



Métodos Estadísticos con R y R Commander

Versión 1.0, junio de 2009

Prof. Dr. Antonio José Sáez Castillo

Dpto de Estadística e Investigación Operativa

Universidad de Jaén

Esta obra está bajo una licencia Reconocimiento-No comercial-Sin obras derivadas 3.0 España de Creative Commons. Para ver una copia de esta licencia, visite <http://creativecommons.org/licenses/by-nc-nd/3.0/es/> o envíe una carta a Creative Commons, 171 Second Street, Suite 300, San Francisco, California 94105, USA.



Métodos Estadísticos con R y R Commander by Antonio José Sáez-Castillo is licensed under a [Creative Commons Reconocimiento-No comercial-Sin obras derivadas 3.0 España License](http://creativecommons.org/licenses/by-nc-nd/3.0/es/).

Métodos Estadísticos
con R y R Commander

Prof. Dr. Antonio José Sáez Castillo
Departamento de Estadística e Investigación Operativa
Universidad de Jaén

Junio de 2009

Prólogo

El presente documento se creó originalmente como *Guión de Prácticas* para la asignatura *Métodos Estadísticos de la Ingeniería*, impartida en distintas especialidades de Ingeniería Técnica Industrial de la Universidad de Jaén. Sin embargo, dada la escasez de documentación en español acerca del uso de R, hemos querido redactar un documento más general que sirva como guía introductoria para usuarios principiantes de R.

Es sabido que existen en el mercado distintos paquetes de software estadístico propietario, como SPSS, Statgraphics, Minitab, Statistica, SAS, S-Plus, etc, que cubren todas las necesidades de cualquier usuario de técnicas estadísticas, básicas o avanzadas. Frente a estas opciones, en los últimos años R ha surgido con fuerza como alternativa de software libre en muy distintos ambientes docentes y de investigación.

En este sentido, antes de tomar la decisión de utilizar R como programa bajo el que desarrollar mis prácticas, he discutido ampliamente (en el buen sentido de la expresión) con compañeros y alumnos al respecto. Como conclusión de esos debates (a veces encendidos), quisiera plasmar varios aspectos que, a mi juicio, avalan la decisión:

- Después de algún tiempo, cada vez que se me cuestiona acerca de *¿por qué usar R?*, he desarrollado una respuesta rápida: *¿y por qué no hacerlo?* Facilita todo lo necesario, luego, como punto de partida, es una solución factible.
- Es software libre, lo que, en primer lugar, es una ventaja desde el punto de vista económico para la Universidad.
- Es software libre, lo que, en segundo lugar, es una ventaja desde el punto de vista económico para nuestro alumnado y para cualquier usuario en general. Supone la posibilidad de acceder al programa desde sus propios equipos fuera del horario de clase y en cualquier momento de su futuro ejercicio profesional.
- Es software libre, lo que a nivel personal, tiene unas connotaciones importantes sobre la manera de entender el papel que juega la Universidad en la sociedad. La construcción del

conocimiento hoy en día me parece muy difícil de entender sin el acceso libre por parte de toda la comunidad a las herramientas fundamentales que lo favorecen. Con ello no quiero decir que esté en contra de que determinadas empresas comercialicen paquetes de software estadístico, sino que me tranquiliza que existan alternativas a estos paquetes que permitan decidir en libertad si quiero comprarlo o no.

Probablemente alguien pensará que R debe tener alguna desventaja en comparación con los programas estadísticos más habituales. Mi experiencia me hace resaltar las siguientes:

1. Su entorno no es *tan bonito* como el de los programas comerciales. Esta cuestión está parcialmente resuelta con la utilización del paquete R Commander, cuya apariencia es muy similar al de otros programas comerciales.
2. Tampoco facilita las salidas, fundamentalmente las tablas de resultados, de forma tan directa como otros paquete comerciales, donde exportar estas tablas a paquetes ofimáticos como Microsoft Word es tan sencillo como *copiar-pegar*.
3. Hemos comprobado que algunas opciones de ventana del paquete R Commander hacen que el programa se cuelgue con frecuencia, obligándonos a reiniciarlo. Mi experiencia es que éste es un problema muy infrecuente en el uso de R desde la consola.

Para finalizar, quiero agradecer especialmente a mi compañero el Prof. Dr. Jesús Navarro Moreno, su ayuda en la depuración del documento y en la redacción de algunos de sus apartados.

Índice general

1. Instalación e introducción a R y R Commander	10
1.1. Introducción	10
1.2. Instalación de R	11
1.3. Instalación e introducción a R Commander	12
2. Manejo de datos	17
2.1. Introducción de datos nuevos	17
2.1.1. La hoja de datos	17
2.1.2. Introducción de una hoja de datos mediante código	18
2.1.3. Introducción de una hoja de datos en R Commander	21
2.1.4. Almacenamiento de un conjunto de datos mediante código. Las funciones <i>save</i> y <i>load</i>	22
2.1.5. Almacenamiento de un conjunto de datos en R Commander	23
2.1.6. Datos faltantes	23
2.2. Importar datos	23
2.2.1. Importar datos de tipo texto	24
2.2.1.1. Mediante código. La función <i>read.table</i>	24
2.2.1.2. Mediante R Commander	25
2.2.2. Importar archivos de tipo Excel	26
2.2.2.1. Mediante código. El paquete <i>RODBC</i>	26
2.2.2.2. Mediante R Commander	27
2.3. Exportar datos	27
2.3.1. Mediante código. Función <i>write.table</i>	27
2.3.2. Mediante R Commander	28
2.4. Recodificación de variables	28
2.4.1. Recodificación de una variable numérica	28

2.4.1.1.	Mediante código. Función <i>recode</i>	29
2.4.1.2.	Mediante R Commander	30
2.4.2.	Recodificación de una variable tipo carácter	30
2.5.	Cálculo de nuevas variables	31
2.5.1.	Mediante código	31
2.5.2.	Mediante R Commander	33
2.6.	Filtrado de datos	33
2.6.1.	Mediante código	33
2.6.2.	Mediante R Commander	34
2.7.	Almacenamiento de instrucciones y resultados	34
2.7.1.	Mediante código. El <i>script</i> y la <i>sesión de trabajo</i>	35
2.7.2.	Mediante R Commander	37
2.8.	Ejercicios propuestos	37
3.	Estadística descriptiva	39
3.1.	Cálculo de medidas de posición, dispersión y forma	39
3.1.1.	Mediante R Commander	39
3.1.2.	Mediante código	40
3.1.3.	Resúmenes por grupos	41
3.2.	Distribuciones de frecuencias	42
3.2.1.	Mediante R Commander	42
3.2.2.	Mediante código	42
3.3.	Diagrama de barras y diagrama de sectores	43
3.3.1.	Diagrama de barras para variables cualitativas	43
3.3.2.	Diagrama de sectores para variables cualitativas	44
3.4.	Histograma para variables continuas y discretas	45
3.4.1.	Histograma para variables continuas	45
3.4.2.	Histograma para variables discretas	46
3.5.	Detección de valores atípicos. Diagrama de caja	47
3.5.1.	Mediante R Commander	47
3.5.2.	Mediante código. La función <i>boxplot</i>	49
3.6.	Ejercicios propuestos	50

4. Manejo de distribuciones de probabilidad	53
4.1. Cálculo de probabilidades	53
4.1.1. Distribuciones discretas	53
4.1.1.1. Distribución binomial	54
4.1.1.2. Distribución de Poisson	55
4.1.1.3. Distribución geométrica	57
4.1.1.4. Distribución binomial negativa	57
4.1.2. Distribuciones continuas	57
4.1.2.1. Distribución uniforme	58
4.1.2.2. Distribución exponencial	58
4.1.2.3. Distribución Gamma	59
4.1.2.4. Distribución normal	60
4.1.3. Ejemplos	60
4.2. Cálculo de percentiles	60
4.2.1. Distribuciones discretas	61
4.2.1.1. Distribución binomial	61
4.2.1.2. Distribución de Poisson	62
4.2.1.3. Distribución geométrica	62
4.2.1.4. Distribución binomial negativa	62
4.2.2. Distribuciones continuas	63
4.2.2.1. Distribución uniforme	63
4.2.2.2. Distribución exponencial	63
4.2.2.3. Distribución Gamma	63
4.2.2.4. Distribución normal	63
4.3. Representaciones gráficas de distribuciones de probabilidad	63
4.3.1. Mediante R Commander	63
4.3.2. Mediante código. La función <i>plot</i>	64
4.4. Simulación de muestras	65
4.4.1. Muestra procedente de una distribución normal	66
4.4.1.1. Mediante R Commander	66
4.4.1.2. Mediante código	66
4.4.1.3. Comparación de la densidad teórica con el histograma de los datos simulados	67

4.4.2.	Muestra procedente de una distribución de Poisson	68
4.4.2.1.	Mediante R Commander	68
4.4.2.2.	Mediante código	69
4.4.2.3.	Comparación de la función masa teórica con el diagrama de barras de los datos simulados	69
4.4.3.	Muestras procedentes de otras distribuciones	70
4.5.	Ejercicios propuestos	70
5.	Estimación puntual y por intervalos de confianza	73
5.1.	Estimación máximo verosímil	73
5.1.1.	Para la distribución de Poisson	75
5.1.1.1.	Estimación de los parámetros	75
5.1.1.2.	Comparación del modelo estimado con la distribución de frecuencias	76
5.1.2.	Para la distribución geométrica	76
5.1.2.1.	Estimación del parámetro	77
5.1.2.2.	Comparación del modelo estimado con la distribución de frecuencias	78
5.1.3.	Para la distribución binomial negativa	78
5.1.3.1.	Estimación de los parámetros	78
5.1.3.2.	Comparación del modelo ajustado con la distribución de frecuencias	80
5.1.4.	Para la distribución exponencial	80
5.1.4.1.	Estimación de los parámetros	81
5.1.4.2.	Comparación del modelo ajustado con los datos muestrales	81
5.1.5.	Para la distribución Gamma	82
5.1.5.1.	Estimación de los parámetros	82
5.1.5.2.	Comparación del modelo ajustado con los datos muestrales	83
5.1.6.	Para la distribución normal	83
5.1.6.1.	Estimación de los parámetros	84
5.1.6.2.	Comparación del modelo ajustado con los datos muestrales	84
5.2.	Estimación por intervalos de confianza	85
5.2.1.	De la media de una distribución normal con varianza desconocida	86
5.2.2.	De la media de una distribución cualquiera, con muestras grandes	86
5.2.3.	De una proporción	87
5.2.4.	De la varianza de una distribución normal	88
5.3.	Ejercicios	88

6. Contraste de hipótesis paramétricas	91
6.1. Introducción	91
6.2. Contrastes sobre medias	91
6.2.1. Contraste sobre la media de una población	91
6.2.1.1. Resolución del problema mediante R Commander	92
6.2.1.2. Consideraciones finales	94
6.2.2. Contraste para la diferencia de medias de poblaciones independientes	94
6.2.2.1. Resolución mediante R Commander	95
6.2.2.2. Consideraciones finales	98
6.2.3. Contraste para la diferencia de medias de poblaciones apareadas	98
6.2.3.1. Resolución del problema mediante R Commander	98
6.2.3.2. Comentarios finales	100
6.2.4. Contrastes para medias mediante código. Función <i>t.test</i>	100
6.3. Contraste para la proporción en una población	101
6.3.1. Enunciado y planteamiento del problema	102
6.3.2. Preparación de los datos y resolución del problema mediante R Commander .	102
6.3.3. Resolución mediante código. La función <i>prop.test</i>	104
6.4. Contraste para la diferencia de proporciones	105
6.4.1. Enunciado y planteamiento del problema	105
6.4.2. Resolución del problema mediante R Commander	106
6.4.3. Resolución mediante código	108
6.5. Contraste para la comparación de varianzas	108
6.5.1. Enunciado y planteamiento del problema	108
6.5.2. Resolución del ejercicio mediante R Commander	109
6.5.3. Resolución mediante código. La función <i>var.test</i>	110
6.6. ANOVA de un factor	111
6.6.1. Enunciado y planteamiento del problema	111
6.6.2. Resolución mediante R Commander	111
6.6.3. Resolución mediante código. La función <i>avov</i>	114
6.7. Ejercicios	115
7. Contrastes de hipótesis no paramétricos	117
7.1. Contrastes de bondad de ajuste	117
7.1.1. Contraste χ^2 de bondad de ajuste	117

7.1.1.1.	Para la distribución de Poisson	119
7.1.1.2.	Para la distribución geométrica	120
7.1.1.3.	Para la distribución binomial negativa	122
7.1.1.4.	Para la distribución binomial	123
7.1.2.	Contraste de Kolmogorov-Smirnoff	124
7.2.	Contraste χ^2 de independencia	125
7.2.1.	Mediante R Commander	125
7.2.2.	Mediante código	127
7.3.	Ejercicios	127
8.	Regresión lineal simple	129
8.1.	Correlación	129
8.1.1.	Mediante R Commander	129
8.1.2.	Mediante código. Las funciones <i>pairs</i> , <i>cors</i> y <i>cor.test</i>	132
8.2.	Ajuste de la recta de regresión	134
8.2.1.	Mediante R Commander	135
8.2.2.	Mediante código. La función <i>lm</i>	137
8.3.	Predicciones, estimaciones e intervalos de confianza para ellas	138
8.3.1.	Mediante R Commander	138
8.3.2.	Mediante código. La función <i>predict</i>	139
8.4.	Algo sobre diagnosis del modelo	141
8.4.1.	Mediante R Commander	141
8.4.2.	Mediante código	142
8.5.	Ejercicios	143

Capítulo 1

Instalación e introducción a R y R Commander

Objetivos:

1. Instalar R y R Commander.
2. Familiarizarnos con la interfaz de R Commander.

1.1. Introducción

R es un lenguaje de programación especialmente indicado para el análisis estadístico. A diferencia de la mayoría de los programas que solemos utilizar en nuestros ordenadores, que tienen interfaces tipo ventana, R es manejado a través de una consola en la que se introduce código propio de su lenguaje para obtener los resultados deseados.

R fue inicialmente diseñado por Robert Gentleman y Ross Ihaka, miembros del Departamento de Estadística de la Universidad de Auckland, en Nueva Zelanda. Sin embargo, una de las grandes ventajas de R es que hoy en día es, en realidad, fruto del esfuerzo de miles de personas en todo el mundo que colaboran en su desarrollo.

Por otra parte, R se considera la versión libre de otro programa propietario, llamado S o S-Plus, desarrollado por los Laboratorios Bell. Aunque las diferencias entre R y S son importantes, la mayoría del código escrito para S funciona en R sin modificaciones.

El código de R está disponible como software libre bajo las condiciones de la licencia GNU-GPL, y puede ser instalado tanto en sistemas operativos tipo Windows como en Linux o MacOS X.

La página principal desde la que se puede acceder tanto a los archivos necesarios para su instalación como al resto de recursos del proyecto R es

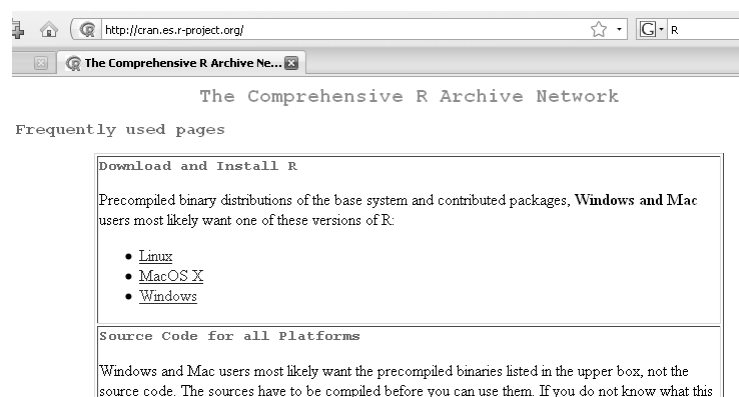


Figura 1.1: Instalación de R (I de III). Clicamos en *Windows*.

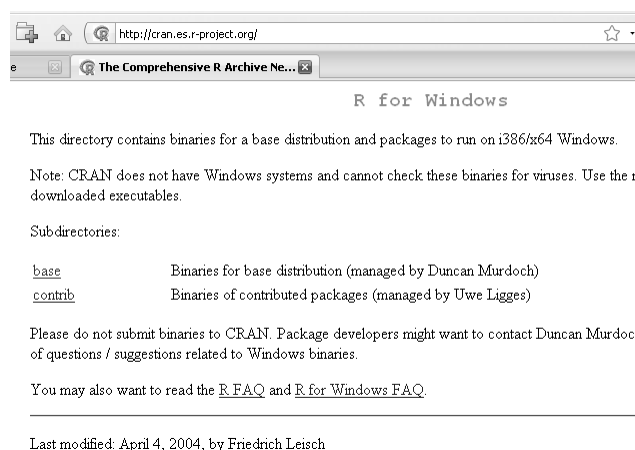


Figura 1.2: Instalación de R (II de III). Clicamos en *base*.

<http://www.r-project.org>.

1.2. Instalación de R

Vamos a explicar aquí como se realiza la instalación en Windows.

La descarga del archivo de instalación se realiza desde

<http://cran.es.r-project.org/>.

En dicha página debemos elegir la instalación en Windows, posteriormente la descarga de *base* y, finalmente, el archivo de instalación (ver Figuras 1.1, 1.2, 1.3). En el momento del inicio de la edición de esta guía la última versión era la 2.7.1. Es recomendable elegir siempre la última versión disponible, incluso actualizar la versión que tenemos instalada si surge una posterior. Con respecto a posibles problemas de incompatibilidades entre versiones, nosotros venimos trabajando con R desde la versión 1.3 y no hemos encontrado ninguna incidencia de esa índole.

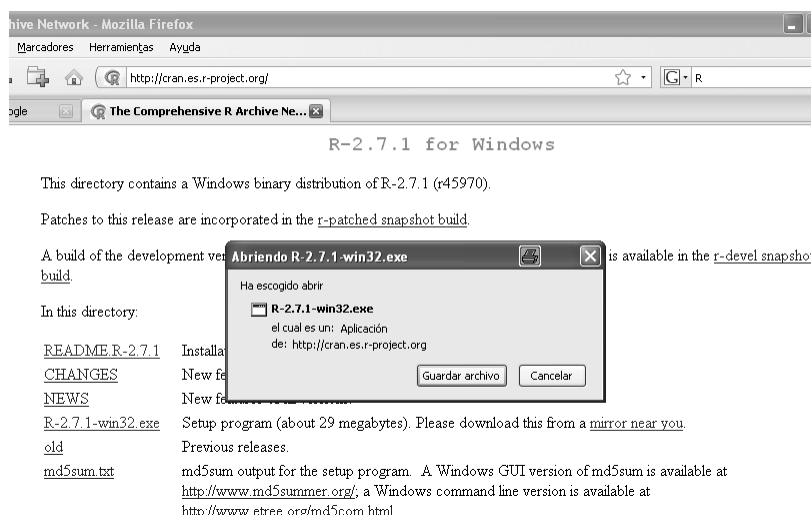


Figura 1.3: Instalación de R (III de III). Clicamos en *R-2.7.1-win32.exe*.

La instalación en sí con el archivo ejecutable es trivial. Tan sólo hay que continuar con las opciones que la instalación proporciona por defecto. Una vez concluida la instalación, podemos ejecutar el programa desde cualquiera de los iconos que nos genera.

Lo primero que nos aparece es una ventana, también llamada consola, donde podemos manejar R mediante la introducción de código.

1.3. Instalación e introducción a R Commander

R Commander es una interfaz tipo ventana que cubre la mayor parte de los análisis estadísticos más habituales en unos menús desplegables a los que estamos bastante acostumbrados, ya que la mayoría de los programas que utilizamos en cualquier sistema operativo son de este tipo. Podemos decir que es una manera de manejar R sin necesidad de aprender su código o casi nada de él, lo cual lo hace muy práctico cuando se está aprendiendo a usarlo.

Además, una de las funcionalidades que destacamos de R Commander es que, a pesar de que permite estos *atajos* mediante sus menús para no utilizar el código de R, escribe el código de las operaciones realizadas en una ventana de sintaxis o *ventana de instrucciones*, de manera que siempre lo veremos en la pantalla y podremos, poco a poco, ir aprendiéndolo, casi sin darnos cuenta.

La instalación de R Commander se realiza en 4 sencillos pasos:

1. En la consola de R seleccionamos *Paquetes* → *Instalar paquete(s)* (ver Figura 1.4).
2. Nos saldrá una ventana solicitando un *mirror* desde el que descargar los paquetes, de entre los cuales elegimos, obviamente, *Spain* (ver Figura 1.5). Hay que decir que ocasionalmente el *mirror* español *se cae*. Observaremos que eso ocurre si no aparece ninguna ventana de paquetes

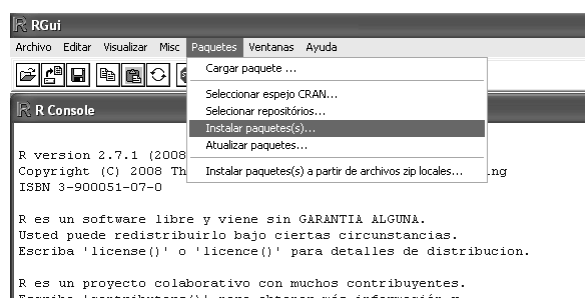


Figura 1.4: Instalación de R Commander (I de III)

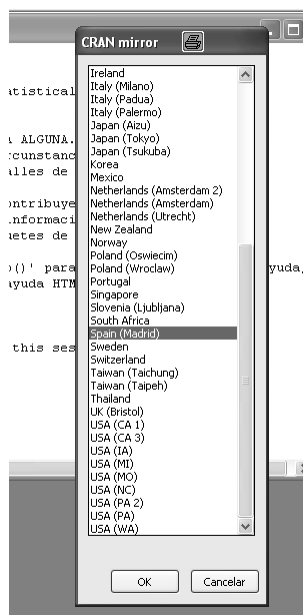


Figura 1.5: Instalación de R Commander (II de III)

o si vemos que los paquetes que aparecen: en ese caso, podemos elegir otro cualquiera de los *mirror* disponibles.

3. Se abrirá una ventana donde aparecen todos los paquetes disponibles para R. Seleccionamos todos los paquetes que comiencen por *Rcmdr* (ver Figura 1.6)¹.
4. A continuación, cargamos R Commander, introduciendo el siguiente código en la consola de R: `library("Rcmdr")`. Esta primera vez que cargamos R Commander nos pedirá la instalación de otros paquetes necesarios: debemos autorizarlo, eligiendo la opción, que aparece por defecto, de descarga desde *CRAN*.

Una vez cargado R Commander² veremos una ventana como la de la Figura 1.7. En ella podemos

¹Para seleccionar varios paquetes de una vez, pulsamos la tecla *Control* y con ella pulsada seleccionamos tantos paquetes como queramos.

²En ocasiones podemos, por error o por necesidad, cerrar la ventana de R Commander. En ese caso, para volver a abrirla sin tener que reiniciar R, ejecutaremos en la consola `Commander()`.

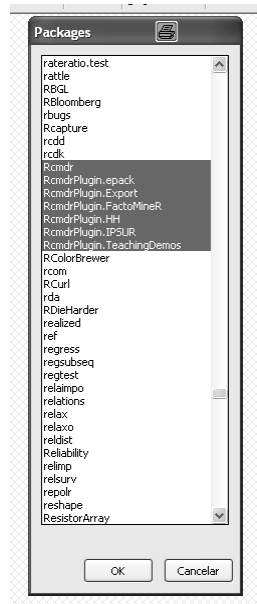


Figura 1.6: Instalación de R Commander (III de III)

distinguir 4 partes:

1. El menú de ventanas desplegable, con las opciones *Fichero*, *Editar*, *Datos*, ...

Es un menú de ventanas con entradas bastante intuitivas, que no requieren conocimientos de R, pero sí de Estadística.

2. La ventana de instrucciones.

Cada vez que ejecutemos alguna acción del menú, R Commander traducirá dicha acción a código de R y lo escribirá en esta ventana. Como decíamos, eso permite ir aprendiendo este código y, además, facilita la posibilidad de volver a ejecutar la misma acción o una ligera variante de la misma retocando el código, sin tener que volver a utilizar el menú.

Por otra parte, esta ventana de instrucciones es equivalente a la consola de R, lo que, por ejemplo, nos sirve para utilizarla como una calculadora. Para ello podemos, por ejemplo, escribir, $2+2$, clicar en el botón de *ejecutar*, obteniendo el resultado. El botón *Ejecutar* es equivalente a *Control+R*.

3. La ventana de resultados.

Si hemos realizado ese sencillo ejemplo en la ventana de instrucciones, habremos visto que el resultado aparece en esta ventana. En general, cualquier resultado de R Commander será mostrado aquí.

4. La ventana de mensajes.

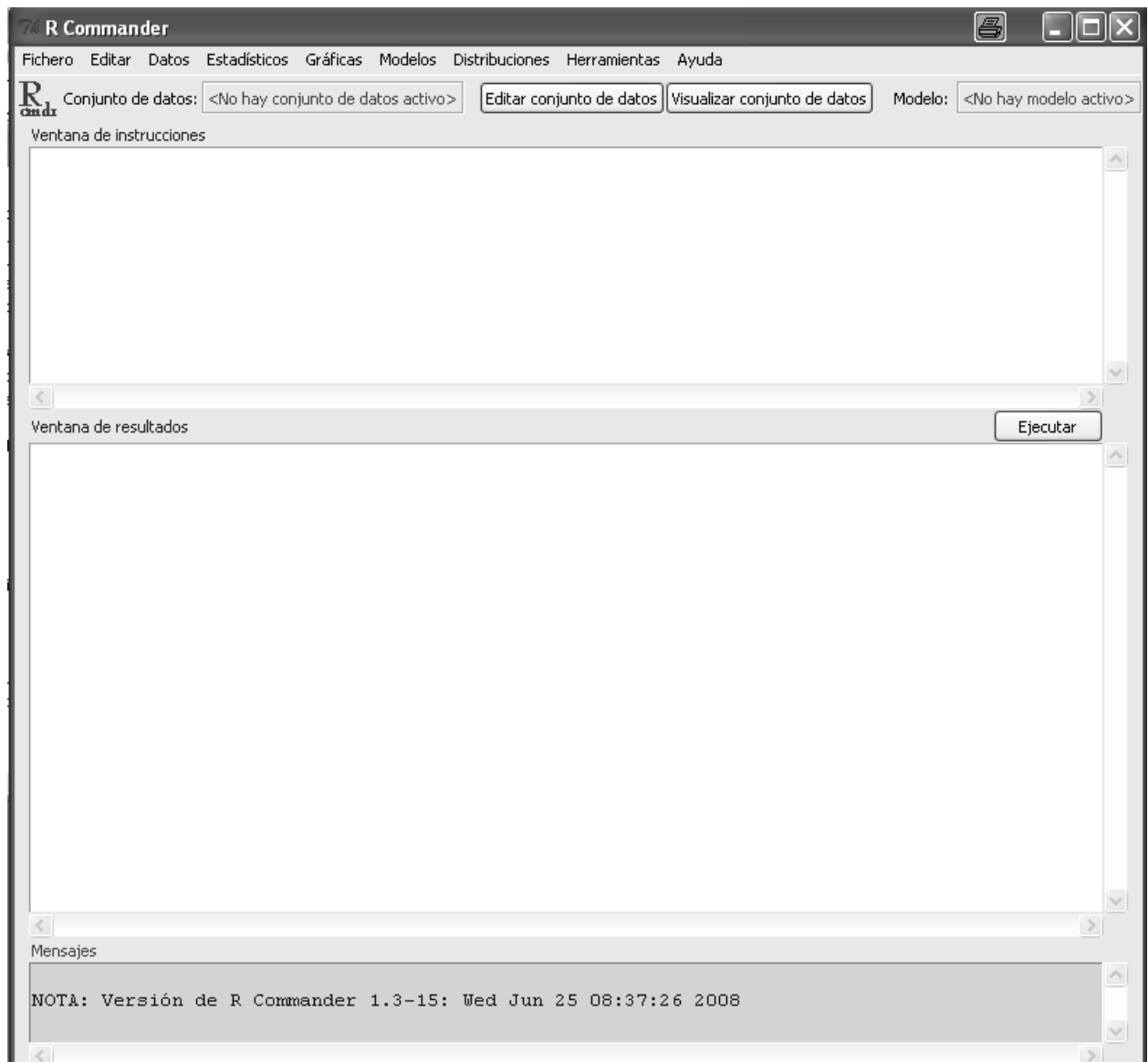


Figura 1.7: Interfaz de R Commander

Es la más inferior de todas y aparece ligeramente sombreada. Sirve para que R Commander nos informe de cualquier aspecto, especialmente de errores cometidos.

Para finalizar esta breve introducción a R Commander, queremos comentar que los paquetes adicionales (*plugins*) que hemos instalado junto con R Commander son complementos que diversos autores de contenidos de R han ido poniendo a disposición de la comunidad de usuarios de R. Cada uno de ellos tiene una funcionalidad concreta, pero la práctica nos hace recomendar que, en lo sucesivo, los carguemos todos sin excepción. Al hacerlo algunas opciones de análisis de los menús desplegable de R Commander cambiarán. Decir por último que para cargar estos *plugins* se debe elegir en el menú de R Commander *Herramientas* → *Cargar plugins de R Commander* y seleccionarlos. Se pedirá reinicializar R Commander, tras lo cual todos ellos están disponibles.

Capítulo 2

Manejo de datos

Objetivos:

1. Introducir y almacenar un nuevo conjunto de datos.
2. Importar datos ya almacenados en distintos formatos.
3. Exportar datos en algún formato estandar.
4. Recodificar variables.
5. Calcular nuevas variables a partir de otras ya existentes.
6. Filtrar casos.

2.1. Introducción de datos nuevos

2.1.1. La hoja de datos

Vamos a ponernos en una situación general en la que tenemos información sobre n individuos, información que se refiere a k variables. En ese caso, la forma en que en Estadística se organiza toda esta información es una matriz de dimensiones $n \times k$ donde cada fila representa un individuo o caso y cada columna representa una variable o dato.

Por ejemplo, consideremos que tenemos la puntuación en una prueba escrita (x) y en una prueba

oral (y), de una muestra de 10 personas. Su matriz de datos es la siguiente:

161	159
203	206
235	241
176	163
201	197
188	193
228	209
211	189
191	169
178	201

En esta matriz la primera columna corresponde a la variable x y la segunda a la variable y , mientras que, por ejemplo, la fila 4ª corresponde a la persona nº 4 de la muestra. En resumen, no lo olvidemos, los individuos están en las filas y las variables en las columnas.

2.1.2. Introducción de una hoja de datos mediante código

De cara a introducir una hoja de datos debemos comenzar comentando que para R existen distintos tipos de objetos. Ahora mismo vamos a centrarnos en los objetos de tipo *vector* y en los objetos de tipo hoja de datos o *data frame*.

En primer lugar, un objeto tipo *vector* estará constituido por un conjunto de elementos que pueden ser números o caracteres. En el caso del ejemplo anterior, los datos de cada variable serían un vector, de dimensión 10, o los datos de cada caso también serían un vector, de dimensión 2.

Para definir un vector se utiliza la función de concatenación $c()$. Su uso es sencillísimo. A modo de ejemplo veamos cómo introducir los datos de las variables x e y en forma de vector:

```
x<-c(161,203,235,176,201,188,228,211,191,178)
y<-c(159,206,241,163,197,193,209,189,169,201)
```

En la primera línea hemos asignado al vector el nombre x mediante el símbolo $<-$, de tal manera que al escribir posteriormente x y pulsar *INTRO* el programa devuelve en pantalla el valor asignado. Lo mismo ocurre al definir y y posteriormente imprimirlo en pantalla.

Por su parte, un objeto de tipo hoja de datos o *data frame* puede ser definido como una unión de vectores de la misma dimensión y corresponde con lo que en el apartado anterior queremos introducir como tal hoja de datos.

La forma de definir una hoja de datos es sencilla: por ejemplo,

```
Datos<-data.frame(Prueba.escrita=x,Prueba.oral=y)
```

Así, hemos llamado a la hoja de datos *Datos*. Por su parte, a la primera variable la hemos llamado *Prueba.escrita* y a la segunda *Prueba.oral*. Después, si escribimos en la consola *Datos*, visualizaremos el resultado.

Tenemos, por tanto, la hoja de datos que se llama *Datos*, que, a su vez, contiene dos variables, *Prueba.escrita* y *Prueba.oral*. Si queremos ahora trabajar con alguna de esas dos variables tenemos dos opciones:

1. Podemos referirnos a cualquiera de ellas poniendo el nombre de la hoja seguido del símbolo *\$* y del nombre de la variable. Es decir,

```
Datos$Prueba.escrita
```

```
Datos$Prueba.oral
```

2. Si no queremos escribir en demasiadas ocasiones *Datos\$*, podemos hacer que la hoja de datos se convierta en la *hoja de datos activa* mediante la función *attach*:

```
attach(Datos)
```

```
Prueba.escrita
```

```
Prueba.oral
```

```
detach(Datos)
```

Observemos que al hacer *attach(Datos)* ya no tenemos que escribir *Datos\$* y a continuación el nombre de las variables. Eso es porque ahora *Datos* es la hoja de datos activa. Podremos referirnos a las variables así hasta que hagamos *detach(Datos)*, momento en el que dejará de ser la hoja de datos activa.

Si queremos referirnos a un elemento concreto de una hoja de datos, podemos identificarlo por su posición dentro de la matriz que constituye la hoja de datos. Por ejemplo, siguiendo con el ejemplo de las pruebas, supongamos que queremos saber el resultado de la prueba escrita (1ª variable) del 5º individuo de la muestra. En ese caso pondríamos en la consola

```
Datos$Prueba.escrita[5]
```

o también,

```
Datos[5,1].
```

Muertes/año	Frecuencia
0	109
1	65
2	22
3	3
4	1

Cuadro 2.1: Distribución de frecuencias

Supongamos que queremos los resultados de la prueba oral (2ª variable) de los individuos 5º, 8º y 10º. Los lograríamos poniendo

```
Datos$Prueba.oral[c(5,8,10)]
```

o bien

```
Datos[c(5,8,10),2].
```

O, por ejemplo, si deseamos los resultados de ambas pruebas de los individuos del 3º al 8º pondríamos

```
Datos[3:8,1:2]
```

o simplemente

```
Datos[3:8,].
```

Otra función muy útil para introducir datos es la función repetición *rep()*. Imaginemos que queremos introducir los datos que se resumen en la distribución de frecuencias que aparece en el Cuadro 2.1. Si lo hacemos con la función *c()* tendríamos que meter 109 ceros, 65 unos, ... En su lugar, podemos introducir la repetición de 109 ceros como *rep(0,109)*, 65 unos como *rep(1,65)*... de manera que los datos quedarían como sigue:

```
muertes<-c(rep(0,109),rep(1,65),rep(2,22),rep(3,3),4).
```

Por último, si queremos introducir los datos de una variable cualitativa debemos utilizar la función *factor()*. “*Factor*” es la forma en que R llama a las variables cualitativas para poder realizar cualquier tipo de análisis en la que interviene una variable de este tipo.

Por ejemplo, supongamos que una variable consiste en datos de 150 personas, de los cuales los 80 primeros son hombres y los 70 siguientes mujeres. La variable, que voy a llamar *genero*, se introduce así:

```
genero<-factor(c(rep(“HOMBRE”,80),rep(“MUJER”,70)))
```

Es decir, repetimos 80 y 70 veces la palabra “HOMBRE” y “MUJER”, las concatenamos y convertimos ese vector en un factor o variable cualitativa mediante la función *factor*.

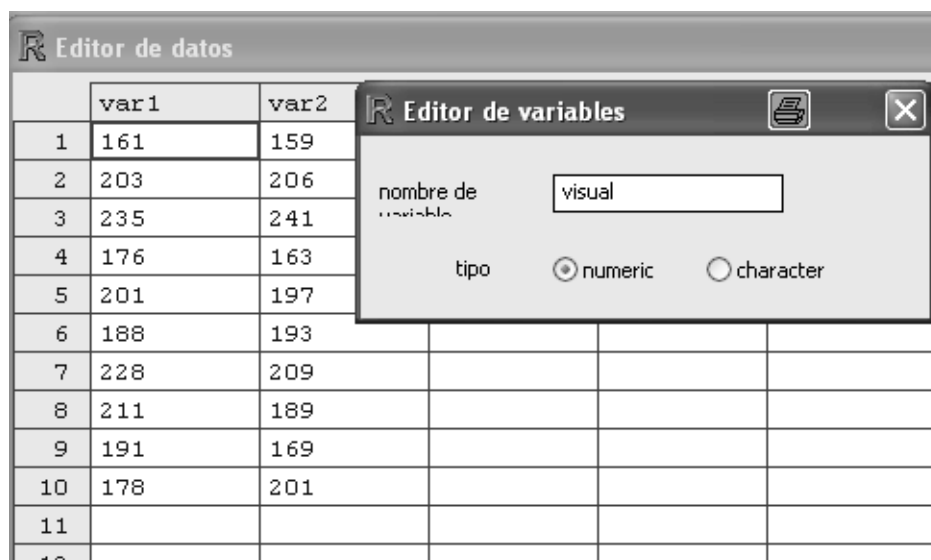


Figura 2.1: Introduciendo datos y renombrando variables

2.1.3. Introducción de una hoja de datos en R Commander

Para introducir los datos de las dos pruebas en R Commander elegimos *Nuevo conjunto de datos* del menú *Datos*. Eso abre el *editor de datos* que, en primer lugar, nos pedirá un nombre para la matriz de datos (ahora hemos elegido *Pruebas*) y a continuación abrirá una ventana con casillas parecida a una hoja de cálculo de Excel. En esta hoja debemos introducir los datos con la misma estructura que tiene la matriz de datos que acabamos de escribir, con los individuos en las filas y las dos variables en dos columnas.

Una vez introducidos los datos, debemos nombrar las variables, es decir, las columnas, con nombres sencillos que nos recuerden a qué variable corresponde cada columna. Para ello clicamos con el ratón sobre la parte superior de cada columna, donde R Commander nombra por defecto las variables como *var1*, *var2*, etc. y escribimos otros nombres más acordes con nuestros datos¹. En este caso he nombrado las variables como *escrita* y *oral* (Figura 2.1).

Por último, fijémonos que en la ventana donde R Commander nos permite cambiar el nombre de la variable aparece señalada la opción *numeric*. Eso es porque al haber introducido datos numéricos, el programa ya ha asimilado que es una variable numérica. Si hubiéramos introducido datos cualitativos mediante caracteres no numéricos, aparecería activada la opción *character*. En ese caso, además, R Commander considerará que una variable tipo *character* es un factor.

Para terminar, cerramos la ventana del editor de datos. En ese momento, R habrá almacenado los datos introducidos convirtiéndolos en lo que R Commander llama el *conjunto de datos activo* y que antes, con el código, hemos logrado con la función *attach()*. Observad que justo encima de

¹No deben introducirse nombres que incluyan espacios, ni caracteres extraños, ni muy largos.

la ventana de instrucciones aparece ahora una pestaña informativa que pone *Conjunto de datos: Pruebas*. Esta ventana especifica que, en efecto, el conjunto de datos activo en este momento es el que nosotros hemos llamado *Pruebas*.

Finalmente, podemos retocar estos datos pulsando la pestaña *Editar conjunto de datos* que hay justo sobre la ventana de instrucciones o simplemente visualizarlos pulsando la pestaña *Visualizar conjunto de datos*.

Como comentario final, debemos advertir que por problemas con el lenguaje de programación en el que está diseñado R y R Commander, es frecuente que al abrir y cerrar el editor de datos, el programa se cuelgue, por lo que recomendamos hacerlo sólo cuando sea imprescindible. Además, también es recomendable cerrar la ventana del editor, evitando dejarla minimizada.

2.1.4. Almacenamiento de un conjunto de datos mediante código. Las funciones *save* y *load*

Debemos tener presente que el conjunto de datos que hemos introducido está sólo almacenado temporalmente, y que si cerramos R serán eliminados. Para que esto no ocurra podemos guardarlos y cargarlos con posterioridad.

En este sentido, la función *save* nos permite almacenar varios objetos, por ejemplo, una o varias hojas de datos, en archivos para poder utilizarlos con posterioridad. Su uso es muy sencillo. Basta con indicarle qué objetos queremos guardar y el nombre del archivo con el que lo queremos guardar. Aunque no es imprescindible, es recomendable que este fichero tenga la extensión propia de los ficheros de datos de R, que es *.RData*. Vamos a aplicarlo a la hoja de datos con los resultados de las dos pruebas:

```
setwd('D:/Asignaturas/Curso R')  
save(Datos,files='Pruebas.RData')
```

1. La función *setwd()* permite cambiar el directorio donde estamos trabajando (*working directory*). Hay que tener en cuenta que los datos se guardarán en ese directorio.
2. Después hemos ejecutado la función *save()* para guardar *Datos* en el fichero *Pruebas.RData*.

Si reiniciamos el programa o borramos todos los objetos mediante *rm(list=ls(all=TRUE))*², al poner *Datos* daría error, porque tal objeto ya no existe. Sin embargo, al cargar el archivo *Pruebas.RData* mediante la función *load()*, recuperamos la hoja de datos:

```
load('Pruebas.RData')
```

²Elegimos en el menú de la consola de R *Misc*→*Remove todos los objetos*.

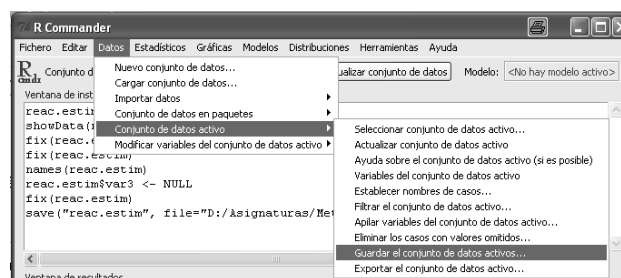


Figura 2.2: Guardando un conjunto de datos activo

2.1.5. Almacenamiento de un conjunto de datos en R Commander

Para guardar una hoja de datos en R Commander, seleccionamos en el menú *Datos* la opción *Conjunto de datos activo* y, dentro de ésta, *Guardar el conjunto de datos activo* (Figura 2.2). A continuación nos pedirá un nombre y un directorio donde almacenar el fichero, cuya extensión por defecto será *.rda*³.

Si posteriormente queremos cargar estos datos, no tenemos más que usar la opción del menú *Datos* → *Cargar conjunto de datos* y buscar el archivo correspondiente mediante la ventana del explorador que se abre.

2.1.6. Datos faltantes

¿Qué ocurre si, por alguna razón, nos falta el dato de una variable referida a un individuo concreto? Eso se conoce en Estadística como *dato faltante* o *missing data*.

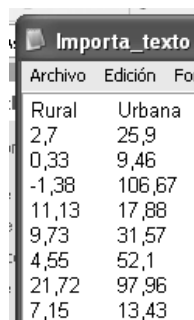
Para R es muy fácil identificar esos datos cuando los introducimos *a mano*: basta con dejar esa casilla vacía, en cuyo caso el editor de datos escribirá en ella NA, acrónimo de *Not Available*.

Si estamos trabajando con código, el carácter para un dato faltante es también *NA*. Por ejemplo, si tengo un vector de 5 datos y el 3º de ellos es un dato faltante, debería escribir, por ejemplo, `c(3,2,NA,2,8)`.

2.2. Importar datos

Hay que decir que introducir datos a mano puede convertirse en una tarea muy pesada a poco que el número de casos o de variables sea medianamente alto. Hoy en día, por otra parte, es bastante común tener los datos almacenados en algún tipo de formato electrónico y la clave del éxito para aprovechar estos recursos y no tener que introducir los datos manualmente radica en hacer que

³La extensión que por defecto asigna R a sus archivos de datos es *RData*, mientras que R Commander utiliza por defecto *rda*. Sin embargo, no hay ningún problema con que desde R se guarden o se lean archivos con extensión *rda* ni para que R Commander guarde o lea archivos con extensión *RData*.



Rural	Urbana
2,7	25,9
0,33	9,46
-1,38	106,67
11,13	17,88
9,73	31,57
4,55	52,1
21,72	97,96
7,15	13,43

Figura 2.3: Archivo de texto. Variables separadas por tabulaciones y nombres de variables incluidos en la primera línea

nuestro programa estadístico, en este caso R, *lea* estos datos.

Los formatos de archivo más habituales en los que nos podemos encontrar unos datos son, en primer lugar, los archivos tipo texto (con extensión *.txt*) y, en segundo lugar, los archivos de Microsoft Excel (con extensión *.xls*). Existen otros muchos formatos, pero casi siempre son convertibles a estos dos tipos. De hecho, el propio Excel o su análogo en OpenOffice, Calc, permiten transformar archivos de texto en archivos *.xls* y viceversa.

2.2.1. Importar datos de tipo texto

Los archivos de tipo texto que contienen datos suelen tener una estructura en la que los individuos están en filas distintas y las variables están separadas por algún tipo de carácter, tales como comas, tabulaciones, espacios u otros.

Además, es posible que la primera fila contenga los nombres de las variables. Estas dos cuestiones, el carácter que separa las variables y el hecho de que el archivo incluya los nombres de las variables, son las necesarias para importar los datos.

Vamos a ver cómo se hace mediante un ejemplo. En el archivo *Importa_texto.txt* aparecen datos relativos al grado de contaminación, en *ppm* de unas muestras de terreno recopiladas en el ámbito rural y urbano, respectivamente. Si abrimos este fichero con el bloc de notas, tiene el aspecto que aparece en la Figura 2.3. En ella podemos ver que, en efecto, se incluye el nombre de las variables y que éstas están separadas por tabulaciones. Además, los decimales están separados por comas, no por puntos.

2.2.1.1. Mediante código. La función *read.table*

La función que R utiliza para importar archivos de tipo texto es *read.table*. Esta función tiene multitud de opciones, pero nosotros vamos a destacar sólo las que creemos que son más importantes. Concretamente, la sintaxis de dicha función, en general, sería la siguiente:

```
read.table(archivo,header=FALSE,sep='',dec='.',na.strings='NA')
```

En esta línea:

- *archivo* sería el nombre del archivo que queremos importar.
- *header* puede tomar el valor *TRUE*, si sabemos que la primera línea del archivo (cabecera) contiene los nombres de las variables, o el valor *FALSE*, si no lo hace.
- *sep* se refiere al carácter que separa los datos. En nuestro ejemplo son tabulaciones, luego deberemos poner “\t”. El valor por defecto es vacío, que corresponde a uno o más espacios en blanco o a tabulaciones.
- *dec* se refiere al carácter que separa los números decimales. Hay que tener cuidado con él porque en español lo correcto es separar con comas, pero en el mundo anglosajón lo es hacerlo con puntos. De hecho, el punto es la opción por defecto.
- *na.strings* se refiere al carácter que en el archivo original identifica a los datos faltantes. Por defecto, se supone que un dato faltante aparecerá como “NA”, pero podemos poner cualquier otro. Si el dato faltante simplemente no aparece en el archivo original, será entendido como tal dato faltante sin necesidad de especificar nada más.

Por ejemplo, en el caso del archivo *Importa_texto.txt*, tendríamos lo siguiente:

```
Datos<-read.table("Importa_texto.txt",header=TRUE,sep='\t',dec=',')
```

Ahora *Datos* ya es una hoja de datos manejable como hemos descrito en los apartados anteriores.

2.2.1.2. Mediante R Commander

Nos vamos a la opción del menú *Datos* → *Importar datos* → *desde archivo de texto o porta-papeles...* Se abre una ventana como la de la Figura 2.4, en la que debemos elegir las opciones del archivo *Importa_texto.txt*:

- Nombre: Por ejemplo, *Datos*.
- Nombre de las variables en el fichero: activado.
- Indicador de valores ausentes: lo dejamos en blanco.
- Separador de campos: tabuladores.
- Carácter decimal: coma.



Figura 2.4: Importando archivos tipo .txt

Después buscamos en la ventana que se abre del explorador dicho archivo y lo seleccionamos. Ahora el conjunto de datos activo es *Datos*. Si lo deseamos, podemos guardar este conjunto de datos activo con formato *.rda* para que la próxima vez no tengamos que importarlo de nuevo.

2.2.2. Importar archivos de tipo Excel

2.2.2.1. Mediante código. El paquete *RODBC*

Para importar archivos de tipo Excel⁴, debemos previamente instalar el paquete *RODBC*, mediante el menú de instalación de paquetes de R.

Vamos a ilustrar el procedimiento de instalación de un archivo llamado *Concentraciones.xls*. Este archivo tiene una única hoja de cálculo, llamada *Hoja1*, que recoge la concentración de ésteres totales de 16 muestras de vino, en *mg/L*. Recordemos que un archivo de Excel puede contener varias hojas de cálculo.

Las sentencias a ejecutar serían las siguientes:

```
library(RODBC)
conexion<-odbcConnectExcel("Concentraciones.xls")
Datos<-sqlQuery(channel=conexion,"select * from [Hoja1$]")
close(conexion)
```

⁴Nosotros trabajamos con versiones de Excel hasta la 2003. Hemos visto en la ayuda que también es posible importar de la versión de Excel 2007, pero no lo hemos probado.

En ese caso, *Datos* pasa a ser una hoja de datos manejable por R. Esencialmente, debemos tener en cuenta el nombre del fichero de Excel y el de la hoja que contiene los datos.

2.2.2.2. Mediante R Commander

En el caso de los archivos tipo Excel, R Commander no necesita que le digamos nada, ya que detecta automáticamente los nombres de las variables si están presentes. No obstante, éstos no deben incluir caracteres extraños, y deben estar todos los nombres de todas las variables o ninguno; en cualquier otro caso, la importación podría ser inválida.

Tan sólo tenemos que utilizar la opción del menú *Datos* → *Importar datos* → *desde conjunto de datos Excel, Access o dBase...*, eligiendo después el archivo a través de la ventana del explorador.

2.3. Exportar datos

Existe la posibilidad de exportar el conjunto de datos activo para que pueda ser leído por cualquier otro programa. El formato más sencillo en que podemos hacerlo mediante R es el formato de texto *.txt*.

2.3.1. Mediante código. Función *write.table*

La función *write.table* permite crear archivos de texto que contienen hojas de datos de R. La sintaxis de dicha función, con las opciones más habituales, es la siguiente:

```
write.table(hoja,file="fichero.txt",sep="\t",na="NA",dec=".",row.names=TRUE,
col.names=TRUE)
```

Vamos a comentar los detalles de cada argumento:

- *hoja* se refiere al nombre de la hoja de datos que queremos exportar.
- *fichero.txt* será el nombre del fichero donde queremos exportar los datos.
- *sep="\t"* quiere decir que los datos estarán separados por una tabulación. También podemos poner una coma, un espacio, ...
- *na="NA"* se refiere a la forma en que se guardarán los datos faltantes. Si queremos que los deje en blanco, pondremos *na=""*.
- *dec="."* indica el carácter con el que se separan los decimales.
- *row.names* indicará si queremos que incluya en el fichero los nombres de las filas.

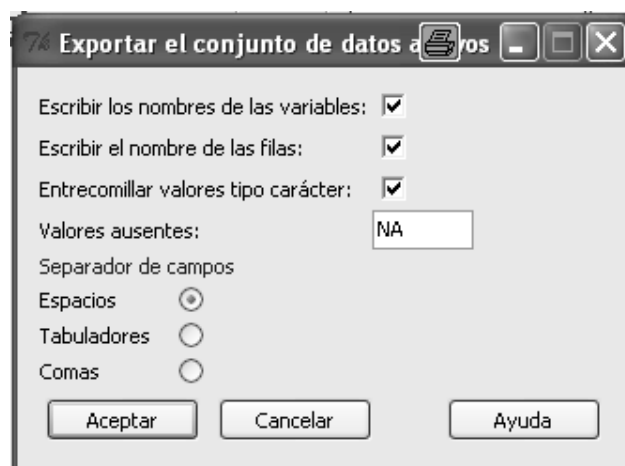


Figura 2.5: Exportar datos

- *col.names* indicará si queremos que se incluyan los nombres de las variables.

Por ejemplo, teníamos una hoja de datos con los resultados de las pruebas de 10 alumnos (se llamaba *Datos*). Imaginemos que queremos exportarlos en un fichero llamado *Pruebas.txt*, con los datos separados por comas y con los nombres de las variables. El código sería:

```
write.table(Datos,file='Pruebas.txt',sep=',',row.names=FALSE,col.names=TRUE)
```

2.3.2. Mediante R Commander

Utilizaremos la opción del menú *Datos* → *Conjunto de datos activo* → *Exportar el conjunto de datos activo*. Las opciones pueden ser modificadas, pero las que aparecen por defecto son las más recomendables (ver Figura 2.5).

2.4. Recodificación de variables

Recodificar una variable consiste en construir una nueva variable mediante la transformación de los valores de otra variable. Esta recodificación es bastante útil en muchas aplicaciones, como iremos viendo. En este apartado vamos a ver un par de ejemplos que nos ayudarán a ver cuál es el procedimiento a seguir.

2.4.1. Recodificación de una variable numérica

En el archivo *Jaencordoba.xls* aparece la población de los municipios de Jaén y Córdoba junto con el código que el INE le asigna a cada uno de ellos, código que coincide con parte del código postal. Si queremos analizar por separado los municipios de ambas provincias, ¿cómo podemos distinguir los que son de Córdoba de los que son de Jaén? El código del INE nos puede ayudar,

ya que todos los códigos de Jaén empiezan por 23 y todos los de Córdoba por 14. Por tanto, la condición que identifica a un municipio de Jaén mediante su código postal es que éste es mayor que 23000 y menor que 24000, mientras que en Córdoba esta condición es ser mayor que 14000 y menor que 15000.

Tenemos así que podremos distinguir los municipios de las dos provincias si construimos una nueva variable a partir del código postal con esas dos condiciones.

2.4.1.1. Mediante código. Función *recode*

En la sintaxis de la función *recode()* debemos especificar cuál es la variable a recodificar, cuáles son los criterios de recodificación y si deseamos que la nueva variable resultante de la recodificación sea un factor o no. Analicemos la forma de hacerlo sobre el ejemplo. Vamos a suponer que el nombre de la hoja de datos es *Datos* y que la variable que contiene los códigos del INE se llama *ine*. La sintaxis sería la siguiente:

```
Datos$Provincia<-recode(Datos$ine,  
  "14000:14999='Córdoba' ; 23000:23999='Jaén'", as.factor.result=TRUE)
```

- En primer lugar hemos escrito el nombre de la nueva variable, *Provincia*. Como queremos que sea una nueva variable de la hoja de datos *Datos*, hemos escrito *Datos\$Provincia*.
- Como primer argumento de *recode()* hemos escrito la variable que vamos a recodificar, *Datos\$ine*.
- A continuación hemos especificado los criterios de recodificación. Deben estar siempre entre comillas⁵. Si hay más de uno, que es lo más habitual, deben estar separados por puntos y comas. Finalmente, los nombres de los niveles de las variables recodificadas deben estar entre comillas simples⁶.

En el ejemplo queremos que todos los códigos entre 14000 y 15000 sean asignados a Córdoba. Escribir *14000:14999* es una forma rápida de generar el vector con todos los enteros entre 14000 y 14999. Así, el primer criterio de recodificación es *14000:14999='Córdoba'*.

- Finalmente, *as.factor.result=TRUE* indica que la nueva variable *Provincia* será un factor. Eso es conveniente para futuros análisis estadísticos.

⁵Las situadas en el 2 de nuestros teclados.

⁶Las situadas justo a la derecha del 0 de nuestros teclados.



Figura 2.6: Recodificando una variable numérica

2.4.1.2. Mediante R Commander

Vamos a ver cómo se hace mediante R commander.

Importamos en primer lugar el archivo. Seleccionamos la opción *Datos → Modificar variables del conjunto de datos activo → Recodificar variables*. Nos aparece la ventana de la Figura 2.6. En ella ya están incluidas las entradas necesarias para nuestra recodificación:

1. La variable a recodificar: *ine*.
2. El nombre de la nueva variable: *provincia*.
3. Las condiciones que determinan la recodificación. Observad que, como comentábamos, para especificar todos los números entre un valor a y otro valor b se debe poner $a : b$. Por otra parte, como queremos que los nuevos valores sean caracteres (Jaén, Córdoba), deben escribirse entre comillas.
4. La opción *Convertir cada nueva variable en factor* se deja activada para que la variable recodificada sea considerada como un factor.

2.4.2. Recodificación de una variable tipo carácter

La única diferencia con la recodificación que hemos descrito de una variable numérica es que los valores de la variable cualitativa o de tipo carácter deben escribirse entre comillas en las instrucciones para la recodificación.

Por ejemplo, si deseamos convertir la variable género, que es cualitativa con valores “Hombre” y “Mujer”, en una variable numérica tipo factor con valores 0 (para los hombres) y 1 (para las

mujeres), deberíamos poner ‘Hombre’=0 y ‘Mujer’=1 en los criterios de recodificación de *recode()* o en la ventana *Introducir directrices para la recodificación* de R Commander.

2.5. Cálculo de nuevas variables

En ocasiones es necesario realizar cálculos sobre las variables del conjunto de datos durante la realización de su análisis estadístico.

Por ejemplo, el archivo *JaenUsosSuelo.xls* contiene la distribución del suelo de los 96 municipios de la provincia de Jaén según su cobertura vegetal. Concretamente, para cada municipio se tienen 13 variables, que consisten en la extensión, en hectáreas, de suelo del municipio cubierto de: barbecho y otras tierras, cultivos herbáceos, cultivos leñosos, prados naturales, pastizales, monte maderable, monte abierto, monte leñoso, erial a pastos, espartizales, terreno improductivo, superficie no agrícola y ríos y lagos.

Lo que pretendemos es analizar el porcentaje de suelo de los municipios de la provincia de Jaén cubierto de monte. Fijaros bien que decimos *porcentaje*, ya que debemos de tener en cuenta que Andújar, uno de los municipios más extensos de la provincia, puede tener una mayor extensión de montes que, por ejemplo, Campillo de Arenas, pero habrá que considerar, no la extensión en términos absolutos de los montes, sino la proporción de éstos sobre la extensión total.

Es decir, lo que nos interesa analizar no está explícitamente en los datos de la matriz original de datos, pero podemos calcularlo a partir de ella. Concretamente, lo que nos interesa analizar es la variable

$$prop.montes = \frac{Monte.maderable + Monte.abierto + Monte.lenoso}{Extension.total} \times 100,$$

expresada en porcentaje, donde *Extension.total* se obtiene, a su vez, como suma de todas las variables presentes en la matriz de datos.

2.5.1. Mediante código

Hemos importado la hoja de datos desde el archivo de Excel, llamándola *Datos*. Para ver el nombre de las variables, escribimos en la consola *names(Datos)*. El resultado es el siguiente:

```
> names(Datos)
[1] "Municipio" "Barbecho.y.otras.tierras" "Cultivos.herbáceos"
[4] "Cultivos.leñosos" "Prados.naturales" "Pastizales"
[7] "Monte.maderable" "Monte.abierto" "Monte.leñoso"
```

```
[10] "Erial.a.pastos" "Espartizales" "Terreno.improductivo"  
[13] "Superficie.no.agrícola" "Ríos.y.lagos"
```

Como podemos ver, los nombres de las variables que se han importado desde el archivo de Excel incluyen caracteres como espacios y tildes, por lo que habrá que escribirlos entre comillas, tal y como aparecen en la consola de R.

Lo que queremos hacer en primer lugar es calcular la extensión total de cada municipio sumando todas las variables referidas a la extensión según los distintos tipos de uso del suelo. Recordemos que para referirnos a una variable de una hoja de datos debemos escribir primero el nombre de la hoja de datos, seguido del símbolo \$ y del nombre de la variable. Vamos a llamar a la variable *Extension.total*. Se calcula de la siguiente forma:

```
Datos$Extension.total<-Datos$"Barbecho y otras tierras"  
+Datos$"Cultivos herbáceos"+Datos$"Cultivos leñosos"  
+Datos$"Prados naturales"+Datos$"Pastizales"  
+Datos$"Monte maderable"+Datos$"Monte abierto"  
+Datos$"Monte leñoso"+Datos$"Erial a pastos"+Datos$"Espartizales"  
+Datos$"Terreno improductivo"+Datos$"Superficie no agrícola"  
+Datos$"Ríos y lagos"
```

Lo que R hace es sumar elemento a elemento los datos correspondientes a cada municipio de la provincia de Jaén.

Una vez calculada la extensión total, vamos a calcular el porcentaje de terreno de cada municipio cubierta de monte, de la siguiente forma:

```
Datos$prop.monte<-100*(Datos$"Monte maderable"  
+Datos$"Monte abierto"  
+Datos$"Monte leñoso")  
/Datos$Extension.total
```

En general, podemos realizar las operaciones aritméticas habituales (suma, diferencia, multiplicación, división, potencias, etc.), con la notación matemática propia de estas operaciones, sin olvidar que tales operaciones se realizan elemento a elemento sobre todos y cada uno de los elementos que constituyen cada variable de la hoja de datos.

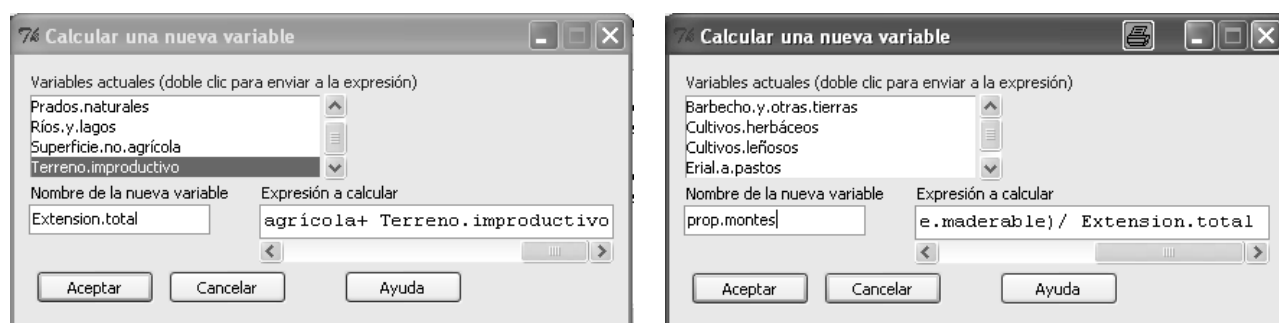


Figura 2.7: Cálculo de la extensión total del municipio (izquierda) y del porcentaje cubierto de monte (derecha)

2.5.2. Mediante R Commander

Una vez importados los datos, utilizamos la opción del menú *Datos* → *Modificar variables del conjunto de datos activo* → *Calcular una nueva variable*. Esto nos abrirá una ventana que funciona básicamente como una calculadora. Lo que vamos a hacer es calcular en un primer paso la extensión total, *Extension.total*, y en un segundo paso la variable *prop.montes*. La Figura 2.7 muestra ambas entradas. En el cálculo de *prop.montes* no se visualiza todo el contenido de la ventana, pero en el interior aparece lo siguiente: `100*(Monte.abierto+ Monte.leñoso+ Monte.maderable)/ Extension.total`.

2.6. Filtrado de datos

En ocasiones es necesario analizar, no todo el conjunto de datos, sino sólo un subconjunto de éste. En ese caso, lo que se hace es filtrar los datos mediante alguna condición dada por uno o varios valores de alguna variable.

Por ejemplo, supongamos que en el archivo de datos contenido en *JaenCordoba.xls* deseamos analizar sólo los datos de los municipios de Jaén.

2.6.1. Mediante código

No olvidemos que la forma de referirnos a los elementos de una variable que conocemos hasta ahora es poniendo la posición que ocupan entre corchetes. Ahora vamos a hacer algo parecido, pero escribiendo entre los corchetes la condición que determina el filtro. Por ejemplo, en el caso que nos ocupa, deseamos quedarnos sólo con los datos relativos a la provincia de Jaén, es decir, deseamos sólo las filas de la provincia de Jaén, y todas las columnas (variables) de la hoja de datos. En ese caso, utilizando la variable *Provincia* que ya construimos, podemos hacerlo de la siguiente forma:

```
Datos.Jaen<-Datos[Datos$Provincia="Jaén",]
```

o también, utilizando el código INE directamente,

```
Datos.Jaen<-Datos[(Datos$ine>23000)&(Datos$ine<24000),].
```

No perdamos de vista que al no poner nada tras la coma que hay dentro del corchete estamos pidiendo a R que mantenga todas las variables.

En resumen, el filtrado se logra identificando las filas que deseamos conservar mediante una expresión lógica.

2.6.2. Mediante R Commander

Lo haremos de la siguiente forma:

1. Seleccionando en el menú *Datos* → *Conjunto de datos activo* → *Filtrar el conjunto de datos activo*.
2. En la ventana emergente (Figura 2.8) podemos seleccionar si deseamos quedarnos con todas las variables o elegir sólo algunas.
3. La casilla más importante es la de *Expresión de selección*: ahí debemos escribir la expresión lógica que determine nuestro filtro. Por ejemplo, en nuestro caso podría ser *ine>23000&ine<24000*, es decir, código INE mayor que 23000 y menor que 24000. Si tuviéramos la variable *Provincia* que recodificamos anteriormente en este conjunto de datos, la expresión sería *Provincia="Jaén"*.
4. Finalmente, es recomendable ponerle un nombre al nuevo conjunto de datos filtrado distinto del original, para evitar que lo sobrescriba. En nuestro caso lo he llamado *Datos.Jaen*.

2.7. Almacenamiento de instrucciones y resultados

Para terminar este capítulo vamos a describir cómo utilizar las opciones de almacenamiento de instrucciones y resultados. El objetivo de este procedimiento es poder acceder a las instrucciones y/o los resultados que hemos implementado en una sesión cualquiera con posterioridad a ella. Aparte de esta función principal, hay que reconocer que R Commander tiene problemas de estabilidad cuando se ejecuta en conjunción con otros programas, de manera que puede ser una buena idea el guardar con cierta frecuencia las instrucciones para evitar que perdamos todo el trabajo ante un cierre inesperado del programa.

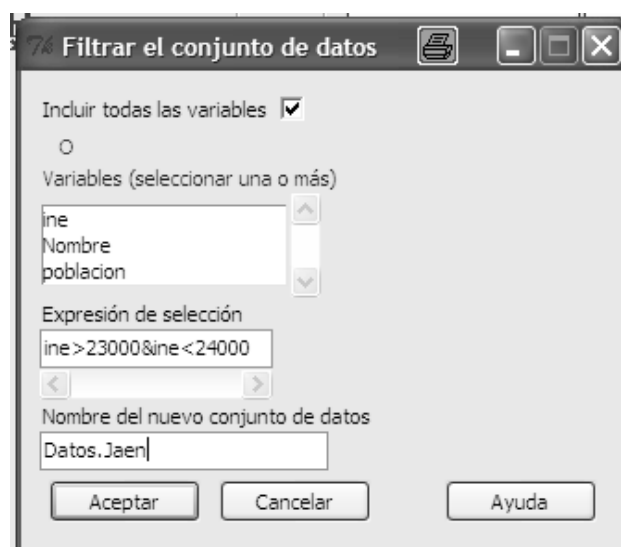


Figura 2.8: Filtrado de casos

2.7.1. Mediante código. El *script* y la *sesión de trabajo*

Hasta ahora hemos escrito todo el código directamente en la consola. Esa no es, sin embargo, la forma más eficiente de trabajar, porque nos dificulta poder corregir errores, ver qué instrucciones ejecutamos con anterioridad, y, por ejemplo, nos obliga a empezar de nuevo si reiniciamos R.

Para evitar estos problemas podemos utilizar lo que R denomina un *script*⁷, que no es más que un fichero modificable desde el mismo R⁸, que contendrá todas las órdenes que hasta ahora hemos introducido en la consola, que podremos salvar y desde donde podemos manejar R de la misma forma que hasta ahora hemos hecho en la consola.

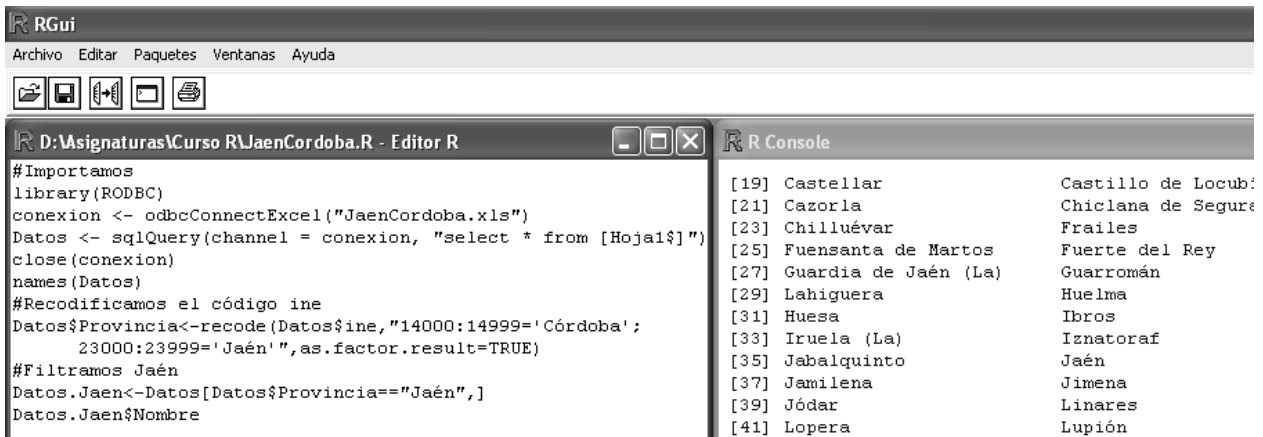
Para abrir un nuevo *script* nos vamos al menú de R y en *Archivo* elegimos *Nuevo script*. Se abrirá una nueva ventana aparte de la consola. Para organizar esta nueva ventana y la consola, suele ser cómodo elegir en el menú *Ventanas* de R la opción *Título*. Eso divide la pantalla principal de la sesión de R en dos partes, la izquierda para el *script* y la derecha para la consola.

Ahora podemos escribir nuestras líneas de código en el *script* y ejecutarlas pulsando *Control+R*. A modo de ejemplo, vamos a retomar todo lo que hemos hecho hasta ahora con el archivo *Jaen-Cordoba.xls* en un *script*:

1. Lo importamos.
2. Obtuvimos una nueva variable, *Provincia*, recodificando la variable *ine*.

⁷La mejor traducción que hemos encontrado de este término es *guión*, y no nos parece especialmente afortunada, así que nos vamos a permitir mantener en lo sucesivo el barbarismo *script*.

⁸También se puede manejar por otros editores que permiten una edición más avanzada y que interactúan con R con mayor comodidad. Entre ellos, recomendamos *Tinn-R*.

Figura 2.9: Ejemplo de *script*

3. Nos quedamos sólo con los datos de Jaén, mediante un filtro.

El *script* sería el que aparece en la Figura 2.9.

Observemos que aparecen algunas líneas con comentarios aclaratorios. Éstas vienen anteceditas del símbolo `#`, para que R las imprima en la consola, pero no las ejecute. Podemos ver cómo al ejecutar la última línea en la consola, a la derecha, aparecen nombres sólo de la provincia de Jaén. Finalmente, también se observa en la parte superior de la ventana del *script* que hemos guardado dicho *script* llamándolo *JaenCordoba.R*. Ésta, *.R*, es la extensión propia de los *script* de R. De esta forma, en el futuro podríamos seguir trabajando con este ejemplo sin necesidad de empezar todo de nuevo.

De la misma forma que podemos estar interesados en guardar el *script* con todas las instrucciones de un análisis, también podríamos estar interesados en guardar los objetos que se van generando en dichos análisis. Por ejemplo, los conjuntos de datos, pero también cualquier resultado que obtenemos y que pueda ser almacenado poniéndole un nombre. Todos esos objetos que se van generando constituyen lo que en R se conoce como la *sesión de trabajo*. Por ejemplo, tras ejecutar el *script* que acabamos de comentar, los únicos objetos que hay en la sesión de trabajo son *conexion*, *Datos* y *Datos.Jaen*.

Para guardar una sesión de trabajo (eso nos evitaría volver a tener que ejecutar el *script*) y almacenar estos objetos, simplemente seleccionamos el menú *Archivo* de R y la opción *Guardar área de trabajo*. Si nos damos cuenta, la extensión del fichero que genera es la de datos de R, *RData*.

Por último, si lo que deseamos es guardar los resultados que van apareciendo en la consola de R, tenemos que clicar sobre la consola, seleccionamos *Archivo* y la opción *Guardar en archivo*. Esto generará un archivo de texto con todas las salidas.

2.7.2. Mediante R Commander

Vamos a ejemplificarlo con el ejemplo que antes hemos trabajado sobre el archivo *JaenUsosSuelo.xls*. Debemos importarlo de nuevo y crear las variables *Extension.total* y *prop.montes*.

A continuación seleccionamos en el menú *Fichero* → *Guardar las instrucciones*. Nos pedirá el nombre y la ruta donde guardar el fichero de instrucciones, que tendrá extensión *.R*. Una buena idea para nombrar los ficheros de instrucciones es ponerles como nombre la fecha del día, por ejemplo, *21_07_08*. No es necesario escribir la extensión: lo hará el propio programa. Podemos y debemos seguir guardando las instrucciones con posterioridad, eligiendo de nuevo *Guardar las instrucciones*, pero ya no nos pedirá de nuevo un nombre, a no ser que elijamos *Guardar las instrucciones como*.

Ahora vamos a reiniciar R Commander y volvemos a cargar el fichero *JaenUsosSuelo.xls*. A continuación elegimos en el menú *Fichero* → *Abrir fichero de instrucciones* y seleccionamos el fichero de instrucciones que antes hemos guardado. Como podemos ver, aparecen las dos líneas con las que hemos creado las variables *Extension.total* y *prop.montes*. Podemos ejecutar estas líneas directamente desde la ventana de instrucciones sin tener que utilizar el menú.

De igual forma, podemos guardar y recuperar con posterioridad todo lo que aparece en la ventana de resultados, mediante la opción *Fichero* → *Guardar los resultados*. Los ficheros de resultados que R Commander crea son ficheros de texto, con extensión *.txt*.

2.8. Ejercicios propuestos

Si decidís utilizar el código de R para hacer los ejercicios, hacedlo usando un *script*. En el caso de que decidais utilizar R Commander, procurad guardar, al menos, el fichero de instrucciones.

El fichero *JaenIndicadores.xls* se refiere a los municipios de la provincia de Jaén en el año 2001, e incluye las siguientes variables:

- Código INE del municipio.
- Nombre del municipio.
- Consumo de energía eléctrica en megavatios por hora.
- Consumo medio de agua en invierno, en metros cúbicos por día.
- Consumo medio de agua en verano, en metros cúbicos por día.
- Destino de los residuos sólidos urbanos: las posibilidades son vertedero controlado, vertedero incontrolado, compostaje.

- Cantidad de residuos sólidos urbanos, en toneladas.
1. Importad el fichero⁹, llamad a la hoja de datos *Jaen* y guardad esta hoja de datos en un archivo de datos de R llamado *JaenIndicadores.RData* o *JaenIndicadores.rda*.
 2. Recodificad la variable *Poblacion* en una variable cualitativa tipo factor llamada *Tamaño* con tres categorías:
 - Si la población es inferior a 2000 habitantes, *Tamaño* será “Pequeño”.
 - Si la población está entre 2000 y 4500 habitantes, *Tamaño* será “Mediano”.
 - Si la población es superior a 4500 habitantes, *Tamaño* será “Grande”.
 3. Calculad los siguientes promedios a partir de las variables existentes:

- Consumo de energía eléctrica por habitante, *elec.hab*, obtenido como

$$\frac{\text{Consumo.de.energia.electrica}}{\text{Poblacion}}.$$

- Consumo medio de agua por habitante y día, *agua.hab*, obtenido como

$$\frac{\text{Consumo.de.agua..Invierno} + \text{Consumo.de.agua..Verano}}{\text{Poblacion}}.$$

- Residuos sólidos urbanos por habitante, *res.hab*, obtenido como

$$\frac{\text{Residuos.solidos.urbanos..Cantidad}}{\text{Poblacion}}.$$

4. Definid una nueva hoja de datos con todas las variables que contienen los datos originales, pero referida sólo a los municipios de tamaño mediano.

⁹Observad con atención que en el archivo *.xls* faltan algunos datos de Linares y Jaén, y en su lugar aparece “_”.

Capítulo 3

Estadística descriptiva

Objetivos:

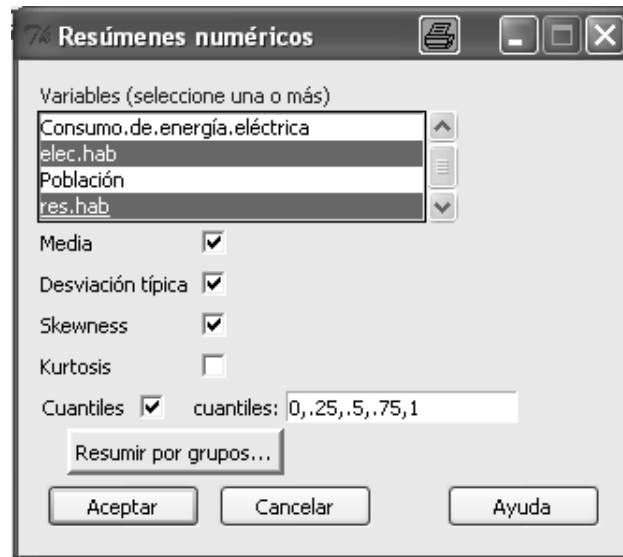
1. Obtener medidas de posición, dispersión y forma de un conjunto de datos.
2. Obtener representaciones gráficas que resuman desde el punto de vista estadístico un conjunto de datos.
3. Detectar valores fuera de rango en un conjunto de datos.

- Al arrancar R Commander de ahora en adelante debemos cargar también todos sus *plug-ins*, lo que amplía las opciones de los análisis. Para ello, en el menú, elegimos *Herramientas* → *Cargar plug-in(s) de Rcmdr* y en la ventana que se abre seleccionamos todos los *plug-ins*. Nos pedirá confirmación para reiniciar R Commander.
- A lo largo del capítulo vamos a utilizar el conjunto de datos *JaenIndicadores.rda* para todos los ejemplos y explicaciones.

3.1. Cálculo de medidas de posición, dispersión y forma

3.1.1. Mediante R Commander

Las medidas de posición, dispersión y forma más comunes, media, mediana, percentiles, desviación típica y coeficiente de asimetría, se hallan en la opción del menú *Estadísticos* → *Resúmenes* → *IPSUR - Numerical summaries*. A modo de ejemplo, vamos a obtener estas medidas para los promedios *elec.hab*, *agua.hab* y *res.hab*. En la Figura 3.1 aparecen las entradas de esa opción del menú. En ella es posible elegir varias variables a la vez, pulsando la tecla *Control* mientras se clican en las variables deseadas.

Figura 3.1: Opción *Resúmenes numéricos* del menú

En el Cuadro 3.1 se muestra la tabla de resultados que aparece. En ésta, *mean* se refiere a la media, *sd* a la raíz de la cuasi-varianza, *skewness* es el coeficiente de asimetría, el percentil 0 es el valor mínimo de la variable, el percentil 50, como ya sabemos, es la mediana y el percentil 100 es el valor máximo de la variable.

	mean	sd	skewness	0 %	25 %	50 %	75 %	100 %
agua.hab	0.53	0.14	1.22	0.14	0.46	0.51	0.56	1.09
elec.hab	2.66	1.40	2.05	0.94	1.75	2.20	3.10	9.19
res.hab	0.23	0.04	1.39	0.18	0.21	0.23	0.24	0.35

Cuadro 3.1: Descriptivos básicos de *elec.hab*, *agua.hab* y *res.hab*

3.1.2. Mediante código

Las funciones *mean()*, *sd()* y *quantile()* proporcionan la media, la desviación típica y los cuantiles de cualquier muestra. Todas estas órdenes responden al mismo tipo de formato. Por ejemplo, si queremos calcular la media de las anteriores variables escribimos

```
mean(Datos[,c("agua.hab", "elec.hab", "res.hab")], na.rm = TRUE)
```

El argumento *na.rm = TRUE* indica que los valores faltantes *NA* se eliminan para realizar los cálculos. Si no se incluye esta opción entonces la media resultante sería *NA*.

De igual forma, la desviación típica se lograría mediante

```
sd(Datos[,c("agua.hab", "elec.hab", "res.hab")], na.rm = TRUE)
```

Finalmente, para obtener los cuantiles necesitamos especificar las variables y las probabilidades de los cuantiles que deseamos mediante el argumento *probs*. Por ejemplo, para obtener los percentiles 5 y 95,

```
quantile(Datos$agua.hab, na.rm = TRUE, probs=c(0.05,0.95))
quantile(Datos$elec.hab, na.rm = TRUE, probs=c(0.05,0.95))
quantile(Datos$res.hab, na.rm = TRUE, probs=c(0.05,0.95))
```

El coeficiente de asimetría sólo viene definido como función en algunos paquetes, pero definirlo mediante código es sencillo. Teniendo en cuenta que se define como

$$\frac{\sum_{i=1}^n (x_i - \bar{x})^3 / n}{s_{n-1}^3},$$

lo obtendríamos mediante el siguiente código:

```
attach(Datos)
sum((Datos$agua.hab-mean(Datos$agua.hab))^3/length(Datos$agua.hab))
/(sd(Datos$agua.hab))^3
detach(Datos)
```

3.1.3. Resúmenes por grupos

¿Y si deseamos hacer el mismo análisis pero en cada uno de los grupos que determina la variable *Tipo*?

Veámoslo en primer lugar mediante R Commander. Dado que esta variable es de tipo factor, este análisis por grupos es bastante sencillo. Utilizamos la misma opción *Estadísticos* → *Resúmenes* → *IP SUR - Numerical summaries*, pero ahora clicamos en la pestaña *Resumir por grupos*. Esta opción abre una ventana (Figura 3.2) donde aparecen como posibilidades todas aquellas variables que pueden dividir al conjunto de datos en grupo. En nuestro caso elegimos la variable *Tipo*. La ventana de resultados mostrará los mismos estadísticos, pero separando cada uno de los tres grupos, para cada variable. Por ejemplo, para la variable *agua.hab* tenemos

	mean	sd	skewness	0 %	25 %	50 %	75 %	100 %
Grande	0.51	0.12	1.45	0.18	0.47	0.51	0.54	1.01
Mediano	0.51	0.07	0.23	0.41	0.45	0.51	0.57	0.65
Pequeño	0.55	0.19	0.70	0.14	0.46	0.49	0.66	1.09

Cuadro 3.2: Descriptivos básicos de *agua.hab* en función de *Tipo*

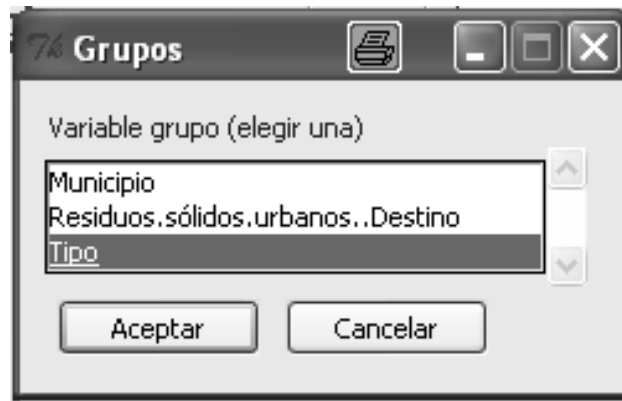


Figura 3.2: Resúmenes por grupos

Mediante código, los anteriores resultados se pueden calcular a través de la orden `tapply()`. Por ejemplo, la media, la desviación típica y los percentiles 5 y 95 de la variable `agua.hab` en función de los grupos de la variable `Tipo` se obtendría de la forma siguiente:

```
tapply(Datos$agua.hab,Datos$Tipo,mean, na.rm=TRUE)
tapply(Datos$agua.hab,Datos$Tipo,sd, na.rm=TRUE)
tapply(Datos$agua.hab,Datos$Tipo,quantile,probs=c(0.05,0.95),na.rm=TRUE)
```

Obsérvese que en la última línea hemos tenido que especificar las probabilidades requeridas a la función `quantile`.

3.2. Distribuciones de frecuencias

Las variables `Tipo` y `Residuos.sólidos.urbanos..Destino` son de tipo cualitativo, por lo que no pueden ser resumidas mediante medidas numéricas. Para este tipo de variables el resumen más conveniente es, simplemente, su distribución de frecuencias.

3.2.1. Mediante R Commander

Para obtener la distribución de frecuencias de una o varias variables de un conjunto de datos mediante R Commander elegimos la opción *Estadísticos* → *Resúmenes* → *Distribución de frecuencias*. En la ventana emergente elegimos las variables que queremos analizar y la tabla aparece en la ventana de resultados, incluyendo las frecuencias absolutas y relativas.

3.2.2. Mediante código

Las funciones de código correspondientes son `table()` y `prop.table()`. Así, para la obtención de las distribuciones de frecuencias (absolutas y relativas) de la variable `Tipo` escribimos



Figura 3.3: Entradas para la construcción de un diagrama de barras

```
Tabla <- table(Datos$Tipo)
Tabla # frecuencias relativas
prop.table(Tabla)# frecuencias absolutas
```

3.3. Diagrama de barras y diagrama de sectores

No obstante, asumiendo el dicho *una imagen vale más que mil palabras*, sabemos que existen dos formas de plasmar en un gráfico la distribución de frecuencias de una variable cualitativa o discreta con pocos valores: el diagrama de barras y el diagrama de sectores.

3.3.1. Diagrama de barras para variables cualitativas

En R Commander este tipo de gráficos están en la opción *Gráficas* → *IPSUR - Bar Graph...*. La ventana de entradas aparece en la Figura 3.3. En esta ventana hemos solicitado un análisis por grupos, según la clasificación de la variable *Tipo*. Si no deseamos esta separación sino un análisis global de la variable, no tenemos que tocar la pestaña *Resumir por grupos*. Es muy importante tener en cuenta que sólo pueden representarse variables cualitativas de tipo factor.

En la Figura 3.4 aparece el diagrama resultante. También se ha capturado la opción que cualquier gráfico generado por R incluye para guardar la imagen en los formatos gráficos más habituales.

La función `barplot()` nos permite obtener la distribución de barras mediante código. La sintaxis para obtener la Figura 3.4 es la siguiente:

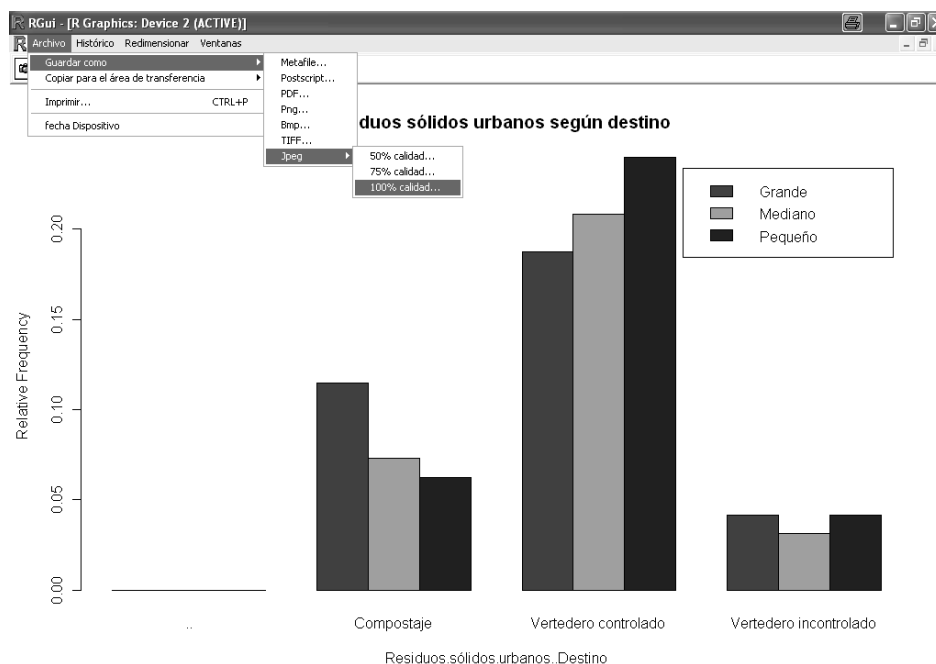


Figura 3.4: Diagrama de barras según el tipo de municipio del destino de los residuos sólidos

```
barplot(prop.table(table(Datos$Tipo),
  main="Residuos sólidos urbanos según su destino",
  xlab="Residuos sólidos urbanos según su destino",
  ylab="Frecuencia relativa"
```

En esta expresión tan sólo hemos incluido la tabla de frecuencias para la que queremos el diagrama y algunos otros detalles:

- *main* se refiere al título del gráfico.
- *xlab* especifica el título del eje X.
- *ylab* especifica el título del eje Y.

Existen otras muchísimas posibilidades en la construcción del gráfico: aquí sólo mostramos aspectos de su sintaxis más básica.

3.3.2. Diagrama de sectores para variables cualitativas

Para realizar un diagrama de sectores mediante R Commander elegiremos la opción *Gráficas* → *Diagrama de sectores*. La ventana emergente sólo permite elegir una variable cualitativa. De nuevo es muy importante tener en cuenta que sólo pueden representarse variables cualitativas de

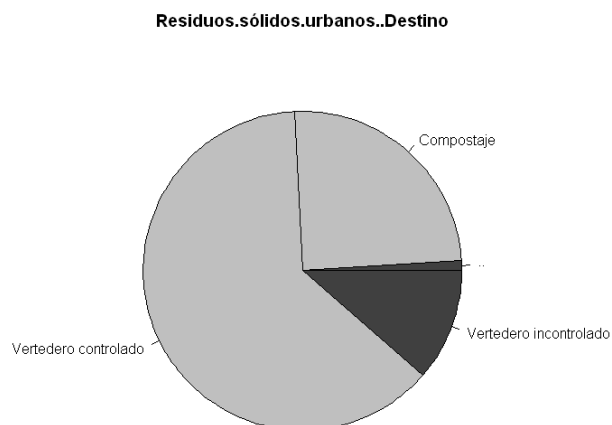


Figura 3.5: Diagrama de sectores del destino de los residuos sólidos urbanos en los municipios de la provincia de Jaén

tipo factor. El diagrama correspondiente al destino de los residuos sólidos de los municipios aparece en la Figura 3.5.

La opción mediante código en su versión más básica es

```
pie(table(Datos$Residuos.sólidos.urbanos..Destino))
```

3.4. Histograma para variables continuas y discretas

3.4.1. Histograma para variables continuas

Como ya sabemos, los diagramas de barras o sectores no son adecuados para datos de variable continuas. Frente a estas representaciones, el histograma aparece como la alternativa válida, ya que obliga a agrupar los valores en intervalos cuya frecuencia sí es relevante.

Para realizar un histograma con R Commander elegimos *Gráficas* → *Histograma*. La ventana de entrada permite elegir sólo una variable para cada análisis, el número de intervalos del histograma y la escala de éste (frecuencias absolutas, porcentajes y densidades).

En el caso de las variables *agua.hab*, *elec.hab* y *res.hab* hemos seleccionado histogramas con escala en porcentajes y 10 intervalos. Los resultados aparecen en la Figura 3.6.

La función *hist()* nos permite representar el histograma mediante código. La sintaxis básica de esta función es la siguiente:

```
hist(x, breaks = "Sturges",  
     freq = NULL, main = paste("Histogram of" , xname),  
     xlab = xname, ylab, TRUE, labels = FALSE)
```

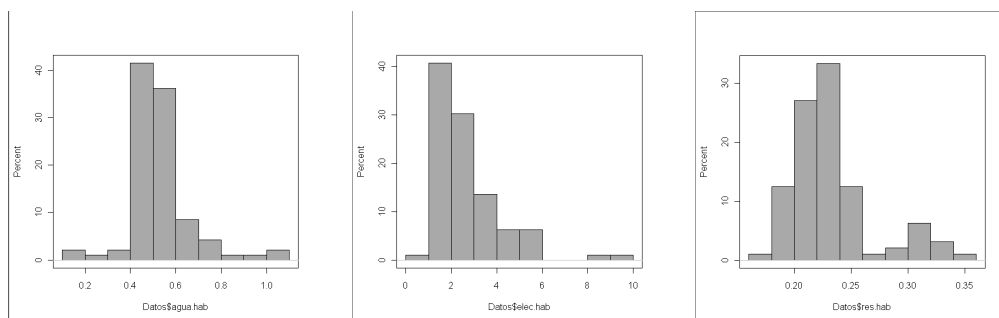


Figura 3.6: De izquierda a derecha, histogramas de *agua.hab*, *elec.hab* y *res.hab*

- *x* es el vector de datos.
- *breaks* puede especificar el número de intervalos que deseamos o los extremos de los intervalos que deseamos considerar, mediante un vector. Por defecto, asigna el número de intervalos por el conocido como método de Sturges.
- *freq* especifica si la escala del histograma es tal que el área de las barras es igual a la proporción de datos en cada intervalo (escala de densidad, con valor `freq = FALSE`) o su altura es simplemente el recuento de las frecuencias (escala de frecuencias, con valor `freq = TRUE`).
- *main* especifica el título del gráfico, mientras que *xlab* e *ylab* especifican el de los ejes.
- *labels* añade una etiqueta a cada barra con el valor de las frecuencias.

Por ejemplo, para calcular el histograma de frecuencias de la variable *agua.hab* debemos escribir

```
hist(Datos$agua.hab, breaks = 10,
     freq = NULL, main = "Histogram de agua.hab",
     labels = TRUE)
```

3.4.2. Histograma para variables discretas

Parece contradictorio hablar de un histograma para variables discretas, ya que una variable discreta debe representarse con un diagrama de barras. Sin embargo, para los ordenadores y los programas estadísticos, la división entre variables discretas y continuas suele no ser habitual. Se entiende que ambas son variables numéricas, al contrario de las variables cualitativas, que son caracteres, y para variables numéricas se ofrece el histograma como representación gráfica.

Lo que podemos hacer es utilizar la función que realiza el histograma para realizar un diagrama de barras de una variable discreta. La idea es muy simple: le pediremos a R a través de Commander

o de la consola el histograma de la variable discreta de manera que las barras del histograma representen las frecuencias de los valores de la variable discreta.

Vamos a verlo con los datos que están contenidos en el archivo *DatosMuertesCoces.rda*. Estos datos, de los que hablaremos más adelante se refieren al número de muertos por coces de caballos en el ejército prusiano en una serie de años y secciones a finales del siglo XIX. Es evidente que es una variable discreta. Para realizar un diagrama de barras de estos datos, haremos lo siguiente:

1. Le pediremos a R Commander un histograma de la variable muertes, eligiendo como escala el recuento de frecuencias. No hace falta retocar el número de intervalos porque lo vamos a hacer a mano a continuación.
2. En la ventana de instrucciones R Commander ha escrito lo siguiente:

```
Hist(muertes$muertes,scale="frequency",breaks="Sturges",col="darkgray")
```

Además, veremos como resultado el histograma que aparece en la Figura 3.7 a la izquierda.

3. Cerraremos ese histograma y retocaremos el código de la siguiente forma. Como deseamos que las barras del histograma cuenten las frecuencias de los valores de la variable, nos fijaremos en que éstos van de 0 a 4, y en la opción breaks escribiremos -0.5:4.5: de esa forma, los intervalos del histograma contarán las frecuencias en los valores 0, 1, 2, 3 y 4. En resumen, el nuevo código será

```
Hist(muertes$muertes,scale="frequency",breaks=-0.5:4.5,col="darkgray")
```

El resultado aparece en la Figura 3.7 a la derecha.

Mediante código es igual de sencillo:

```
hist(muertes$muertes,breaks=-0.5:4.5,freq=TRUE,labels=TRUE)
```

3.5. Detección de valores atípicos. Diagrama de caja

Un método común para la detección de valores atípicos es el llamado diagrama de caja o *boxplot*. Este método es válido para cualquier conjunto de datos, independientemente de la forma de su distribución de frecuencias.

3.5.1. Mediante R Commander

Vamos a obtener el diagrama de caja de las variables *elec.hab* y *res.hab* e identificar los municipios atípicos en cuanto al consumo de energía eléctrica y de los residuos generados por habitante. Para

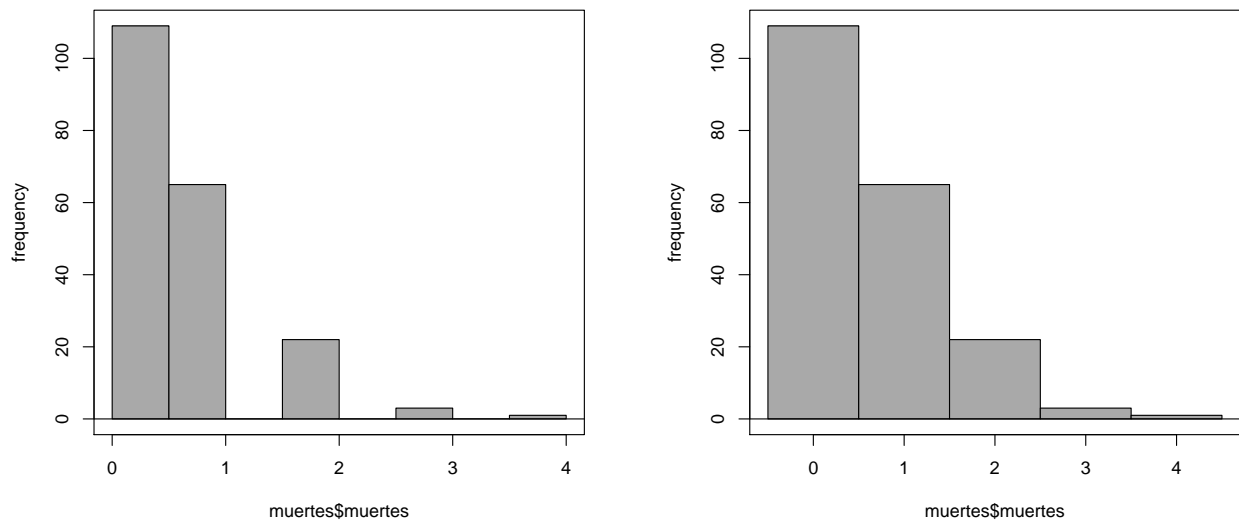


Figura 3.7: Histogramas de una variable discreta

ello, elegimos la opción *Gráficas* → *Diagrama de caja* o la opción *Gráficas* → *IPSUR - Boxplot*¹. Las dos ventanas de entradas son muy parecidas: en ellas tenemos que elegir la variable que queremos analizar y existen dos opciones muy interesantes:

1. Podemos elegir un análisis por grupos, sin más que clicar en la pestaña *Resúmenes por grupos*. En esta ocasión no es necesario, pero podríamos hacerlo con los grupos generados por la variable *Tipo*.
2. Podemos elegir la opción *Identificar atípicos con el ratón*, que es la forma más fácil de señalar los municipios atípicos². Recordemos que en un diagrama de caja los municipios atípicos se señalan con círculos, y los atípicos extremos con asteriscos.

En la Figura 3.8, a la izquierda aparece el diagrama de caja de la variable *elec.hab*. En ella podemos ver gráficamente la asimetría a la derecha de la distribución, ya que el lado a la derecha de la mediana, que separa la caja, es más grande. Por su parte, también podemos ver que hemos detectado como municipios atípicos a la derecha de la distribución, es decir, que destacan por su fuerte consumo por habitante, a los municipios 35, 53, 56, 57 y 88 (estos dos últimos de forma muy clara), que corresponden con Guarromán, Lupión, Martos, Mengíbar y Villanueva de la Reina. Una forma eficiente de ver sus nombres es introducir el siguiente código en la ventana de instrucciones: `Datos$Municipio[c(35,53,56,57,88)]`.

¹Ambas ventanas suelen *colgar* R Commander.

²Sólo hay que tener un poquito de buen pulso cuando los valores están muy próximos entre sí.

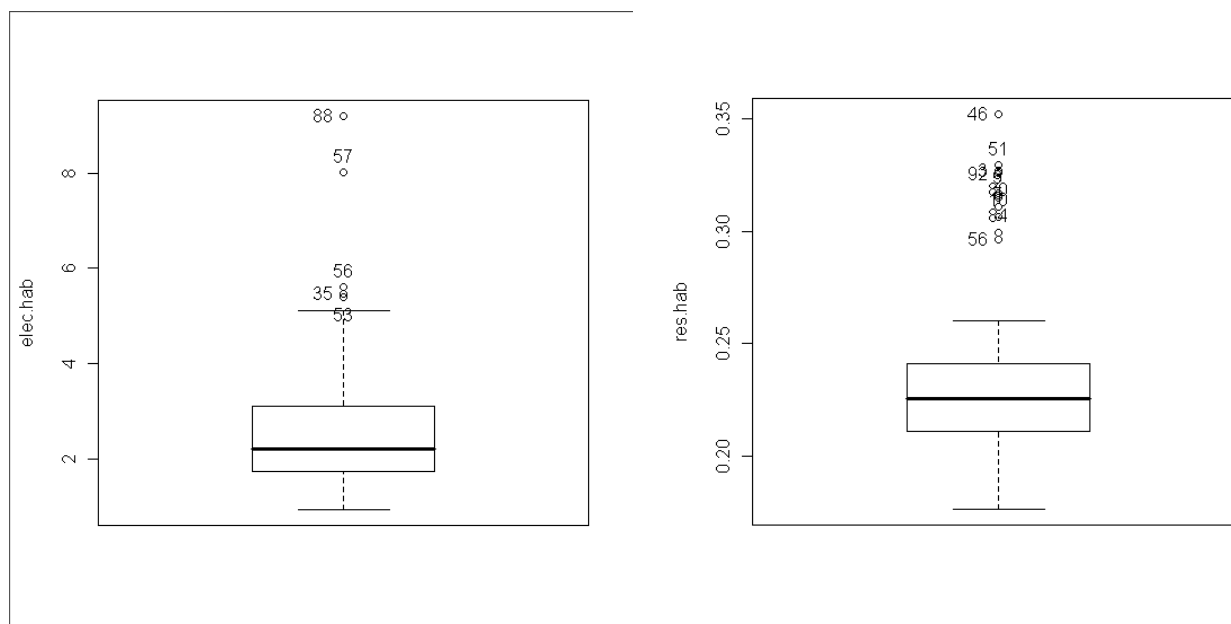


Figura 3.8: Diagramas de caja de las variables *elec.hab* (izquierda) y *res.hab* (derecha)

3.5.2. Mediante código. La función *boxplot*

La sintaxis básica de la función *boxplot* obliga simplemente a especificar el conjunto de datos. Por ejemplo,

```
boxplot(Datos$elec.hab)
```

También es posible añadir un título al gráfico y a los ejes, como en el caso de *hist*.

Por su parte, si no cerramos el gráfico, también podemos especificar los atípicos mediante la función *identify*. Vamos a verlo en el ejemplo:

```
identify(rep(1, length(Datos$elec.hab)),
        Datos$elec.hab, rownames(Datos))
```

El primer argumento, `rep(1, length(Datos$elec.hab))`, está especificando que la coordenada *x* de todos los puntos es 1 (lugar donde se sitúan todos los valores). `Datos$elec.hab` especifica la coordenada *y* de los puntos (ya que el diagrama es vertical), mientras que `rownames(Datos)` especifica cómo identificar los puntos, en este caso con el número de fila. Podríamos cambiar esta opción por `Datos$Municipio` y al clicar nos daría directamente el nombre del municipio (aunque si señalamos varios, probablemente se solaparán).

3.6. Ejercicios propuestos

Los datos que se encuentran en el fichero *EjercicioDescriptiva.xls* han sido obtenidos desde la página web del Instituto de Estadística de Andalucía. Se refieren a todos los municipios de Andalucía, en relación a las siguientes variables:

- Código INE.
 - Municipio.
 - Tasa de actividad en 2001.
 - Nº de líneas ADSL en funcionamiento en 2007.
 - Edad media del municipio en 2007.
 - Renta familiar disponible por habitante. Aparece agrupado en varias categorías de renta. Hay numerosos datos faltantes que aparecen señalados como “..”.
 - Crecimiento vegetativo en 2006. El crecimiento vegetativo es la diferencia entre el número de nacidos y el número de fallecidos.
 - Número de parados en 2007.
 - Población total del municipio en 2001, 2003, 2006 y 2007.
1. A partir de la variable código INE, construir una variable tipo factor que distinga la provincia de pertenencia de cada municipio. Obtener la distribución de frecuencias, un diagrama de barras y un diagrama de sectores de esta variable tipo factor. ¿Qué provincia tiene más municipios? ¿Cuál tiene menos?
 2. Obtener un resumen descriptivo de la variable tasa de actividad que incluya medidas numéricas, histograma y diagrama de caja. En función de este resumen, contestar a las siguientes preguntas:
 - a) ¿Cuál es la tasa media de actividad de los municipios andaluces?
 - b) ¿Cómo valoras la homogeneidad de los valores de la tasa de actividad en los municipios andaluces?
 - c) ¿En ese sentido, qué municipios andaluces destacan significativamente del resto (como atípicos) por su alta tasa de actividad y por su baja tasa de actividad? ¿Se te ocurre alguna explicación al respecto?

- d) ¿Cómo valoras la simetría de la distribución de frecuencias?
3. La variable nº de líneas ADSL por municipio está claramente afectada por la población del mismo. Para analizar la presencia de líneas ADSL en términos relativos a nivel municipal, construir una tasa de líneas por cada 100 habitantes³ como

$$tasa.lineas.ADSL = \frac{N \text{ de lineas ADSL}}{Poblacion.2007} \times 100.$$

Responder a las siguientes preguntas:

- a) ¿Cuál es la tasa promedio de líneas ADSL por cada 100 habitantes en los municipios andaluces?
- b) ¿Cómo valoras la homogeneidad de la presencia de líneas ADSL en los municipios andaluces? ¿Qué explicación encuentras al respecto?
- c) ¿Cómo valoras la simetría de la distribución de frecuencias? Interpreta los resultados de una forma aplicada, en relación con la variable que estamos estudiando.
- d) ¿Qué municipios destacan de una forma especialmente significativa (como atípicos) por su alto número de líneas ADSL por habitante? ¿Se te ocurre alguna explicación al respecto?
4. Obtener un resumen descriptivo de la edad media de la población por provincias que incluya medidas numéricas, histograma y diagrama de caja. En función de este resumen, contestar a las siguientes preguntas:
- a) ¿Cuál es el promedio de la edad media de la población, en cada provincia, de los municipios andaluces?
- b) ¿Cómo valoras, en cada provincia, la homogeneidad de los valores de la edad media de la población en los municipios andaluces?
- c) ¿En ese sentido, qué municipios andaluces destacan como atípicos sobre el resto, en cada provincia, por su alta o por su baja edad media de la población? ¿Se te ocurre alguna explicación al respecto?
- d) ¿Cómo valoras la simetría de la distribución de frecuencias en cada provincia?
5. Proporcionar la distribución de frecuencias de la renta familiar disponible por habitante y su representación mediante un diagrama de barras y un diagrama de sectores.

³Lo ideal habría sido hacer una tasa en relación al nº de hogares, no de habitantes, pero ese dato es inexistente desde 2001, fecha del último censo.

6. Transformar la variable crecimiento vegetativo en 2006 de manera que tenga en cuenta la población del municipio, construyendo una tasa de crecimiento vegetativo de la siguiente forma:

$$tasa.crecimiento.vegetativo.2006 = \frac{crecimiento.vegetativo.2006}{poblacion.2006} \times 100.$$

Obtener un resumen numérico y mediante representaciones gráficas de la distribución de frecuencias de esta variable y responder a las siguientes preguntas:

- a) ¿Crees que el crecimiento vegetativo en los municipios andaluces es un indicador muy homogéneo? Justifica la respuesta y trata de explicar porqué.
 - b) En términos generales, ¿crees que la población en Andalucía crece o decrece? ¿Y por provincias?
 - c) ¿Qué municipios andaluces destacan como atípicos sobre el resto por su alto o su bajo valor de la tasa de crecimiento vegetativo? ¿Se te ocurre alguna explicación al respecto?
7. En relación al nº de parados en 2007, construir una variable de tasa de paro con respecto al total de la población, dada por

$$tasa.paro.2007 = \frac{Numero.parados.2007}{Poblacion.2007} \times 100.$$

Utilizando esta variable, responder a las siguientes cuestiones:

- a) ¿Cuál es la tasa media de paro de los municipios andaluces? ¿En qué medida es esta tasa media representativa del conjunto de los municipios?
- b) Valora la simetría de la distribución de frecuencias e interpreta los resultados.
- c) ¿Qué municipios destacan como atípicos sobre el resto por su alta o por su baja tasa de paro? Trata de encontrar una explicación al respecto.

Capítulo 4

Manejo de distribuciones de probabilidad

Objetivos:

1. Calcular probabilidades asociadas a las distribuciones de probabilidad más habituales.
2. Calcular percentiles de las distribuciones de probabilidad más habituales.
3. Obtener representaciones gráficas de las distribuciones de probabilidad más habituales.
4. Simular muestras procedentes de las distribuciones de probabilidad más habituales.

4.1. Cálculo de probabilidades

R y cualquier otro programa de software estadístico proporcionan una excelente manera de obtener probabilidades asociadas a las distribuciones de probabilidad más comunes. Debemos recordar que, incluso los modelos más sencillos, como el binomial o el de Poisson, presentan dificultades en cuanto a lo tedioso que resulta realizar los cálculos. En otras distribuciones, como la Gamma o la normal, el problema es que es imposible *hacer las cuentas a mano*, siendo absolutamente necesario la utilización de algún software matemático para obtener aproximaciones precisas de las probabilidades.

4.1.1. Distribuciones discretas

En el caso de las distribuciones discretas, tenemos, básicamente, dos tipos de probabilidades:

1. Las probabilidades que proporciona la función masa, que podríamos llamar *probabilidades simples*, del tipo $P[X = x]$.
2. *Probabilidades acumuladas* (dadas en términos de la función de distribución), del tipo $P[X \leq x]$.

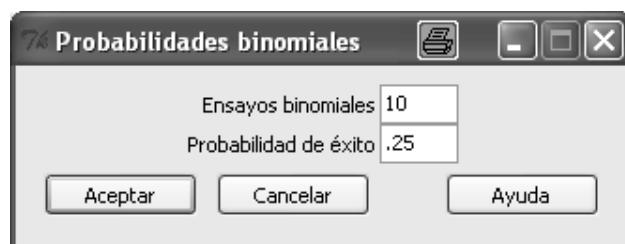


Figura 4.1: Ventana de entradas para probabilidades de la distribución binomial

Es evidente que las probabilidades acumuladas se pueden calcular a partir de las probabilidades simples, sin más que tener en cuenta que $P[X \leq x] = \sum_{x_i \leq x} P[X = x_i]$.

Vamos a ir viendo a continuación cómo obtener estas probabilidades para 4 de las distribuciones discretas más comunes. En esta ocasión hemos preferido ir intercalando la forma de operar mediante código y R Commander.

4.1.1.1. Distribución binomial

Si queremos probabilidades simples mediante R Commander debemos seleccionar *Distribuciones* → *Distribuciones discretas* → *Distribución binomial* → *Probabilidades binomiales*. La ventana de entrada (Figura 4.1) requiere que introduzcamos los parámetros n , ensayos binomiales, y p , probabilidad de éxito. Lo que aparece en la ventana de resultados es toda la distribución de probabilidad, es decir, la función masa para todos los valores posibles de la variable. En el caso de la Figura 4.1 aparece como resultado el Cuadro 4.1.

x	Pr
0	0.06
1	0.19
2	0.28
3	0.25
4	0.15
5	0.06
6	0.02
7	0.00
8	0.00
9	0.00
10	0.00

Cuadro 4.1: Función masa de la B(10,0.25)

Si lo que queremos es una probabilidad de tipo acumulado, la ventana de entrada (Figura 4.2) permite adicionalmente que especifiquemos para qué valores queremos dichas probabilidades acumuladas. Los hemos puesto todos, en cuyo caso aparece el siguiente resultado:

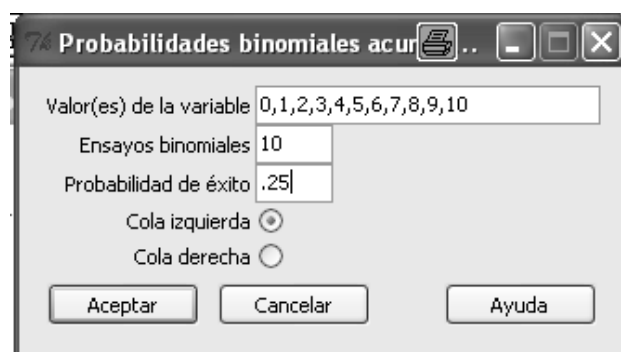


Figura 4.2: Ventana de entradas para probabilidades acumuladas de la distribución binomial

```
0.05631351 0.24402523 0.52559280 0.77587509 0.92187309 0.98027229 0.99649429
0.99958420 0.99997044 0.99999905 1.00000000
```

Vamos a fijarnos de nuevo en el Cuadro 4.1 para entender este resultado. Por ejemplo, el 4º valor, 0.77587509, corresponde a $P[X \leq 3] = P[X = 0, 1, 2, 3]$ que, teniendo en cuenta el valor de la función masa es

$$0.06 + 0.19 + 0.28 + 0.25 = 0.78,$$

que es el valor redondeado de 0.77587509. Es decir, las probabilidades acumuladas se pueden obtener rápidamente a partir de las simples.

La ventana de instrucciones de R Commander ya nos facilita el código necesario para realizar los cálculos directamente desde la consola de R:

- Si queremos probabilidades simples, podemos utilizar la función `dbinom`, que requiere tres datos: de qué valores queremos las probabilidades, el parámetro n y el parámetro p . Por ejemplo, las probabilidades de 0, 1, 2 y 3 para los mismos parámetros de antes las obtenemos escribiendo lo siguiente en la ventana de instrucciones y ejecutándolo: `dbinom(0:3,10,0.25)` o `dbinom(c(0,1,2,3),10,0.25)`. Y si queremos sumarlo, `sum(dbinom(0:3,10,0.25))`.
- Si queremos probabilidades acumuladas, podemos utilizar la función `pnbinom`, que requiere las mismas entradas que `dbinom`. Por ejemplo, el 4º valor que hemos obtenido desde el menú, el valor de $P[X \leq 3]$, 0.77587509, es `pnbinom(3,10,0.25)`.

4.1.1.2. Distribución de Poisson

Las respectivas ventanas de entrada para las probabilidades simples y acumuladas (Figura 4.3) ahora requieren, lógicamente, el parámetro de la distribución de Poisson, que es su media.

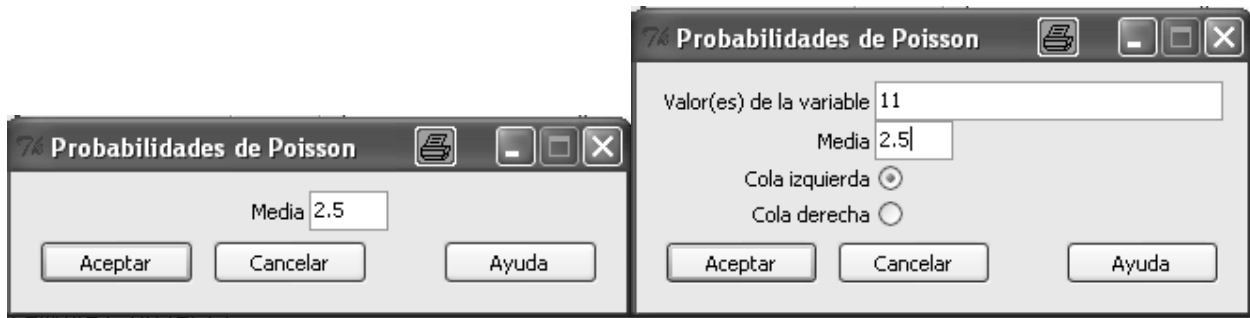


Figura 4.3: Ventanas de entrada para las probabilidades de una distribución de Poisson

Sin embargo, hay una diferencia importante con la distribución binomial. Observad que R Commander ofrece las probabilidades simples de 0 a 11. Vemos, además, que el valor 11 ya tiene probabilidad inferior a 0.00001, pero ¡no tiene probabilidad cero! No olvidemos que la distribución de Poisson puede tomar cualquier valor desde 0 a ∞ , aunque es cierto que una *Poisson*(2.5) es muy improbable que tome valores superiores a 11. De hecho, la probabilidad acumulada hasta 11 resulta ser 0.9999874, luego existe una probabilidad de

$$1 - 0.9999874 = 1.259846 \times 10^{-5}$$

de que salgan valores mayores que 11.

¿Qué podemos hacer, entonces, si queremos la probabilidad de que la distribución valga 12? Podemos usar el código que, como siempre, aparece en la ventana de instrucciones cuando ejecutamos la opción de probabilidades simples desde el menú. Cuando lo hacemos, aparece lo siguiente en la ventana de instrucciones:

```
.Table <- data.frame(Pr=round(dpois(0:11, lambda=2.5), 4))
rownames(.Table) <- 0:11
.Table
remove(.Table)
```

De ese código, lo que realmente proporciona la función masa es `dpois(0:11, lambda=2.5)`. El resto es para que el resultado aparezca como una tabla redondeando las probabilidades con 4 decimales. Si queremos la probabilidad del valor 12, escribimos `dpois(12, lambda=2.5)` o simplemente `dpois(12, 2.5)`.

En el caso de que deseáramos probabilidades acumuladas, deberíamos utilizar la función *ppois*, con los mismos argumentos que la función *dpois* para probabilidades simples.

4.1.1.3. Distribución geométrica

Sólo cabe hacer los mismos comentarios que en el caso de la distribución de Poisson. En esta ocasión la ventana de entradas pedirá el parámetro p de la distribución geométrica, y debemos también recordar que ésta puede tomar todos los valores de 0 a ∞ .

Si queremos emplear el código, las probabilidades simples las proporciona la función *dgeom* y las probabilidades acumuladas *pgeom*.

4.1.1.4. Distribución binomial negativa

Con esta distribución ocurre algo que debemos comentar. Recordemos que, en general, la distribución binomial negativa tiene parámetros $a > 0$ y $p \in [0, 1]$. Como caso particular, a puede ser un número entero; entonces, esa distribución binomial negativa puede verse como la del número de fracasos que serán necesarios hasta lograr a éxitos de probabilidad p .

Lo que ocurre si utilizamos R Commander para realizar los cálculos es que no permite que a no sea entero. De hecho, en la ventana de entrada se pide el parámetro *número de éxitos* y, aunque pongamos, por ejemplo, 2.5 éxitos, el programa redondea a 2.

¿Cómo podemos, entonces, utilizar R para calcular probabilidades de, por ejemplo, una binomial negativa de parámetros $a = 2.5$ y $p = 0.75$? De nuevo mediante la introducción de código en la ventana de instrucciones. Pongo algunos ejemplos referentes a una variable $X \rightarrow BN(2.5, 0.75)$ en el que se entenderá la sintaxis de las funciones *dnbinom* y *pnbinom*:

- $P[X = 2] = \text{dnbinom}(2, 2.5, 0.75)$.
- $P[X \leq 5] = \text{pnbinom}(5, 2.5, 0.75)$.
- $P[X = 3, 6, 7] = \text{sum}(\text{dnbinom}(c(3, 6, 7), 2.5, 0.75))$.
- $P[X > 7] = 1 - P[X \leq 6] = 1 - \text{pnbinom}(6, 2.5, 0.75)$.
- $P[X \geq 2] = 1 - P[X < 2] = 1 - P[X \leq 1] = 1 - \text{pnbinom}(1, 2.5, 0.75)$.

4.1.2. Distribuciones continuas

En el caso de las distribuciones de tipo continuo sabemos que los valores concretos de la variable tienen probabilidad cero o, dicho de otra forma, no tienen masa de probabilidad, sino densidad de probabilidad. En estas variables no tiene sentido preguntarse por probabilidades del tipo $P[X = x]$ porque todas son cero. En su lugar, lo que nos preguntamos es por las probabilidades de que las

variables proporcionen valores en intervalos, es decir, probabilidades del tipo $P[a < X < b]$, y donde las desigualdades pueden ser estrictas o no, ya que el resultado final no varía.

Siguiendo con este recordatorio, sabemos que las probabilidades del tipo $P[a < X < b]$ se calculan como

$$P[a < X < b] = \int_a^b f(x) dx = F(b) - F(a),$$

donde $f(x)$ es la función de densidad de la variable y $F(x) = \int_{-\infty}^x f(t) dt$ es una primitiva suya, conocida como función de distribución. En resumen, podremos calcular probabilidades del tipo $P[a < X < b]$ siempre que podamos obtener los valores de la función de distribución $F(x)$.

En todos los modelos que a continuación vamos a considerar R permite calcular el valor de la función de distribución o el valor de $P[X > x] = 1 - F(x)$ mediante código o mediante R Commander. Cuando se trate del valor de la función de distribución diremos que la probabilidad es la de la cola de la izquierda, y cuando queramos el valor de $P[X > x]$ diremos que queremos la probabilidad de la cola de la derecha. De nuevo iremos intercalando la manera de proceder.

4.1.2.1. Distribución uniforme

Los argumentos que requiere la ventana de entrada en R Commander son:

- Los valores de la variable para los que queremos las probabilidades acumuladas.
- El mínimo (parámetro a) y el máximo (parámetro b).
- El sentido de la cola de la probabilidad: a la izquierda (función de distribución, $P[X \leq x]$) o a la derecha ($P[X > x]$).

En código, la función de distribución *punif()* tan sólo requiere que especifiquemos en qué valor se evalúa y los dos extremos del intervalo.

4.1.2.2. Distribución exponencial

En el caso de la distribución exponencial hay que comenzar comentando algo importante con respecto a la forma de expresar dicha distribución.

Para nosotros una distribución exponencial de parámetro λ es una distribución con función de densidad

$$f(x) = \lambda e^{-\lambda x}$$

sólo para valores positivos, en cuyo caso su media es $\frac{1}{\lambda}$. Sin embargo, en muchos libros y programas

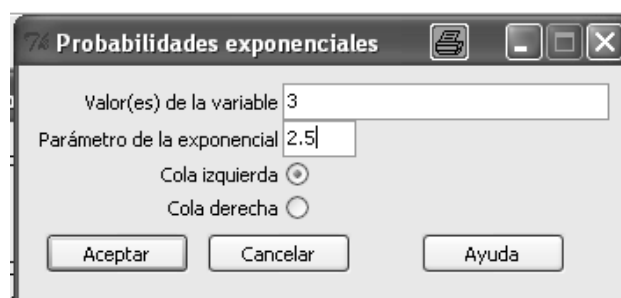


Figura 4.4: Ventana de entrada para el cálculo de probabilidades de una distribución exponencial

estadísticos se expresa como

$$f(x) = \frac{1}{\mu} e^{-\frac{x}{\mu}},$$

de manera que μ es la media. Obviamente, las dos expresiones son equivalentes, teniendo en cuenta que $\mu = \frac{1}{\lambda}$.

El problema surge cuando no tenemos claro si el programa que estamos usando o el libro que estamos leyendo usan una u otra parametrización. Si nos hablan de una distribución $\exp(2.5)$, ¿2.5 es μ o λ ? Para evitar esta confusión al parámetro λ se le llama parámetro de razón o *rate* y al parámetro μ se le llama parámetro de escala¹ o *scale*.

R Commander, por su parte, requiere en la ventana de entrada el parámetro de razón (λ). En la Figura 4.4 aparecen las entradas necesarias para calcular la probabilidad de que una distribución $\exp(2.5)$, donde 2.5 es el parámetro de razón, sea menor o igual a 3. El código que proporciona este resultado es `pexp(3,2.5)`.

4.1.2.3. Distribución Gamma

La distribución Gamma tiene el mismo problema con su parámetro λ , llamado parámetro de razón o *rate*, ya que se puede expresar su densidad como

$$f(x) = \frac{\lambda^a x^{a-1} e^{-\lambda x}}{\Gamma(a)}, \quad x \geq 0$$

o como

$$f(x) = \frac{x^{a-1} e^{-x/\mu}}{\mu^a \Gamma(a)}, \quad x \geq 0,$$

donde al parámetro μ se la llama también parámetro de escala o *scale*. Al parámetro a se le llama de forma o *shape*.

Al contrario que ocurre con la distribución exponencial, R Commander nos preguntará el pará-

¹Se le llama parámetro de escala porque es la media, con las mismas unidades que la variable, mientras que la otro se le llama parámetro de razón porque sus unidades son inversas a las de la variable.

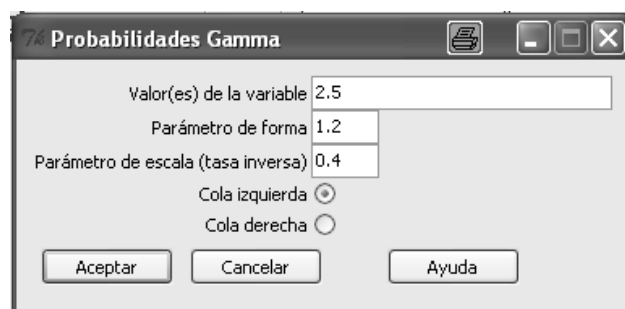


Figura 4.5: Ventana de entrada para el cálculo de probabilidades de una distribución Gamma

metro de escala en la ventana de entrada (Figura 4.5), aunque esta vez al menos nos lo aclara en la propia ventana. En la figura se está calculando la probabilidad de que (en nuestra notación) una $\text{Gamma}(1.2, 1/0.4)$ sea menor o igual a 2.5.

El código que proporciona el resultado de la ventana de la Figura 4.5 es

```
pgamma(2.5, 1.2, scale=0.4)
```

aunque ahora sí existe la posibilidad de expresarlo con el parámetro de razón, como

```
pgamma(2.5, 1.2, rate=1/0.4).
```

4.1.2.4. Distribución normal

No existe ningún problema con la parametrización de la distribución normal. Se requiere en la ventana de entradas los valores de la variable para los que queremos las probabilidades acumuladas, la media, la desviación típica y el sentido de la cola de la probabilidad: a la izquierda ($P[X \leq x]$) o a la derecha ($P[X > x]$).

4.1.3. Ejemplos

En el Cuadro 4.2 incluimos algunos ejemplos con el código que los resuelve y el resultado.

4.2. Cálculo de percentiles

Recordemos que los percentiles son medidas de posición relativas. El percentil $100p$ (siendo $100p$ un n° entre 0 y 100) de una distribución de probabilidad es aquél que deja por debajo de sí una probabilidad p ; es decir, el percentil $100p$ es el que, si pudiéramos hipotéticamente ordenar todos los valores que puede tomar la variable aleatoria de menor a mayor, ocuparía la posición $100p$. Eso es lo que permite interpretarlos como medidas de posición relativas, ya que permite analizar si un dato es *alto*, *medio* o *bajo* en su distribución.

Modelo	Probabilidad	Código	Resultado
$B(12, 0.65)$	$1 \leq X < 6$	<code>sum(dbinom(1:5, 12, 0.65))</code>	0.08462869
$Poisson(3.8)$	$X > 3$	<code>1-ppois(3, 3.8)</code>	0.5265152
$Geo(0.45)$	$2 < X < 6$	<code>sum(dgeom(3:5, 0.45))</code>	0.1386944
$BN(5, 0.75)$	$5 < X \leq 8$	<code>sum(dnbinom(6:8, 5, 0.75))</code>	0.01873858
$exp(1.2)$	$X < 2.5$	<code>pexp(2.5, 1.2)</code>	0.950213
$Gamma(1.2, 3.8)$	$X = 1.5$	¿Para qué?	0
$Gamma(1.2, 1/3.8)$	$1.05 < X < 5$	<code>pgamma(5, 1.2, scale=1/3.8) - pgamma(1.05, 1.2, scale=1/3.8)</code>	0.02771233
$N(2.5, 1.1)$	$2.2 \leq X \leq 3$	<code>pnorm(3, 2.5, 1.1) - pnorm(2.2, 2.5, 1.1)</code>	0.2827504
$N(2.2, 0.5)$	$2.2 < X < 3$	<code>pnorm(3, 2.2, sqrt(0.5)) - pnorm(2.2, 2.2, sqrt(0.5))</code>	0.3710505

Cuadro 4.2: Ejemplos de cálculo de probabilidades mediante R

Desde el punto de vista del cálculo, observemos que en la sección anterior hemos aprendido a calcular los valores $p = P[X \leq x]$. Lo que ahora tenemos que hacer es justo lo contrario, es calcular los valores x tales que $P[X \leq x] = p$.

Decir, por último, que los percentiles también se conocen, de una forma más general, como *cuantiles*, en inglés *quantiles*. Los cuantiles se asocian a la probabilidad que dejan a su izquierda. Por ejemplo, el percentil 95 es el cuantil 0.95. Otra forma de llamar a los cuantiles es *puntos porcentuales*.

4.2.1. Distribuciones discretas

A la hora de hablar de percentiles en distribuciones discretas hay que recordar que es posible que no podamos encontrar algunos percentiles concretos, precisamente por el carácter discreto de la variable. Es por eso que en el caso de distribuciones discretas la definición de percentil debe afinar un poco más. Hablamos del percentil $100p$ o del cuantil p como del mayor valor x tal que $P[X \leq x] \geq p$.

4.2.1.1. Distribución binomial

Para que entendamos bien esto último, vamos a recordar el ejemplo de la función masa de una distribución $B(10, 0.25)$. Dicha función masa aparece en el Cuadro 4.3.

El percentil 25 es 1, ya que $P[X \leq 1] = 0.06 + 0.19 = 0.25$, pero también es el percentil 20, ya que es el primer valor tal que $P[X \leq x] \leq 0.2^2$.

Para calcular percentiles (en realidad, cuantiles) de la distribución binomial mediante R Com-

²En realidad, el percentil 25 es 2, pero por los redondeos aplicados en la tabla, de ella se deduce que es 1.

x	Pr
0	0.06
1	0.19
2	0.28
3	0.25
4	0.15
5	0.06
6	0.02
7	0.00
8	0.00
9	0.00
10	0.00

Cuadro 4.3: Función masa de la $B(10,0.25)$

mander podemos utilizar la opción del menú *Distribuciones* \rightarrow *Distribuciones discretas* \rightarrow *Distribución binomial* \rightarrow *Cuantiles binomiales*. La ventana de entradas simplemente nos pide los parámetros de la distribución y la(s) probabilidad (es) de la(s) que queremos los cuantiles.

Decir, por último, que el código asociado a esta ventana, el que proporciona cuantiles de la distribución binomial es `qbinom`. Por ejemplo, `qbinom(0.25,10,0.75)` devuelve el percentil 25 de una $B(10,0.75)$.

4.2.1.2. Distribución de Poisson

La ventana de entradas correspondiente requiere los parámetros habituales. El código equivalente para, por ejemplo, obtener el percentil 95 de una *Poisson* (3.2) es `qpois(0.95,3.2)`.

4.2.1.3. Distribución geométrica

La ventana de entradas correspondiente requiere los parámetros habituales. El código equivalente para, por ejemplo, obtener el percentil 5 de una *Geo* (0.32) es `qgeom(0.05,0.32)`.

4.2.1.4. Distribución binomial negativa

La ventana de entradas correspondiente requiere los parámetros habituales, si bien hay que tener en cuenta que sólo permite que el parámetro a de la distribución sea un entero. El código equivalente para, por ejemplo, obtener el percentil 50 de una *BN* (3.2, 0.67) es `qnbinom(0.50,3.2,0.67)`. Dicho código sí permite que a no sea entero.

4.2.2. Distribuciones continuas

En el caso de las distribuciones continuas, dado $p \in (0,1)$ siempre existe un valor x tal que $P[X \leq x]$ es exactamente igual a p , y ese valor es el percentil $100p$ o el cuantil p .

4.2.2.1. Distribución uniforme

La ventana de entradas correspondiente requiere los parámetros habituales. El código equivalente para, por ejemplo, obtener el percentil 95 de una $U(3.2, 4.5)$ es `qunif(0.95, 3.2, 4.5)`.

4.2.2.2. Distribución exponencial

La ventana de entradas correspondiente requiere los parámetros habituales. Hay que tener en cuenta que el parámetro que hay que especificar es el parámetro de razón, λ . El código equivalente para, por ejemplo, obtener el percentil 25 de una $exp(4.5)$ es `qexp(0.25, 4.5)`.

4.2.2.3. Distribución Gamma

La ventana de entradas correspondiente requiere los parámetros habituales. En esa ventana podemos ver que hay que especificar el parámetro de forma y el parámetro de escala, $\mu = 1/\lambda$. El código equivalente para, por ejemplo, obtener el percentil 5 de una $Gamma(1.2, 2.5)$ es

```
qgamma(0.05, shape=1.2, rate=2.5)
```

aunque también puede especificarse como `qgamma(0.05, shape=1.2, scale=1/2.5)`.

4.2.2.4. Distribución normal

La ventana de entradas correspondiente requiere los parámetros habituales. El código equivalente para, por ejemplo, obtener el percentil 95 de una $N(0, 4.5)$ es `qnorm(0.95, 0, 4.5)`.

4.3. Representaciones gráficas de distribuciones de probabilidad

4.3.1. Mediante R Commander

Las distintas opciones del menú *Distribuciones*, para cada una de las que hemos estudiado, como por ejemplo, *Distribuciones* \rightarrow *IPSUR - Continuous distributions* \rightarrow *Distribución normal* \rightarrow *Gráfica de la distribución normal*, ofrecen la posibilidad de representar gráficamente las características de cualquiera de las distribuciones que hemos estudiado. En estas opciones del menú tan sólo tenemos que especificar el valor de los parámetros de la distribución y si deseamos graficar la función masa

(probabilidades simples) o densidad, la función de distribución (probabilidades acumuladas) o los cuantiles de la distribución.

4.3.2. Mediante código. La función *plot*

Vamos a aprovechar la necesidad de comentar el código que permite representar gráficamente una distribución de probabilidad para describir una de las funciones más utilizadas en representaciones gráficas con R, la función *plot()*.

La función *plot()* ejecuta una simple representación de un conjunto de puntos caracterizados mediante coordenadas cartesianas. Su sintaxis básica es la siguiente:

```
plot(x,y,type="l",main="Título",sub="Subtítulo",xlab="Eje X",ylab="Eje Y")
```

En esta expresión,

- *x* se refiere a las coordenadas en el eje de abscisas de los puntos que queremos representar, expresadas como un vector.
- *y* son las coordenadas en el eje de ordenadas.
- *type* especifica el tipo de gráfico que queremos. Las opciones más habituales son "*p*" si queremos simplemente puntos, "*l*" si queremos que una los puntos con una línea o "*b*" , si queremos que haga ambas cosas.
- *main* es el título del gráfico.
- *sub* es el subtítulo.
- *xlab* especifica el título del eje X.
- *ylab* especifica el título del eje Y.

Por tanto, la representación de una distribución de probabilidad mediante *plot()* tan sólo requerirá que le digamos *qué* queremos representar (en el eje Y; probabilidades simples o densidades, probabilidades acumuladas o cuantiles) y *dónde* (en el eje X). Las demás opciones sólo añadirán información al gráfico.

Por ejemplo, supongamos que deseamos representar la función de densidad de una distribución normal $N(\mu = 100, \sigma = 10)$:

1. Empecemos preguntándonos *dónde* queremos la representación. Dado que el 99 % de los valores de una normal están en un intervalo de su media menos/más tres desviaciones típicas, vamos

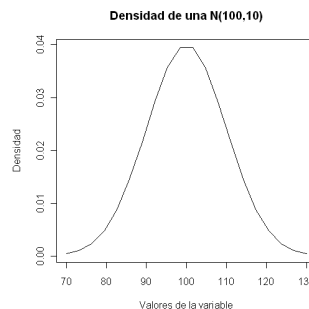


Figura 4.6: Densidad de una $N(100, 10)$

a representarlo entre 70 y 130. Por su parte, vamos a elegir un número razonable de puntos, por ejemplo, 20. Con esto, podemos especificar que las coordenadas en el eje X de los puntos que queremos representar son

```
x<-seq(70,130,length.out=20).
```

2. Queremos representar la función de densidad, luego las coordenadas en el eje Y deben ser

```
y<-dnorm(x,100,10).
```

3. Finalmente, el código sería el siguiente, donde hemos añadido etiquetas y títulos:

```
x<-seq(70,130,length.out=20)
y<-dnorm(x,100,10)
plot(x,y,type='l',main='Densidad de una N(100,10)',
xlab='Valores de la variable',ylab='Densidad')
```

El resultado aparece en la Figura 4.6.

4.4. Simulación de muestras

Una de las aplicaciones más comunes de la Estadística es la de proporcionar un marco teórico para poder realizar simulaciones de procesos más o menos complejos. En esas simulaciones existe la necesidad de contar con *datos inventados*, pero *inventados* según un modelo proporcionado por una distribución de probabilidad que se suponga adecuado para el fenómeno que estamos simulando.

Es por ello que la mayoría de los paquetes de software estadístico, entre ellos R, facilitan la posibilidad de obtener muestras aleatorias simples de las distribuciones más usuales.

Lo que vamos a hacer a continuación es proporcionar dos ejemplos, uno de una distribución continua y otro de una discreta, de cómo se obtienen muestras aleatorias. Además, vamos a realizar una comparación gráfica para comprobar que, en efecto, esos datos simulados en las muestras corresponden con los modelos que supuestamente los generan.

4.4.1. Muestra procedente de una distribución normal

4.4.1.1. Mediante R Commander

Para obtener muestras seleccionamos la opción del menú *Distribuciones* \rightarrow *Sampling distributions* \rightarrow *Continuous distributions* \rightarrow *Normal population*³. La ventana de entradas de esta opción (Figura 4.7) requiere:

- El nombre que le vamos a asignar al conjunto de datos que se genera con la muestra. No se permiten espacios en este nombre, ni caracteres extraños.
- La media de la distribución normal.
- La desviación típica de la distribución normal.
- El *número de muestras (filas)*. Esta entrada se refiere al tamaño de la(s) muestra(s) que vamos a generar, es decir, al número de casos o de observaciones que contiene la(s) muestra(s) y que, por tanto, corresponde al n^o de filas del conjunto de datos generado.
- El *número de observaciones (columnas)*. Se refiere al n^o de muestras que queremos generar. Por ejemplo, normalmente sólo queremos generar una muestra, en cuyo caso el valor de esta entrada será 1 y nos devolverá como resultado un conjunto de datos formado por una sola variable (columna). Pero podría ocurrir que queramos varias muestras a la vez, que aparecerían como columnas en el conjunto de datos.

Adicionalmente, si generamos más de una columna, la ventana permite incorporar varios resúmenes a los resultados, entre ellos, la media y la desviación típica de las muestras generadas para cada fila.

En este ejemplo, la Figura 4.7 genera una sola muestra de 250 casos de una distribución $N(1.2, 2.3)$.

4.4.1.2. Mediante código

La simulación viene dada por la función *rnorm()*. La letra *r* se usa como acrónimo de *random* que en inglés significa aleatorio. La sintaxis es muy sencilla: tan sólo tenemos que especificarle el

³También se puede hacer desde *Distribuciones* \rightarrow *IPSUR - Discrete distributions* \rightarrow *Distribución de Poisson* \rightarrow *Muestra de una distribución de Poisson*.

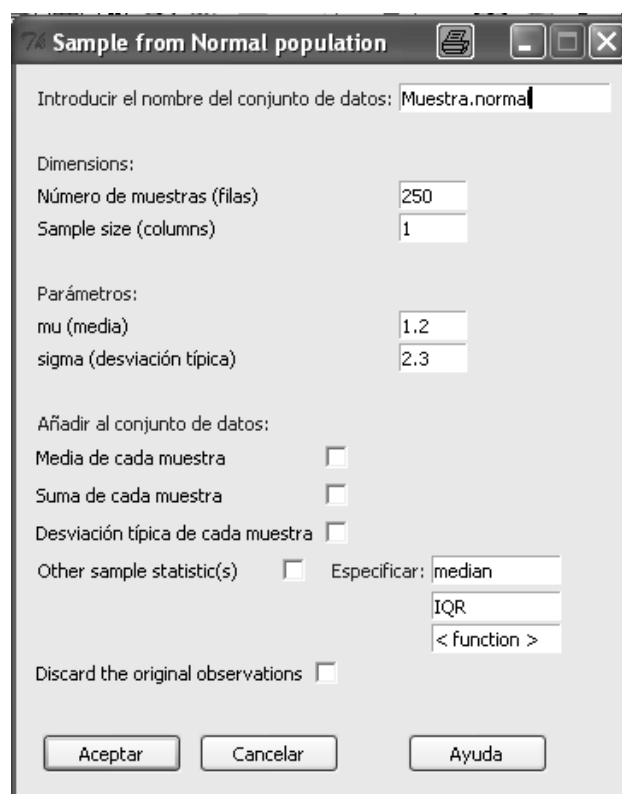


Figura 4.7: Ventana de entradas para simular una muestra de una distribución normal

número de valores que queremos simular y los parámetros de la distribución. Por ejemplo, el ejemplo que hemos realizado mediante R Commander se escribiría simplemente como

```
rnorm(250,100,10) .
```

4.4.1.3. Comparación de la densidad teórica con el histograma de los datos simulados

Vamos a comprobar que, en efecto, esta muestra procede de una $N(1.2, 2.3)$. Para ello podemos comparar el histograma de la muestra con la función de densidad de esa $N(1.2, 2.3)$. Si, como debería ocurrir, los datos simulados proceden de esa normal, el histograma y la función de densidad deberían parecerse bastante. El resultado aparece en la Figura 4.8. Hay que decir que el parecido es notable, aunque sería aún más notable si hubiéramos elegido aún más datos (más de 250).

El código es el siguiente:

1. `muestra<-sort(rnorm(250,100,10))` genera la muestra de tamaño 250 de la $N(100, 10)$ y la ordena mediante la función `sort()`⁴.
2. `hist(muestra,freq=FALSE)` grafica el histograma de los datos de la muestra en escala de densidad.

⁴¿Qué ocurre si no ordenamos los datos? Realizar el gráfico sin hacerlo y explicar porqué ocurre lo que ocurre.

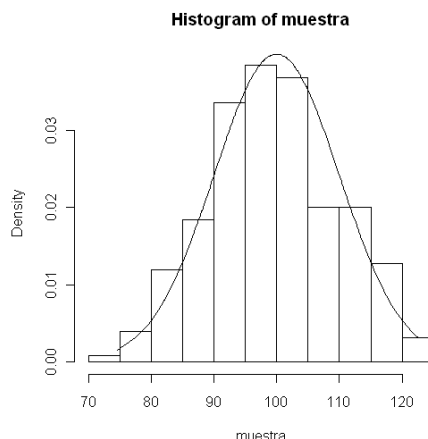


Figura 4.8: Histograma de una muestra junto con la función de densidad del modelo del que procede

3. Finalmente `lines(muestra,dnorm(muestra,100,10))` añade al histograma la gráfica de la función de densidad evaluada en los valores de la muestra.

Hemos de hacer un par de comentarios a este código a propósito de la función `lines()`. Se trata de una función muy similar a `plot()` en su estructura y utilidad, cuya principal diferencia radica en que `plot()` es lo que en R se conoce como una función gráfica de primer nivel y `lines()` lo es de segundo nivel. Que una función gráfica sea de primer nivel quiere decir que por sí misma abrirá una ventana gráfica y mostrará su resultado. Que una función sea de segundo nivel implica que por sí misma no tiene autoridad para abrir una ventana de gráficos; tan sólo puede añadirse a una ventana ya abierta.

`hist()` es una función gráfica de primer nivel, por lo que realiza el histograma y lo muestra en una ventana de gráfico. A esa ventana se añade la función de densidad graficada mediante `lines()`. ¿Qué hubiera ocurrido si lo hubiéramos hecho mediante `plot()`? Que al ser esta una función de primer nivel, habría anulado el histograma creando una nueva ventana sólo para ella.

4.4.2. Muestra procedente de una distribución de Poisson

4.4.2.1. Mediante R Commander

En esta ocasión seleccionamos, por ejemplo⁵, la opción del menú *Distribuciones* → *IPSSUR - Discrete distributions* → *Distribución de Poisson* → *Muestra de una distribución de Poisson*. La ventana de entradas de esta opción (Figura 4.9) requiere:

- El nombre que le vamos a asignar al conjunto de datos.

⁵También se puede hacer desde *Distribuciones* → *Sampling distributions* → *Discrete distributions* → *Poisson population*.

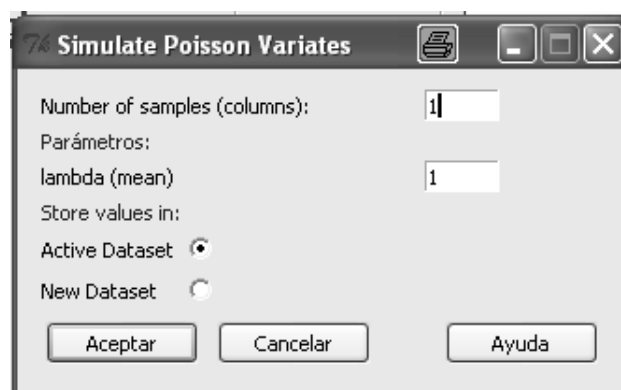


Figura 4.9: Ventana de entradas para simular una muestra de una distribución de Poisson

- La media de la distribución.
- El *número de observaciones (columnas)*.

4.4.2.2. Mediante código

Se realiza mediante la función `rpois()` cuya sintaxis simplemente requiere el número de muestras y el parámetro de la distribución.

4.4.2.3. Comparación de la función masa teórica con el diagrama de barras de los datos simulados

De nuevo vamos a utilizar el código para plasmar cómo los datos simulados obedecen a la distribución que los generó:

1. Empezamos generando una muestra de 250 valores de una *Poisson* (2.5) mediante `muestra<-rpois(250,2.5)`.
2. Representamos un diagrama de barras para esos datos valiéndonos de la función `hist()`. Como el valor máximo que ha resultado en la muestra es 9, introducimos `hist(muestra,breaks=-0.5:9.5)`.
3. Añadimos a esa gráfica las frecuencias esperadas según la función masa de una *Poisson* (2.5) para los valores de 0 a 9 en una muestra de 250 datos mediante

```
lines(0:9,dpois(0:9,2.5)*250) .
```

El resultado aparece en la Figura 4.10.

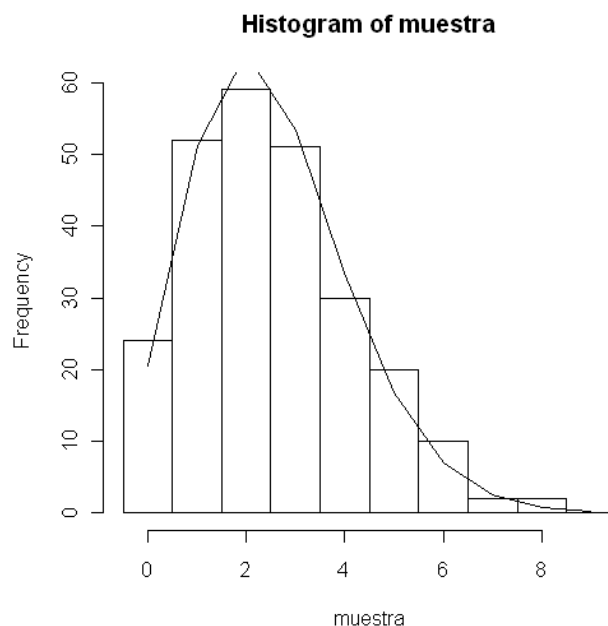


Figura 4.10: Comparación del diagrama de barras de una muestra generada según una $Poisson(2.5)$ con las frecuencias esperadas según ese modelo.

4.4.3. Muestras procedentes de otras distribuciones

Es evidente que las ventanas de entrada de las correspondientes opciones del menú para el resto de las distribuciones requieren los parámetros de éstas, como habitualmente han sido introducidos. Las funciones que en el código de R proporcionan esas opciones son *rbinom*, *rgeom*, *rnbinom*, *runif*, *rexp* y *rgamma*.

4.5. Ejercicios propuestos

1. Consideremos una variable aleatoria que sigue una distribución $B(15, 0.33)$. Se pide:
 - a) Calcular la probabilidad de que la variable sea mayor que 3 y menos o igual que 7.
 - b) Calcular la probabilidad de que sea mayor que 5.
 - c) ¿Qué valor de la variable deja por debajo de sí el 75 % de la probabilidad?
 - d) Calcular el percentil 95 % de la distribución.
 - e) Obtener una muestra de tamaño 1000 de esta distribución, representarla gráficamente mediante un diagrama de barras y comparar éste con las frecuencias esperadas según el modelo que genera los datos.
2. Consideremos una variable aleatoria que sigue una distribución $P(7.2)$. Se pide:

- a) Calcular la probabilidad de que sea mayor o igual que 10.
 - b) Calcular la probabilidad de sus valores mayores o iguales a 2 y menores o iguales a 8.
 - c) Obtener el percentil 75 de la distribución.
 - d) ¿Qué valor es el que deja por debajo de sí el 5 % de los valores más bajos de la variable?
 - e) Obtener una muestra de tamaño 500 de la distribución, representarla gráficamente mediante un diagrama de barras y comparar éste con las frecuencias esperadas según el modelo que genera los datos.
3. Sea X una variable aleatoria con distribución $Geo(0.56)$. Se pide:
- a) Calcular $P[X < 2]$.
 - b) Calcular $P[3 \leq X < 9]$.
 - c) Obtener el percentil 50, es decir, la mediana.
 - d) ¿Cuál es el valor de la variable por debajo del cual queda el 25 % de los valores más bajos?
 - e) Obtener dos muestras de tamaño 100 y los diagramas de barras de ambas muestras. ¿Son iguales? ¿Por qué?
4. Sea $Y \hookrightarrow BN(5, 0.25)$. Se pide:
- a) Calcular $P[2 < Y < 9]$.
 - b) Calcular $P[Y > 5]$.
 - c) Calcular el percentil 5 y el percentil 95.
 - d) ¿Qué valor de la variable se encuentra justo en el centro de la distribución?
 - e) Obtener una muestra de tamaño 1000 de la distribución, representarla gráficamente mediante un diagrama de barras y comparar éste con las frecuencias esperadas según el modelo que genera los datos.
5. Consideremos una variable aleatoria X que sigue una distribución exponencial de media 2. Se pide:
- a) $P[X = 5]$.
 - b) $P[2 < X < 8.5]$
 - c) El percentil 5 y el percentil 95.

- d) ¿Qué valor de la variable queda justo en el centro de la distribución?
 - e) Obtener una muestra de tamaño 1000 de la distribución.
 - f) Representar la función de densidad de esta distribución y compararla con el histograma de la muestra obtenida en el apartado anterior.
6. Sea una variable aleatoria Y con distribución Gamma cuyo parámetro de forma es 1.5 y cuyo parámetro de escala es 5.2. Se pide:
- a) $P[X = 5, 6, 7]$.
 - b) $P[X < 5]$
 - c) El percentil 5 y el percentil 95.
 - d) ¿Qué valor de la variable deja por encima de sí el 15 % de los valores más altos de la variable?
 - e) Obtener una muestra de tamaño 1000 de la distribución.
 - f) Representar la función de densidad de esta distribución y compararla con el histograma de la muestra obtenida en el apartado anterior.
7. Consideremos una variable aleatoria W con distribución $N(250, 13)$. Se pide:
- a) $P[240 < W \leq 245.5]$
 - b) $P[W \geq 256]$.
 - c) Si queremos desechar el 5 % de valores más altos de la distribución y el 5 % de valores más bajos, ¿con qué intervalo de valores nos quedaremos?
 - d) Obtener una muestra de tamaño 1000 de la distribución.
 - e) Representar la función de densidad de esta distribución y compararla con el histograma de la muestra obtenida en el apartado anterior.

Capítulo 5

Estimación puntual y por intervalos de confianza

Objetivos:

1. Estimar mediante el método de máxima verosimilitud los parámetros que ajustan las distribuciones que hemos manejado hasta ahora a un conjunto de datos.
2. Obtener una representación gráfica que permita valorar, al menos visualmente, la bondad del ajuste logrado mediante el ajuste de una distribución a unos datos.
3. Obtener intervalos de confianza para medias, proporciones y varianzas.

A lo largo de este capítulo vamos a emplear funciones de R que no están implementadas como opciones del menú de R Commander, de manera que tendremos que evaluarlas mediante la introducción de código.

5.1. Estimación máximo verosímil

Como punto de partida a lo largo de todo el capítulo, vamos a suponer que tenemos una muestra aleatoria simple de datos de una variable. En esta sección lo que vamos a aprender es a obtener una estimación de los parámetros que ajustan las distribuciones que venimos manejando a estos datos.

El método de estimación que vamos a utilizar, quizá el más utilizado a todos los niveles, es el método de máxima verosimilitud. Recordemos que éste se basa en la optimización de la llamada función de verosimilitud, que depende de los parámetros de la distribución. R es una magnífica herramienta para programar esta optimización mediante el empleo de métodos numéricos desarrollados por los matemáticos, cuando esto es necesario.

No obstante, nosotros no vamos a programar el método de máxima verosimilitud, sino que vamos a utilizar un paquete de R, *MASS*, que ya trae incorporadas las estimaciones de los parámetros de las distribuciones más usuales. Para ello, veamos primero cómo se instala y se carga el paquete:

1. Abrimos R (sin cargar R Commander aún).
2. Elegimos la opción del menú *Paquetes* \rightarrow *Instalar paquete(s)*.
3. Se abrirá una ventana para que elijamos un *mirror* desde el que descargar el paquete. Por cercanía, elegimos *Spain*.
4. A continuación se abrirá una ventana titulada *Packages* en la que aparecen todos los paquetes que podemos instalar de R. Elegimos *MASS (VR)*.
5. Finalmente, elegimos en R la opción *Paquetes* \rightarrow *Cargar paquete*. Aparecen todos los paquetes que tenemos incorporados en nuestra instalación de R. De entre ellos, elegimos *MASS*. Ahora ya podemos cargar R Commander.

El paquete *MASS* contiene una función llamada *fitdistr* que proporciona las estimaciones máximo verosímiles y los errores estándares asociados a esas estimaciones de las distribuciones más usuales dados unos datos. La sintaxis de esta función es muy simple; sólo necesita que especifiquemos:

- Los datos de la muestra. Sólo debemos tener en cuenta que estos datos no pueden incorporar datos faltantes (que aparecen como *NA*).
- La distribución de la cuál queremos los estimadores de sus parámetros. Las opciones que a nosotros nos interesan son las siguientes: "Poisson", "geometric", "negative binomial", "exponential", "gamma" y "normal". No aparece el estimador del parámetro p de una distribución binomial, probablemente porque es tan simple que los autores del paquete creyeron que no merecía la pena hacerlo.

Es importante comentar que la función *fitdistr* devuelve el valor exacto de los estimadores para las distribuciones de Poisson, geométrica, exponencial y normal, mientras que para las distribuciones binomial negativa y Gamma utiliza métodos numéricos para proporcionar valores aproximados de las estimaciones, ya que no existen fórmulas explícitas de los estimadores para esas distribuciones.

Vamos a ir viendo en una serie de ejemplos cómo se concreta esta sintaxis para cada uno de los modelos que conocemos.

5.1.1. Para la distribución de Poisson

5.1.1.1. Estimación de los parámetros

Uno de los ejemplos más clásicos sobre el uso de la distribución de Poisson es el del ajuste del número de muertes por coces de caballos en el ejército prusiano. Von Borkiewicz publicó y analizó en 1898 el número de muertes al año por coces de caballo que se habían producido en 10 secciones del ejército de Prusia durante 20 años. La distribución de frecuencias de estos datos aparece en el Cuadro 5.1¹

Muertes/año	Frecuencia
0	109
1	65
2	22
3	3
4	1

Cuadro 5.1: Distribución de frecuencias en el ejemplo del ejército prusiano

Lo que se planteaban en aquella época es si este tipo de accidentes mortales se producían por puro azar en las distintas secciones o si había alguna causa que implicara un mayor número de muertes en algunos casos concretos. Esa cuestión del *puro azar* equivale a plantearse si la distribución de Poisson es adecuada para la variable considerada.

Los datos están en el fichero *DatosMuertesCoces.rda*. Al cargarlo tenemos un conjunto de datos activo (que en mi caso se llama *muertes*), dentro del cual hay una variable que se refiere al nº de muertes al año en cada una de las 20 secciones y los 10 años (la variable en mi conjunto de casos activo se llama *muertes*). Lo que vamos a hacer es estimar el parámetro de la distribución de Poisson. Es tan simple como introducir en la consola el código siguiente:

```
fitdistr(muertes$muertes,"Poisson")
```

El resultado es :

```
lambda  
0.6100000  
(0.0552268)
```

Es decir, el parámetro λ de la distribución de Poisson, que es su media, es estimado en un valor $\hat{\lambda} = 0.61$, y el error estándar de esa estimación es de 0.052268.

¹Los datos se pueden introducir con el siguiente código:
`muertes<-c(rep(0,109),rep(1,65),rep(2,22),rep(3,3),4)`

5.1.1.2. Comparación del modelo estimado con la distribución de frecuencias

Así pues, los datos cuya distribución de frecuencias aparece en el Cuadro 5.1 han sido ajustados mediante una distribución *Poisson* (0.61), pero hasta ahora nadie ha hecho nada por confirmar si ese modelo Poisson es realmente bueno para esos datos.

Lo que vamos a hacer nosotros para poder decir algo al respecto es representar en una misma figura la distribución de frecuencias de la muestra junto con las frecuencias que determina una distribución *Poisson* (0.61) para 200 datos, que son los que tiene la muestra.

En primer lugar, para ver estas frecuencias según la distribución de Poisson ajustada, no tenemos más que introducir el código `200*dpois(0:4,0.61)`, lo que nos devuelve los siguientes resultados:

```
108.6701738 66.2888060 20.2180858 4.1110108 0.6269291
```

Ahora, por tanto, lo que queremos es un diagrama de barras que represente la distribución de frecuencias y, además, dibujar estas frecuencias que determina la distribución de Poisson ajustada a los datos:

- `hist(muertes$num.muertes,breaks=-0.5:4.5)` dibuja la distribución de frecuencias. No debes cerrar ese gráfico.
- `lines(0:4,200*dpois(0:4,0.61))` añade al gráfico anterior las frecuencias según el modelo ajustado.

El gráfico resultante lo tenemos en la Figura 5.1. Resulta interesante ver cómo el modelo ajustado casi *clava* la distribución de frecuencias. Eso quiere decir que parece que la distribución de Poisson sí es adecuada para esos datos².

5.1.2. Para la distribución geométrica

En tráfico de telecomunicaciones los datos se transmiten en unas unidades de información llamadas *paquetes*. En un protocolo de transmisión cualquiera el tiempo que pasa entre un paquete de información y el siguiente es variable, mejor dicho, es aleatorio, ya que depende de multitud de factores difícilmente controlables, la mayoría de los cuales se refieren a las condiciones de la red en el momento de la transmisión. Los datos que vamos a manejar se refieren a los tiempos (en *ms*) que transcurren entre una muestra aleatoria de 250 paquetes en una determinada transmisión. Están en el fichero *tráfico.telem.rda*. Puede resultar extraño que tratándose de tiempos los analicemos como una variable discreta en vez de continua, pero a este nivel (es decir, al nivel de los milisegundos) los ordenadores suelen trabajar con el tiempo como si fuera una variable discreta.

²Pero no podemos por ahora afirmar nada categóricamente al respecto. Veremos esto con rigor más adelante.

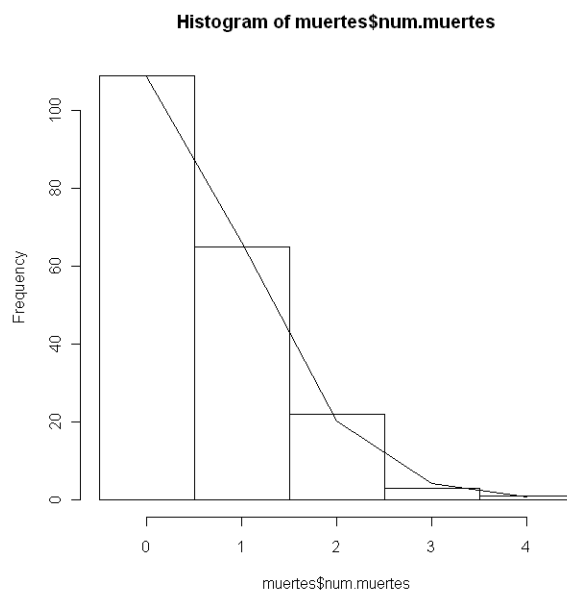


Figura 5.1: Diagrama de barras y frecuencias del modelo de Poisson ajustado para los datos del número de muertes por coces en el ejército prusiano

Una de las aplicaciones más usuales de la Estadística en este campo se refiere a la posibilidad de proporcionar muestras simuladas. Para ello, previamente se necesita un modelo determinado por una distribución de probabilidad. En esta cuestión es donde vamos a considerar nuestro objetivo: queremos proporcionar una distribución de probabilidad adecuada para esos datos.

Uno de los más importantes organismos internacionales de Telecomunicaciones, la ETSI, recomienda que para datos como estos se utilice la distribución geométrica. Vamos, por ahora, a hacerle caso a la ETSI y ajustar una distribución geométrica a estos datos estimando su parámetro por máxima verosimilitud.

5.1.2.1. Estimación del parámetro

Para ello, cargamos en primer lugar el conjunto de datos, que se encuentra en el fichero *trafico.telem.rda*, y cuyo nombre es también *trafico.telem*. Para ver el nombre de la variable podemos usar la opción *Visualizar conjunto de datos*. Vemos que el nombre de la variable es *tiempos.entre.paquetes*. Finalmente, el código para obtener la estimación es

```
fitdistr(trafico.telem$tiempos.entre.paquetes,"geometric")
```

Este código, ejecutado en la ventana de instrucciones, devuelve lo siguiente en la ventana de resultados:

```
prob
```

```
0.117591722  
(0.006986208)
```

Por lo tanto, el ajuste por el método de máxima verosimilitud de los datos mediante una distribución geométrica lo es con un parámetro $p = 0.1176$, con un error típico para esta estimación de casi 0.007.

5.1.2.2. Comparación del modelo estimado con la distribución de frecuencias

Como en el anterior ejemplo, vamos ahora a obtener una representación gráfica simultánea de la distribución de frecuencias y de las frecuencias que se esperan según una distribución *Geo* (0.1176) para 250 datos.

Los valores que se observan en la muestra van de 0 a 32. Las probabilidades según una *Geo* (0.1176) de los valores de 0 a 32 los proporciona el código `dgeom(0:32,0.117591722)`. Debemos multiplicar estas probabilidades por 250, que es el número de datos de la muestra, para obtener las frecuencias esperadas según el modelo. Finalmente, el código para la representación gráfica es el siguiente:

- `hist(trafico.telem$tiempos.entre.paquetes,breaks=-0.5:32.5,freq=TRUE)` dibuja la distribución de frecuencias. No debeis cerrar ese gráfico.
- `lines(0:32,250*dgeom(0:32,0.117591722))` añade al gráfico anterior las frecuencias según el modelo ajustado.

El resultado aparece en la Figura 5.2. Ahora no es tan fácil valorar si el ajuste es bueno o malo, pero si nos fijamos bien, las frecuencias de los primeros valores (0 y 1 sobre todo) están bastante mal ajustadas, y hay algunas otras frecuencias intermedias que tampoco están bien ajustadas. No podemos decir que el ajuste sea malo, pero tampoco que es excelente. Más adelante aprenderemos a ser más objetivos y precisos en esta cuestión.

5.1.3. Para la distribución binomial negativa

Vamos a continuar con los datos de los tiempos entre paquetes, pero como no estamos muy convencidos del ajuste mediante la distribución geométrica y puesto que sabemos que la distribución binomial negativa es una generalización de ésta, vamos a proporcionar un ajuste mediante una binomial negativa.

5.1.3.1. Estimación de los parámetros

Simplemente ejecutamos el siguiente código:

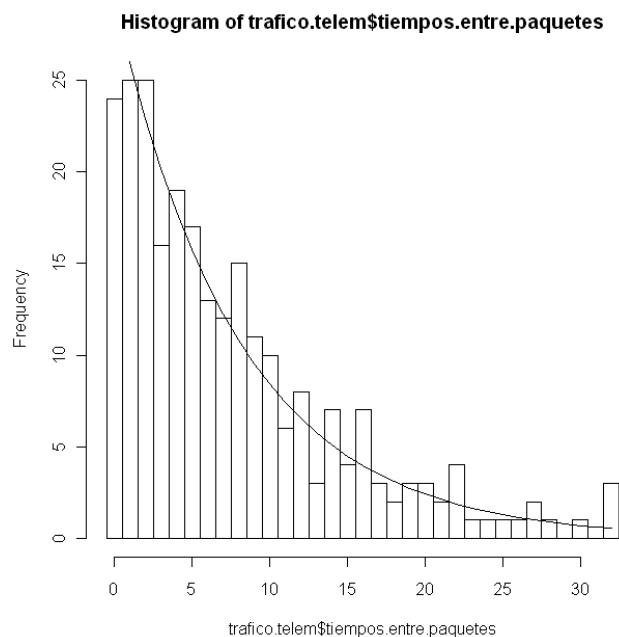


Figura 5.2: Ajuste de los datos de tiempos entre paquetes mediante una distribución geométrica

```
fitdistr(trafico.telem$tiempos.entre.paquetes,
'negative binomial')
```

El resultado es el siguiente

```
size mu
1.1903584 7.5039917
(0.1265281) (0.4682259)
```

Vamos a interpretarlo:

- El parámetro *size* es el parámetro que nosotros hemos llamado a en una $BN(a, p)$, luego $\hat{a} = 1.19$, con un error típico de 0.1265.
- El parámetro μ es la media. Nosotros sabemos que la media es $\mu = a \frac{1-p}{p}$, luego despejando tenemos que

$$p = \frac{a}{a + \mu}.$$

Si aplicamos esto a nuestras estimaciones,

$$\hat{p} = \frac{\hat{a}}{\hat{a} + \hat{\mu}} = \frac{1.1903584}{1.1903584 + 7.5039917} = 0.1369117.$$

Por lo tanto, el ajuste es una distribución $BN(1.19, 0.137)$.

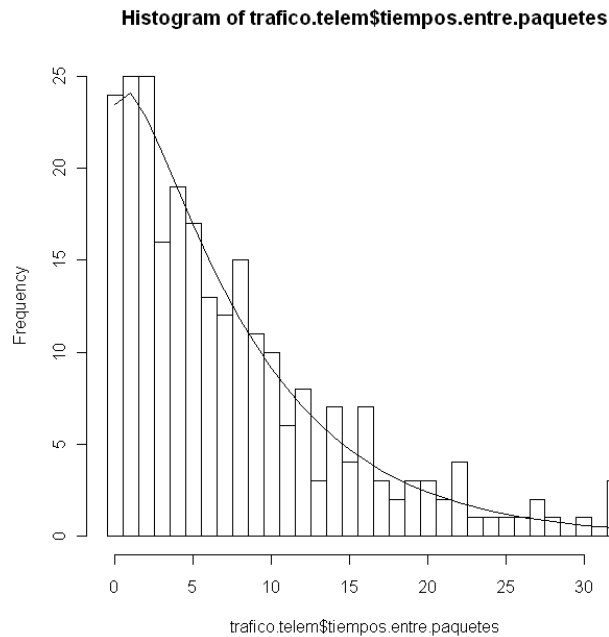


Figura 5.3: Ajuste de los datos de tiempos entre paquetes mediante una distribución binomial negativa

5.1.3.2. Comparación del modelo ajustado con la distribución de frecuencias

Para construir el gráfico que nos permite realizar la comparación del modelo ajustado con la distribución de frecuencias, consideramos el siguiente código:

- `hist(tráfico.telem$tiempos.entre.paquetes,breaks=-0.5:32.5,freq=TRUE)` dibuja la distribución de frecuencias. No debeis cerrar ese gráfico.
- `lines(0:32,250*dnbinom(0:32,1.19,0.137))` añade al gráfico anterior las frecuencias según el modelo ajustado.

El resultado aparece en la Figura 5.3. Podemos ver que el ajuste es mejor que el proporcionado por la geométrica. Fijémonos sobre todo en las frecuencias del 0 y del 1, que ahora están mucho mejor ajustadas.

5.1.4. Para la distribución exponencial

Como conjunto de datos para ejemplificar el procedimiento vamos a considerar 300 datos correspondientes al tiempo hasta el fallo de unas determinadas componentes electrónicas usadas en un proceso industrial. En principio, el histograma se asemeja bastante al de una distribución exponencial, así que no parece descabellado tratar de obtener un ajuste mediante esta distribución para estos datos.

5.1.4.1. Estimación de los parámetros

Los datos se encuentran en el fichero *tiempos.fallo.rda*. Al cargarlos como conjunto de datos activo, vemos que la variable también se llama *tiempos.fallo*. Con esta información, el código para obtener el estimador máximo-verosímil del parámetro λ de una distribución exponencial que ajuste los datos es el siguiente:

```
fitdistr(tiempos.fallo$tiempos.fallo,"exponential")
```

El resultado es el siguiente:

```
rate  
0.49185399  
(0.02839720)
```

Tenemos, por tanto, que el ajuste por el método de máxima verosimilitud de la distribución binomial de estos datos arroja como modelo el de una *exp* (0.492). El error estándar estimado de la estimación de λ es 0.028.

5.1.4.2. Comparación del modelo ajustado con los datos muestrales

Como en los ajustes anteriores, vamos ahora a tratar de obtener una representación gráfica de cómo de preciso es el ajuste. Esta vez se trata de una variable continua, por lo que ya no podemos utilizar frecuencias absolutas ni relativas, sino densidades. Lo que vamos a hacer es comparar el histograma, utilizando una escala de densidad, con la función de densidad de la distribución *exp* (0.492):

1. Para obtener ese histograma utilizamos

```
hist(tiempos.fallo$tiempos.fallo,breaks=18,freq=FALSE).
```

 Observad que he elegido 18 intervalos, que es aproximadamente la raíz cuadrada del número de datos que tenemos, 300. Esta gráfica no debeis cerrarla.

2. Ahora vamos a añadirle el gráfico de la función de densidad de la distribución *exp* (0.492). Vamos a dibujar esta densidad en el mismo dominio que tiene el histograma, de 0 a 10. Vamos a dibujarla seleccionando, por ejemplo, 101 puntos entre 0 y 10, con la misma distancia entre ellos. La forma más fácil de hacerlo es mediante el código *0:100/10*³. El código es

```
lines(0:100/10,dexp(0:100/10,0.49185399))
```

³*0:100* proporciona los números de 0 a 100. Al dividirlos por 10 obtenemos números de 0 a 10 a intervalos de longitud 0.1.

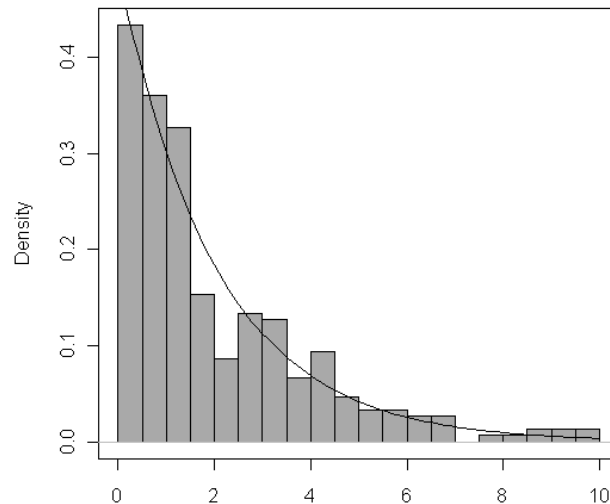


Figura 5.4: Comparación del histograma de los tiempos de fallo con la densidad exponencial ajustada

El gráfico resultante aparece en la Figura 5.4. Podemos ver a simple vista bastante parecido entre el histograma y la función de densidad del modelo ajustado.

5.1.5. Para la distribución Gamma

Vamos a tratar de ajustar ahora los mismos datos, relativos a los tiempos hasta el fallo de 300 componentes electrónicas, mediante una distribución Gamma.

5.1.5.1. Estimación de los parámetros

El código ahora es

```
fitdistr(tiempos.fallo$tiempos.fallo,"gamma")
```

El resultado que aparece es el siguiente:

```
shape rate
0.96639591 0.47532653
(0.06926486) (0.04404461)
```

Podemos comentar algo importante en estos parámetros en relación al ajuste anterior, el que realizamos con la distribución exponencial: fijémonos en que el parámetro a o parámetro *de forma* (*shape*), se estima en 0.966. Recordemos que una distribución Gamma con $a = 1$ es, en realidad, una distribución exponencial. Lo que nos está diciendo esta estimación es que la distribución Gamma resultante es casi una distribución exponencial. De hecho, fijémonos también en que la estimación del parámetro de razón, λ , es 0.475, cuando en el caso de la exponencial era 0.492. Son valores muy

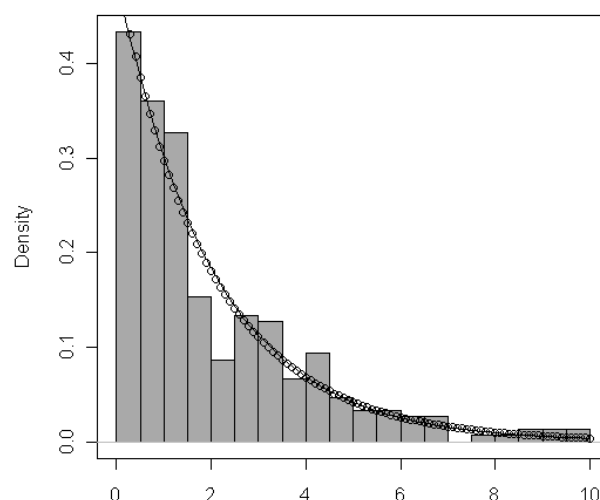


Figura 5.5: Comparación del histograma de los tiempos de fallo con las densidades ajustadas exponencial y Gamma

parecidos. Estas estimaciones van a provocar que la densidad exponencial antes ajustada y la de la *Gamma* (0.966, 0.475) sean muy parecidas.

5.1.5.2. Comparación del modelo ajustado con los datos muestrales

Para poner esto de manifiesto, vamos a añadir al gráfico anterior, el que aparece en la Figura 5.4, la representación de la densidad de la *Gamma* (0.966, 0.475). Para ello introducimos el siguiente código:

```
lines(0:100/10,dgamma(0:100/10,0.96639591,0.47532653),"p")
```

Hemos añadido “p” a la función *lines* para que dibuje la densidad Gamma con puntos ya que, de otra forma es casi imposible distinguirla de la densidad exponencial que antes habíamos dibujado. El resultado aparece en la Figura 5.5.

5.1.6. Para la distribución normal

Como ejemplo con el que vamos a aprender a realizar la estimación, consideremos los datos relativos al tiempo en segundos que tarda cada uno de los operarios cualificados de un colectivo de 158 operarios en ejecutar una tarea sencilla dentro de un proceso industrial. Los datos aparecerán resumidos en su histograma en la Figura 5.6, junto al ajuste mediante una distribución normal. Se encuentran en el fichero *operarios.tarea.rda*; el nombre del conjunto de datos es *muestra* y el nombre de la variable es *tiempos*. Lo primero que debemos hacer, obviamente, es cargar el conjunto de datos.

5.1.6.1. Estimación de los parámetros

Para obtener la estimación de los parámetros μ y σ de la distribución normal por máxima verosimilitud podemos usar el siguiente código:

```
fitdistr(muestra$tiempo,'normal')
```

El resultado es el siguiente:

```
mean sd
42.3531760 3.6633968
( 0.2914442) ( 0.2060822)
```

Es decir, el ajuste mediante el método de máxima verosimilitud viene dado por una $N(42.353, 3.663)$. Podemos ver, además, las estimaciones de los errores estándares de estas estimaciones, 0.291 y 0.206.

5.1.6.2. Comparación del modelo ajustado con los datos muestrales

De nuevo vamos a representar en una misma gráfica el histograma asociado a los datos y la densidad del modelo ajustado. Eso nos permitirá valorar, al menos de forma visual, la precisión del ajuste:

1. En primer lugar, obtenemos con el histograma con la escala de densidad. Como tenemos 158 datos en la muestra, un número adecuado de intervalos puede ser 13.
2. Sin cerrar el gráfico del histograma, y observando que el eje X va aproximadamente de 35 a 55, ejecutamos el siguiente código⁴:

```
lines(350:550/10,dnorm(350:550/10,42.3531760,3.6633968))
```

El resultado es el que aparece en la Figura 5.6. Podemos estar tentados a decir que el ajuste es muy bueno, pero observemos que el histograma es ligeramente asimétrico a la derecha, mientras que la densidad sólo puede ser totalmente simétrica. Por eso tenemos la densidad ligeramente por encima del histograma en la cola de la izquierda y ligeramente por debajo a la derecha. En la práctica, eso quiere decir que el modelo puede estar sobrevalorando la cola de la izquierda e infravalorando la de la derecha. No olvidemos que la cola de la derecha corresponde con los operarios que más tardan en ejecutar la tarea.

⁴En él, `350:550` proporciona todos los valores enteros entre 350 y 550. Al dividir éstos por 10 obtenemos valores entre 35 y 55 de 0.1 en 0.1.

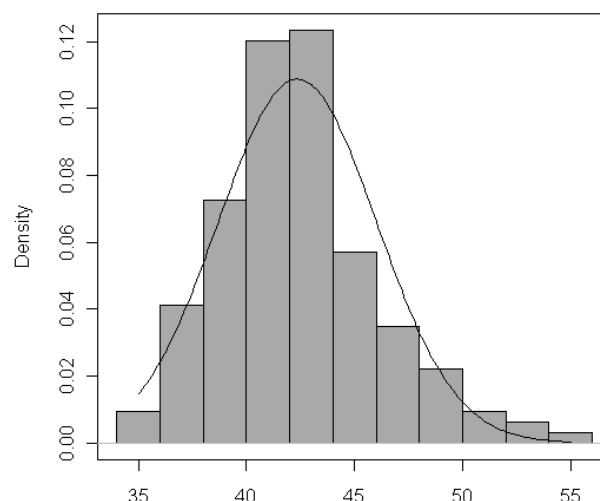


Figura 5.6: Comparación del histograma de los tiempos de realización de la tarea con la densidad normal ajustada

5.2. Estimación por intervalos de confianza

Los intervalos de confianza guardan una relación biunívoca con los contrastes de hipótesis estadísticas, que estudiaremos en el siguiente capítulo. Ese es el motivo por el que R Commander no facilita un menú específico para la construcción de intervalos de confianza, sino que éstos son proporcionados como parte de los resultados vinculados a los contrastes de hipótesis. Por este motivo, no vamos a comentar aquí cómo se realizan los intervalos de confianza mediante R Commander, sino en el capítulo siguiente.

Sin embargo, es casi trivial la posibilidad de usar R como una simple calculadora para aplicar las fórmulas de los intervalos de confianza que hemos visto en clase, y eso es lo que vamos a hacer aquí.

A lo largo de toda esta sección vamos a plasmar con ejemplos cómo podemos obtener algunos intervalos de confianza bilaterales. Lo vamos a hacer exclusivamente con código y sin la ayuda de ningún paquete adicional, para lo cual es bueno recordar la sintaxis de algunas funciones y describir la de otras que aún no hemos visto:

- `mean(datos)` devuelve la media muestral de *datos*.
- `sd(datos)` devuelve la desviación típica muestral de *datos*.
- `sqrt(x)` devuelve \sqrt{x} .
- `qnorm(α)` devuelve z_α .
- `qt(α, v)` devuelve $t_{\alpha, v}$.

- $qchisq(\alpha, v)$ devuelve $\chi^2_{\alpha, v}$.

5.2.1. De la media de una distribución normal con varianza desconocida

Recordemos que si notamos x_1, \dots, x_N a una muestra de una distribución $N(\mu, \sigma)$, ambas desconocidas, un intervalo de confianza con nivel de significación α para μ viene dado por

$$\left(\bar{x} \mp t_{1-\frac{\alpha}{2}; n-1} s_{n-1} / \sqrt{N} \right).$$

Vamos a considerar de nuevo el ejemplo del tiempo de los tiempos de ejecución de una tarea por parte de trabajadores cualificados, suponiendo que esta variable tenga una distribución normal. Teníamos una muestra de 158 operarios y sus tiempos. El archivo es *operarios.tarea.rda*, el nombre del conjunto de datos es *muestra* y el nombre de la variable es *tiempos*. Queremos un intervalo de confianza para el tiempo medio de ejecución de la tarea, con $\alpha = 0.05$. El código es el siguiente:

- `ci<-mean(muestra$tiempos)-qt(0.975,157)*sd(muestra$tiempos)/sqrt(158)`. Esto calcula la cota inferior del intervalo, llamándola *ci*.
- `cs<-mean(muestra$tiempos)+qt(0.975,157)*sd(muestra$tiempos)/sqrt(158)`. Esto calcula la cota superior del intervalo, llamándola *cs*.
- `c(ci,cs)`. Eso aglutina en un vector la cota inferior y la cota superior, haciéndolas aparecer en la ventana de resultados.

El resultado es 41.77569 42.93066. Es decir, la probabilidad de que el intervalo (41.776, 42.931) contenga a la media de los tiempos de ejecución de la tarea es del 95 %.

5.2.2. De la media de una distribución cualquiera, con muestras grandes

Antes hemos tratado de obtener un modelo para la variable tiempos hasta el fallo de 300 componentes electrónicas. Probamos la distribución exponencial y la Gamma.

Ahora nos da igual si estos modelos son o no adecuados; lo que queremos hacer es obtener un intervalo de confianza para la media μ desconocida, de la variable.

Sabemos que, visto el histograma, no es admisible pensar que la variable sigue una distribución normal, pero tenemos 300 datos, suficientes para poder aplicar el resultado basado en el teorema central del límite que determina que un intervalo para μ a un nivel de confianza α es

$$\left(\bar{x} \mp z_{1-\frac{\alpha}{2}} s_{n-1} / \sqrt{N} \right),$$

siendo N el tamaño de la muestra. El código es el siguiente:

- `ci<-mean(tiempos.fallo$tiempos.fallo)`
`-qnorm(0.975)*sd(tiempos.fallo$tiempos.fallo)/sqrt(300)`. Esto calcula la cota inferior del intervalo, llamándola *ci*.
- `cs<-mean(tiempos.fallo$tiempos.fallo)`
`+qnorm(0.975)*sd(tiempos.fallo$tiempos.fallo)/sqrt(300)`. Esto calcula la cota superior del intervalo, llamándola *cs*.
- `c(ci,cs)`. Eso aglutina en un vector la cota inferior y la cota superior, haciéndolas aparecer en la ventana de resultados.

El resultado es 1.806714 2.259533. Es decir, la probabilidad de que el intervalo (1.807, 2.260) contenga a la media del tiempo de fallo de las componentes electrónicas es del 95 %.

5.2.3. De una proporción

Supongamos que una empresa envasadora de nueces comprueba que en una muestra de 300 nueces, 21 están vacías. La marca quiere proporcionar un intervalo de confianza al 95 % para el porcentaje de nueces vacías en las bolsas que saca al mercado. Vamos a calcularlo.

El intervalo, para un nivel α viene dado por

$$\left(\hat{p} \mp z_{1-\frac{\alpha}{2}} \sqrt{\frac{\hat{p}(1-\hat{p})}{N}} \right)$$

donde \hat{p} es la proporción muestral (en nuestro caso, $\frac{21}{300}$) y N es el tamaño de la muestra (en nuestro caso, 300). El código para obtener el intervalo es el siguiente:

- `ci<-21/300-qnorm(0.975)*sqrt((21/300)*(1-21/300)/300)`. Para la cota inferior.
- `cs<-21/300+qnorm(0.975)*sqrt((21/300)*(1-21/300)/300)`. Para la cota superior.
- `c(ci,cs)`. Para que aparezcan en la ventana de resultados.

El resultado es 0.04112793 0.09887207, luego un intervalo de confianza al 95 % para el porcentaje de nueces vacías de la marca es (4.11 %, 9.89 %).

5.2.4. De la varianza de una distribución normal

Finalmente, vamos a obtener un intervalo de confianza para la varianza de la variable de tiempo de ejecución de la tarea por parte de operarios cualificados.

En clase hemos visto que dicho intervalo viene dado por

$$\left(\frac{\sum_{i=1}^N (x_i - \bar{x})^2}{\chi_{1-\frac{\alpha}{2}; N-1}^2}, \frac{\sum_{i=1}^N (x_i - \bar{x})^2}{\chi_{\frac{\alpha}{2}; N-1}^2} \right).$$

Teniendo en cuenta que $s_{N-1}^2 = \frac{\sum_{i=1}^N (x_i - \bar{x})^2}{N-1}$, otra forma de expresarlo es

$$\left(\frac{(N-1) s_{N-1}^2}{\chi_{1-\frac{\alpha}{2}; N-1}^2}, \frac{(N-1) s_{N-1}^2}{\chi_{\frac{\alpha}{2}; N-1}^2} \right).$$

El código es el siguiente:

- `ci<-157*sd(muestra$tiempos)/qchisq(0.975,157)`. Para la cota inferior.
- `cs<-157*sd(muestra$tiempos)/qchisq(0.025,157)`. Para la cota superior.
- `c(ci,cs)`. Para que aparezcan en la ventana de resultados.

El resultado es 2.980518 4.645565, lo que quiere decir que la probabilidad de que el intervalo (2.981, 4.646) contenga a la varianza del tiempo de ejecución de la tarea es del 95 %.

5.3. Ejercicios

1. El fichero *muestra1.rda* contiene un conjunto de datos de valores de una variable discreta. Se pide:
 - a) Ajustar una distribución binomial negativa a esos datos mediante el método de máxima verosimilitud.
 - b) Representar gráficamente el diagrama de barras de los datos junto con la función masa de la distribución del ajuste.
 - c) ¿Crees que la distribución resultante es un buen ajuste para los datos? ¿Por qué? ¿Cuál crees que es el problema?
2. El fichero *muestra2.rda* contiene un conjunto de datos de valores de una variable discreta. Se pide:

- a) Ajustar una distribución de Poisson a esos datos mediante el método de máxima verosimilitud.
 - b) Representar gráficamente el diagrama de barras de los datos junto con la función masa de la distribución del ajuste.
 - c) ¿Crees que la distribución resultante es un buen ajuste para los datos? ¿Por qué?
3. El fichero *muestra3.rda* contiene un conjunto de datos de valores de una variable discreta. Se pide:
 - a) Ajustar una distribución geométrica y una distribución binomial negativa a esos datos mediante el método de máxima verosimilitud.
 - b) Representar gráficamente el diagrama de barras de los datos junto con las dos funciones masa de las distribuciones ajustadas.
 - c) ¿Cuál de las dos distribuciones crees que ajusta mejor los datos?
4. El fichero *muestra4.rda* contiene un conjunto de datos de valores de una variable continua. Se pide:
 - a) Ajustar una distribución exponencial y una distribución Gamma a esos datos mediante el método de máxima verosimilitud.
 - b) Representar gráficamente el histograma de los datos junto con las dos funciones de densidad de las distribuciones ajustadas.
 - c) ¿Cuál de las dos distribuciones crees que ajusta mejor los datos?
5. El fichero *muestra5.rda* contiene un conjunto de datos de valores de una variable continua. Se pide:
 - a) Ajustar una distribución normal a esos datos mediante el método de máxima verosimilitud.
 - b) Representar gráficamente el histograma de los datos junto con la función de densidad de la distribución del ajuste.
 - c) ¿Crees que la distribución resultante es un buen ajuste para los datos? ¿Por qué?
6. Obtener intervalos de confianza al 95 % para la media y la varianza de la distribución de los datos del ejercicio anterior, suponiendo que siguen una distribución normal.

7. Obtener un intervalo de confianza al 95 % para la media de la distribución de los datos que aparecen en el fichero *muestra4.rda*.
8. Una empresa realiza un sondeo telefónico para estimar el porcentaje de ventas que tendrá al sacar un nuevo producto de venta telefónica al mercado. 65 de las 1000 llamadas realizadas concluyen con la venta del producto. Proporcionar, con esos datos, un intervalo de confianza al 95 % para la proporción de ventas que tendrá la empresa de ese producto cuando lo saque al mercado.

Capítulo 6

Contraste de hipótesis paramétricas

Objetivos:

- Realizar contrastes de hipótesis sobre la media de una y dos poblaciones.
- Realizar contrastes de hipótesis sobre la proporción en una y dos poblaciones.
- Realizar contrastes de hipótesis sobre varianzas en dos poblaciones.
- Realizar contrastes sobre la media en más de dos poblaciones (ANOVA).
- Obtener intervalos de confianza mediante R Commander.

6.1. Introducción

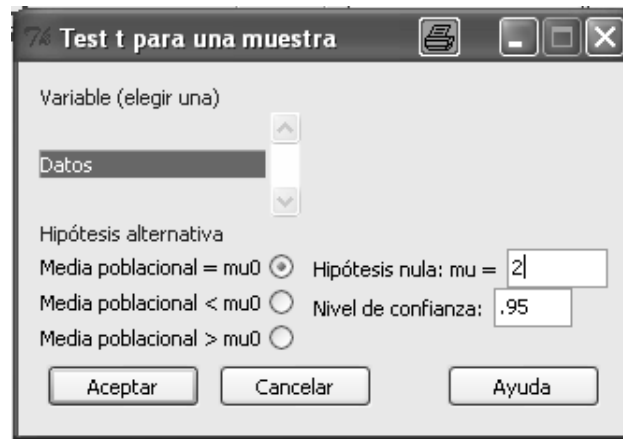
A lo largo de este tema vamos a abordar la realización de contrastes de hipótesis paramétricas a través de diversos ejemplos. Como siempre, vamos a comentar las posibilidades de ejecutar dichos tests directamente a través de la consola de R y mediante las opciones de R Commander.

6.2. Contrastes sobre medias

6.2.1. Contraste sobre la media de una población

Consideremos el siguiente enunciado:

Un ingeniero industrial ha diseñado una máquina que envasa bolsas de cebollas de dos kilos. Sin embargo, debido a diversas razones, como los diferentes pesos de las cebollas, problemas en el llenado, etc. es consciente de que el peso final de la bolsa de cebollas no será exactamente de dos kilos, sino que se producirán variaciones aleatorias con

Figura 6.1: Test t para una muestra

respecto a esta cantidad. Para comprobar si la máquina está bien calibrada, toma una muestra de 45 bolsas llenas de cebollas y contabiliza su peso. Con esta información, ¿tiene razones el ingeniero para pensar que la máquina está mal calibrada? (Utilícese un nivel de significación del 5 %).

Los datos se encuentran en el fichero *cebollas.txt*. Vamos a empezar planteando el problema y posteriormente veremos cómo resolverlo.

Fijémonos que nos piden claramente que confirmemos una afirmación: *la máquina está mal calibrada*. Eso obliga a plasmar dicha afirmación en la hipótesis alternativa, que es la única a la que podemos asignar la carga del nivel de confianza. Por lo tanto, si notamos μ al peso medio de las bolsas, tenemos que queremos contrastar $H_0 : \mu = 2$ frente a $H_1 : \mu \neq 2$.

Dado que el tamaño muestral es generoso (45, superior a 30), no necesitamos la hipótesis de normalidad de los datos. Dicho esto, vemos que se trata de un contraste sobre la media de una distribución normal, contraste bilateral.

6.2.1.1. Resolución del problema mediante R Commander

Debemos importar los datos desde el fichero *cebollas.txt* para manejar la variable en cuestión.

A continuación elegimos la opción del menú *Estadísticos* \rightarrow *Medias* \rightarrow *Test t para una muestra*. Esta opción abrirá la ventana que aparece en la Figura 6.1. Fijémonos con detalle en ella:

- Nos pide en primer lugar que elijamos una (sólo una) variable, que debe ser aquella cuya media estemos analizando.
- Nos pide que indiquemos cuál es la hipótesis alternativa. En nuestro caso hemos elegido la opción de un test bilateral.

- Nos pide que especifiquemos el valor del valor hipotético con el que estamos comparando la media, en nuestro caso, 2.
- Nos pide, por último, que especifiquemos un nivel de confianza. En realidad este nivel de confianza no lo es para el contraste, que se resolverá a través del p-valor, sino para el intervalo de confianza asociado al problema. El enunciado no dice nada, por lo que ponemos la opción habitual del 95 %.

El resultado es el siguiente:

```
One Sample t-test
data: Datos$Datos
t = -1.8415, df = 44, p-value = 0.0723
alternative hypothesis: true mean is not equal to 2
95 percent confidence interval:
 1.994974  2.000227
sample estimates:
mean of x
 1.9976
```

Analicemos el resultado con detalle:

- En primer lugar, nos recuerda que estamos analizando la variable `Datos$Datos`.
- A continuación nos informa del valor del estadístico de contraste ($t = -1.8415$), de los grados de libertad ($df = 44$) y del p-valor ($p\text{-value} = 0.0723$). Ya podemos, por tanto, concluir:
Dado que el p-valor no es inferior al 5 %, no tenemos suficientes evidencias en los datos para rechazar la hipótesis nula ($\mu = 2$) en favor de la alternativa ($\mu \neq 2$), es decir, con los datos de la muestra no tenemos suficientes evidencias de que el peso medio de las bolsas sea distinto de 2.
- Nos recuerda cuál era la hipótesis nula que habíamos planteado: `alternative hypothesis: true mean is not equal to 2`.
- A continuación proporciona un intervalo de confianza unilateral a la derecha, con un nivel de confianza del 95 %, para la media de la distribución normal que se le supone a los datos: `95 percent confidence interval: 1.994974 2.000227`. Lo que quiere decir el resultado es que

$$P[\mu \in (1.994974, 2.000227)] = 0.95.$$

La relación que guarda el intervalo de confianza con el contraste de hipótesis es la siguiente: fijémonos que el valor hipotético que hemos considerado para la media, 2, está dentro de este intervalo, luego éste es un valor *de confianza* para μ . Es otra forma de concluir que no hay datos que avalen que la media de la variable es significativamente distinta de 2, ya que éste es un valor bastante plausible para esta media. Si los datos fueran tales que el intervalo de confianza para μ dejara fuera al valor 2, tendríamos razones para pensar que el valor de μ es significativamente distinto de 2, pero no es el caso.

- Finalmente, proporciona los estadísticos muestrales utilizados, en este caso, la media muestral:
`sample estimates: mean of x 1.9976.`

6.2.1.2. Consideraciones finales

Por lo tanto, y a modo de conclusión, podemos decir que no hay evidencias de que el peso medio de las bolsas de cebollas sea distinto de dos, lo que implicaría que la máquina está mal calibrada. Quizá deberíamos recomendar a los expertos que deseaban contrastar la hipótesis que aumenten el tamaño de la muestra, ya que los datos parecen apuntar en esa dirección (el p-valor sólo es ligeramente superior a 0.05), aunque no de forma suficientemente significativa.

6.2.2. Contraste para la diferencia de medias de poblaciones independientes

Nos vamos a centrar ahora en el siguiente enunciado:

Un ingeniero industrial ha sintetizado en el laboratorio una feromona con la que pretende luchar contra una plaga de insectos. La feromona se aplica en trampas donde caen los insectos masivamente. Hasta ahora se trabajaba introduciendo otro producto que se supone que atraía al insecto, por lo que el ingeniero desearía demostrar que su feromona sintetizada es más efectiva que dicho producto. Para probar si esto ocurre, prepara 100 trampas con el producto tradicional y 100 con su feromona y las distribuye, contabilizando el número de insectos atrapados en cada una de las 200 trampas. Con esos datos, ¿puede concluir el ingeniero que su feromona es más efectiva que el producto tradicional? (Utilícese un nivel de significación del 5 %).

Vamos a llamar μ_V a la media de las capturas con el viejo producto y μ_N a la media de las capturas con la nueva feromona. Lo que nos piden en el enunciado es que contrastemos la hipótesis nula $H_0 : \mu_V = \mu_N$ frente a la alternativa $H_1 : \mu_V < \mu_N$:

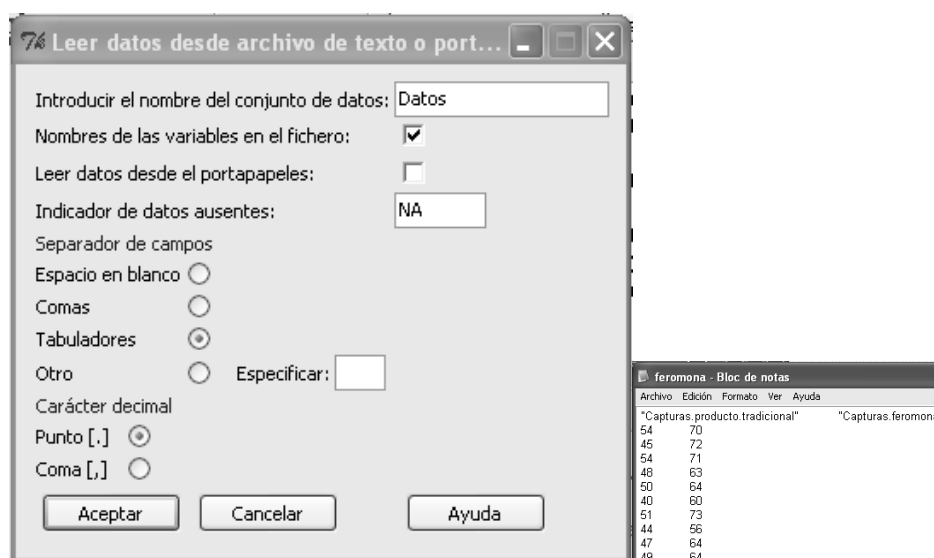


Figura 6.2: Importando los datos de los diámetros de los cojinetes

- En primer lugar, podemos suponer que las muestras son independientes. Nada hace pensar que los datos de la muestra bajo el producto antiguo hayan tenido nada que ver en la muestra bajo el producto nuevo ni al contrario.
- Con respecto al tamaño muestral, debe preocuparnos la hipótesis de normalidad: recordemos que si el tamaño de la muestras es pequeño, éstas deberían proceder de una distribución normal, pero no es el caso: ambas muestras tienen tamaños superiores a 30.
- Finalmente, deberemos plantearnos si podemos suponer o no que las varianzas son iguales.

6.2.2.1. Resolución mediante R Commander

Lo primero que tenemos que hacer para importar los datos, que se encuentran en un fichero de tipo texto es ver cómo están almacenados: si lo abrimos, por ejemplo, con el bloc de notas, vemos que están separados por tabulaciones y que los nombres de las variables están en la primera fila. La Figura 6.2 a la izquierda muestra la ventana con la que importamos los datos, mientras que la parte de la derecha muestra cómo se ven con el bloc de notas.

Fijémonos que los datos de las dos muestras aparecen en dos columnas paralelas. Esa es una forma no demasiado correcta de especificarlas, ya que parece que cada dato de una de las muestras está relacionado con otro dato de la otra muestra y, en realidad, las muestras son independientes (de hecho podrían tener distinto tamaño muestral).

Por este motivo, tenemos que preparar los datos para que R Commander entienda que se trata de dos muestras independientes. Lo que tenemos que hacer es juntar o apilar las dos muestras en

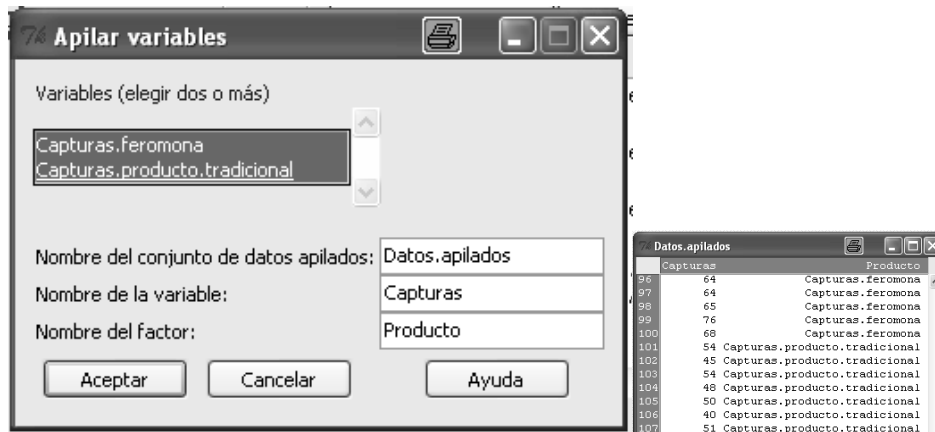


Figura 6.3: Apilando los datos de las dos muestras

una sola columna, indicando en una segunda columna si el dato es de una muestra u otra.

Esta operación se realiza mediante la opción *Datos* → *Conjunto de datos activo* → *Apilar variables del conjunto de datos activo*. Lo que tenemos que indicar en esta ventana (Figura 6.3 a la derecha) es cuáles son las variables que queremos apilar, el nombre del nuevo conjunto de datos, el nombre de la nueva variable y el nombre del factor que separa los datos de las dos muestras. En la parte de la derecha de la Figura 6.3 aparecen los datos tal y como quedan tras apilarlos.

Observemos que, en efecto, ha creado una primera columna con todos los datos y una segunda columna con un factor que especifica cuáles de ellos son del producto viejo y cuáles del nuevo.

Hay un último aspecto muy importante: los datos son ordenados según el factor, y éste ordena sus categorías por orden alfabético. Eso en nuestro ejemplo hace que aparezcan primero los datos según el nuevo producto y después según el viejo.

Ahora ya tenemos los datos preparados para ser analizados. Elegimos la opción *Estadísticos* → *Medias* → *Test t para muestras independientes*. Esta opción abre la ventana de entradas que aparece en la Figura 6.4. En ella podeis ver que tenemos que especificar el factor que separa las dos muestras, la variable que estamos analizando, la hipótesis alternativa, el nivel de confianza requerido y si podemos suponer varianzas iguales. En nuestro caso:

- El factor es el único que aparece, que hemos llamado *Producto*.
- La variable es la única numérica del conjunto de datos, que hemos llamado *Capturas*.
- Es un test unilateral: ¿cómo sabemos cuál es la muestra 1 y cuál la muestra 2? Lo sabemos porque el factor se ordena alfabéticamente, luego las capturas con la feromona es la muestra 1 y con el viejo producto la muestra 2. Como queremos contrastar $H_1 : \mu_N > \mu_V$, señalamos *Diferencia* > 0.



Figura 6.4: Test t para muestras independientes. Ventana de entradas

- El nivel de confianza exigido es del 95 %, que va a ser el que consideremos para el intervalo de confianza asociado.
- No tenemos razones que avalen que las varianzas deban ser consideradas iguales.

El resultado que aparece en la ventana de resultados es el siguiente:

```
Welch Two Sample t-test
data: Capturas by Producto
t = 20.4367, df = 197.952, p-value < 2.2e-16
alternative hypothesis: true difference in means
is greater than 0
95 percent confidence interval:
14.25580 Inf
sample estimates:
mean in group Capturas.feromona
64.94
mean in group Capturas.producto.tradicional
49.43
```

Vamos a analizarlo punto por punto:

- Especifica que se trata de un test t para la variable *Capturas* separada por el factor *Producto*.
- Proporciona el valor del estadístico de contraste ($t = 20.4367$), los grados de libertad ($df = 197.952$), y el p-valor ($p\text{-value} < 2.2e-16$). Dado que es inferior a 0.05, ya podemos concluir que tenemos evidencias en los datos para afirmar con un 95 % de confianza que la media de las capturas con la feromona es superior a la de las capturas con el viejo producto.

- Explicita cuál es nuestra hipótesis alternativa.
- Proporciona un intervalo de confianza para la diferencia de las medias. En este caso, la probabilidad de que el intervalo (14.25580 Inf) contenga a la diferencia de las medias es del 95 %. El cero no es, por tanto, un valor bastante plausible y por ello hemos rechazado la hipótesis nula en favor de la alternativa.
- Proporciona las dos medias muestrales.

6.2.2.2. Consideraciones finales

En resumen, hemos concluido que podemos afirmar con un 95 % de confianza que la feromona es más efectiva que el viejo producto al aumentar significativamente el promedio de capturas.

6.2.3. Contraste para la diferencia de medias de poblaciones apareadas

El problema que vamos a utilizar a modo de ejemplo es el siguiente:

El encargado de formación de una empresa ha diseñado un curso de especialización con el que pretende mejorar el rendimiento de los trabajadores que realizan una determinada tarea. La mejora consistiría en que tardaran menos tiempo en realizar la tarea. Para comprobar la eficacia del curso antes de implantarlo en todos los trabajadores que realizan esta tarea, elige al azar una muestra de 35 trabajadores y para cada uno de ellos contabiliza el tiempo medio (en segundos) que tarda en ejecutar la tarea¹ en un turno, antes y después de recibir el curso de especialización.

Basándose en los datos de esa muestra, ¿puede concluir el encargado de formación que el curso es efectivo? (Utilícese un nivel de significación del 5 %).

Los datos se encuentran en el fichero *curso.txt*.

Vamos a notar por μ_{DC} al promedio de los tiempos medios que tardan los trabajadores en realizar la tarea después del curso y por μ_{AC} al mismo promedio antes del curso. Nos piden contrastar $H_0 : \mu_{DC} = \mu_{AC}$ frente a $H_1 : \mu_{DC} < \mu_{AC}$.

6.2.3.1. Resolución del problema mediante R Commander

En primer lugar debemos cargar los datos. Debemos realizar un contraste de igualdad de medias en poblaciones apareadas. En este caso el conjunto de datos tiene exactamente la forma que R

¹Un mismo trabajador realiza la tarea muchas veces a lo largo de su turno, por lo que contabiliza el tiempo medio de ejecución a lo largo de un turno.

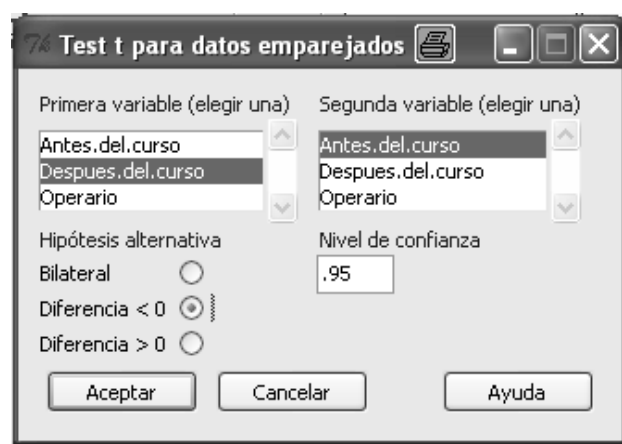


Figura 6.5: Prueba de igualdad de medias en poblaciones apareadas

Commander necesita para ello, ya que tenemos dos variables, correspondientes a los tiempos medios antes y después del curso.

Elegimos, por tanto, la opción *Estadísticos* → *Medias* → *Test t para datos relacionados*, lo cual abrirá una ventana de entradas como la de la Figura 6.5. En ella ya he marcado las opciones requeridas por nuestro análisis. El resultado es el siguiente:

```
Paired t-test
data: Datos$Despues.del.curso and Datos$Antes.del.curso
t = -0.1221, df = 34, p-value = 0.4518
alternative hypothesis: true difference in means is less than 0
95 percent confidence interval:
-Inf 0.002225738
sample estimates:
mean of the differences
-0.0001731764
```

Vamos a analizar los resultados con detalle:

- En las dos primeras líneas se nos informa que estamos realizando un test t para muestras apareadas sobre los datos relativos a las variables `Despues.del.curso` y `Antes.del.curso` del conjunto de datos `Datos`. A propósito de ello, ¿no hemos dicho nada acerca de la hipótesis requerida en dicho test! Dado que la muestra es suficientemente amplia, podemos considerar que la diferencia media entre las dos variables sigue una distribución normal.
- En la siguiente línea aparece el valor del estadístico de contraste ($t = -0.1221$), de los grados de libertad ($df = 34$) y el p-valor ($p\text{-value} = 0.4518$). Visto el valor de éste podemos concluir

que no existen evidencias en los datos de la muestra de que el curso disminuya el promedio que tardan los operarios en realizar la tarea. Observad que el p-valor es muy alto, luego las diferencias detectadas son mínimas, en absoluto significativas.

- A continuación aparece el tipo de hipótesis alternativa que hemos elegido.
- Posteriormente aparece un intervalo de confianza al 95 % para la diferencia de las medias. El hecho de que el cero sea un valor posible de dicho intervalo es otra forma de ver que no podemos rechazar la hipótesis nula en favor de la alternativa.
- Finalmente aparece el valor muestral de la diferencia de las medias.

6.2.3.2. Comentarios finales

En resumen, los datos no muestran el más mínimo indicio de que el curso disminuya el promedio que tardan los operarios en realizar la tarea.

6.2.4. Contrastes para medias mediante código. Función *t.test*

En los apartados anteriores hemos visto cómo resolver problemas que involucran a la media de una población comparándola con un valor hipotético o a la media de dos poblaciones (independientes o apareadas), comparándolas entre sí. Lo que tienen en común estas pruebas es que todas ellas se basan en un estadístico de contraste que sigue una distribución *t* de Student.

Por esta razón, la resolución de ese tipo de problemas mediante código en la consola de R se realiza mediante la misma función, la función *t.test()*. Su sintaxis básica es la siguiente:

```
t.test(x, y = NULL, alternative = c("two.sided", "less", "greater"),  
mu = 0, paired = FALSE, var.equal = FALSE, conf.level = 0.95)
```

- *x* es un vector de datos correspondiente a una de las muestras o a la única muestra del problema. Si estamos haciendo un test sobre la media de una población, *x* contendrá la única muestra; si estamos realizando un test de comparación de medias, *x* será la primera de las dos muestras.
- *y* correspondería con la segunda muestra en un test de comparación de medias. Si no es el caso y estamos en un test sobre una sola población, simplemente no se incluye.
- *alternative* especifica la dirección de la hipótesis alternativa. Como puede verse, tiene 3 posibles valores, "two.sided" (bilateral), "less" (unilateral a la izquierda) y "greater" (unilateral a la derecha).

- *mu* es el valor hipotético con el que se compara la media o la diferencia de medias en el contraste.
- *paired* especifica si las dos muestras *x* e *y*, en caso de que aparezcan, son apareadas o no.
- En el caso en el que aparecen dos muestras, *var.equal* especifica si podemos suponer varianzas iguales o no.
- *conf.level* es el nivel de confianza de los intervalos que se mostrarán asociados al test.

En el caso del problema descrito en el apartado 6.2.1, debemos importar los datos, que se encuentran en el fichero *cebollas.txt*, y especificar cómo especificaríamos el test. Los resultados son exactamente los mismos que comentamos como salidas de R Commander:

```
Datos<-read.table(cebollas.txt'',header=TRUE)
t.test(Datos$Datos,alternative='two.sided',mu=2)
```

De igual forma, para el problema que aparece en el apartado 6.2.2, veamos el código que proporciona las mismas salidas que R Commander:

```
Datos<-read.table('feromona.txt'',header=TRUE)
t.test(x=Datos$Capturas.feromona,
y=Datos$Capturas.producto.tradicional,
alternative='greater',mu=0)
```

Finalmente, el problema del apartado 6.2.3 es de comparación de medias en poblaciones apareadas, por lo que el código sería el siguiente:

```
Datos<-read.table('curso.txt'',header=TRUE)
t.test(x=Datos$Antes.del.curso,
y=Datos$Despues.del.curso,alternative='less',
mu=0,paired=TRUE)
```

6.3. Contraste para la proporción en una población

R realiza este tipo de contrastes de dos formas: mediante una prueba tipo χ^2 o mediante una prueba binomial exacta.

En el primero de los casos, el de las pruebas tipo χ^2 lo que se hace es comparar las frecuencias de casos favorables en la muestra de los datos (frecuencias observadas, O_i) con la muestra de casos

favorables que habría en una muestra con el mismo número de datos si la hipótesis nula fuera cierta (frecuencias esperadas, E_i). El estadístico sería

$$\chi^2 = \sum_i \frac{(O_i - E_i)^2}{E_i},$$

que, bajo el supuesto de que ninguna frecuencia esperada E_i es inferior a 5, se distribuye según una distribución χ^2 . En el caso de que alguna frecuencia esperada sea inferior a 5 se suele utilizar la corrección por continuidad de Yates, en la que el estadístico es

$$\chi^2 = \sum_i \frac{(|O_i - E_i| - 0.5)^2}{E_i}.$$

La prueba binomial exacta parte del hecho de que, si la hipótesis nula fuera cierta, la distribución del número de casos favorables en la muestra sería binomial. Valorando el número observado de casos favorables dentro de la distribución binomial que se daría bajo la hipótesis nula, se obtiene el p-valor de la prueba. De todas formas, por brevedad, sólo vamos a comentar los tests basados en pruebas χ^2 .

6.3.1. Enunciado y planteamiento del problema

El artículo “*Refinement of Gravimetric Geoid Using GPS and Leveling Data*” (W. Thurston, en Journal of Surveying Engineering, 2000:27-56) presenta un método para medir las alturas ortométricas por encima del nivel del mar. Para una muestra de 1225 puntos de partida, 926 dieron resultados que están dentro del espíritu de la clase C nivelando los límites de tolerancia. ¿Se puede concluir que este método produce resultados dentro de los límites de tolerancia más del 75 % de las veces?

Si notamos p a la proporción de resultados dentro de los límites de tolerancia, se nos está pidiendo que contrastemos $H_0 : p = 0.75$ frente a $H_1 : p > 0.75$.

6.3.2. Preparación de los datos y resolución del problema mediante R Commander

Antes de comenzar, conviene que carguemos todos los plugins de R Commander.

Tenemos dos formas de proporcionar los datos para realizar el test y dos formas de realizarlo:

1. Si tenemos los datos en un archivo, los cargamos y seleccionamos la opción del menú *Estadísticos* \rightarrow *Proporciones* \rightarrow *Test de proporciones para una muestra*.

Por ejemplo, los datos del problema están en el fichero *prop.rda*. Los cargamos y seleccionamos

la opción del menú. La ventana de entrada emergente aparece en la Figura 6.6 a la izquierda. En ella tenemos que especificar:

- a) El valor hipotético en la hipótesis nula. En nuestro caso 0.75.
 - b) El sentido de la hipótesis alternativa. En nuestro caso unilateral a la derecha.
 - c) Un nivel de confianza para el intervalo de confianza resultante.
 - d) El tipo de prueba. Dejamos la opción por defecto, que es la prueba χ^2 sin corrección por continuidad.
2. Si no tenemos los datos en un archivo basta con que sepamos cuál es el número de éxitos y fracasos en la muestra. En nuestro caso, 926 éxitos y 299 fracasos. Seleccionamos la opción del menú *Estadísticos* \rightarrow *Proporciones* \rightarrow *IPSUR Enter table for single-sample*. En la ventana emergente (Figura 6.6 a la derecha) tenemos que especificar:

- a) El número de éxitos (926) y fracasos (299).
- b) El valor hipotético en la hipótesis nula. En nuestro caso 0.75.
- c) El sentido de la hipótesis alternativa. En nuestro caso unilateral a la derecha.
- d) Un nivel de confianza para el intervalo de confianza resultante.
- e) El tipo de prueba. Dejamos la opción por defecto.

En ambos casos la ventana de resultados muestra lo mismo:

```
1-sample proportions test without continuity correction
data: rbind(.Table), null probability 0.75
X-squared = 0.2288, df = 1, p-value = 0.3162
alternative hypothesis: true p is greater than 0.75
95 percent confidence interval:
0.7351821 1.0000000
sample estimates:
p
0.7559184
```

Vamos a analizarlos:

- Especifica en primer lugar que se trata de un test para la proporción en una muestra.
- Especifica a continuación el valor hipotético en la hipótesis nula.

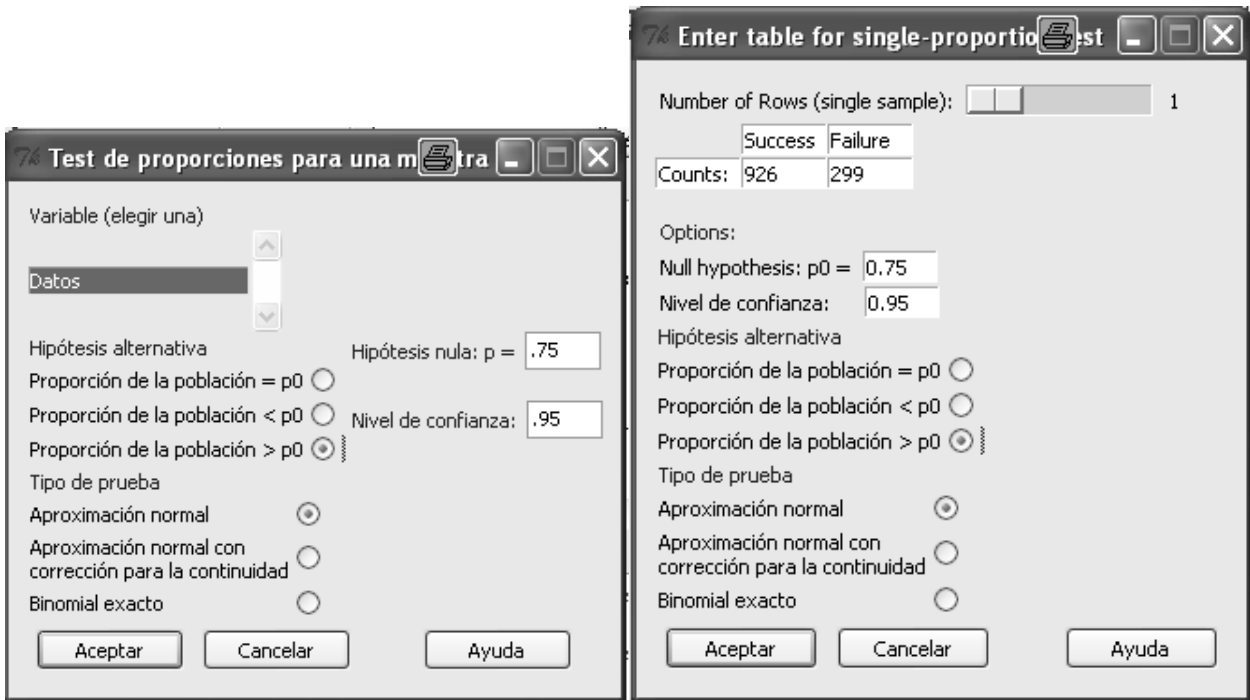


Figura 6.6: Test para una proporción

- Proporciona el valor del estadístico de contraste y, lo que más nos interesa, el p-valor.

En este caso el p-valor es bastante superior a 0.05, luego a la luz de estos datos no podemos rechazar la hipótesis nula en favor de la alternativa, es decir, no podemos concluir que el porcentaje de resultados dentro de los límites de tolerancia esté por encima del 75 %.

- A continuación recuerda la hipótesis alternativa.
- Facilita un intervalo de confianza (en este caso unilateral a la derecha) para la proporción.
- Finalmente, muestra la proporción muestral.

6.3.3. Resolución mediante código. La función *prop.test*

Vamos a partir del hecho de que conocemos el número de éxitos y fracasos en la muestra. Si no fuera así, sino que tenemos los datos en una hoja de datos, podemos rápidamente tabularla mediante la función *table()* a la que sólo hay que especificarle la hoja de datos a tabular y, si ésta tuviera más de una variable, cuál de ellas queremos tabular.

La sintaxis de la función *prop.test* es la siguiente. Dicha sintaxis también nos servirá para los contrastes de comparación de dos proporciones:

```
prop.test(x, n, p = NULL, alternative = c("two.sided", "less", "greater"),
conf.level = 0.95, correct = TRUE)
```

Vamos a comentar con detalle cada uno de los argumentos de la función:

- x puede especificar dos cosas. O bien simplemente el número de éxitos, o bien, mediante una matriz de dos columnas, el número de éxitos y de fracasos en cada muestra.
- n especifica el número de datos de la muestra en el caso en que x sea el número de éxitos, y es ignorado en el caso en que x proporcione también el número de fracasos.
- p es el vector de probabilidades de éxito bajo la hipótesis nula. Debe ser un vector de la misma dimensión que el número de elementos especificado en x .
- *alternative* especifica la dirección de la hipótesis alternativa, tomando los valores *"two.sided"*, *"greater"* o *"less"*.
- *conf.level* es el nivel de confianza de los intervalos que se muestran entre los resultados.
- *correct* especifica si se usa la corrección por continuidad de Yates. Obsérvese que la opción por defecto es que sí se use esta corrección.

El ejemplo anterior podemos ejecutarlo rápidamente de dos formas:

1. `prop.test(x=926,n=1225,p=0.75,alternative="greater",correct=FALSE)` o bien
2. `prop.test(x=cbind(926,299),p=0.75,alternative="greater",correct=FALSE)`. En esta línea, la función *cbind()* sirve para concatenar los valores que aparecen entre paréntesis como columnas de una matriz.

6.4. Contraste para la diferencia de proporciones

Las pruebas que R utiliza para este tipo de contrastes de nuevo se basan en el uso del estadístico χ^2 , comparando las frecuencias observadas en ambas muestras con las que aparecerían bajo la hipótesis nula. De hecho, la función que utiliza es la misma que en el caso de una muestra: *prop.test*.

6.4.1. Enunciado y planteamiento del problema

Vamos a trabajar sobre el siguiente enunciado:

A raíz de la alarma creada entre la opinión pública por la repercusión que tuvo el caso de un bloque de edificios con un transformador en su planta baja y donde un gran porcentaje de vecinos sufrió cáncer, se decide realizar un estudio para tratar de encontrar relación entre la cercanía de un transformador eléctrico y la incidencia del cáncer.

Para ello, se eligió una muestra aleatoria de edificios con transformadores en su planta baja durante un periodo de más de 10 años, contabilizando el número de habitantes en ellos, 2150, y todos los casos de cáncer detectados en los 5 últimos años, 37. Por su parte, se recolectó otra muestra aleatoria de control con edificios que no tuvieran ningún transformador eléctrico cercano, contabilizando también el número de personas, 2200, y el número de casos de cáncer en ellos en los últimos 5 años, 33. En ambas muestras, los expertos procuraron eliminar la posibilidad de ruidos, es decir, la presencia de otros factores que pudieran incidir en variar la incidencia del cáncer en alguna de las muestras. Los datos de ambas muestras se encuentran en el fichero correspondiente al ejercicio².

A la luz de los datos de este estudio, ¿podemos afirmar que la cercanía de un transformador eléctrico aumenta la proporción de casos de cáncer? (Utilícese un nivel de significación del 5 %)

Si llamamos p_{CT} a la proporción de casos de cáncer en los edificios con transformador cercano y p_{ST} a la proporción análoga en los edificios sin transformador, nos piden que contrastemos $H_0 : p_{CT} = p_{ST}$ frente a $H_1 : p_{CT} > p_{ST}$.

6.4.2. Resolución del problema mediante R Commander

Antes de comenzar hay que decir que, al igual que en el ejemplo anterior, podríamos tener los datos en un archivo y tratarlos directamente desde ahí³, mediante la opción *Estadísticos* → *Proporciones* → *Test de proporciones para dos muestras*. Sin embargo, tan sólo es necesario conocer el número de éxitos y fracasos en cada una de las dos muestras, utilizando la opción *Estadísticos* → *Proporciones* → *IPSUR Enter table for independent samples*. En la ventana emergente (Figura 6.7) tenemos que especificar:

- Número de éxitos y fracasos en la primera muestra. En nuestro caso, 37 y 2113.
- Número de éxitos y fracasos en la segunda muestra. En nuestro caso, 33 y 2167.
- Un nivel de significación para el intervalo de confianza.
- El sentido de la hipótesis alternativa. En nuestro caso, unilateral a la derecha.
- El tipo de test. Dejamos la opción por defecto.

²Aunque parezca obvio, quiero dejar claro que tanto el enunciado como los datos del ejercicio son ficticios. Con el enunciado sólo quiero describir someramente, de una forma muy esquemática, cómo podría realizarse un estudio de este tipo.

³Tendrían que estar en una única columna, con una segunda variable tipo factor que distinga a qué muestra pertenece cada dato. Recordemos la opción *Apilar* que ya hemos trabajado para eso.

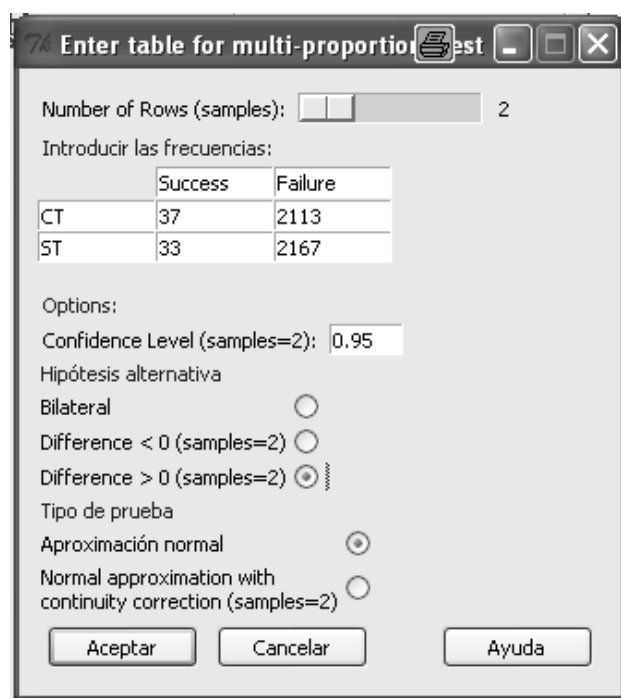


Figura 6.7: Test para la comparación de proporciones

Los resultados son los siguientes:

```
2-sample test for equality of proportions
without continuity correction
data: .Table
X-squared = 0.3352, df = 1, p-value = 0.2813
alternative hypothesis: greater
95 percent confidence interval:
-0.004071904 1.000000000
sample estimates:
prop 1 prop 2
0.01720930 0.01500000
```

Lo que realmente nos interesa es el p-valor, que aparece en la tercera línea, 0.2813, y que indica que no se puede rechazar la hipótesis nula en favor de la alternativa, es decir, no podemos, con los datos existentes, asegurar con un 95 % de confianza que la proporción de casos de cáncer sea superior en los edificios que tengan un transformador eléctrico en su planta baja.

6.4.3. Resolución mediante código

El contraste se realiza de nuevo mediante la función *prop.test*, pero previamente debemos comentar algo acerca de cómo introducir los datos.

Los éxitos y los fracasos de cada muestra deben ir en dos filas de una matriz de dos columnas, es decir, con una estructura como la siguiente:

Nº de éxitos de la muestra 1 (n_{11})	Nº de fracasos de la muestra 1 (n_{12})
Nº de éxitos de la muestra 2 (n_{21})	Nº de fracasos de la muestra 2 (n_{22})

Para crear una matriz así se utiliza la función *matrix*, a la que tenemos que especificarle mediante un vector los elementos de la matriz, las dimensiones de la matriz y el sentido en el que vienen especificados los elementos. En nuestro caso sería

- `matrix(c(n_{11} , n_{12} , n_{21} , n_{22}), 2, 2, byrow=TRUE)` o bien
- `matrix(c(n_{11} , n_{21} , n_{12} , n_{22}), 2, 2, byrow=FALSE)`.

En el ejemplo, sería de la siguiente forma:

```
tabla<-matrix(c(37,2113,33,2167),2,2,byrow=TRUE)
```

Finalmente, la aplicación de la función *prop.test* sería la siguiente:

```
prop.test(tabla, alternative='greater', correct=FALSE)
```

6.5. Contraste para la comparación de varianzas

6.5.1. Enunciado y planteamiento del problema

En una comercializadora de aceite de oliva se envasan botellas de 1 litro pero, como ocurre en cualquier proceso industrial, el volumen de llenado de cada botella no es exactamente de un litro, sino que se producen variaciones aleatorias en el volumen real con el que se llenan. El ingeniero responsable del control de calidad es consciente del problema que supone esta variabilidad en el volumen de llenado.

De cara a tratar de reducirla, la empresa se plantea cambiar la máquina de llenado, pero previamente quiere probarla para garantizarse que la inversión merece la pena. Para ello calibra la máquina para llenados de 1 litro y recopila los datos de llenado de 100 botellas. Un análisis exploratorio inicial no le hace pensar que estos datos se salgan de la hipótesis de normalidad. También realiza un muestreo de otras 100 botellas con la vieja máquina,

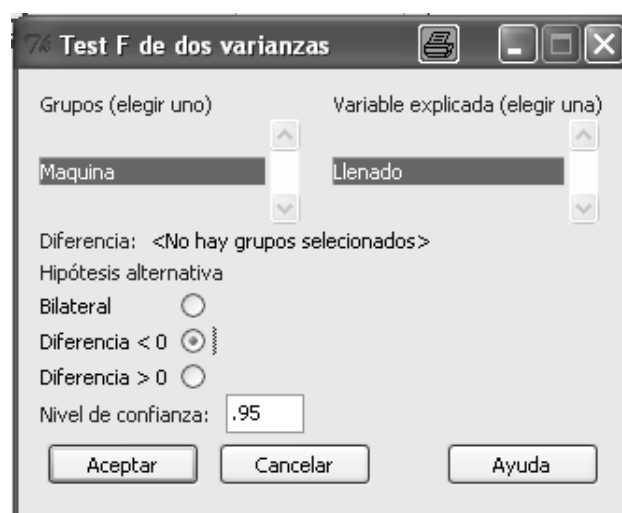


Figura 6.8: Test de igualdad de varianzas

en las mismas condiciones (horario, operarios que extraen las muestras, etc.) que las 100 botellas muestreadas con la nueva máquina. De nuevo no hay evidencias de que los datos se separen de la normalidad.

A la luz de esos datos, ¿puede concluir el ingeniero de control de calidad que la nueva máquina de llenado disminuye la variabilidad del volumen con el que se llenan las botellas? (Utilícese un nivel de significación del 5 %).

Los datos se encuentran en el fichero *aceite.txt*.

Lo que vamos a plantear para resolver el problema es un contraste de igualdad de varianzas (o de desviaciones típicas). Si notamos σ_V a la desviación típica del volumen de llenado bajo el viejo proceso y σ_N a la desviación típica del volumen de llenado bajo el nuevo proceso, se trata de contrastar $H_0 : \sigma_V = \sigma_N$ frente a $H_1 : \sigma_V > \sigma_N$.

De nuevo vamos a asumir desde el principio que los datos proceden de distribuciones normales.

6.5.2. Resolución del ejercicio mediante R Commander

Al igual que hicimos allí, necesitamos que los datos se encuentren apilados en una única variable y que una segunda variable tipo factor distinga de qué muestra procede cada dato. Eso ya lo hicimos anteriormente.

Elegimos la opción *Estadísticos* \rightarrow *Varianzas* \rightarrow *Test F para dos varianzas*. En la ventana de entradas hay que especificar:

- La variable que define los grupos (*Maquina* en nuestro caso) y la variable explicada (*Llenado*).

- La hipótesis alternativa. Aquí es importante que recordemos que el factor (*Maquina*) se ordena alfabéticamente, luego la primera muestra es la del proceso nuevo y la segunda la del viejo. Por tanto, dado que queremos la alternativa $H_1 : \sigma_V > \sigma_N$, tenemos que elegir la opción *Diferencias* < 0. Eso es porque *Diferencias* se refiere a la diferencia entre el primer grupo (nueva) y el segundo grupo (vieja).
- El nivel de confianza para el intervalo de confianza.

Los resultados son los siguientes:

```
F test to compare two variances
data: Llenado by Maquina
F = 1.1016, num df = 99, denom df = 99, p-value = 0.6844
alternative hypothesis: true ratio of variances is less than 1
95 percent confidence interval:
0.000000 1.535685
sample estimates:
ratio of variances
1.101590
```

En la tercera línea podemos ver que aparece el valor del estadístico F , los grados de libertad en el numerador y el denominador y el p-valor. Ese $p = 0.6844$ indica que no hay suficientes evidencias en los datos para concluir que el proceso nuevo disminuya significativamente la varianza de los diámetros. De hecho, observemos que las propias varianzas muestrales indican que la varianza de la nueva máquina es mayor que la de la vieja máquina.

6.5.3. Resolución mediante código. La función *var.test*

Esta función tiene una sintaxis muy parecida a las anteriores:

```
var.test(x, y, ratio = 1, alternative = c("two.sided", "less", "greater"),
conf.level = 0.95)
```

- x corresponde al vector de datos de la primera muestra.
- y es el vector de datos de la segunda muestra.
- *ratio* es el cociente hipotético con el que se compara. Habitualmente deseamos contrastar que las varianzas son distintas, o lo que es lo mismo, que su cociente es 1, así que la opción por defecto es precisamente *ratio*=1.

- *alternative* especifica la hipótesis alternativa: "two.sided" para $H_1 : \sigma_1 \neq \sigma_2$, "less" para $H_1 : \sigma_1 < \sigma_2$ y "greater" para $H_1 : \sigma_1 > \sigma_2$.
- *conf.level* es el nivel de confianza del intervalo para el cociente de varianzas que se muestra en las salidas.

Para la resolución del ejemplo anterior, tendríamos el siguiente código:

```
Datos<-read.table("aceite.txt",header=TRUE,sep="\t",dec=',')
var.test(Datos$Nueva.maquina,Datos$Vieja.maquina,
         alternative="less")
```

6.6. ANOVA de un factor

6.6.1. Enunciado y planteamiento del problema

Una compañía química recoge información sobre las concentraciones máximas por hora (en $\mu\text{g}/\text{m}^3$) de SO_2 para cuatro de sus plantas de energía. ¿Los resultados permiten concluir a la compañía que hay diferencias entre las concentraciones máximas por hora entre las cuatro plantas? (Utilícese un nivel de significación del 5 %).

Los datos se encuentran en el fichero *SO2.txt*. Se refieren a 4 lugares, y se nos pide valorar si se detectan diferencias (significativas) entre ellos. Asumiendo que se trate de datos procedentes de distribuciones normales con varianzas iguales, vamos a abordar el problema mediante una prueba ANOVA.

6.6.2. Resolución mediante R Commander

En primer lugar, como es obvio, debemos importar los datos. Si abrimos el fichero *SO2.txt* con el bloc de notas veremos que los nombres de los lugares aparecen en la primera fila y que los datos están separados por tabulaciones. Esta es toda la información que necesitamos para importar los datos (Figura 6.9 a la izquierda). El aspecto que tiene el conjunto de datos tras la importación aparece en la Figura 6.9, a la derecha.

Observemos que algunos de los valores aparecen como *NA*, es decir, como no disponibles, como datos faltantes. Eso es debido a que la muestra no es del mismo tamaño en los 4 lugares donde se muestrea, por lo que el editor tiene que *rellenar* con valores *NA* las casillas donde no hay valor muestral.

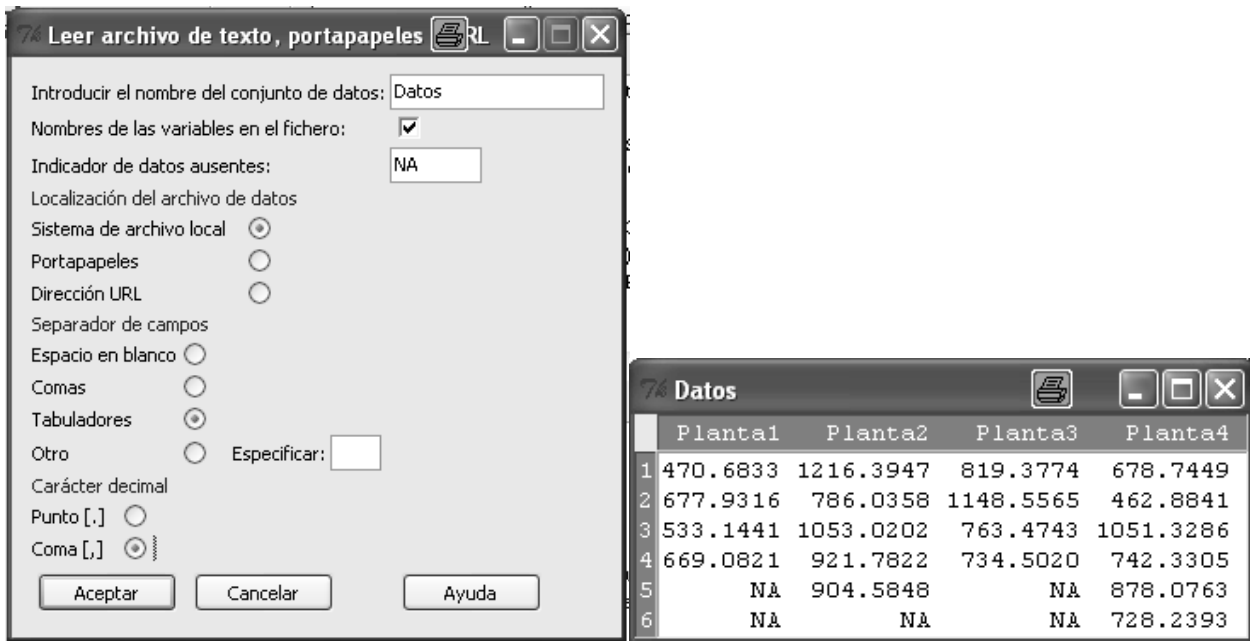


Figura 6.9: Importando los datos del contenido de TiO_2 del carbón

Eso nos hace ver que aquí tenemos el mismo problema que con el contraste de muestras independientes: hay que transformar los datos para que R Commander realice el análisis. Lo que tenemos que hacer es lo mismo que hicimos allí, apilar los datos para que aparezcan todos ellos en una misma columna y con una nueva variable que indique el lugar de donde procede el valor muestra. En la Figura 6.10 a la izquierda aparece la ventana de entradas de la opción *Datos* → *Conjunto de datos activo* → *Apilar variables del conjunto de datos*. A la derecha aparecen los datos apilados. Si nos fijamos, hay algunos valores *NA*. No tenemos que hacerles caso, ya que son aquellos que han sido apilados de los originales, pero R los ignorará.

Los datos ya están dispuestos para ser analizados. Seleccionamos la opción *Estadísticos* → *Medias* → *ANOVA de un factor* y se abre una ventana de entradas como la de la Figura 6.11. En ella hemos de seleccionar la variable que estamos analizando así como el factor que distingue los grupos, en nuestro caso, los lugares donde se tomaron las muestras.

Los resultados aparecen en la Figura 6.12:

- La variable de respuesta, es decir, la variable que estamos analizando como posiblemente afectada por el factor, es *SO2*.
- A continuación aparecen los grados de libertad, la suma de los cuadrados y la media de los cuadrados entre grupos y dentro de los grupos.
- Aparece en la penúltima columna el valor del estadístico de contraste *F*, 4.1043.

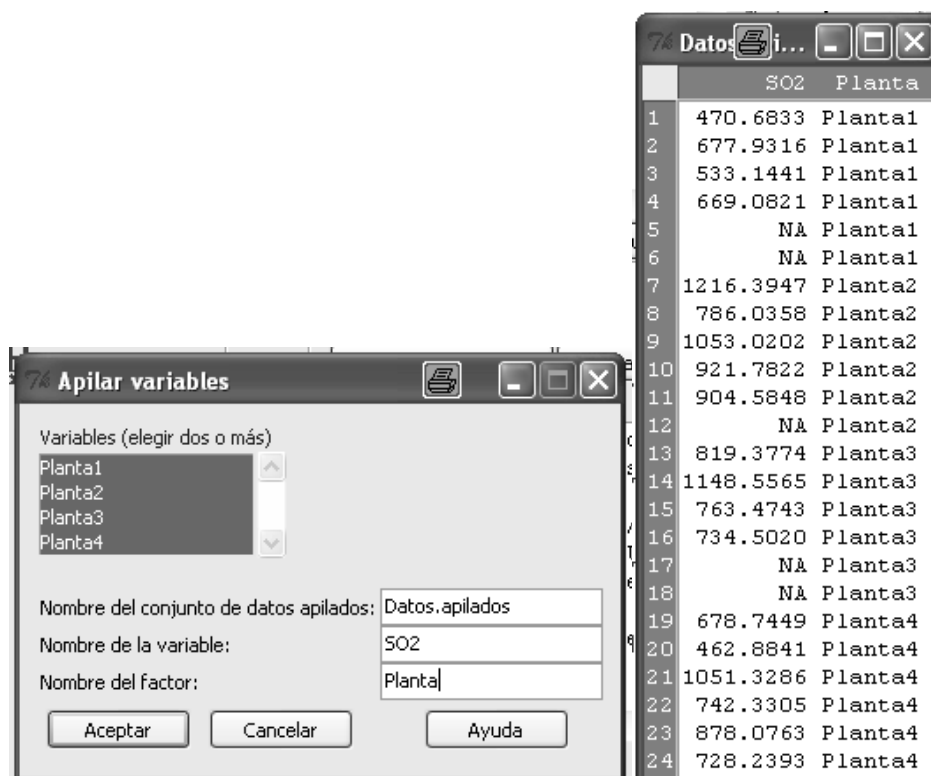
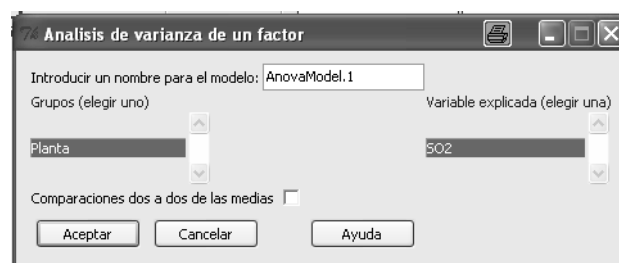
Figura 6.10: Apilando los datos del contenido de TiO_2 del carbón

Figura 6.11: Contraste de igualdad de medias en poblaciones apareadas

```

      Df Sum Sq Mean Sq F value Pr(>F)
Planta    3 364508   121503   4.1043 0.02595 *
Residuals 15 444055    29604
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
5 observations deleted due to missingness

> numSummary(Datos.apilados$SO2 , groups=Datos.apilados$Planta,
+  statistics=c("mean", "sd"))
      mean      sd n NA
Planta1 587.7103 102.3622 4  2
Planta2 976.3635 164.2005 5  1
Planta3 866.4776 191.3236 4  2
Planta4 756.9340 197.4635 6  0

```

Figura 6.12: Resultados del ANOVA

- En la última columna aparece el p-valor del contraste, 0.02595.

Visto este p-valor, podemos concluir que con los datos muestrales aportados se detectan diferencias significativas entre los niveles medios de *SO2* medidos en los 4 lugares.

- Aparecen al final los valores muestrales de las medias, las desviaciones típicas y el número de datos de cada muestra.

6.6.3. Resolución mediante código. La función *aov*

La sintaxis de esta función varía ligeramente de las anteriores porque obliga a presentar los datos en forma de modelo, a través de una fórmula. Concretamente, necesitamos, como en R Commander, que todos los datos de todas las muestras estén contenidos en una única variable, y que una segunda variable de tipo factor especifique a qué grupo pertenece cada dato.

Imaginemos que x_1 es un vector de dimensión n_1 conteniendo la primera muestra, x_2 es un vector de dimensión n_2 que contiene la segunda muestra y así sucesivamente hasta los k grupos del problema. Una forma muy simple de generar los datos necesarios para aplicar *aov* y que es equivalente a la opción *Apilar datos* de R Commander es

```
Datos<-data.frame(Variable=c(x1,x2,...,xk),
Grupo=factor(c(rep(1,n1),rep(2,n2),...,rep(k,nk))))
```

Una vez que tenemos preparados los datos, veamos cómo es la sintaxis básica de *aov*:

```
aov(Variable~Grupo,data=Datos))
```

Como podemos ver, sólo tenemos que especificar el nombre de la variable que contiene los datos (*Variable*), el nombre del factor que distingue a qué grupo pertenece cada dato (*Grupo*) y el nombre de la hoja de datos (*Datos*). La expresión *Variable~Grupo* se conoce en Estadística como la *fórmula del modelo*. Lo que viene a decir es que se trata de explicar la variabilidad de la variable *Variable* mediante el conocimiento de la variable *Grupo*⁴.

Finalmente, hay que decir que al ejecutar la función *aov* sólo se muestra en pantalla la partición de la varianza que resulta del ANOVA, pero no la tabla de éste que permite concluir el contraste. Para obtener la tabla de ANOVA es necesario aplicar la función *summary()* al resultado de *aov()*. Concretamente, habría que ejecutar

```
ANOVA<-aov(Variable~Grupo,data=Datos))
summary(ANOVA)
```

⁴El símbolo ~ se obtiene mediante la combinación de teclas ALT+126 o ALT GR+4.

o simplemente

```
summary(aov(Variable~Grupo,data=Datos))
```

Con todo, la resolución del ejemplo se lograría de la siguiente forma:

```
Datos<-read.table("S02.txt",header=TRUE,sep="\t",dec=',')
x1<-Datos$Planta1[is.na(Datos$Planta1)==0]
n1<-length(x1)
x2<-Datos$Planta2[is.na(Datos$Planta2)==0]
n2<-length(x2)
x3<-Datos$Planta3[is.na(Datos$Planta3)==0]
n3<-length(x3)
x4<-Datos$Planta4[is.na(Datos$Planta4)==0]
n4<-length(x4)
Datos<-data.frame(Variable=c(x1,x2,x3,x4),
Grupo=factor(c(rep(1,n1),rep(2,n2),rep(3,n3),rep(4,n4))))
summary(aov(Variable~Grupo,data=Datos))
```

Recordemos que al importar los datos, se han colado algunos valores ausentes, ya que no todas las muestras tienen el mismo número de datos. Evitamos cualquier confusión cuando definimos los vectores x_1 , x_2 , x_3 , x_4 con sólo los datos que no son *NA*, a través de la función *is.na()*. Esta función es un operador lógico que será 1 o TRUE si el argumento es *NA* y será 0 o FALSE si no lo es.

6.7. Ejercicios

1. Es un tópico que las mujeres conducen peor que los hombres. Un ingeniero mecánico que trabaja en cuestiones relativas a seguridad vial quiere realizar una comprobación al respecto en la población que le atañe. Concretamente, se interesa por el porcentaje de varones causantes de accidentes de tráfico. En una muestra aleatoria de n accidentes, descubre que en k de ellos fue un varón el causante. Sabiendo que el porcentaje de varones en la población es del 49%, ¿tiene evidencias el ingeniero que existan diferencias entre hombres y mujeres como causantes de accidentes de tráfico? (Utilícese un nivel de significación del 5%).
2. En una empresa los operarios de planta constituyen un colectivo de 528 empleados, de los cuales 79 sufren problemas de espalda. Los administrativos, por el contrario, son 32, de los

cuáles 7 sufren problemas de espalda. ¿Se tienen evidencias de que los administrativos sufren más problemas de espalda que los operarios de planta? (Utilícese un nivel de significación del 5 %).

3. El archivo *EjercicioDescriptiva.xls* contiene datos relativos a la edad media de todos los municipios de Andalucía. Con base en estos datos, ¿podemos concluir que la existen diferencias en los promedios de la edad media de los municipios de Jaén y Málaga? (Utilícese un nivel de significación del 5 %).
4. El mismo archivo también contiene datos relativos a la tasa de actividad en los municipios andaluces. Contrastar si existen diferencias significativas en los promedios de las tasas de actividad de los municipios de las 8 provincias. (Utilícese un nivel de significación del 5 %).
5. Una empresa se plantea comprar una herramienta que podría ayudar a sus empleados en la ejecución de las tareas más comunes, pero sólo le resultará rentable si, en promedio, mejora el rendimiento en más de dos minutos por cada 100 tareas. Para contrastar este hecho, contabiliza el tiempo (en minutos) que 40 trabajadores necesitan para realizar 100 tareas con y sin la nueva herramienta. Los datos aparecen en el fichero *herramienta.txt*. ¿Debe la empresa comprar la nueva herramienta? (Utilícese un nivel de significación del 5 %).

Capítulo 7

Contrastes de hipótesis no paramétricos

Objetivos:

1. Realizar contrastes χ^2 de bondad de ajuste para evaluar los ajustes mediante distribuciones discretas.
2. Realizar contrastes de Kolmogorov-Smirnoff de bondad de ajuste para evaluar los ajustes mediante distribuciones continuas.
3. Realizar contrastes χ^2 de independencia para valorar la relación existente entre variables cualitativas.

7.1. Contrastes de bondad de ajuste

R Commander no dispone de opciones de menú para realizar los contrastes de bondad de ajuste que hemos estudiado. Es por ello que debemos utilizar el código de R directamente para dichos contrastes.

7.1.1. Contraste χ^2 de bondad de ajuste

En el caso del test χ^2 de bondad de ajuste debemos instalar y cargar un paquete adicional de R llamado *vcd*. Recordemos que para ello debemos hacer lo siguiente:

1. Ejecutar la opción del menú de R *Paquetes* \rightarrow *Instalar paquete(s)*. Nos pedirá un *mirror* desde el que descargar el paquete, eligiendo *Spain*. Después buscamos el paquete *vcd* en la lista emergente.
2. Ejecutar la opción del menú de R *Paquetes* \rightarrow *Cargar paquete*, eligiendo *vcd*.

Este paquete permite ajustar los parámetros de las distribuciones mediante un método distinto al de máxima verosimilitud, llamado método de la mínima χ^2 , y realizar el test χ^2 de bondad de ajuste para las distribuciones de Poisson, binomial y binomial negativa. Indirectamente también permite la distribución geométrica, considerando que ésta es un caso particular de la binomial negativa.

La función para ello es *goodfit*, cuya sintaxis básica es:

```
ajuste<-goodfit(datos, type="distribución", par, method='MinChisq')
```

Fijaros que le hemos puesto nombre al resultado de la función (*ajuste*). Eso es para poder sacar de él varios resultados, como veremos enseguida. Vamos a especificar los argumentos con detalle:

1. **datos** es el vector de datos de la muestra.
2. **type** es la distribución que consideramos en el ajuste. Los valores posibles son "*poisson*", "*binomial*" y "*nbinomial*".
3. **par** es la lista de los parámetros de la distribución del ajuste. Sólo lo usaremos para el caso de la geométrica. Teniendo en cuenta que una *Geo*(*p*) es una *BN*(1,*p*), si queremos el contraste para un ajuste mediante una distribución *Geo*(*p*), tenemos que poner **type='nbinomial'** y **par=list(size=1,prob=p)**.
4. **method='MinChisq'** es la opción para que realice el test χ^2 . Debemos dejarlo fijo siempre.

Los resultados que *goodfit* proporciona son los siguientes:

1. **summary(ajuste)** dará el valor del estadístico χ^2 , los grados de libertad y el p-valor del test. Hay que decir que no agrupa casillas con frecuencias esperadas inferiores a 5. Si hay alguna casilla así, aparecerá un mensaje de alarma.
2. **plot(ajuste)** dibuja la función masa junto con el diagrama de barras del ajuste, en escala de raíz cuadrada. Las barras del diagrama aparecen alineadas con la función masa, de manera que la falta de ajuste se aprecia en la base de las barras.
3. **ajuste** devuelve la tabla de frecuencias observadas y esperadas.
4. **ajuste\$par** devuelve la estimación de los parámetros por el método de la mínima χ^2 .

Vamos a volver sobre los mismo ejemplos que utilizamos en la práctica sobre estimación por máxima verosimilitud. Recordad que allí calculamos las estimaciones y dibujamos las funciones masa junto con los diagramas de barras, pero comentamos que aún no podíamos decidir si los ajustes eran *buenos* o *malos*. Ahora es el momento de terminar con esa cuestión.

7.1.1.1. Para la distribución de Poisson

En el Cuadro 7.1 aparecen de nuevo los datos sobre el número de muertes en las 20 compañías del ejército prusiano debidas a coces de caballos. Esos datos están en el fichero *DatosMuertesCoces.rda*. En su momento ajustamos para esos datos una distribución *Poisson*(0.61). Si ese ajuste fuese bueno, podríamos concluir que las muertes se producen por puro azar en las distintas compañías.

Muertes/año	Frecuencia
0	109
1	65
2	22
3	3
4	1

Cuadro 7.1: Distribución de frecuencias en el ejemplo del ejército prusiano

Carguemos los datos del fichero. El conjunto de datos y la variable del conjunto de datos se llaman ambos *muertes*. Para realizar el test χ^2 de bondad de ajuste debemos ejecutar el siguiente código:

1. `ajuste.poisson <- goodfit(muertes$muertes, type = "poisson", method="MinChisq")`. Esto generará el objeto llamado `ajuste.poisson` que contiene los resultados del test χ^2 .

2. `ajuste.poisson$par` devuelve lo siguiente:

```
$lambda
```

```
[1] 0.6139969
```

Por lo tanto, el ajuste por el método de la mínima χ^2 es una *Poisson*(0.614). La diferencia entre la estimación máximo-verosímil y la de la mínima χ^2 es de menos de 4 milésimas.

3. `summary(ajuste.poisson)`. Devuelve el siguiente resultado:

```
Goodness-of-fit test for poisson distribution
```

```
X^2 df P(> X^2)
```

```
Pearson 0.594649 3 0.8976563
```

Tenemos, por tanto, que el valor del estadístico de contraste es 0.594, los grados de libertad son 3 y el p-valor es 0.8976563. Ese p-valor nos permite aceptar la hipótesis nula, es decir, concluir que no existen evidencias en los datos en contra de que estos se ajusten a una *Poisson*(0.614). De hecho, el p-valor es altísimo, lo que indica que el ajuste es excelente.

4. `plot(ajuste.poisson)`. Devuelve la Figura 7.1. Pueden verse sólo ligeros desajustes en las bases de las barras.

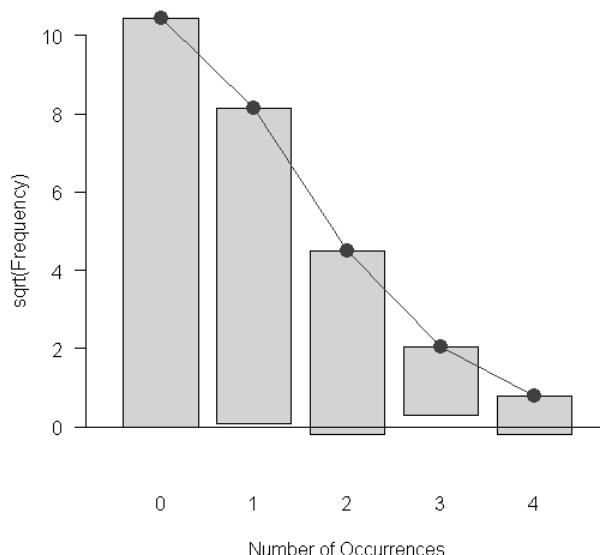


Figura 7.1: Diagrama de barras y ajuste de Poisson en el ejemplo de las muertes por ceces de caballos

5. `ajuste.poisson` devuelve la siguiente tabla:

```
Observed and fitted values for poisson distribution
with parameters estimated by 'MinChisq'
count observed fitted
0 109 108.2366935
1 65 66.4569976
2 22 20.4021963
3 3 4.1756286
4 1 0.6409558
```

Podemos ver que las dos últimas frecuencias esperadas son inferiores a 5. Por ello, en la ventana de mensajes aparece lo siguiente:

```
AVISO: Warning in summary.goodfit(ajuste.poisson) : Chi-squared approximation may
be incorrect
```

7.1.1.2. Para la distribución geométrica

Los datos que manejamos se referían a los tiempos (en *ms*) que transcurren entre una muestra aleatoria de 250 paquetes en una determinada transmisión telemática, datos que se encuentran en el fichero *trafico.telem.rda*.

Cargamos los datos. El conjunto de datos se llama *trafico.telem* y la variable *tiempos.entre.paquetes*. Para realizar el ajuste ejecutamos el siguiente código:

1. `ajuste.geom <- goodfit(trafico.telem$tiempos.entre.paquetes, type = "nbinomial", method='MinChisq', par=list(size=1)).`
2. `ajuste.geom$par` devuelve lo siguiente:

```
$size
[1] 1
$prob
[1] 0.1129637
```

Por lo tanto, el ajuste por el método de la mínima χ^2 es una *Geo* (0.113). En su momento vimos que la estimación por máxima verosimilitud de p era 0.117591722. De nuevo la diferencia entre ambas estimaciones es del orden de milésimas.

3. `summary(ajuste.geom)`. Devuelve el siguiente resultado:

```
Goodness-of-fit test for nbinomial distribution
X^2 df P(> X^2)
Pearson 14.66274 29 0.9874704
```

Tenemos, por tanto, que el valor del estadístico de contraste es 0.594, los grados de libertad son 3 y el p-valor es 0.987. Ese p-valor nos permite aceptar la hipótesis nula, es decir, concluir que no existen evidencias en los datos en contra de que estos se ajusten a una *Poisson* (0.614). De hecho, el p-valor es altísimo, lo que viene a decir que el ajuste es excelente.

4. `plot(ajuste.geom)`. Devuelve la Figura 7.2. A pesar de que el p-valor indica que el ajuste es excelente, se observan bastantes desajustes, incluso en los valores iniciales que, al tener frecuencias mayores, deberían estar mejor ajustados.
5. `ajuste.geom` devuelve la tabla de frecuencias observadas y esperadas. Es demasiado larga para ponerla aquí, pero podeis ver que hay multitud de casillas que deberían ser agrupadas. Eso puede estar provocando que el p-valor no corresponda con la realidad de la distribución del estadístico de contraste.

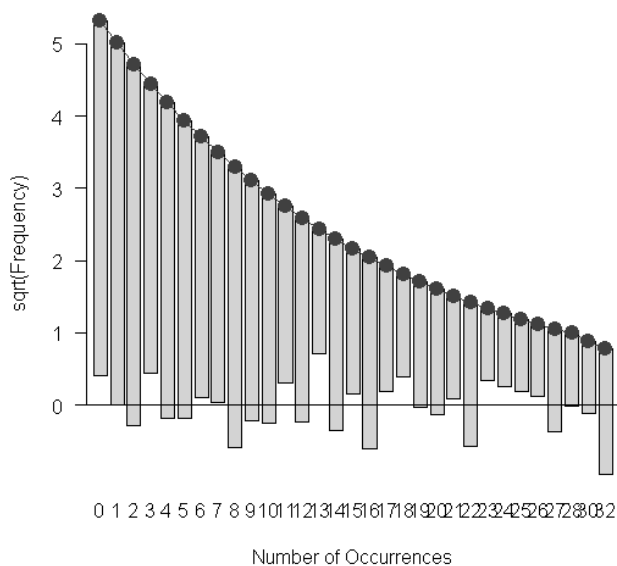


Figura 7.2: Diagrama de barras y ajuste geométrico de los datos de tráfico telemático

7.1.1.3. Para la distribución binomial negativa

Con los mismos datos tratamos de ajustar una distribución binomial negativa. Vamos a hacerlo de nuevo aquí:

```
1. ajuste.nbinom <- goodfit(trafico.telem$tiempos.entre.paquetes,
  type = "nbinomial",method="MinChisq")
```

```
2. ajuste.nbinom$par devuelve
```

```
$size
[1] 1.165172
$prob
[1] 0.1301135
```

Si recordamos las estimaciones por máxima verosimilitud, vemos de nuevo que no son muy distintas de éstas.

```
3. summary(ajuste.nbinom) devuelve
```

```
Goodness-of-fit test for nbinomial distribution
X^2 df P(> X^2)
Pearson 12.65186 28 0.9943238
```

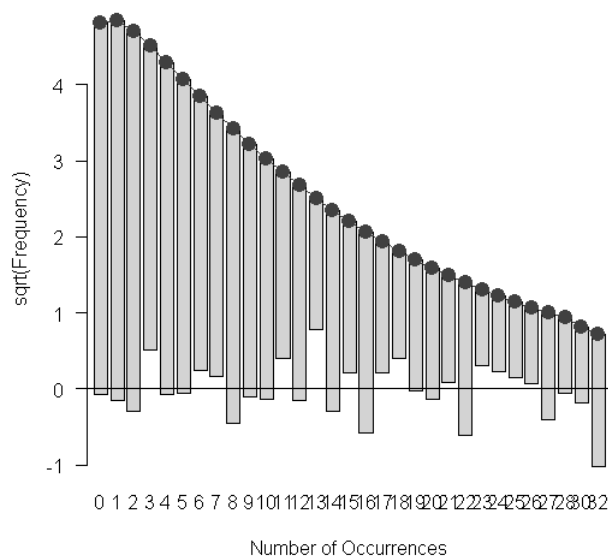


Figura 7.3: Diagrama de barras y ajuste geométrico de los datos de tráfico telemático

Tampoco hay evidencias en contra del ajuste mediante una $BN(1.165, 0.130)$. Además, el p-valor es superior aún al del ajuste mediante la distribución geométrica, luego éste es mejor que aquel. No obstante, al igual que antes, también hay muchas casillas con frecuencias esperadas inferiores a 5.

4. `plot(ajuste.nbinom)` devuelve la Figura 7.3. Los desajustes son ligeramente inferiores.

7.1.1.4. Para la distribución binomial

Para finalizar, consideremos el siguiente ejemplo. Se analizan camadas de ratas que dan a luz a 20 congéneres. Se les suministra una droga para analizar su toxicidad y se contabilizan el número de crías que sobreviven hasta los dos meses. Los datos, referentes a 150 camadas de 20 crías, se encuentran en el fichero *crias.camadas.rda*, donde el conjunto de datos activo tiene el mismo nombre, *crias.camadas* y la variable se llama *crias.supervivientes*. Vamos a ver si una distribución binomial es adecuada para ajustar esos datos, en cuyo caso podríamos inferir que el comportamiento de las camadas es independiente y que la probabilidad de supervivencia al fármaco es constante. Querríamos, además, una estimación de esta probabilidad de supervivencia.

Cargamos el fichero con los datos. El código a ejecutar es el siguiente:

1. `ajuste.binom <- goodfit(crias.camadas$crias.supervivientes, type = "binomial", method="MinChisq", par=list(size=20))`. Aquí es importante incluir el parámetro $n = 20$, ya que, de no hacerlo, R consideraría como valor de n el máximo valor de la variable, que podría no ser 20 (de hecho no lo es).

2. `ajuste.binom$par` devuelve

```
$size
[1] 20

$prob
[1] 0.1607786
```

Es decir, la estimación por el método de la mínima χ^2 del parámetro p es $\hat{p} = 0.161$. La estimación por el método de máxima verosimilitud es $\frac{\bar{x}}{n} = 0.158$. En nuestro caso, el ajuste de los datos viene dado por una distribución $B(20, 0.161)$.

3. `summary(ajuste.binom)` devuelve

```
Goodness-of-fit test for binomial distribution

X^2 df P(> X^2)

Pearson 11.40937 7 0.1217325
```

El p-valor superior a 0.05 indica que no hay evidencias en contra del ajuste de los datos mediante la distribución $B(20, 0.161)$. No obstante, de nuevo hay casillas con frecuencias esperadas inferiores a 5.

4. `plot(ajuste.binom)` devuelve la Figura 7.4. Debemos destacar, al menos, dos cuestiones:

- a) Hay un desajuste entre lo observado y lo esperado bastante importante en la frecuencia que constituye la moda, el valor 3.
- b) La gráfica termina en el valor 8 porque no se observan valores en la muestra superiores a 8. Según el modelo dado por la distribución $B(20, 0.161)$, la probabilidad de que se den valores superiores a 8 es 0.002.

7.1.2. Contraste de Kolmogorov-Smirnoff

Para el ejemplo con el que vamos a describir el contraste, consideremos los datos del fichero *tiempos.fallo.rda*, que en su momento tratamos de ajustar mediante una distribución exponencial. La variable también se llamaba *tiempos.fallo*, por lo que si queremos referirnos a los datos concretos debemos escribir *tiempos.fallo\$tiempos.fallo*.

La distribución resultante del ajuste fue *exp*(0.492). Vamos a contrastar ahora la bondad de este ajuste. El código es el siguiente:

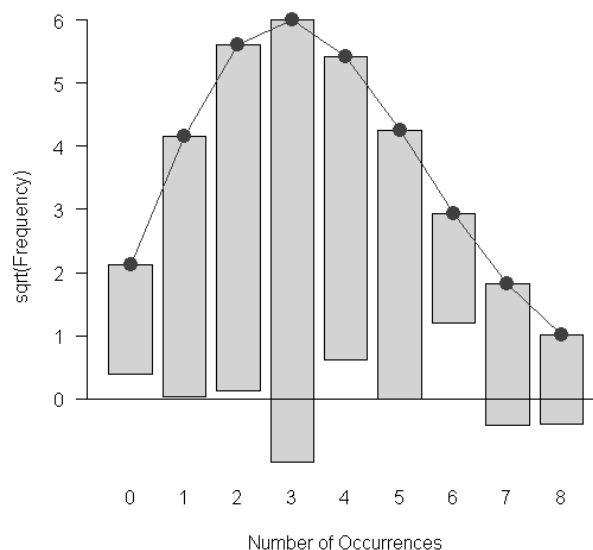


Figura 7.4: Ajuste de los datos de las crías supervivientes al fármaco mediante una distribución binomial

```
ks.test(tiempos.fallo$tiempos.fallo,"pexp",0.492)
```

El resultado es

```
One-sample Kolmogorov-Smirnov test
data: tiempos.fallo$tiempos.fallo
D = 0.0485, p-value = 0.4805
alternative hypothesis: two-sided
```

Por lo tanto, no hay evidencias en los datos en contra del ajuste considerado mediante la distribución exponencial.

Fijaros que en el código para usar la función *ks.test* sólo hay que introducir lo siguiente:

1. Los datos (`tiempos.fallo$tiempos.fallo`).
2. La función de distribución del modelo del ajuste, entre comillas. Por ejemplo, *"pexp"*, *"pgamma"*, o *"pnorm"* (en nuestro caso, la distribución exponencial).
3. Los parámetros de la distribución ajustada (en nuestro caso, $\lambda = 0.492$).

7.2. Contraste χ^2 de independencia

7.2.1. Mediante R Commander

Recordemos que este contraste sirve para valorar si existe relación significativa entre un par de variables cualitativas.

	Raza			
Grupo	Blancos	Negros	Mestizos	Total
A	510	78	175	763
B	116	48	62	226
AB	37	10	18	65
O	578	141	314	1033
Total	1241	277	569	2087

Cuadro 7.2: Distribución de frecuencias observadas en función de la raza y el grupo sanguíneo

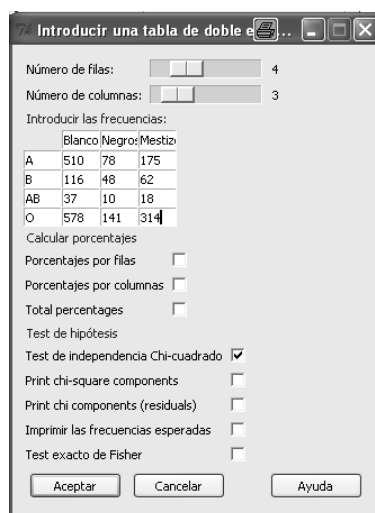


Figura 7.5: Ventana de entradas para tablas de contingencia

A modo de ejemplo, en la Revista Cubana de Hematología, Inmunología y Medicina Transfusional (1997, Vol. 13(2), pp. 122-31), el artículo *Frecuencia de los grupos sanguíneos A1, A2, Aint, Ael, B y O en donantes de sangre* se plantea si el grupo sanguíneo está o no relacionado con la raza. Para ello analizan una muestra de individuos donantes de sangre en relación a su grupo sanguíneo (clasificado en nuestro caso en A, B, AB y O) y su raza: descendientes de europoides (blancos), descendientes de africanoides (negros) y mestizos (mezcla entre ambos)

Para realizar el contraste con R Commander, elegimos la opción *Estadísticos* → *Tablas de contingencia* → *Introducir y analizar una tabla de doble entrada*. Aparece una ventana como la de la Figura 7.5. En ella tenemos que especificar lo siguiente:

1. El número de filas y de columnas, mediante los *scrolls* que aparecen en la parte superior y que podemos mover en un sentido o en otro según queramos más o menos filas o columnas.
2. Los valores en sí de la tabla.
3. Si lo deseamos, podemos pedirle que saque en la ventana de resultados la tabla en porcentajes, calculados por filas, por columnas o sobre el total de la tabla.

4. El tipo de test que queremos. La versión más simple, la que nosotros hemos considerado, es la de la del Test de independencia χ^2 .

El resultado es el siguiente:

```
Pearson's Chi-squared test  
data: .Table  
X-squared = 37.1616, df = 6, p-value = 1.638e-06
```

Vemos el valor del estadístico de contraste (37.1616), los grados de libertad (6) y el p-valor, que permite concluir que con esos datos podemos confirmar que existe relación altamente significativa entre el grupo sanguíneo y la raza.

7.2.2. Mediante código

De nuevo utilizamos la función *chisq.test*, proporcionando simplemente la matriz de frecuencias observadas:

```
Tabla<-matrix(c(510,78,175,116,48,62,37,10,18,578,141,314),4,3,byrow=TRUE)  
chisq.test(Tabla,correct=FALSE)
```

7.3. Ejercicios

1. El fichero *muestra1.rda* contiene un conjunto de datos de valores de una variable discreta que ya ajustamos en el capítulo de estimación puntual. Se pide ajustar una distribución binomial a esos datos mediante el método de mínima χ^2 y valorar si el ajuste es adecuado a los datos mediante un test χ^2 de bondad de ajuste.
2. Ajustar una distribución de Poisson a los datos que contiene el fichero *muestra2.rda* mediante el método de mínima χ^2 y valorar si el ajuste es adecuado a los datos mediante un test χ^2 de bondad de ajuste.
3. El fichero *muestra3.rda* contiene un conjunto de datos de valores de una variable discreta que también ajustamos en su momento mediante estimadores de máxima verosimilitud. Se pide:
 - a) Ajustar una distribución geométrica y una distribución binomial negativa a esos datos mediante el método de mínima χ^2 .
 - b) Según el test χ^2 de bondad de ajuste ¿cuál de las dos distribuciones que ajusta mejor los datos?

	Hombre	Mujer
Diestro	43	36
Zurdo	44	16

Cuadro 7.3: Tabla de contingencia

	Hombre	Mujer
Fuma	120	112
No fuma	88	130

Cuadro 7.4: Tabla de contingencia

4. El fichero *muestra4.rda* contenía un conjunto de datos de valores de una variable continua que ajustamos mediante máxima verosimilitud con una distribución exponencial y una Gamma. ¿Cuál de los dos ajustes, según el test de Kolmogorov-Smirnoff, es más preciso?
5. El fichero *muestra5.rda* contiene un conjunto de datos de valores de una variable continua que ajustamos con una distribución normal mediante el método de máxima verosimilitud. Según el test de Kolmogorov-Smirnoff, ¿ese ajuste es razonable?
6. El Cuadro 7.3 muestra los datos de una muestra de hombres y mujeres según si son diestros o zurdos en una tabla de contingencia. A la luz de esos datos, ¿podemos afirmar que existe una relación significativa entre el género y el ser diestro o zurdo?
7. El Cuadro 7.4 incluye una tabla de contingencia sobre el hábito de fumar de una muestra de hombres y mujeres. ¿Se puede inferir, a raíz de ella, que hay una relación estadísticamente significativa entre el género y el hábito de fumar?
8. El Cuadro 7.5 contiene una tabla de contingencia basada en los datos de una muestra de hombres y mujeres clasificados como *obesos* y *no obesos*. ¿Se puede decir, a la luz de esos datos, que existe una relación significativa entre el género y esa clasificación?

	Hombre	Mujer
Obeso	35	33
No obeso	19	13

Cuadro 7.5: Tabla de contingencia

Capítulo 8

Regresión lineal simple

Objetivos:

1. Contrastar si la relación entre pares de variables es estadísticamente significativa.
2. Ajustar la recta de regresión de una variable dependiente dada una variable independiente.
3. Realizar predicciones y estimaciones a partir de la recta de regresión.
4. Obtener intervalos de confianza para dichas predicciones y estimaciones.

8.1. Correlación

A lo largo de esta sección vamos a considerar los indicadores de consumo de agua y electricidad y de generación de residuos de los municipios de la provincia de Jaén recogidos en el fichero *JaenIndicadores.rda*.

Vamos a comenzar recordando que la manera que tenemos de identificar el grado de relación lineal entre pares de variables es analizando el coeficiente de correlación lineal entre dichos pares de variables.

En el ejemplo de los municipios de la provincia de Jaén, vamos a comenzar preguntándonos por el sentido y la fortaleza de las asociaciones lineales entre los indicadores que en su momento comentamos: consumo de agua, de electricidad y residuos generados por cada 100 habitantes. Para ello necesitamos los coeficientes de correlación lineal entre ellos.

8.1.1. Mediante R Commander

En el menú de R Commander elegimos la opción *Estadísticos* \rightarrow *Resúmenes* \rightarrow *Matriz de correlaciones*. En la ventana emergente debemos seleccionar las variables *agua.hab*, *elec.hab* y *res.hab*.

	agua.hab	elec.hab	res.hab
agua.hab	1.00	-0.05	-0.01
elec.hab	-0.05	1.00	0.16
res.hab	-0.01	0.16	1.00

Cuadro 8.1: Coeficientes de correlación lineal entre los indicadores de consumo de los municipios de la provincia de Jaén

El resto de las opciones las dejamos en sus valores predeterminados. El resultado es la matriz que aparece en el Cuadro 8.1.

Vamos a comentar con detalle los resultados:

- En la diagonal, obviamente, aparecen unos, ya que el coeficiente de correlación lineal de una variable consigo misma es uno.
- La matriz es simétrica, ya que el coeficiente de correlación lineal también lo es, es decir, el coeficiente de la variable X con la variable Y es idéntico al de la variable Y con la variable X .
- Los coeficientes entre la variable relativa al consumo de agua por habitante con las otras dos variables son prácticamente cero, indicando que el grado de relación lineal es prácticamente nulo.
- El coeficiente de correlación lineal entre el consumo de energía eléctrica y la cantidad de residuos producidos por habitante es ligeramente positivo, lo que podría indicar cierto grado de relación directa entre ambas variables. Este dato vendría a decir, si se confirma, que parece haber una ligera tendencia a que los municipios con mayor consumo eléctrico sean además los que más residuos generen.

De todas formas, hasta ahora sólo hemos analizado estos coeficientes de una forma descriptiva, cuando lo más interesante es responder con claridad a la pregunta de si existe o no una relación **estadísticamente significativa** entre esos indicadores. Dicho de otra forma, refiriéndonos, por ejemplo, al coeficiente 0.164, ¿es suficientemente grande como para que indique una relación estadísticamente significativa o su valor podría deberse al azar? Esa misma pregunta es válida para los otros dos coeficientes que, en principio, hemos dicho que son *demasiado pequeños* para tenerlos en cuenta: ¿será eso cierto?

Lo que implícitamente estamos haciendo con estos comentarios es plantear la necesidad de contrastar si estos valores muestrales de los coeficientes son capaces o no de demostrar que los coeficientes de correlación poblacionales son significativamente distintos de cero.

Tenemos que hacerlo de dos en dos, no con las tres variables a la vez. Elegimos la opción *Estadísticos* \rightarrow *Resúmenes* \rightarrow *Test de correlación* y en la ventana emergente seleccionamos dos de

las variables. Tenemos, además, la opción de elegir si el test es bilateral o unilateral. En nuestro caso la hipótesis alternativa es que es distinto de cero, luego elegimos bilateral.

Para los datos relativos al consumo de agua y de electricidad los resultados son los siguientes:

```
Pearson's product-moment correlation
data: Datos$agua.hab and Datos$elec.hab
t = -0.4562, df = 92, p-value = 0.6493
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
-0.2477396 0.1566172
sample estimates:
cor
-0.04750735
```

Fijémonos que aparecen las variables que hemos elegido, el valor del estadístico t , los grados de libertad y, lo que es más importante, el p-valor. En este caso, es muy superior a 0.05, así que no podemos rechazar la hipótesis nula (el coeficiente es cero) en favor de la alternativa (el coeficiente es distinto de cero) o, dicho de otra forma, no hemos encontrado indicios de que exista una relación estadísticamente significativa entre el consumo de energía eléctrica y el consumo de agua por habitante.

Aparece además, un intervalo de confianza al 95 % para el coeficiente de correlación lineal (que contiene al cero, como cabía esperar) y el valor muestral del coeficiente.

En el caso del consumo de agua y la cantidad de residuos generados los resultados son los siguientes:

```
Pearson's product-moment correlation
data: Datos$agua.hab and Datos$res.hab
t = -0.0778, df = 92, p-value = 0.9381
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
-0.2103830 0.1948254
sample estimates:
cor
-0.008111786
```

Las evidencias contra la hipótesis nula son aún menores. El p-valor indica que no hay en absoluto evidencias de relación entre el consumo de agua por habitante y el volumen de residuos generados por habitante.

Finalmente, los resultados relativos al consumo de energía eléctrica y el volumen de residuos son los siguientes:

```
Pearson's product-moment correlation
data: Datos$elec.hab and Datos$res.hab
t = 1.7571, df = 94, p-value = 0.08216
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
-0.02298577 0.36573257
sample estimates:
cor
0.1783219
```

Como podemos ver, el p-valor (0.082) indica que existen leves indicios de relación, pero no los suficientes para afirmar que exista una relación estadísticamente significativa entre las dos variables.

Para finalizar, hemos cometido un pequeño desliz que aún estamos a tiempo de corregir, ya que directamente hemos analizado las posibles relaciones entre las variables como relaciones de tipo lineal. Ya comentamos en clase que previamente era muy recomendable realizar un diagrama de dispersión o nube de puntos para detectar otro tipo de relaciones.

En R Commander esto podemos hacerlo mediante la opción *Gráficas* → *Matriz de diagramas de dispersión*. Este análisis es bastante rico, pero nosotros no hemos visto la mayoría de las opciones que permite, por lo que seleccionamos sólo el análisis más simple (ver Figura 8.1 a la izquierda). El resultado aparece en la Figura 8.1, a la derecha. No aparecen patrones que hagan pensar en ningún otro tipo de relación.

8.1.2. Mediante código. Las funciones *pairs*, *cors* y *cor.test*

La matriz con los diagramas de dispersión puede realizarse trivialmente mediante la función *pairs* en cuya sintaxis básica tan sólo hay que facilitarle como entrada la matriz que constituyen las muestras de las variables cuyas relaciones estamos analizando¹:

```
pairs(Datos[,c("agua.hab", "res.hab", "elec.hab")])
```

¹La función que R Commander utiliza para la matriz de diagramas de dispersión es *scatterplot.matrix*, del paquete *car*.

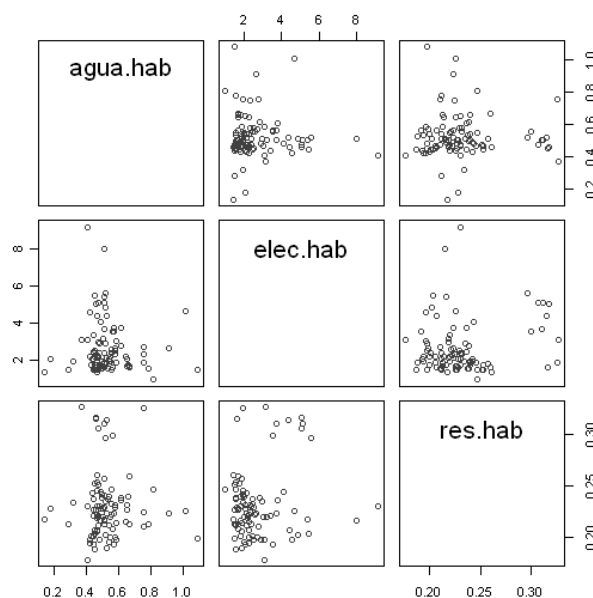
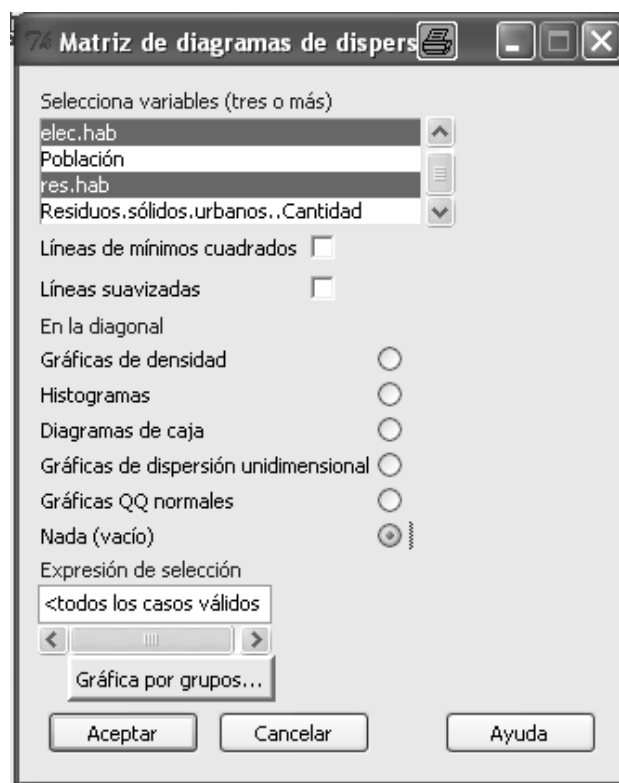


Figura 8.1: Ventana de entradas para la obtención de los diagramas de dispersión

La matriz de coeficientes de correlación la facilita la función *cor*, en cuya sintaxis básica de nuevo hay que facilitarle como entrada la matriz de datos y especificar, si es necesario, qué hacer con las observaciones faltantes. Por ejemplo, en el caso que nos ocupa,

```
cor(Datos[,c("agua.hab", "res.hab", "elec.hab")],  
     ,use="complete.obs")
```

Debido a que hay datos faltantes, si no ponemos `use="complete.obs"` los coeficientes de correlación no serán calculados. Lo que especifica ese argumento es que utilice sólo los casos que tengan todos los valores de las variables disponibles.

Finalmente, el test de correlación se realiza mediante la función *cor.test*. Su sintaxis básica es la siguiente:

```
cor.test(x, y, alternative = c("two.sided", "less", "greater"),  
         conf.level = 0.95)
```

Por ejemplo, para realizar los tres test de nuestro caso tendríamos

```
cor.test(Datos$agua.hab,Datos$elec.hab)  
cor.test(Datos$agua.hab,Datos$res.hab)  
cor.test(Datos$res.hab,Datos$elec.hab)
```

8.2. Ajuste de la recta de regresión

En el ejemplo que acabamos de ver no se detecta relación lineal entre las variables, luego tiene poco sentido que nos planteemos un ajuste, es decir, un modelo, para dicha relación. Vamos a considerar otro ejemplo donde sí tiene sentido analizarlo.

En el capítulo de Estadística Descriptiva manejamos, entre otras, variables con las que calculamos la tasa de líneas ADSL en 2007 y, además, analizamos la variable relativa a la edad media de cada municipio en 2007 en Andalucía. Los datos se encuentran en el fichero *EjercicioDescriptiva.rda*.

Si calculamos, como anteriormente, el coeficiente de correlación lineal entre estas dos variables, el resultado es -0.643 . El p-valor asociado a este valor muestral para contrastar si el coeficiente de correlación es distinto de cero es inferior a 2.2×10^{-16} . Obviamente, esto indica que hay una relación fortísimamente significativa desde el punto de vista estadístico entre las variables. Desde un punto de vista más práctico, podemos afirmar, por tanto, que existe una tendencia lineal bastante intensa a que los municipios de más edad tengan menos líneas ADSL por habitante.

Por otra parte, preguntémonos por si podemos considerar si una de las variables es la causa y otra el efecto en esta relación detectada. Hay ocasiones en la que esto no es posible, porque ambas son causa y efecto una de la otra. En este caso, sin embargo, resulta difícil pensar que la tasa de líneas ADSL cause cambios en la edad media de un municipio, y sí es más creíble pensar que es la edad media del municipio la que afecta a la tasa de líneas ADSL, de una forma negativa. Así pues, desde el punto de vista de la regresión, la tasa de líneas ADSL sería la variable dependiente y la edad media del municipio, la variable independiente.

¿Qué sentido puede tener ajustar la recta de regresión para estas variables?

- Podríamos plantearnos, por ejemplo, analizar en qué medida se vería afectada la implantación de líneas ADSL ante un cambio en la edad media poblacional.
- Si se desconociera el dato de implantación de líneas ADSL en un municipio, podríamos tratar de predecirlo utilizando la recta.
- Podremos analizar las diferencias entre lo observado y lo esperado, y tratar de explicar estas diferencias. Si, por ejemplo, una población muy joven tiene menos implantación de líneas ADSL de lo que se espera según la recta, una empresa proveedora de este servicio podría en primer lugar preguntarse cuál es la causa y, en segundo lugar, iniciar una campaña de captación de clientes, entendiendo que el municipio es una potencial mina de nuevos contratos.

8.2.1. Mediante R Commander

Vamos manos a la obra. Para obtener el ajuste de la recta de regresión entre las dos variables, elegimos la opción *Estadísticos* → *Ajuste de modelos* → *Regresión lineal*. La ventana emergente (Figura 8.2) nos pide que especifiquemos la variable explicada (o variable dependiente) y la variable explicativa (o variable independiente)². Adicionalmente, nos permite ponerle un nombre al modelo resultante.

Los resultados que aparecen en la ventana son los siguientes:

Call:

```
lm(formula = tasa.lineas.ADSL.2007 ~ Edad.media.2007  
, data = Datos)
```

Residuals:

```
Min 1Q Median 3Q Max
```

²En realidad, se pueden especificar varias variables explicativas, pero en ese caso ya no sería un modelo de regresión lineal simple, sino un modelo de regresión lineal múltiple, que excede los contenidos de este curso.

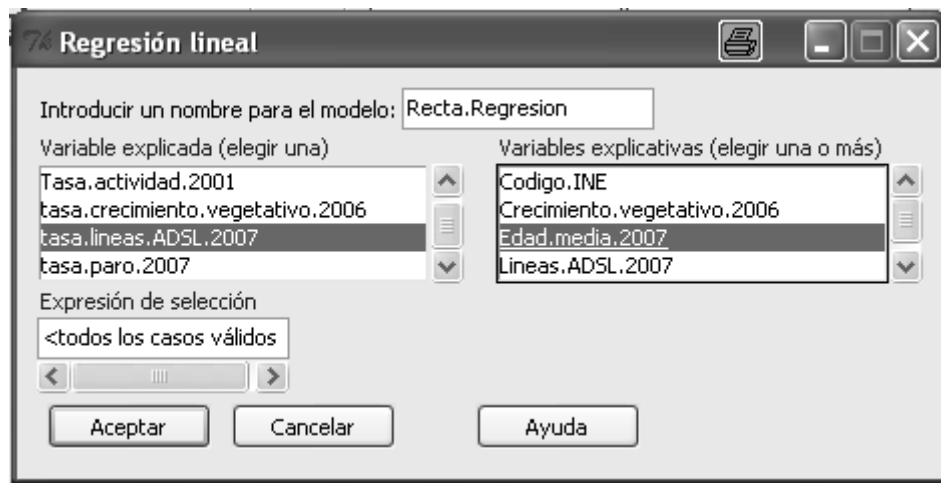


Figura 8.2: Ventana emergente para el ajuste de la recta de regresión

```
-10.7585 -2.5384 -0.2040 1.8205 31.3879
```

```
Coefficients:
```

```
Estimate Std. Error t value Pr(>|t|)
```

```
(Intercept) 38.35290 1.36416 28.11 <2e-16 ***
```

```
Edad.media.2007 -0.76192 0.03271 -23.29 <2e-16 ***
```

```
---
```

```
Signif. codes: 0 *** 0.001 ** 0.01 * 0.05 . 0.1 1
```

```
Residual standard error: 4.035 on 768 degrees of freedom
```

```
Multiple R-squared: 0.4139, Adjusted R-squared: 0.4132
```

```
F-statistic: 542.5 on 1 and 768 DF, p-value: < 2.2e-16
```

Algunos de estos resultados requerirían conocimientos avanzados de regresión, pero busquemos los que nos interesan:

1. La estimación del valor del parámetro β_0 (*intercept*) es 38.353. Hipotéticamente, se interpretaría como el número estimado o el promedio de la tasa de líneas ADSL si la edad media fuera 0, lo que, claro está, no tiene sentido.

A continuación lo que aparece es: el error estandar de esa estimación (1.364), el valor del estadístico de contraste (28.11) y el p-valor del contraste de $H_0 : \beta_0 = 0$ frente a $H_1 : \beta_0 \neq 0$ (inferior a 2×10^{-16}).

2. La estimación de β_1 es -0.762, con un error estandar de la estimación de 0.033. El contraste de $H_0 : \beta_1 = 0$ frente a $H_1 : \beta_1 \neq 0$ arroja un valor del estadístico de -23.29 y un p-valor también inferior a 2×10^{-16} .

La recta ajustada aparece, por tanto, especificada a través de sus dos coeficientes: el término independiente o *intercept* y la pendiente de la recta:

$$tasa.lineas.ADSL.2007 = 38.353 - 0.762 \times Edad.media.2007.$$

Así pues, por cada año en que se incremente la edad media de los municipios, el promedio de líneas ADSL por cada 100 habitantes se reducirá en 0.762.

3. El error estandar del ajuste tiene un valor de 4.035.
4. El coeficiente R^2 , aunque es obvio de calcular, aparece en la penúltima línea. Su valor, 0.414, indica que casi el 41.4% de toda la variabilidad que tiene el fenómeno relativo a la tasa de líneas ADSL por cada 100 habitantes puede ser explicado por la edad media del municipio. Sin ser un porcentaje muy elevado, resulta una afirmación sorprendente.

Aparte del propio ajuste de los parámetros β_0 y β_1 , podemos también obtener intervalos de confianza al nivel que deseemos de los mismos, sin más que clicar en *Modelos* \rightarrow *Intervalos de confianza*. En la ventana emergente sólo tenemos que elegir el nivel de confianza deseado. En nuestro caso, el resultado que aparece es el siguiente:

```
2.5% 97.5%  
(Intercept) 35.6749790 41.0308115  
Edad.media.2007 -0.8261374 -0.6977012
```

8.2.2. Mediante código. La función *lm*

La sintaxis básica de la función *lm* es la siguiente:

```
lm(formula, data, subset)
```

- *formula* es la expresión que define el modelo. Ya hemos hablado de este tipo de expresiones, en las que debe aparecer la variable dependiente seguida del símbolo \sim y la variable independiente.
- *data* es una opción adicional que puede especificar la hoja de datos que queremos manejar.
- *subset* es también un parámetro opcional en el que podemos especificar si sólo queremos utilizar un subconjunto de casos para ajustar el modelo.

Por ejemplo, en el caso que nos ocupa tendríamos

```
ajuste<-lm(tasa.lineas.ADSL.2007~Edad.media.2007, data = Datos)  
summary(ajuste)
```

Eso devolvería los mismos resultados que R Commander.

8.3. Predicciones, estimaciones e intervalos de confianza para ellas

Recordemos que los valores que proporciona la recta de regresión para un valor dado de la variable independiente pueden interpretarse como predicciones del valor de la variable o como estimaciones de su promedio.

A modo de ejemplo, consideremos en primer lugar que un municipio andaluz tiene una edad media de 45 años, pero no se sabe la tasa de líneas ADSL por habitante. En ese caso, podemos *predecirla* con la recta de regresión.

De igual forma, pensemos ahora en la población hipotética o real de municipios andaluces de 45 años. ¿Cuál será la media de la tasa de líneas ADSL en estos municipios? También podemos *estimarla* con la recta de regresión.

Finalmente, tanto para estas *predicciones* como para estas *estimaciones*, podemos proporcionar intervalos de confianza al nivel que se considere apropiado, normalmente al 95 %.

8.3.1. Mediante R Commander

Para hacer todo esto con R Commander en primer lugar seleccionamos la opción del menú *Modelos* → *Selecciona el modelo activo*. Eso cargará el modelo de la recta de regresión que hemos calculado antes y lo utilizará para todos los cálculos posteriores, hasta que construyamos otro modelo distinto.

A continuación, seleccionamos *Modelos* → *Prediction intervals (HH)*. La ventana emergente (Figura 8.3) permite calcular el valor de la recta de regresión (*point estimate only*), el intervalo de confianza para el promedio dado un valor de la variable independiente (*confidence interval for mean*) o el intervalo de predicción para el valor dado de la variable independiente (*prediction interval for individual*). Estos análisis pueden realizarse para uno o varios valores, utilizando el *scroll* de la parte superior de la ventana.

En nuestro caso, para 45 años, el valor predecido o el valor promedio de líneas ADSL por cada 100 habitantes es de 4.066.

Si consideramos este valor como valor de predicción de un municipio con esa edad media, un intervalo de predicción que con un 95 % contiene a la verdadera tasa de líneas ADSL es $(-3.863147, 11.9962)$. Nosotros sabemos que es imposible que la tasa sea negativa, por lo que corregimos ese intervalo a $(0, 11.9962)$. ¡Es amplísimo! Proporciona poca información porque el objetivo de incluir al verdadero

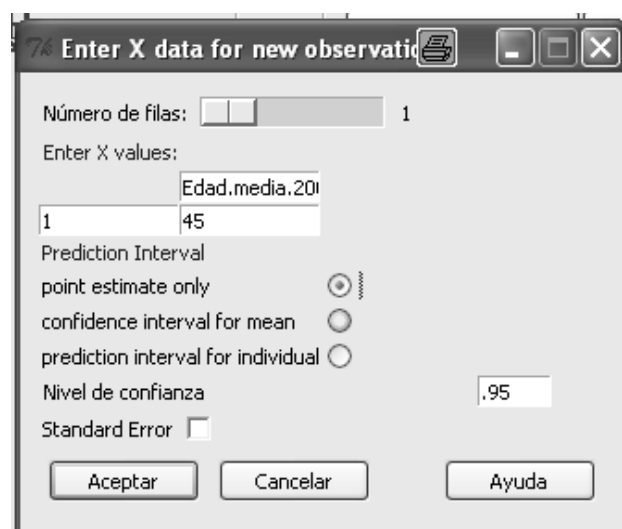


Figura 8.3: Obteniendo predicciones, estimaciones y sus intervalos de confianza

valor de la tasa con un 95 % de probabilidad es demasiado ambicioso, dadas las características de este ajuste. Necesitaríamos un R^2 superior para que el intervalo fuera más estrecho.

Finalmente, el intervalo de confianza al 95 % para el promedio de las líneas ADSL de los municipios de 45 años es (3.701712, 4.43134).

Hay una forma visual muy ilustrativa de observar las predicciones-estimaciones de la recta de regresión y los intervalos de predicción-confianza. Para verlo, elijamos la opción *Modelos* → *Confidence intervals plot*. En la ventana emergente debemos elegir de nuevo la variable dependiente y la variable independiente. El resultado es el que aparece en la Figura 8.4.

En el gráfico aparecen:

- Los valores observados en la muestra, en forma de un diagrama de dispersión.
- Los valores ajustados según el modelo, es decir, la recta de regresión.
- Los intervalos de confianza (al 95 %) para el promedio ajustado de la tasa de líneas ADSL dado cada valor de la edad media. En la leyenda se muestra que el color es verde, pero no se percibe en blanco y negro. Sin embargo, recordemos que estos intervalos son más estrechos que los de predicción, así que podemos deducir que son los más próximos a la recta de regresión.
- Los intervalos de predicción (al 95 %) para cada valor de la edad media. Los extremos de estos intervalos configuran las curvas más exteriores.

8.3.2. Mediante código. La función *predict*

La sintaxis básica de esta función es la siguiente:

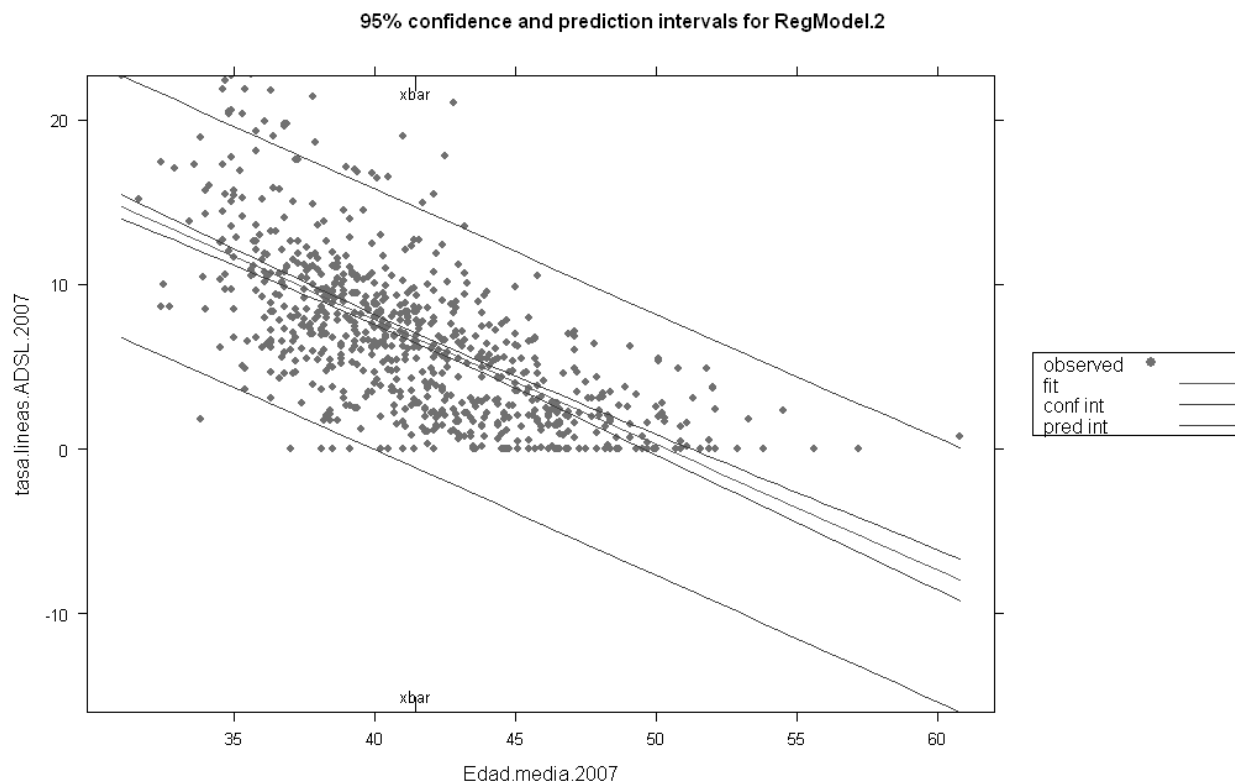


Figura 8.4: Gráfico de los intervalos de predicción y de confianza

```
predict(modelo, newdata, interval, level = 0.95)
```

- *modelo* se refiere al resultado devuelto por la función *lm*.
- *newdata* debe ser una hoja de datos que especifique los valores de la variable dependiente para los que queremos las estimaciones o predicciones.
- *interval* especifica si queremos intervalos de confianza (para el promedio estimado) o de predicción (para el valor predicho), mediante las opciones “confidence” o “prediction”.
- *level* es el nivel de significación de estos intervalos.

Por ejemplo, en nuestro caso sería

```
predict(ajuste, data.frame(Edad.media.2007=c(45)),
interval='prediction')
```

o

```
predict(ajuste, data.frame(Edad.media.2007=c(45)),
interval='confidence')
```

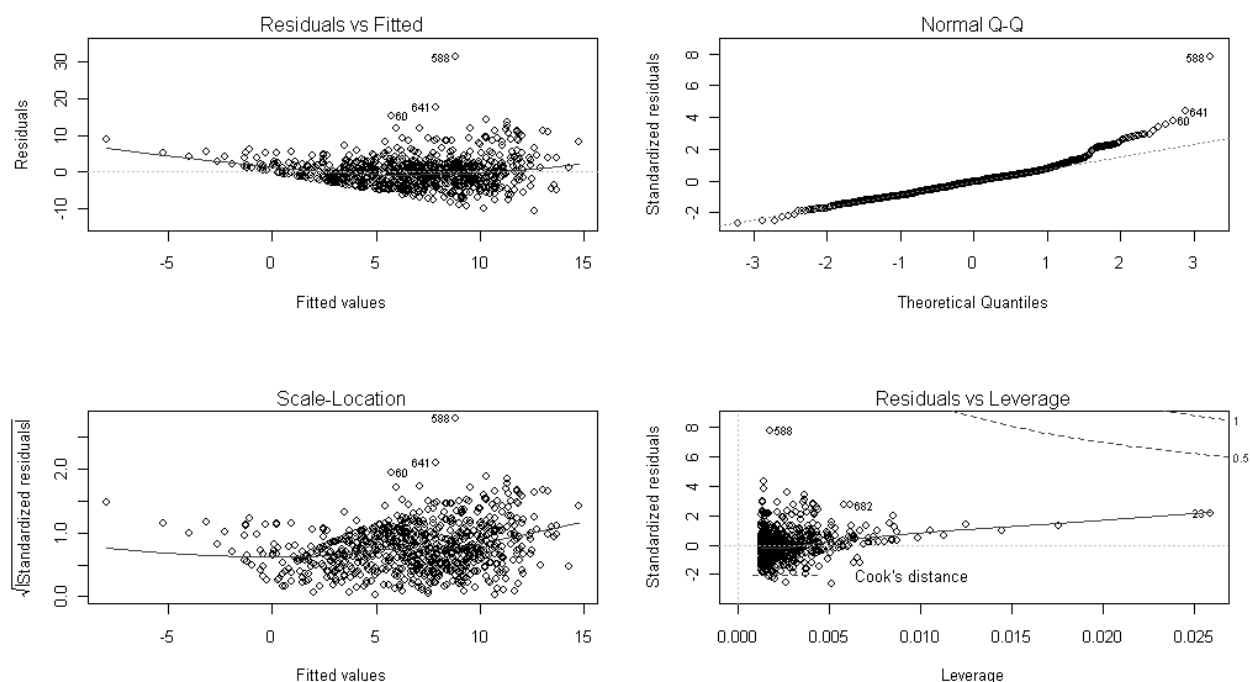



Figura 8.5: Diagn sis gr fica del modelo de regresi n lineal simple

A trav s de las salidas de `predict` se puede construir mediante c digo una gr fica como la que facilita R Commander en la Figura 8.4. Lo dejamos como ejercicio al lector interesado.

8.4. Algo sobre diagn sis del modelo

Para finalizar, resulta necesario concluir un cap tulo dedicado a la regresi n hablando, al menos de una forma introductoria, sobre la cuesti n de c mo evaluar las hip tesis b sicas que requiere el modelo para ser v lido. No vamos a detenernos con detalle sobre ello, pero s  podemos comentar algo al respecto, al menos brevemente.

8.4.1. Mediante R Commander

Las opciones m s b sicas para ello est n en *Modelos* \rightarrow *Gr ficas* \rightarrow *Gr ficas b sicas de diagn stico*.

De las 4 gr ficas que aparecen (ver Figura 8.5), s lo quiero comentar aqu  la de la esquina superior izquierda, la **gr fica de residuos frente a valores ajustados**. Se trata de una representaci n donde el eje X incluye los valores ajustados de la recta para cada municipio y el eje Y los residuos de dichos ajustes. Por ejemplo, Abila tiene una edad media de 44.3 a os y una tasa de l neas ADSL por cada 100 habitantes de 2.906. El valor ajustado de la recta para sus 44.3 a os es 4.59987, por lo que el residuo es $2.906 - 4.59987 = -1.69387$.

En el gráfico además, aparecen los valores atípicos del modelo y una curva, consistente, básicamente, en las medias muestrales de los residuos para pequeños grupos de valores individuales de los datos.

¿Qué sabemos que debe ocurrir con el modelo de regresión lineal simple?

- Las medias de los residuos deben ser cero para cualquier valor de la variable independiente. En nuestro gráfico vemos que la curva de medias no coincide exactamente con el eje X, que es el valor cero de los residuos. Eso supone una pequeña violación de la hipótesis requerida.
- Debe haber una tendencia lineal. En el diagrama de dispersión no se observa ningún patrón extraño no lineal en nuestro caso.
- La varianza de los residuos debía ser constante para todos los valores de la variable independiente. En el caso que analizamos hay claramente mayor dispersión (y por tanto, mayor varianza) en la parte de la derecha, lo que viola la hipótesis del modelo.

¿Qué debemos hacer, por tanto, teniendo en cuenta estos aparentemente decepcionantes resultados? En primer lugar, no lo son tanto, en realidad. Sólo la hipótesis de la varianza constante es claramente incumplida. Por otra parte, existen técnicas que permiten corregir este tipo de deficiencias de la construcción de los modelos, aunque exceden los contenidos de un curso como al que van dirigido estas notas.

Para terminar, observemos que aparecen señalados 3 datos, los municipios 60, 641 y 588. Esos son los que se consideran como valores atípicos del modelo debido a que sus residuos son muy elevados. Esos municipios son Mojácar (Almería), Ojén (Málaga) y (el que más destaca) Benahavís (Málaga). Desconocemos las peculiaridades de estos municipios, pero se trata de municipios pequeños que destacan sobre el resto por el elevado residuo, es decir, porque tienen muchísimas más líneas por habitante de lo que se espera según su edad media.

8.4.2. Mediante código

La función *lm* que proporciona el ajuste permite obtener las representaciones ligadas a la diagnosis del modelo de forma trivial mediante la función *plot*. Por ejemplo, en nuestro caso sería de la siguiente manera:

```
ajuste<-lm(Lineas.ADSL.2007~Edad.media.2007, data=Datos)
plot(ajuste)
```

Aparecerá una pantalla de gráficos. Clicando el ratón irán apareciendo los 4 gráficos.

Si lo que deseamos es que aparezcan los 4 gráficos juntos en una sola ventana, tendríamos que ejecutar las siguientes líneas:

```
ajuste<-lm(Lineas.ADSL.2007~Edad.media.2007, data=Datos)
par(mfrow=c(2,2))
plot(ajuste)
```

La línea `par(mfrow=c(2,2))` determina como parámetros gráficos la generación de una matriz de gráficos definidos por filas de dos filas y dos columnas.

8.5. Ejercicios

1. El director de una empresa quiere analizar, mediante un modelo de regresión lineal simple, la relación entre el número de trabajadores de cada sección de dicha empresa y las pérdidas por robo de producción. Para ello, a lo largo de 20 semanas contabilizó el número de trabajadores en distintos turnos de distintas secciones y las pérdidas. Los datos aparecen en el fichero *reg_ej1.txt*. Se pide:
 - a) Ajustar dicho modelo.
 - b) ¿Puede concluir el director que existe relación estadísticamente significativa entre el número de trabajadores y las pérdidas por robo? Utilícese un 5 % de significación para la respuesta.
 - c) Calcular el porcentaje de bondad de ajuste (R^2) y decidir si el modelo proporcionaría predicciones muy fiables de las pérdidas en función del número de trabajadores.
2. Una empresa está planteándose cambiar la maquinaria con la que fabrica un determinado producto y desea conocer el tiempo que necesitan sus empleados para aprender a usarla. Para ello selecciona 20 empleados con aproximadamente las mismas habilidades en la producción y los hace participar en cursos de entrenamiento de distinta duración. En los datos del ejercicio se contabilizan las horas de entrenamiento junto con el tiempo que, tras dicho entrenamiento, tardaron en realizar un proyecto estándar con la nueva maquinaria. Los datos aparecen en el fichero *reg_ej2.txt*. Se pide:
 - a) Ajustar un modelo de regresión lineal simple para explicar el tiempo para realizar el proyecto en función del número de horas de entrenamiento.

- b) ¿Podemos concluir, con un 95 % de confianza, que el tiempo de entrenamiento es relevante para el tiempo que se tarda en ejecutar el proyecto con la nueva maquinaria?
 - c) Realizar una predicción del tiempo que tardará un empleado que entrene 28 horas en ejecutar el proyecto, mediante un valor puntual y mediante un intervalo de predicción al 95 %.
 - d) La empresa decide finalmente que los empleados recibirán un entrenamiento de 30 horas. Realizar una estimación del promedio que tardarán los empleados en ejecutar los proyectos, mediante un valor puntual y mediante un intervalo de confianza al 95 %.
3. Una gran multinacional del sector de la automoción intuye que sus trabajadores de la cadena de montaje sufren problemas de espalda con el paso de los años. Para tratar de corroborar esta hipótesis, analiza la proporción de trabajadores con problemas diagnosticados de espalda en función del número de años que llevan trabajando en la cadena de montaje. Los datos aparecen en el fichero *reg_ej3.txt*. Se pide:
- a) A la luz de los datos, ¿podemos concluir con un 95 % de confianza que, en efecto, la proporción de trabajadores con problemas de espalda está relacionada significativamente con el número de años trabajados?
 - b) Si un trabajador lleva 30 años en la cadena de montaje, establecer una predicción puntual y mediante un intervalo al 95 % para la probabilidad de que padezca problemas de espalda.
 - c) ¿En qué medida (expresese en porcentaje) explica el número de años trabajados la probabilidad de sufrir daños de espalda?
4. Dada una muestra de hígado, medir su contenido en proteínas resulta caro y difícil. Por ello, los laboratorios utilizan el hecho de que la cantidad de proteína está relacionada con la cantidad de luz que absorbe la muestra para establecer una estimación de forma indirecta y más simple de la cantidad de proteína. En los datos del ejercicio aparece la cantidad real de proteínas de 20 muestras de hígado junto con la cantidad de luz que absorben. Los datos aparecen en el fichero *reg_ej4.txt*. Se pide:
- a) Establecer una medida del grado y el sentido de la relación lineal entre la cantidad de proteínas y la cantidad de luz absorbida.
 - b) ¿Es esta relación estadísticamente significativa? (Utilícese un 5 % de significación)

- c) Establecer un intervalo de predicción al 95 % para la cantidad de proteína que tendría una muestra con 1.00 de luz absorbida.
- d) Establecer un intervalo de confianza al 95 % para la cantidad promedio de proteína que contienen las muestras de hígado que absorben 1.00 de luz.

Bibliografía

- [1] Crawley, M. J. (2007). The R Book. Wiley.
- [2] Everitt, B. S. & Hothorn, T. (2006). A Handbook of Statistical Analyses Using R. Chapman & Hall/CRC.
- [3] Maindonald, J. & Braun, J. (2007). Data Analysis and Graphics Using R. Cambridge University Press.
- [4] Mendenhal, W & Sincich, T. (1997). Probabilidad y Estadística para Ingeniería y Ciencias (4ª edición). Prentice Hall.
- [5] Navidi, W. (2006). Estadística para ingenieros y científicos. McGraw-Hill.