar el tamaño muestral el error de muestreo disminuye.

```
#Generamos 1000 muestras de tamaño 2 de una población N(3,2)
valores1<-matrix(rnorm(2000,3,2),nrow=1000)
x1<-rowSums(valores1)/2 #Vector de medias muestrales

split.screen(c(1,2))
screen(1)
hist(x1,prob=T,xlim=c(-2,8))
lines(density(x1))
#Generamos 1000 muestras de tamaño 20 de una población N(3,2)
valores2<-matrix(rnorm(20000,3,2),nrow=1000)
x2<-rowSums(valores2)/20 #Vector de medias muestrales
screen(2)
hist(x2,prob=T,xlim=c(-2,8))
lines(density(x2))
```



Con un tamaño de muestra mayor (20 respecto de 2) la desviación típica del estimador (error de muestreo), que en este caso es la media muestral, disminuye sensiblemente, como se puede apreciar con claridad en las figuras.

9.13 MUESTREO CON Y SIN REEMPLAZAMIENTO

En esta prueba nos planteamos estudiar la diferencia que existe entre estimar un parámetro con y sin reemplazamiento. El parámetro, en este caso, representa la proporción de bolas blancas respecto del total en una urna (población finita) con 100 bolas blancas y 900 bolas negras. El estimador adecuado en ambos casos es la proporción muestral, x/n, siendo x el número de bolas blancas en la muestra y n el número total de bolas extraídas. Comprobaremos que en el muestreo sin reemplazamiento el error de muestreo es menor o, dicho de otro modo, que en el muestreo sin reposición se necesita un tamaño de muestra menor para cometer el mismo error que en el caso del muestreo con reposición.

```
> urna<-c(rep(1,100),rep(0,900)) #1 representa bola blanca y 0 bola negra
> #Sea theta la proporción de bolas blancas en la urna
> #Extraemos una muestra de tamaño n CON reemplazamiento y obtenemos x blancas. El estimador adecuado de theta es x/n
> n<-500 #Exageramos el tamaño muestral para evidenciar mejor la diferencia
> CR<-replicate(100,
+ {
+ muestra<-sample(urna,n,replace=T)
+ sum(muestra)
+ #Nuestra estimación de theta es la siguiente
+ estim.theta<-sum(muestra)/n
+ estim.theta
+ }
+ )
> mean(CR);sd(CR)
[1] 0.09968
[1] 0.01259058
> #Verdadero valor del parámetro: 0.1

> #Extraemos una muestra de tamaño n SIN reemplazamiento y obtenemos x blancas. El estimador adecuado de theta es x/n
> SR<-replicate(100,
+ {
+ muestra<-sample(urna,n)
+ sum(muestra)
+ #Nuestra estimación de theta es la siguiente
+ estim.theta<-sum(muestra)/n
+ estim.theta
```

```
+ }
+ )
> mean(SR);sd(SR)
[1] 0.10002
[1] 0.009257244
> #Verdadero valor del parámetro: 0.1
```

Como se advierte en la prueba llevada a cabo, la estimación del parámetro se aproxima a 0,1 bastante en ambos casos (0.09968 y 0.10002), siendo menor el error de muestreo en el caso "SR" (sin reemplazamiento). Evidentemente, y dado que se trata de una simulación concreta, podría haber ocurrido lo contrario, si bien es más improbable.

9.14 ESTIMACIÓN DEL NÚMERO DE TAXIS DE UNA CIUDAD

Se trata de estimar el número total de taxis que hay en una ciudad mediante la observación de los números de una pequeña fracción de ellos. Se supone que los taxis tienen una numeración correlativa desde 1 hasta el valor que representa la cantidad total. Se demuestra que el estimador centrado de varianza mínima (sin entrar en detalles podemos decir que es un buen estimador de ese valor) es: $max(x_i)+max(x_i)/n-1$, siendo $max(x_i)$ el valor máximo de la muestra. Por ejemplo, si la muestra obtenida fuera {1,6,45,23,15} la estimación sería: 45+45/5-1=53.

Comprobaremos, a continuación, que el valor anterior da una buena estimación en una ciudad en la que, por ejemplo, hay 15450 taxis (aunque nosotros no conocemos este dato). Así pues, para resolver el problema planteado, nos situamos en la calle, apuntamos los números de unos cuantos vehículos (probaremos con 10, 20 y 500) y finalmente obtendremos la estimación de la cantidad total de taxis:

```
> x<-1:15450
> set.seed(8)
> a10<-sample(x,10)
> a10
[1]  7205  3211 12354 10070  4967 11104
[7]  4493 14398 11878  9952
> max(a10)+max(a10)/10-1
[1] 15836.8

> set.seed(8)
> a20<-sample(x,20)
> a20
[1]  7205  3211 12354 10070  4967 11104
[7]  4493 14398 11878  9952  7057  1379
[13]  6676  8413  2134 14321    21  4082
[19]  4268  8042
```

```
> max(a20)+max(a20)/20-1
[1] 15116.9

> set.seed(8)
> a500<-sample(x,500)

> max(a500)+max(a500)/500-1
[1] 15479.9

> #El verdadero valor del número total de taxis es 15540
```

Se observa que con solo apuntar los números de 10 taxis se obtiene una estimación bastante aceptable.

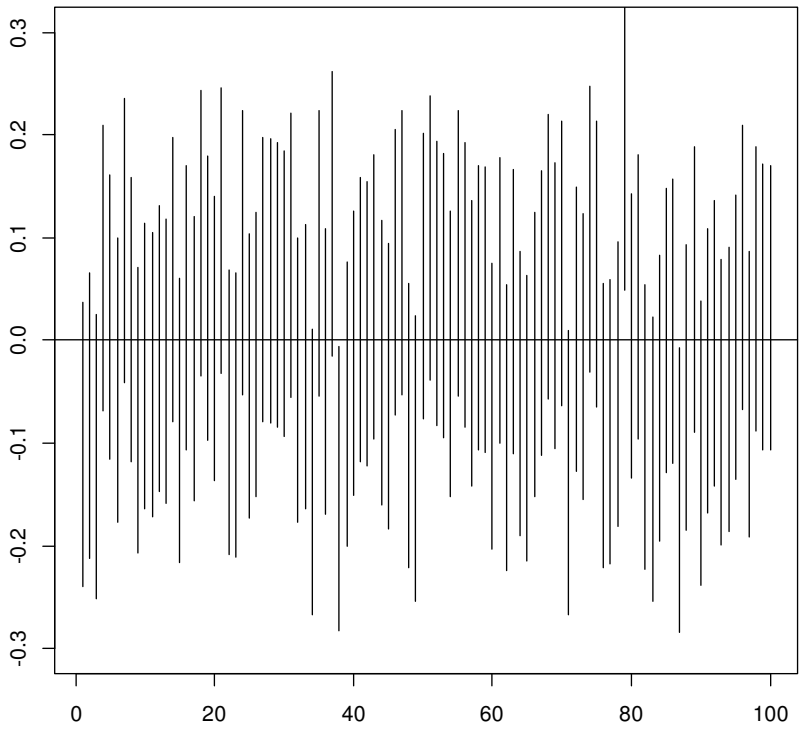
9.15 SIGNIFICADO DE LOS INTERVALOS DE CONFIANZA

En esta experiencia vamos a generar 100 muestras de tamaño 200 provenientes de una variable aleatoria N(0,1), para calcular los correspondientes intervalos de confianza para la media al nivel del 95%. Veremos el número de ellos que no contienen a la media. La expresión para calcular, en este caso, el intervalo de confianza es:

$$(\overline{x} \pm z_{\alpha/2}\, \sigma / \sqrt{n})$$

```
> #Generaremos en primer lugar las 100 muestras; tendremos en total 20000 valores aleatorios de una N(0,1)
> valores.aleatorios<-rnorm(20000)

> #Organizamos los valores anteriores en una matriz de 100 filas (100 muestras)
> x<-matrix(valores.aleatorios,nrow=100)
> #Formamos un vector con todas las medias muestrales
> medias.muestrales<-margin.table(x,1)/200
> xraya<-c(medias.muestrales)
> xraya
  [1] -0.082281578 -0.039925947 -0.038924876  0.055208037  0.111785215
  [6] -0.024319005 -0.013515022 -0.020781209  0.066281579  0.241937906
  ......
 [96] -0.057047812 -0.058831532  0.067255427 -0.070909111  0.031170621

> #Calculamos los límites inferiores (li) y superiores (ls) de los IC al nivel del 95%
> li<-c(xraya-qnorm(0.975)*1/sqrt(200))
> ls<-c(xraya+qnorm(0.975)*1/sqrt(200))


> #Formamos todos los intervalos de confianza
```

```
> IC<-matrix(c(li,ls),nrow=100)
> IC
            [,1]         [,2]
  [1,] -0.22087196  0.056308804
  [2,] -0.17851633  0.098664435
  ......
  [100,] -0.10741976  0.169761004

> plot(li,type="n",ylim=c(-0.3,0.3),xlab=" ",ylab=" ")
> #Con la opción type="n" no dibujamos el gráfico, solo los ejes
> #A continuación dibujamos los segmentos que representan los diferentes IC
> coordx<-1:100
> segments(coordx,li,coordx,ls)
> abline(h=0) #Dibujamos la línea de la media (cero)

> #El gráfico de los 200 IC es
```




```
> #Calculemos cuántos IC no contienen a la media
> a<-c(rep(0,100))
> for (i in 1:100) {if (IC[i,1]>0 | IC[i,2]<0) a[i]<-1 else 0}
```

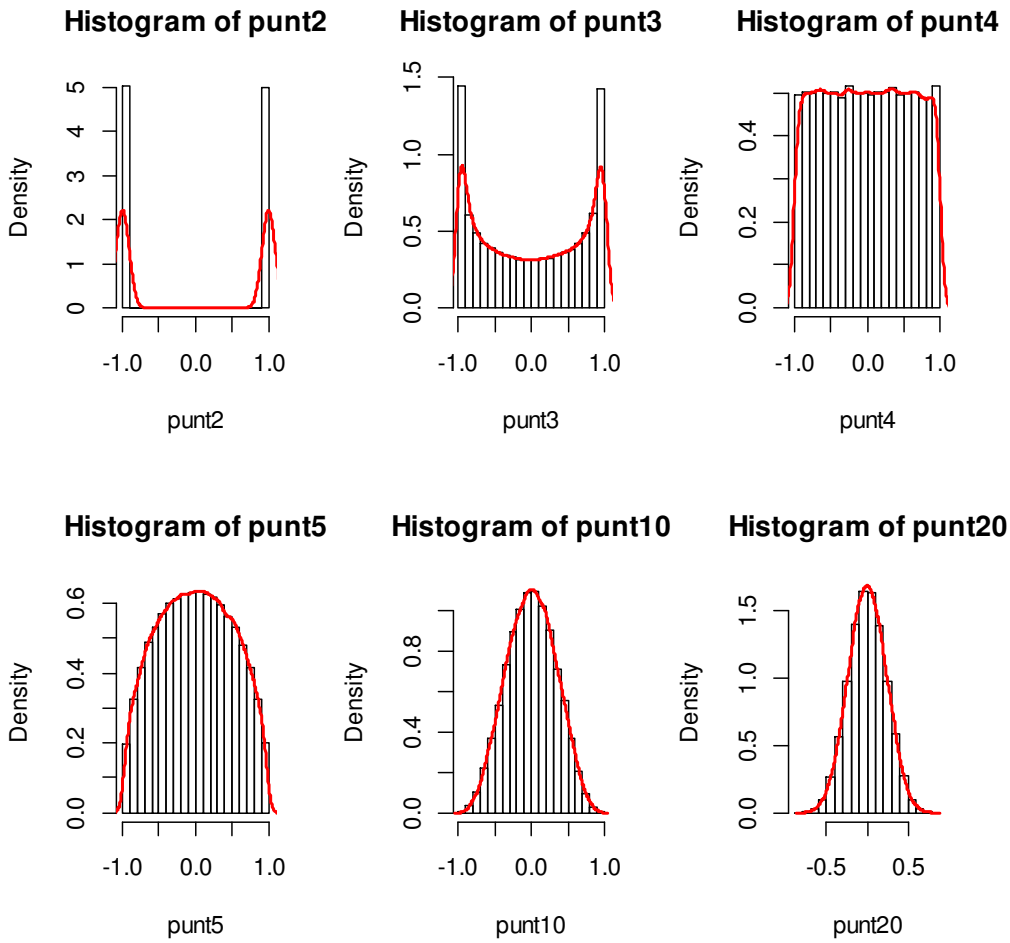
```
> a
[1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
[21] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0
[41] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
[61] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0
[81] 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0

> #El número de IC que no contienen a la media es, en este caso:
> sum(a)
[1] 3
```

9.16 COEFICIENTE DE CORRELACIÓN EN FUNCIÓN DEL NÚMERO DE PUNTOS ELEGIDOS AL AZAR

Considérese una nube de puntos en el plano cuyas coordenadas corresponden a las realizaciones de dos variables aleatorias independientes. En tal situación se produce un fenómeno muy curioso según cuántos puntos se tomen a la hora de calcular el coeficiente de correlación entre las dos variables.

Si se escogen dos puntos al azar, el valor que toma el coeficiente de correlación, evidentemente, solo puede ser -1 o 1; si se toman 3 puntos o más, como veremos a continuación, se puede obtener cualquier valor entre -1 y 1. Teniendo en cuenta que cuando se eligen puntos al azar el coeficiente de correlación entre las variables que representan las coordenadas de esos puntos es una variable aleatoria, la distribución de probabilidad correspondiente adquiere diversas formas, según el número de puntos elegido. A medida que ese número va aumentando, la distribución se va semejando cada vez más a una normal.

```
> split.screen(c(2,3))
[1] 1 2 3 4 5 6

> punt2<- replicate(100000,cor(c(runif(2)),c(runif(2))))
> screen(1);hist(punt2,freq=F);lines(density(punt2),col="red",lwd=2)
> punt3<- replicate(100000,cor(c(runif(3)),c(runif(3))))
> screen(2);hist(punt3,freq=F);lines(density(punt3),col="red",lwd=2)
> punt4<- replicate(100000,cor(c(runif(4)),c(runif(4))))
> screen(3);hist(punt4,freq=F);lines(density(punt4),col="red",lwd=2)
> punt5<- replicate(100000,cor(c(runif(5)),c(runif(5))))
> screen(4);hist(punt5,freq=F);lines(density(punt5),col="red",lwd=2)
> punt10<- replicate(100000,cor(c(runif(10)),c(runif(10))))
> screen(5);hist(punt10,freq=F);lines(density(punt10),col="red",lwd=2)
> punt20<- replicate(100000,cor(c(runif(20)),c(runif(20))))
> screen(6);hist(punt20,freq=F);lines(density(punt20),col="red",lwd=2)
```



9.17 ERROR DE TIPO II

Para poner de manifiesto el comportamiento del error de tipo II en un contraste de hipótesis, efectuaremos un test sobre la media de una población normal. El problema concreto sobre el que vamos a realizar el contraste es el que se enuncia a continuación.

El peso de los recién nacidos es una variable aleatoria normal de media *mu* y desviación típica 0.5 kg. Procederemos variando el valor de *mu* entre 3.1 y 3.5 kg e iremos extrayendo muestras aleatorias simples de tamaño 50 de las poblaciones correspondientes para contrastar, mediante un test de la t de Student, si la media es 3. Veremos en cuántos de esos test se acepta la hipótesis nula de que la media es 3 y, por tanto, cuándo se comete error de tipo II; es decir, cuándo se acepta la hipótesis nula siendo falsa.

```
> #Cada uno de los test será similar al siguiente
> A<-t.test(rnorm(50,100,2),mu=100)
> #Para extraer el p-valor de este test hacemos lo siguiente:
> A[["p.value"]]
[1] 0.8996734
> #Para comprobar si el p-valor es mayor que 0.01 y, por tanto, si se acepta H0 hacemos
> ifelse(A[["p.value"]]>0.01,1,0)
[1] 1

> #Probamos el test con 1000 muestras de tamaño 50 y vemos en cuántos casos se obtiene un p-valor mayor que 0.01
> media<-seq(3.1,3.5,0.05)
> media
[1] 3.10 3.15 3.20 3.25 3.30 3.35 3.40 3.45
[9] 3.50

> for (i in 1:length(media))
+ {
+ options("scipen"=100) #Para no usar notación exponencial
+ M<-numeric(length(media))
+ set.seed(2)
+                                                    M[i]<-
sum(replicate(1000,ifelse(t.test(rnorm(50,media[i],0.5),mu=3)[["p.value"]]>0.01,
1,0)))
+ print(c(media[i],M[i]))
+ }
[1]   3.1 865.0
[1]   3.15 680.00
[1]   3.2 427.0
[1]   3.25 218.00
[1]  3.3 67.0
[1] 3.35 15.00
[1] 3.4 4.0
[1] 3.45 1.00
[1] 3.5 0.0
```

La segunda columna representa el número de casos, entre 1000, en los que se tomaría una decisión incorrecta: aceptar que la media es 3. Como se ve, a medida que nos vamos alejando de 3 decrece la probabilidad de cometer error de tipo II. Si volvemos a repetir la experiencia con muestras de mayor tamaño, n=100, veremos que este error disminuye mucho más rápidamente.

```
> #Ahora probamos el test con 1000 muestras de tamaño 100
> for (i in 1:length(media))
+ {
+ options("scipen"=100) #Para no usar notación exponencial
+ M<-numeric(10)
+ #set.seed(2)
+                                                    M[i]<-
sum(replicate(1000,ifelse(t.test(rnorm(100,media[i],0.5),mu=3)[["p.value"]]>0.01
,1,0)))
+ print(c(media[i],M[i]))
+ }
[1]   3.1 713.0
[1]   3.15 338.00
[1]   3.2 102.0
[1]  3.25 14.00
[1] 3.3 0.0
[1] 3.35 0.00
[1] 3.4 0.0
[1] 3.45 0.00
[1] 3.5 0.0
```

9.18 TEST DE BONDAD DE AJUSTE

En esta experiencia se trata de que el lector compruebe, a simple vista primero y utilizando un test ji-cuadrado después, si se puede aceptar que unos valores simulados se ajustan a una distribución uniforme discreta. Se supone que se ha lanzado 100 veces un dado, por lo que la suma de los valores de la segunda fila debe ser 100. Supongamos, por ejemplo, que disponemos de los datos siguientes:

1	2	3	4	5	6
20	14	19	21	15	11

```
> prob.nul<-c(1/6,1/6,1/6,1/6,1/6,1/6)
> valores<-c(20,14,19,21,15,11)
> chisq.test(valores,p=prob.nul) #En este caso, al ser las probabilidades iguales,
se podría prescindir del argumento p=prob.nul
Chi-squared test for given
probabilities
data:  valores
X-squared = 4.64, df = 5, p-value =
0.4614
```

Como para los valores de la tabla obtenemos un p-valor muy alto no se puede rechazar la hipótesis nula de que siguen una distribución uniforme discreta. Pero dejemos ahora a *R* que elija esos seis valores y tratemos de ver a simple vista si se puede aceptar la hipótesis nula de que los resultados son equiprobables:

```
> val<-20:80 #Para que no haya excesivas diferencias entre los valores elegiremos entre los naturales de 20 a 80
> A<-sample(val,6,replace=T)
> B<-round(100*A/sum(A)) #Para obtener valores enteros que sumen aproximadamente 100

> #Las siguientes sentencias tienen por objeto conseguir que los seis valores sumen exactamente 100
> ifelse(sum(B)==100,B<-B,B[6]<-B[6]+1)
[1] 25
> ifelse(sum(B)==100,B<-B,B[6]<-B[6]-2)
[1] 25
> sum(B);B
[1] 100
[1] 25 25  8 20 11 11
```

Ahora deberíamos tratar de adivinar, de forma intuitiva, si los valores 25, 25, 8, 20, 11 y 11 se ajustan o no a una distribución uniforme discreta.

```
> chisq.test(B)
  Chi-squared test for given
  probabilities
data:  B
X-squared = 17.36, df = 5, p-value =
0.003865
```

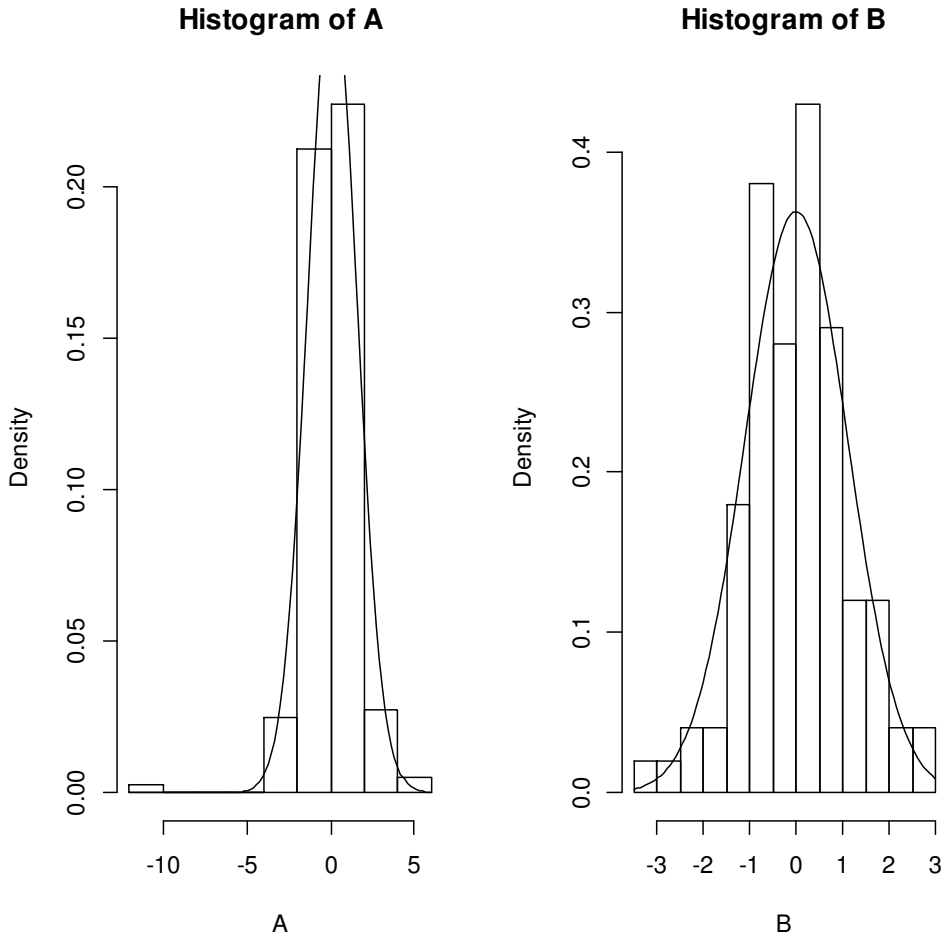
Como muestra la salida de este test, en este caso se rechaza la hipótesis de que los valores simulados provienen de una distribución uniforme discreta.

9.19 APROXIMACIÓN DE LA DISTRIBUCIÓN t DE STUDENT POR UNA NORMAL ESTÁNDAR CON n GRANDE

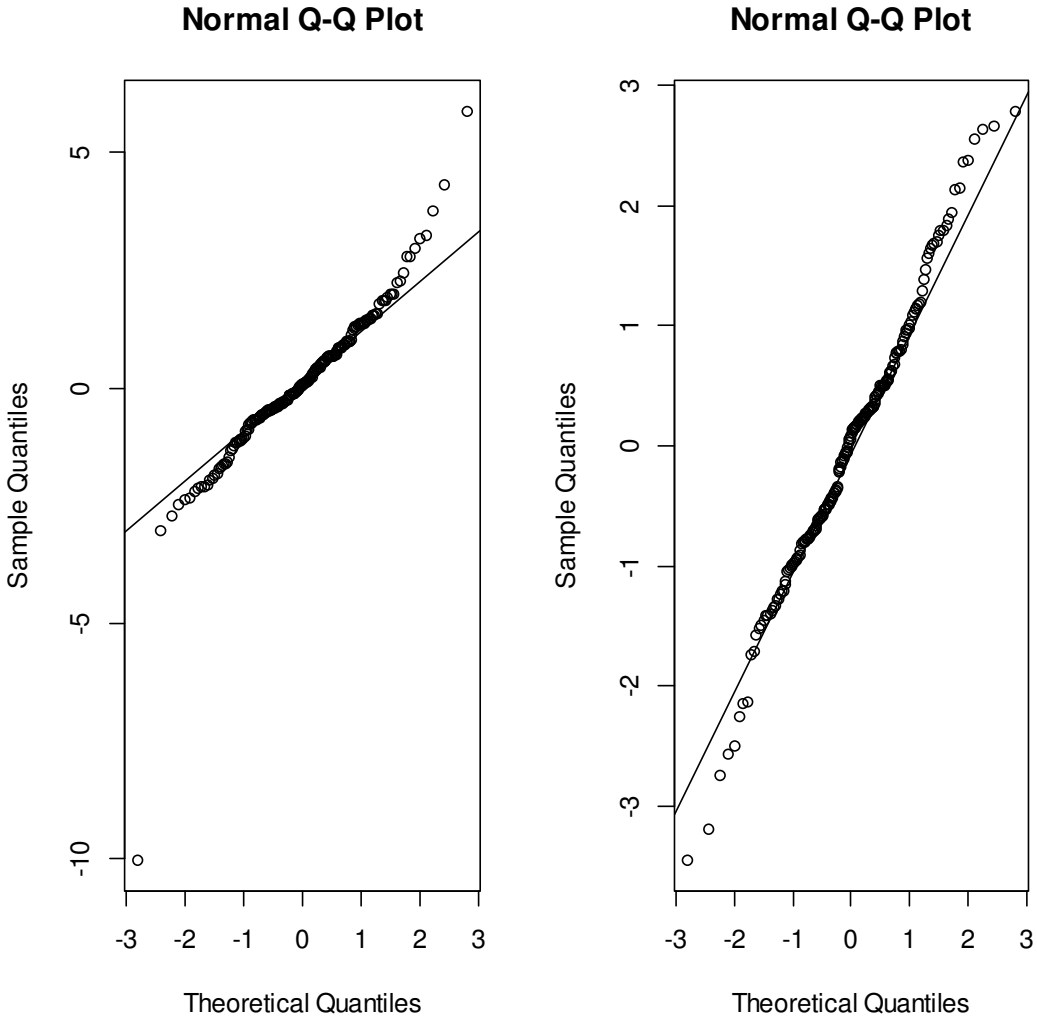
Vamos a estudiar la normalidad de los datos provenientes de una distribución t de Student de 3 grados de libertad y de otra de 40 para poner de manifiesto que, a diferencia de la primera, la distribución t de Student de 40 grados de libertad se ajusta muy bien a una normal estándar.

```
> A<-rt(200,3)
> B<-rt(200,40)
```

```
> split.screen(c(1,2))
[1] 1 2
> screen(1)
> hA<-hist(A,plot=F) #Se genera el histograma de los datos A pero no se dibuja
> #La sentencia siguiente tiene por objeto no cortar la parte superior de la función de densidad de la normal
> ylimA<-range(0,hA$density,dnorm(mean(A),sd(A)))
> hist(A,freq=F,ylim=ylimA)
> curve(dnorm(x,mean(A),sd(A)),add=T) #Superponemos al histograma una normal con media y desviación típicas las de los valores A

> screen(2)
> hB<-hist(B,plot=F)
> ylimB<-range(0,hB$density,dnorm(mean(B),sd(B)))
> hist(B,freq=F,ylim=ylimB)
> curve(dnorm(x,mean(B),sd(B)),add=T)
```



```
> dev.off() #Eliminamos el gráfico actual
null device
      1
> split.screen(c(1,2))
[1] 1 2

> screen(1)
> qqnorm(A)
> qqline(A)

> screen(2)
> qqnorm(B)
> qqline(B)
```



```
> #A continuación efectuamos los test de normalidad
> shapiro.test(A)
  Shapiro-Wilk normality test
data:  A
W = 0.8942, p-value = 0.0000000001094
> shapiro.test(B)
     Shapiro-Wilk normality test
data:  B
W = 0.9897, p-value = 0.1621

> #Cargamos el paquete nortest
> library(nortest)
> #Realizamos el test de normalidad de Anderson-Darling perteneciente a este paquete
> ad.test(A)
   Anderson-Darling normality test
data:  A
A = 2.446, p-value = 0.000003333
> ad.test(B)
     Anderson-Darling normality test
data:  B
A = 0.6012, p-value = 0.1171
```

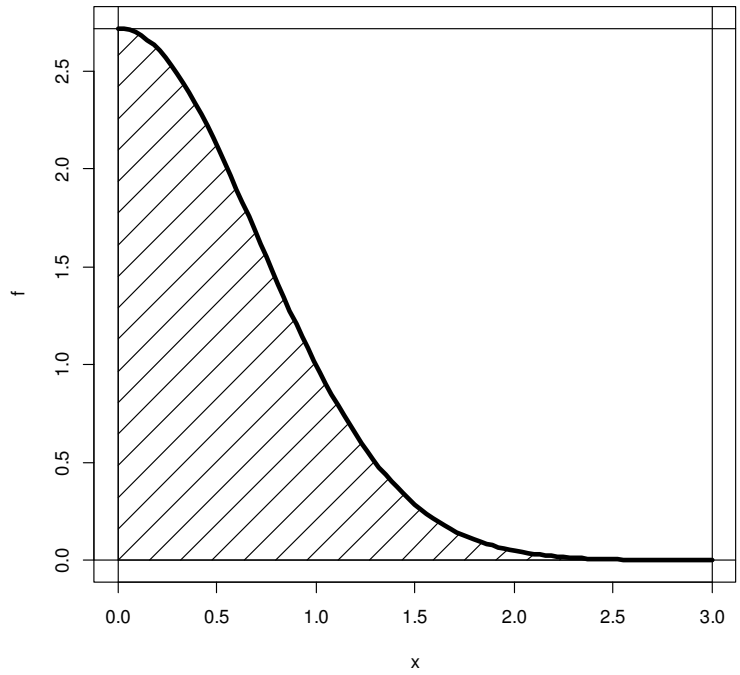
En esta experiencia se observa que el histograma y el gráfico de normalidad se ajustan muy bien a una distribución normal en la t_{40} pero no en la t_3. En los test de normalidad se obtienen p-valores altos en la t_{40} y muy pequeños en la t_3, lo que confirma que para grados de libertad grandes (mayores que 30) la distribución t de Student se aproxima muy bien por la N(0,1).

9.20 SIMULACIÓN DE MONTE CARLO: CÁLCULO DE UN ÁREA

En ocasiones no se pueden calcular áreas bajo ciertas curvas mediante la regla de Barrow, al no ser posible obtener una primitiva expresable como combinación de funciones elementales. En tales casos se hace necesario utilizar métodos numéricos para computar el área. Entre esos métodos cabe destacar el método de Monte Carlo, donde se emplean números aleatorios, en concreto valores obtenidos al azar de una distribución uniforme.

El problema que se acaba de describir surge, por ejemplo, al tratar de calcular el área comprendida entre la curva $y=exp(-x^2+1)$, las rectas verticales en $x=0$ y $x=3$ y el eje de abscisas. Para ello se efectúan lanzamientos de "dardos aleatorios" en un rectángulo que incluye al recinto en estudio. Después, se evalúa la fracción de lanzamientos que caen en el área a calcular y finalmente se da como estimación del área esa fracción multiplicada por la superficie del rectángulo.

El área que buscamos es la correspondiente a la zona rayada de la figura:

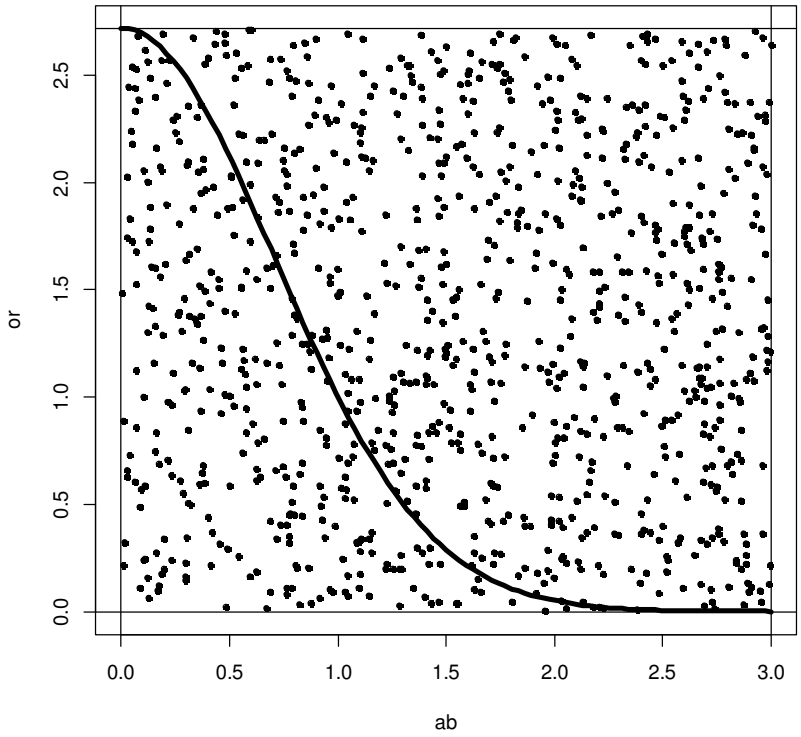



```
> #Definimos la función a integrar
> f<-function(x) exp(-x^2+1)
> #Definimos los límites del rectángulo
> lim.inf.x<-0
> lim.sup.x<-3
> lim.inf.y<-0
> lim.sup.y<-f(0)
>
> #Lanzamos n dardos
> n<-1000

> #Generamos las abscisas aleatorias de los dardos
> ab<-runif(n,lim.inf.x,lim.sup.x)
> #Generamos las ordenadas aleatorias de los dardos
> or<-runif(n,lim.inf.y,lim.sup.y)

> #Formamos la matriz de lanzamientos
> LANZ<-matrix(c(ab,or),nrow=n)

> #Se define una función para discriminar si un dardo cae en el área a calcular
> #V representa un vector de coordenadas las del dardo aleatorio
> g<-function(V) if(f(V[1])>=V[2]) 1 else 0
```

> #Aplicamos a cada dardo la función g para saber si ha dado en el blanco
> aciertos<-apply(LANZ,1,g)

> #Calculamos el área
> S<-(sum(aciertos)/n)*(lim.sup.x-lim.inf.x)*(lim.sup.y-lim.inf.y)
> S;integrate(f,0,3) #Valor que se obtiene con la función integrate() directamente
[1] 2.340441
2.408961 with absolute error < 8.6e-11

> #Efectuamos un gráfico para ver qué dardos han caído en el área en estudio y cuáles fuera de ella
> plot(ab,or,pch=20)
> curve(f(x),from=lim.inf.x,to=lim.sup.x,lwd=4,add=T)
> abline(h=0);abline(h=f(0))
> abline(v=0);abline(v=3)



En el ejemplo se ha realizado una única simulación; lo que se suele hacer, normalmente, es efectuar una serie de pruebas y calcular la media. Se simula un alto número de veces para disminuir el error de muestreo.

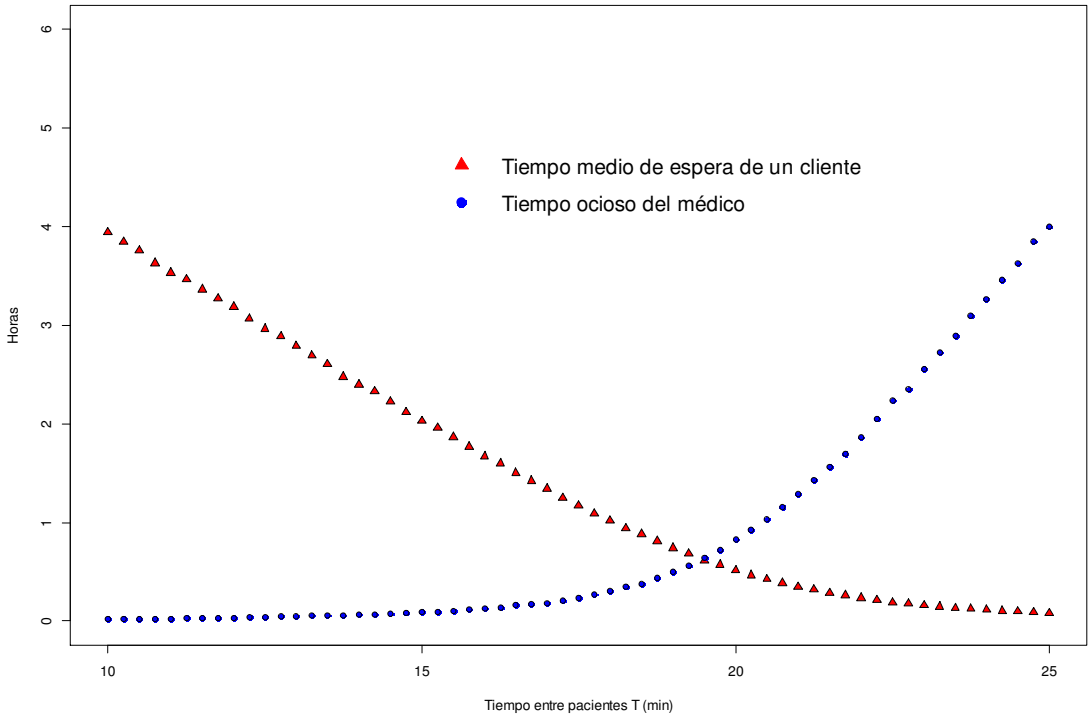
9.21 TIEMPO DE ESPERA EN LA CONSULTA DEL MÉDICO

Un médico pasa consulta entre las 8 de la mañana y las 8 de la noche de forma ininterrumpida. Por estudios previos sabe que de media tarda 20 minutos con cada paciente, pero no tiene claro cada cuánto tiempo debe citar a cada uno de ellos. Se supone que los pacientes llegan exactamente a la hora concertada. El tiempo que pasa un paciente en la consulta es una variable aleatoria normal (de media 20 y desviación típica 10). Se trata de simular el comportamiento del sistema (para cada valor del tiempo entre pacientes se realizarán 5000 simulaciones) y, a partir de ahí, tratar de optimizarlo, de modo que, por una parte, el tiempo que el médico esté sin atender a nadie (ocioso) sea pequeño y, por otra, los pacientes no tengan que esperar demasiado.

```
> #Efectuaremos el estudio para los primeros 48 pacientes

> n<-50 #Número de replicaciones (5000 días)
> mu<-20 #Media, en minutos, del tiempo de atención a cada paciente
> des<-10 #Desviación típica del tiempo de atención a cada paciente

> #El médico quiere decidir cada cuánto tiempo T debe citar a quienes acuden a
su consulta
> T<-seq(10,25,0.25) #Considera estos posibles valores de T en minutos
> T.hor<-T/60 #Valor de T en horas

> #Preparamos el gráfico que aparecerá al final
>plot(0,0,type="n",xlim=c(min(T),max(T)),ylim=c(0,6),xlab="Tiempo entre
pacientes T (min)",
+ ylab="Horas")

> for (s in T.hor)
+ {
+ A<-replicate( #Inicio de replicate
+ n,
+ { #Inicio función a replicar
+ #Numeramos a cada uno de los pacientes que llegan
+ paci<-seq(1:48)

+ t.entre.paci<-c(0,rep(s,47)) #Tiempo entre pacientes

+ hora.llegada<-rep(8,48)+cumsum(t.entre.paci)

+ t.at<-rnorm(48,mu/60,des/60) #Tiempo de atención a cada paciente
```

```
+ #Establecemos hora de inicio y fin de consulta y tiempo de espera de paciente y ocioso del médico para el paciente 1
+ in.cons<-numeric(48) #Hora de inicio de la consulta
+ in.cons[1]<-8

+ fin.cons<-numeric(48) #Hora de finalización de la consulta
+ fin.cons[1]<-8+t.at[1]

+ t.espera<-numeric(48) #Tiempo que espera cada paciente antes de empezar a ser atendido
+ t.espera[1]<-0

+ t.ocioso<-numeric(48) #Tiempo que el médico está ocioso
+ t.ocioso[1]<-0

+ #Establecemos hora de inicio y fin de consulta y tiempo de espera de paciente y ocioso del médico para los pacientes 2 a 48
+ for(i in 2:48)
+ { #Inicio del for de i
+ if(hora.llegada[i]>fin.cons[i-1])
+ {
+ in.cons[i]<-hora.llegada[i]
+ fin.cons[i]<-in.cons[i]+t.at[i]
+ t.espera[i]<-in.cons[i]-hora.llegada[i]
+ t.ocioso[i]<-in.cons[i]-fin.cons[i-1]
+ }
+ else
+ {
+ in.cons[i]<-fin.cons[i-1]
+ fin.cons[i]<-in.cons[i]+t.at[i]
+ t.espera[i]<-in.cons[i]-hora.llegada[i]
+ t.ocioso[i]<-in.cons[i]-fin.cons[i-1]
+ }
+ } #Fin del for de i

+ ESPERA<-sum(t.espera)/48;OCIOSO<-sum(t.ocioso)
+ c(ESPERA,OCIOSO) #Tiempos medidos en horas
+ } #Fin de función a replicar
+ ) #Fin de replicate

+ points(s*60,mean(A[1,]),pch=24,bg="red")
+ points(s*60,mean(A[2,]),pch=21,bg="blue")

+ } #Fin del for de s
```

```
> #Por último generamos la leyenda del gráfico
> legend(15,5,c("Tiempo medio de espera de un cliente","Tiempo ocioso del médico"),
+ cex=1.5,bty="n",pch=c(24,21),col=c("red","blue"),pt.bg=c("red","blue"))
```



En el gráfico se puede apreciar como a medida que el intervalo de tiempo entre pacientes aumenta, el tiempo medio que deben esperar hasta que empiezan a ser atendidos disminuye, pero a costa de aumentar el tiempo ocioso del médico. Este último podrá determinar un intervalo de tiempo óptimo para citar a los pacientes valiéndose de este gráfico.

BIBLIOGRAFÍA

- Crawley M.J. (2005): *Statistics. An Introduction using R.* New York: Wiley.

- Dalgaard P. (2002): *Introductory Statistics with R*. New York: Springer.

- Dallas E. Johnson (2000): *Métodos multivariados aplicados al análisis de datos.* México: Thompson Editores.

- Everitt Brian S. and Hothorn Torsten (2006): *A Handbook of Statistical Analysis Using R.* Boca Raton, Florida: Chapman & Hall/CRC.

- Fox John (2002): *An R and S-PLUS Companion to Applied Regression*. Thousand Oaks, California: Sage Publications, Inc.

- Grima Pere (2010): *La certeza absoluta y otras ficciones. Los secretos de la estadística (El mundo es matemático).* RBA Coleccionables, S.A.

- Guisande Cástor y Vaamonde Antonio (2012): *Gráficos estadísticos y mapas con R.* Madrid: Díaz de Santos.

- Hair J.F., Anderson R.E., Tatham R.L. y Black W.C. (1995): *Análisis multivariante.* Madrid: Pearson Educación S.A.

- Kay Steven M. (2006): *Intuitive Probability and Random Processes using MATLAB.* New York: Springer.

- Maindonald J. and Braun J. (2003): *Data Analysis and Graphics Using R.* Cambridge: Cambridge University Press.

- Murrell Paul (2006): *R Graphics*. Boca Raton, Florida: Chapman & Hall/CRC.

- Palmer Pol Alfonso Luis (1999): *Análisis de datos. Etapa exploratoria.* Madrid: Ediciones Pirámide.

- R Development Core Team (2000): *Introducción a R.* http://cran.r-project.org/doc/contrib/R-intro-1.1.0-espanol.1.pdf

- Robert C.P. and Casella G. (2010): *Introducing Monte Carlo Methods with R.* New York: Springer.

- Spector Phil (2008): *Data Manipulation with R.* Berkeley, California: Springer.