



Introducción al uso de R-commander.

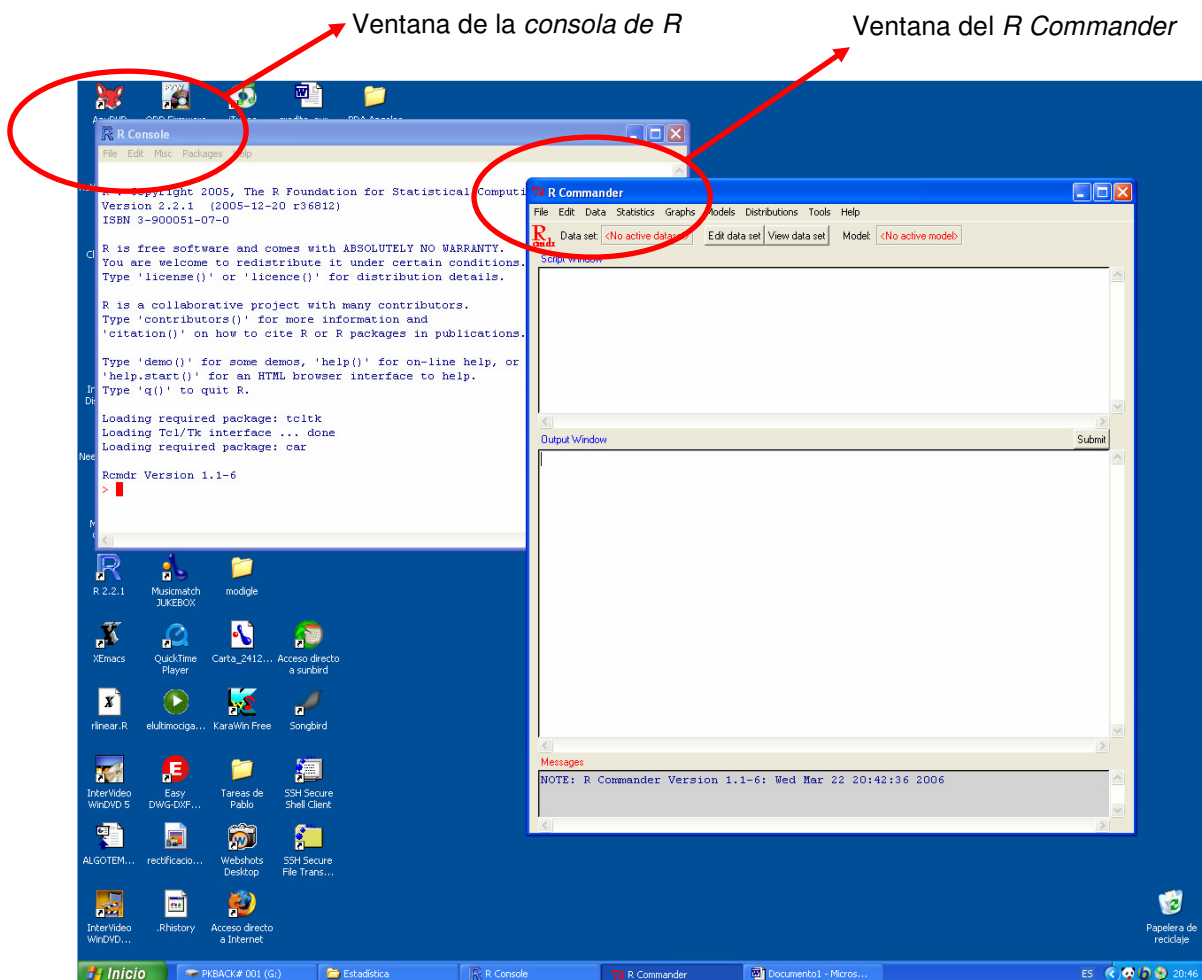
Angelo Santana

Índice

EL ENTORNO R-COMMANDER.....	2
LECTURA DE DATOS DESDE UN FICHERO EXTERNO.....	4
ESTADÍSTICOS DESCRIPTIVOS BÁSICOS CON R-COMMANDER	9
RESÚMENES POR GRUPOS: FACTORES.....	11
AMPLIACIÓN DE LAS CAPACIDADES DE R: BÚSQUEDA Y CARGA DE PAQUETES.....	14
OTRAS FUENTES DE AYUDA EN R:.....	21
UN EJEMPLO DE IMPLEMENTACIÓN DE RUTINAS DE CÁLCULO EN R..	24
GRÁFICOS EN R-COMMANDER.....	26
TABLAS DE FRECUENCIAS: VARIABLES DISCRETAS.....	27
TABLAS DE FRECUENCIAS: VARIABLES CONTINUAS	28
CÁLCULO DE LA RECTA DE REGRESIÓN	29

El entorno R-commander

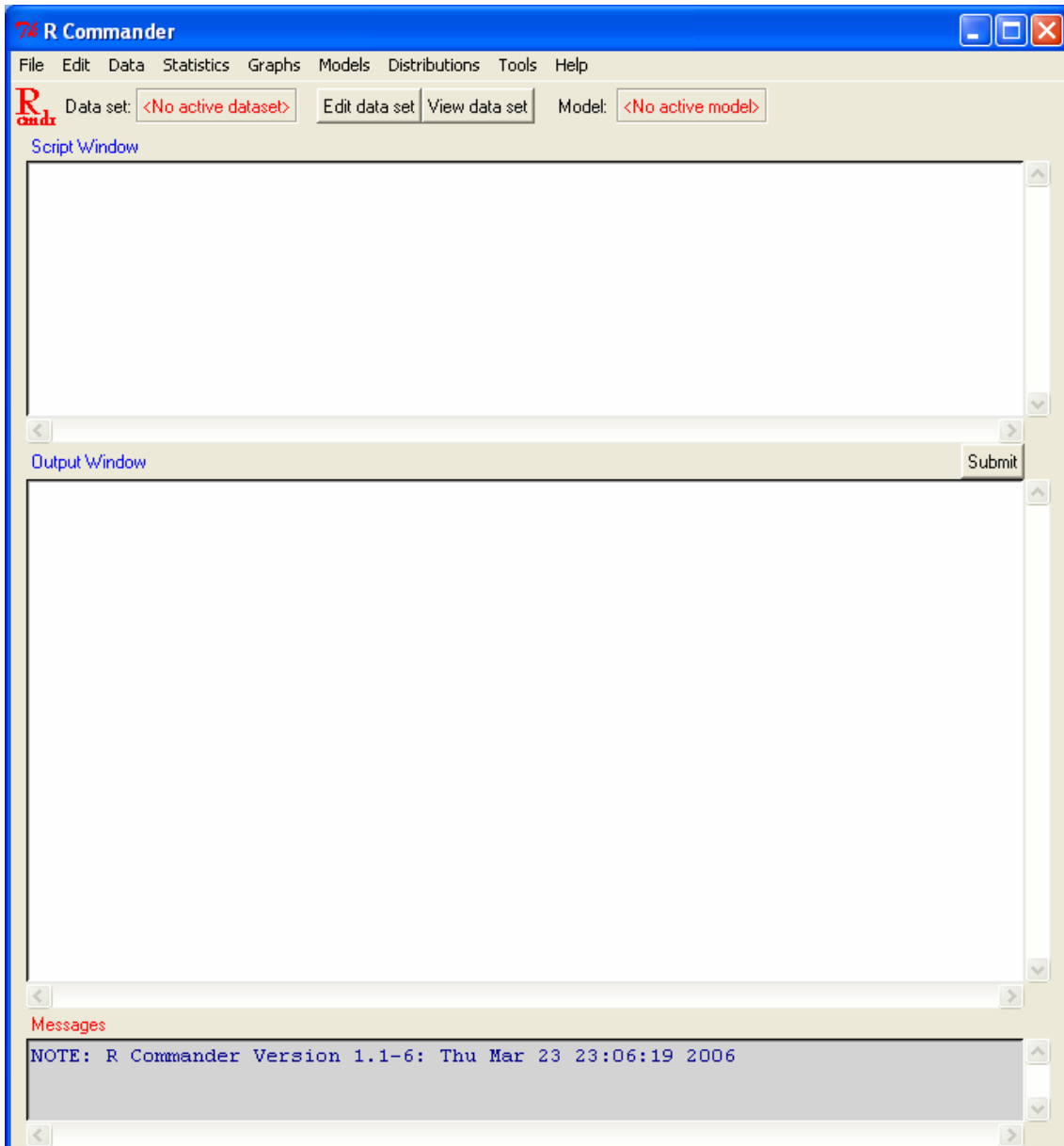
R-Commander es una Interfaz Gráfica de Usuario (GUI en inglés), creada por John Fox, que permite acceder a muchas capacidades del entorno estadístico R sin que el usuario tenga que conocer el lenguaje de comandos propio de este entorno. Al arrancar R-Commander, se nos presentan dos ventanas:



La ventana de la izquierda es la *consola* de R. Aquí podremos ejecutar comandos de R, para lo cual necesitamos conocer el lenguaje R y su sintaxis. La ventana de la derecha corresponde al entorno de *R-commander*, que nos evita precisamente tener que usar dicho lenguaje de comandos, al menos para las tareas que se encuentran implementadas dentro de dicho entorno.

No obstante, R-Commander no pretende ocultar el lenguaje R. Si observamos de cerca la ventana de R-Commander, vemos que se divide en tres subventanas: *script*, *output* y *messages*. Cada vez que, a través de los menús de R-commander accedamos a las capacidades de R (gráficos, procedimientos estadísticos, modelos, etc.), en la ventana *script* se mostrará el comando R que ejecuta la tarea que hayamos solicitado, y en la ventana *output* se mostrará el resultado de dicho comando. De este modo, aunque el usuario no conozca el

lenguaje de comandos de R, simplemente observando lo que va apareciendo en la ventana *script* se irá familiarizando (y con un poco de interés, también aprendiendo) con dicho lenguaje. Es más, el usuario puede introducir comandos directamente en dicha ventana, y tras pulsar el botón **Submit** dichos comandos serán ejecutados¹ y su resultado mostrado en la ventana *Output*. El *script* puede guardarse y volver a ser ejecutado directamente otras veces con otros conjuntos de datos diferentes, sin que el usuario tenga que desplazarse por todo el sistema de menús para volver a realizar las mismas tareas.



El acceso a las funciones implementadas en R-commander es muy simple y se realiza utilizando el ratón para seleccionar, dentro del menú situado en la

¹ Si se desea ejecutar un único comando basta con situar el cursor en cualquier punto del comando y pulsar **Submit**. Si se desean ejecutar varios comandos conjuntamente hay que seleccionarlos todos con el ratón y a continuación pulsar **Submit**.

primera línea de la ventana, la opción a la que queramos acceder. Las opciones son:

- **File:** para abrir ficheros con instrucciones a ejecutar, o para guardar datos, resultados, sintaxis, etc.
- **Edit:** las típicas opciones para cortar, pegar, borrar, etc.
- **Data:** Utilidades para la gestión de datos (creación de datos, importación desde otros programas, recodificación de variables, etc.)
- **Statistics:** ejecución de procedimientos propiamente estadísticos
- **Graphs:** gráficos
- **Models:** definición y uso de modelos específicos para el análisis de datos.
- **Distribution:** probabilidades, cuantiles y gráficos de las distribuciones de probabilidad más habituales (Normal, t de Student, F de Fisher, binomial, etc.)
- **Tools:** carga de librerías y definición del entorno.
- **Help:** ayuda sobre R-commander (en inglés).

Lectura de datos desde un fichero externo

Supongamos que hemos creado los datos con EXCEL (o equivalente), y que los hemos guardado desde EXCEL en formato CSV (Comma Separated Values)². Este formato es simplemente un formato de texto en el que los datos se guardan tal como se han introducido en EXCEL, separados por punto y coma, y sin que se añada ninguna información adicional (negritas, cursivas, colores de las letras, etc.). Para que R (y cualquier otro paquete estadístico) pueda utilizar los datos, éstos deben introducirse de modo que cada variable figure en una columna, y cada fila represente un caso. Asimismo, es conveniente que cada columna esté encabezada con el nombre de la variable. A modo de ejemplo, vemos la pantalla de EXCEL correspondiente a una muestra de peces, de cada uno de los cuales se ha determinado su estado de madurez sexual (0 = inmaduro, 1 = maduro) y sexo (1 = hembra, 2 = macho), midiéndose además su longitud y peso. Como se ve, cada columna corresponde a una variable, y cada fila corresponde a un sujeto (un pez en este caso).

A veces no se dispone del valor de alguna variable o variables en alguno o varios casos. Por ejemplo, pudiera haber peces cuyo estado de madurez o sexo no haya podido determinarse, o cuya longitud o peso se haya podido

² En EXCEL al guardar el fichero hay que seleccionar “guardar como” y elegir como tipo de fichero CSV (delimitado por comas). EXCEL nos informará de que sólo se guardará la hoja activa (deberemos aceptar), y a continuación nos avisará que el libro puede tener características no compatibles con CSV Deberemos elegir la opción SI, para que guarde definitivamente el fichero en ese formato

medir. Cuando falte algún dato, conviene introducir el valor NA, que R interpretará como *Not Assigned* (valor no asignado). En el ejemplo que se muestra a continuación vemos que hay valores perdidos en varias variables y varios casos, que han sido consignados con el valor NA.

	A	B	C	D	E	F	G	H
1	madurez	sexo	longitud	peso				
2		1	1	26,6670663	440,306603			
3		0 NA	1	25,6062287	419,851698			
4		0	1	16,1578212	312,124633			
5		1	1	20,4586932	351,303699			
6		1	2	28,4984989	486,283266			
7		1	1	27,2485734	455,881339			
8		0	2	22,2125252	387,45439			
9		1	1	27,2195466	435,584034			
10		1	1	30,4580328	511,879427			
11		0	1	32,6018949	485,360247			
12		1	2	37,0100762	588,687296			
13		1	2	23,3616035	436,202406			
14	NA		1	20,7847016	378,73805			
15		1 NA	1	31,5931125	498,453294			
16		0	2	25,0964858	414,105419			
17		1	1	22,6102399	422,253188			
18		0	1	26,8893059	NA			
19		1	2	26,9110293	416,978543			
20		0	2	28,9897047	440,772369			
21		0	2	27,9192934	431,317516			
22								
23								

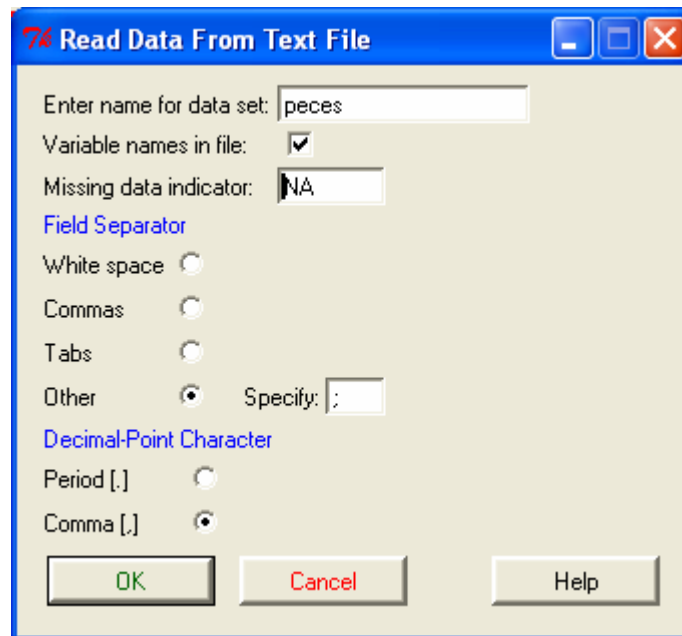
Para leer estos datos con R commander, una vez situados en la ventana de este programa hay que picar con el ratón en la opción *Data*, luego *Import data* y por último *from text file ...*:

Data > Import data > from text file ...

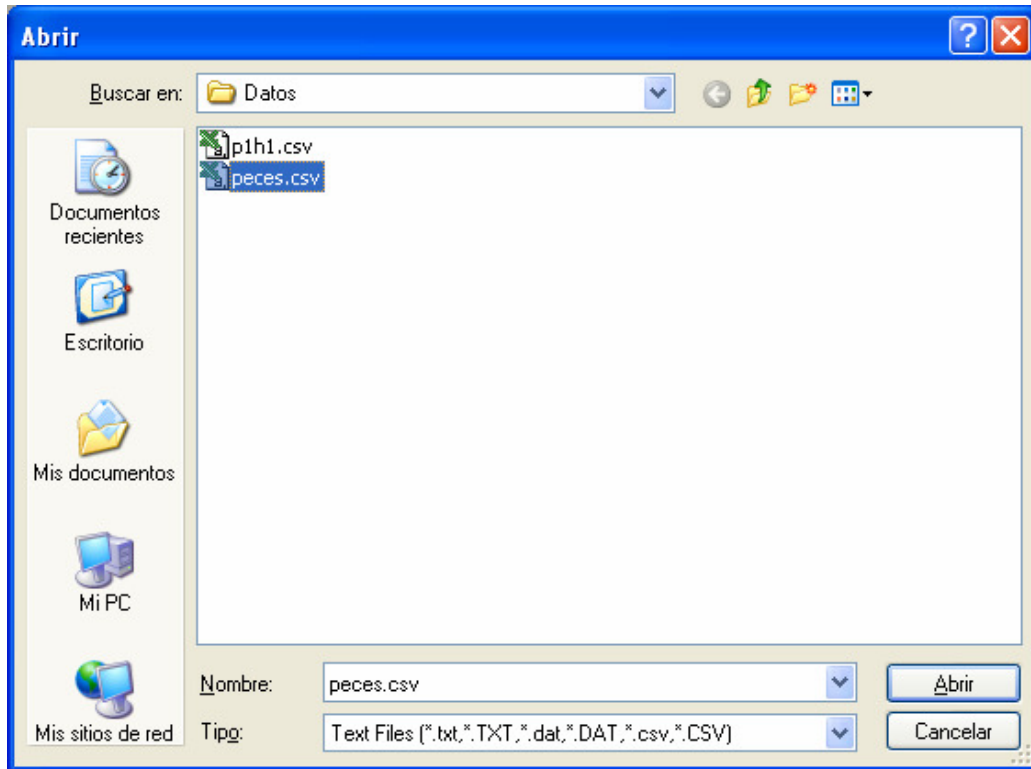
Nos aparecerá entonces el cuadro que se muestra en la página siguiente, en el que hay que especificar:

- Nombre que le queremos poner al conjunto de datos (**data set**) que vamos a analizar. Este nombre no tiene por qué coincidir con el nombre del fichero de datos. En este caso vamos a llamar **peces** a nuestro conjunto de datos.
- Como en el fichero están los nombres de las variables en la primera fila, marcamos la casilla **Variable names in file**.

- Por defecto en la casilla **Missing data indicator** aparece **NA**. Si hemos codificado de esta forma los valores perdidos, no hay que modificar esta asignación.
- Hay que especificar qué carácter separa los campos (**Field Separator**). Si hemos guardado los datos con EXCEL en formato CSV, el separador es el punto y coma. Hay que marcar **Other** y en el cuadro **Specify** poner ;
- Por último en **Decimal-Point Character** hay que especificar cual es el separador de cifras decimales. En los ordenadores con Windows en español el separador decimal es la coma.

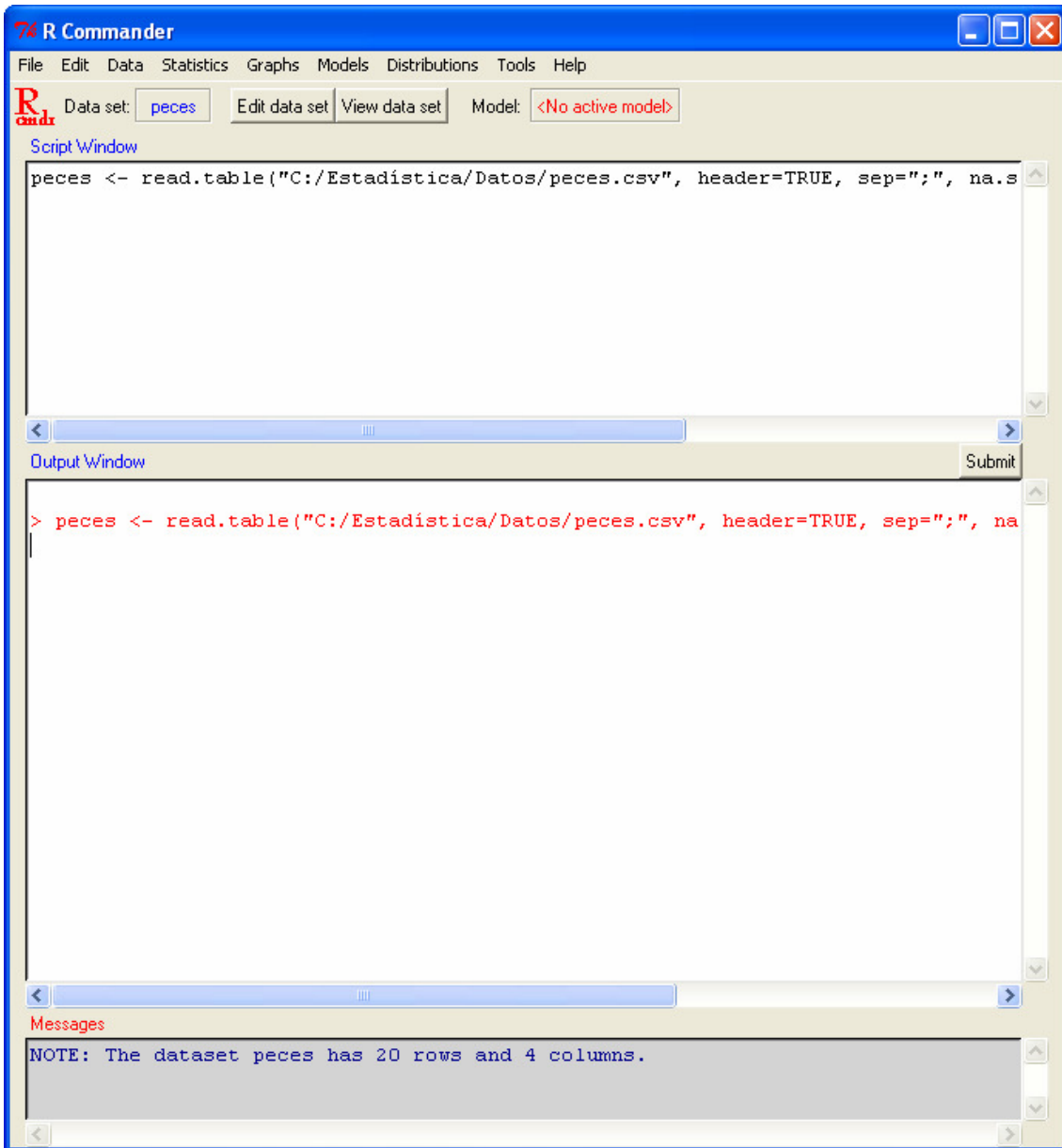


Una vez introducida esta información, picando en OK, nos aparece la ventana para abrir ficheros de Windows. Buscamos el directorio en que hemos guardado nuestro fichero de datos y lo seleccionamos:



En la pantalla de R-commander ha aparecido un comando en la ventana superior (script) y el mismo comando repetido en la ventana inferior (Output window). Este comando es concretamente:

```
peces <- read.table("C:/Estadística/Datos/peces.csv",  
header=TRUE, sep=";", na.strings="NA", dec="," ,  
strip.white=TRUE)
```



La sintaxis de este comando es fácil de entender: en el *data set* (conjunto de datos) *peces* se introduce el resultado de leer (`read.table`) el fichero `c:/Estadística/Datos/peces.csv`. Se indica que el fichero contiene los nombres de las variables en la cabecera (`header=TRUE`), que el separador de los datos es el punto y coma (`sep=";"`), que los valores perdidos se han codificado como NA (`na.strings="NA"`), que el separador de cifras decimales es la coma (`dec=","`) y que en caso de leer variables de tipo carácter (variables cuyos valores son alfanuméricos, por ejemplo, nombres de islas) se eliminen los espacios anteriores y posteriores al valor registrado en dichas variables (`strip.white=TRUE`).

En la ventana de salida (*Output*) no se observa ningún resultado ya que solamente se ha procedido a la lectura de los datos. Esto se nos indica en la ventana inferior (*Messages*) donde aparece una nota señalando que se ha leído el *dataset* *peces*, y que éste tiene 20 filas y 4 columnas.

Obsérvese también que en la ventana de R-commander, arriba a la izquierda, en el recuadro denominado **Data set** aparece el nombre de nuestro conjunto de datos activo, **peces**. Si deseamos ver los datos que se han leído, picamos el recuadro **View data set**, y si queremos editarlos para introducir alguna modificación picamos en el recuadro **Edit data set**

Estadísticos descriptivos básicos con R-Commander

Puede obtenerse un resumen de los estadísticos descriptivos elementales de todas las variables del conjunto de datos activo picando en:

Statistics > Summaries > Active data set

Para cada variable se nos muestra el valor mínimo de esta variable, el primer y tercer cuartil, la mediana y la media:

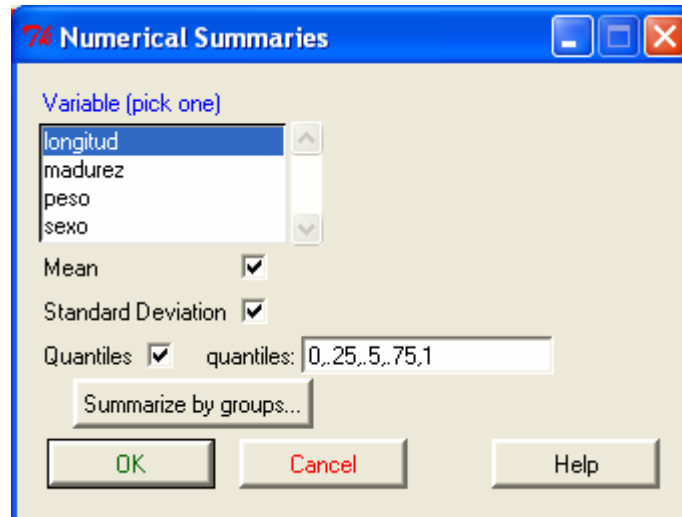
```
> summary(peces)
```

madurez	sexo	longitud	peso
Min. :0.000	Min. :1.000	Min. :16.16	Min. :312.1
1st Qu.:0.000	1st Qu.:1.000	1st Qu.:23.17	1st Qu.:415.5
Median :1.000	Median :1.000	Median :26.90	Median :435.6
Mean :0.579	Mean :1.444	Mean :26.41	Mean :437.6
3rd Qu.:1.000	3rd Qu.:2.000	3rd Qu.:28.62	3rd Qu.:470.6
Max. :1.000	Max. :2.000	Max. :37.01	Max. :588.7
NA's :1.000	NA's :2.000		NA's : 1.0

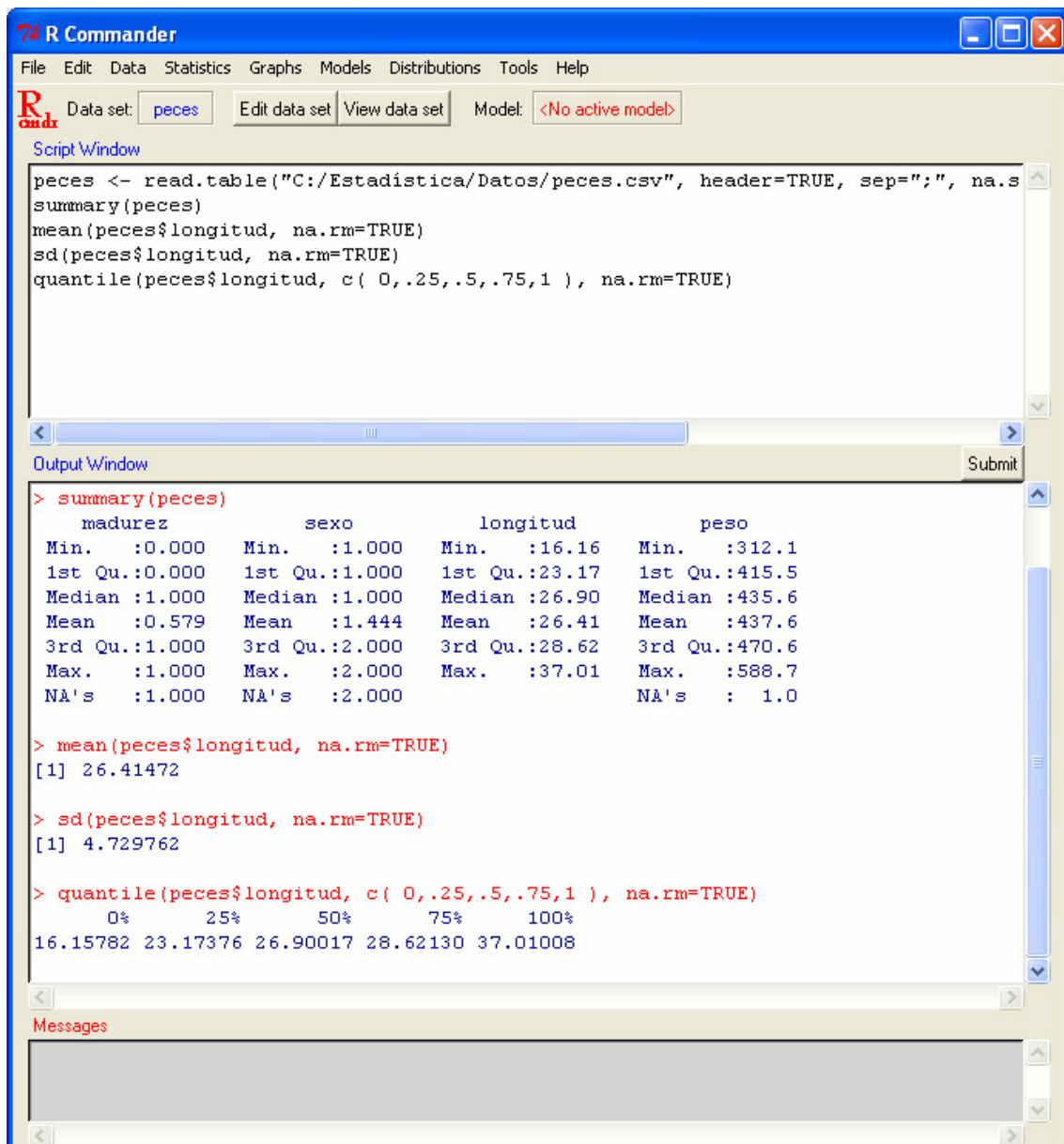
Si elegimos la opción:

Statistics > Summaries > Numerical summaries

nos pide elegir una variable (elegimos, por ejemplo, `longitud`). Por defecto nos calculará la media, la desviación típica y los cuantiles que le indiquemos en el recuadro correspondiente:



El resultado en este caso es:



Los valores obtenidos han sido:

```
> mean(peces$longitud, na.rm=TRUE)
[1] 26.41472

> sd(peces$longitud, na.rm=TRUE)
[1] 4.729762

> quantile(peces$longitud, c( 0,.25,.5,.75,1 ), na.rm=TRUE)
      0%      25%      50%      75%     100%
16.15782 23.17376 26.90017 28.62130 37.01008
```

Observemos la sintaxis del comando para el cálculo de la media:

```
mean(peces$longitud, na.rm=TRUE)
```

Como puede verse, para calcular la media se utiliza el comando `mean` (media en inglés). Como argumentos del comando figuran:

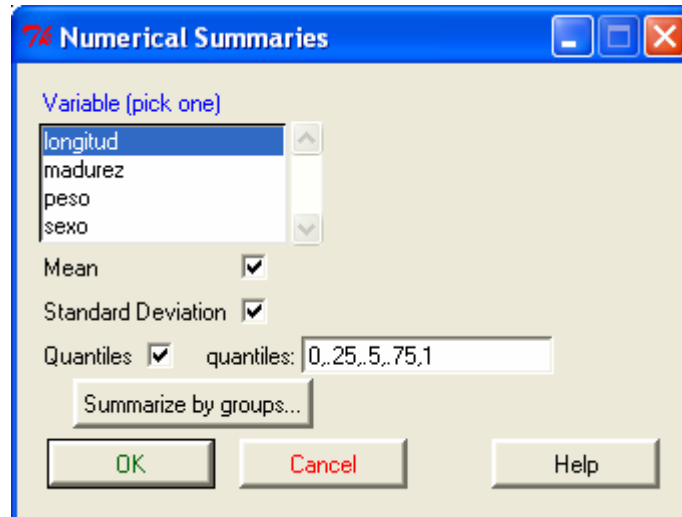
- el nombre de la variable cuya media se desea calcular, en este caso `peces$longitud`. Nótese que el nombre de la variable está compuesto por:
 - o El nombre del conjunto de datos: `peces`
 - o El símbolo `$`
 - o El nombre específico de la variable: `longitud`
- La expresión `na.rm=TRUE`. Esta expresión simplemente indica a R que si encuentra valores perdidos en la variable los elimine (`na` indica “no asignado”, `rm` significa “remove” (eliminar), y por tanto `na.rm=TRUE` viene a significar “Eliminar valores no asignados = True (verdad)”) y no los tenga en cuenta a la hora de calcular la media.

La sintaxis de los otros dos comandos es similar y no requiere mayor explicación.

Observemos en la pantalla anterior como la ventana superior (*script*) se va llenando con todos los comandos R necesarios para ejecutar las tareas que le hemos pedido. En la ventana inferior (*output window*) se repiten estos comandos y se muestran los resultados que generan.

[Resúmenes por grupos: Factores](#)

Supongamos ahora que quisiéramos obtener los estadísticos descriptivos anteriores desglosados por sexo y por grado de madurez. Si volvemos a picar en **Statistics > Summaries > Numerical summaries**, y en la pantalla que se nos muestra:



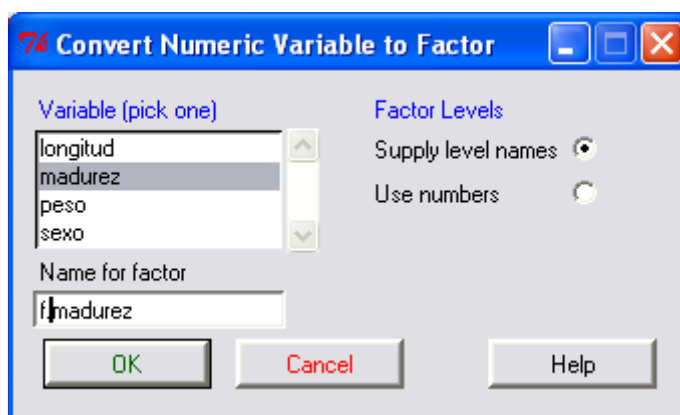
picamos en el recuadro `Sumarize by groups ...`, se produce un error y se nos muestra el siguiente mensaje en la ventana **Messages** :

ERROR: There no factors in the active data set.

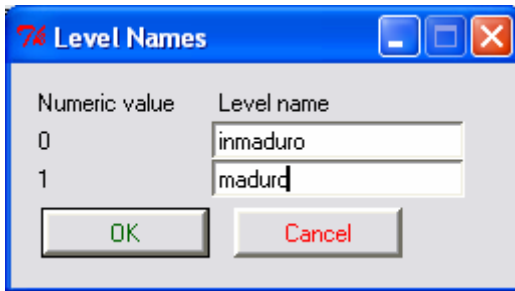
¿Qué ha ocurrido?. Simplemente que en este momento no hay ninguna variable dentro del conjunto de datos que R *entienda* que sirve para distinguir grupos de casos, por ejemplo machos y hembras. Las variables que sirven para clasificar los datos en grupos reciben el nombre de **factores**, y tienen un tratamiento especial en R. En primer lugar hay que informar al programa qué variables son las que debe considerar como factores. Para ello habremos de picar con el ratón en:

Data > Manage variables in active data set > Convert numeric variable to factor ...

y seleccionar las variables a convertir en factores. Seleccionamos primero la variable *madurez*, y como nombre para el factor especificamos *f.madurez*:



Si dejamos marcada la casilla `Supply level names`, tras picar OK nos solicita los nombres asociados a los niveles del factor:



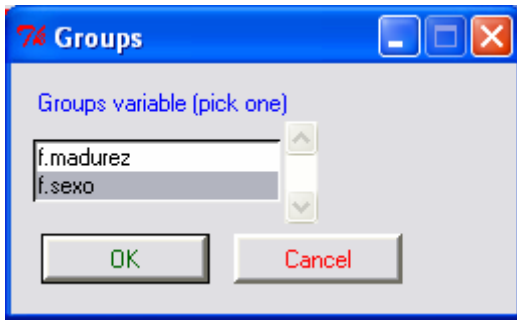
Procedemos de modo análogo con la variable *sexo*, creando el factor *f.sexo*, y asociando el valor 1 a las hembras y el 2 a los machos. Si ahora picamos en `View data set`, veremos que se han creado los nuevos factores:

	madurez	sexo	longitud	peso	f.madurez	f.sexo
1	1	1	26.66707	440.3066	maduro	hembra
2	0	NA	25.60623	419.8517	inmaduro	<NA>
3	0	1	16.15782	312.1246	inmaduro	hembra
4	1	1	20.45869	351.3037	maduro	hembra
5	1	2	28.49850	486.2833	maduro	macho
6	1	1	27.24857	455.8813	maduro	hembra
7	0	2	22.21253	387.4544	inmaduro	macho
8	1	1	27.21955	435.5840	maduro	hembra
9	1	1	30.45803	511.8794	maduro	hembra
10	0	1	32.60189	485.3602	inmaduro	hembra
11	1	2	37.01008	588.6873	maduro	macho
12	1	2	23.36160	436.2024	maduro	macho
13	NA	1	20.78470	378.7381	<NA>	hembra
14	1	NA	31.59311	498.4533	maduro	<NA>
15	0	2	25.09649	414.1054	inmaduro	macho
16	1	1	22.61024	422.2532	maduro	hembra
17	0	1	26.88931	NA	inmaduro	hembra
18	1	2	26.91103	416.9785	maduro	macho
19	0	2	28.98970	440.7724	inmaduro	macho
20	0	2	27.91929	431.3175	inmaduro	macho

Si ahora volvemos a

Statistics > Summaries > Numerical summaries

picamos en el recuadro `Sumarize by groups ...`, y en la ventana que aparece elegimos como variable de grupo *f.sexo*:



obtenemos como resultado las medias, desviaciones típicas y cuartiles por sexo:

```
> by(peces$longitud, peces$f.sexo, mean, na.rm=TRUE)
```

```
INDICES: hembra
[1] 25.10959
```

```
-----
INDICES: macho
[1] 27.4999
```

```
> by(peces$longitud, peces$f.sexo, sd, na.rm=TRUE)
```

```
INDICES: hembra
[1] 5.011954
```

```
-----
INDICES: macho
[1] 4.551939
```

```
> by(peces$longitud, peces$f.sexo, quantile, na.rm=TRUE, probs=c(
0, .25, .5, .75, 1 ))
```

```
INDICES: hembra
  0%    25%    50%    75%   100%
16.15782 21.24109 26.77819 27.24132 32.60189
```

```
-----
INDICES: macho
  0%    25%    50%    75%   100%
22.21253 24.66277 27.41516 28.62130 37.01008
```

[Ampliación de las capacidades de R: Búsqueda y carga de paquetes](#)

Quizás la mayor ventaja de R en comparación con otros paquetes estadísticos es su capacidad para “crecer”, incrementando el número de funciones, modelos, métodos, gráficos, etc. disponibles. Ello se consigue a través del concepto de **package** (paquete o librería de programas). Un *package* es básicamente un conjunto de funciones que se incorporan a R a través del comando `library()`. El número de paquetes disponibles para R se ha incrementado de forma exponencial en los últimos años, a medida que cada vez más usuarios desarrollan sus propios procedimientos, los incluyen en un paquete y los ponen a disposición pública en la página web de R (www.R-project.org).

En la actualidad es frecuente que los autores de artículos científicos en los que se describan nuevas metodologías estadísticas desarrollen el paquete de programas correspondiente para R, y lo pongan (de manera absolutamente gratuita) a disposición de todo el mundo a través de la citada web. En cualquier caso, debe señalarse que los paquetes disponibles no son necesariamente desarrollo de nuevas metodologías. El propio R-Commander es un paquete cuyo objetivo es presentar una interfaz gráfica simple para comunicarse con R. Otros paquetes son ampliaciones con funciones o gráficos estándar que en su momento no se incluyeron en el núcleo o paquete base de R.

A modo de ejemplo, podemos observar que en los menús de estadística descriptiva de R-commander no figuran ni la asimetría ni la curtosis, ya que estas medidas de síntesis de datos no se incluyeron en su día en el paquete base de R, ni en otros paquetes que usa R-commander (de manera invisible para el usuario). ¿Cómo podemos averiguar si existe algún paquete que contenga estas funciones? Y si existe, ¿cómo accedemos a él? ¿Cómo conseguimos que R-commander ejecute estas funciones?

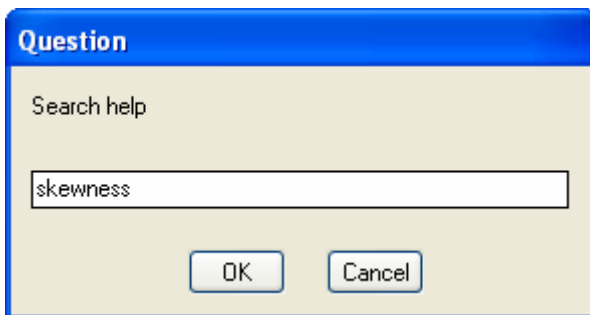
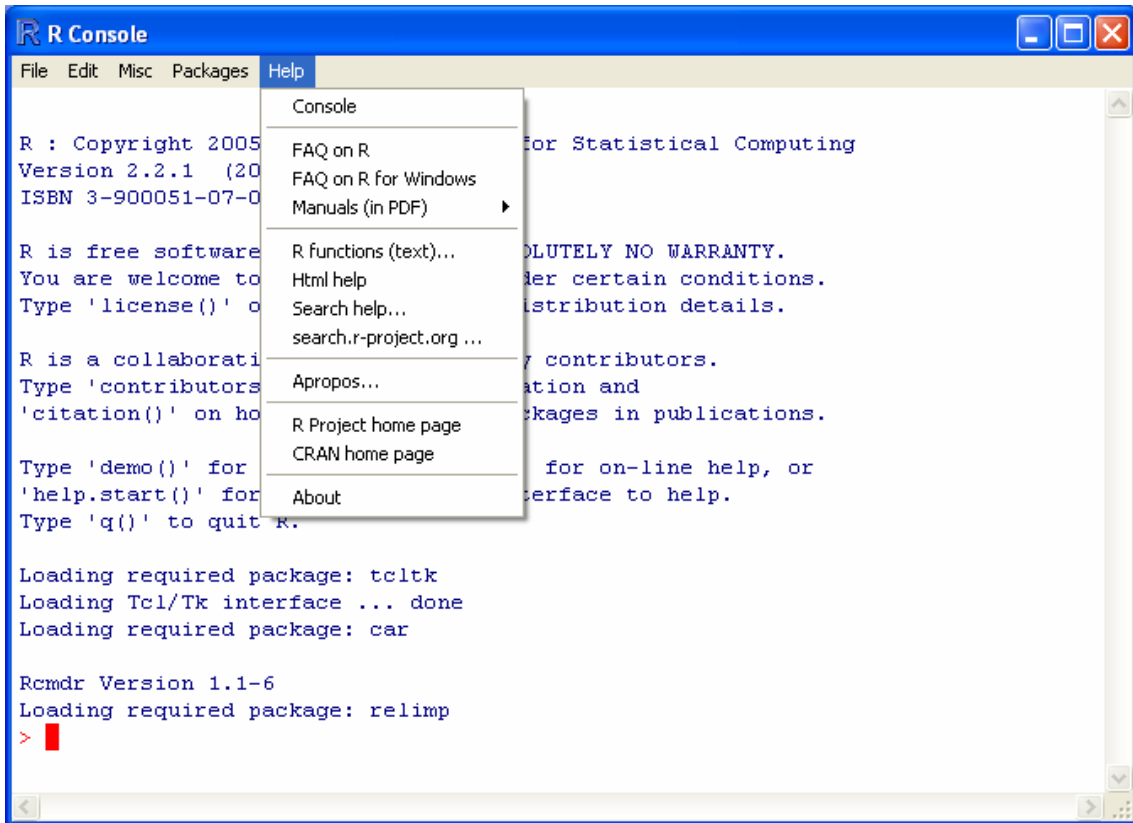
Para responder a la primera pregunta (¿cómo saber si existe un paquete con estas funciones?) es necesaria cierta dosis de intuición, cierto conocimiento del idioma inglés, y a veces acceso a internet, ya que la documentación y el sistema de ayuda que trae R por defecto pueden resultar insuficientes. El primer lugar donde podemos buscar es en la *consola* de R (la primera ventana que se abrió al arrancar R-Commander y que no hemos usado para nada hasta el momento). Supongamos que sabemos (o buscamos en un diccionario) que la palabra inglesa para asimetría es *skewness*. En principio, para averiguar si hay algún paquete que calcule el coeficiente de asimetría, tenemos dos opciones de búsqueda:

1. En el *prompt* de la consola (el símbolo `>`, donde R espera que introduzcamos un comando) introducimos el comando

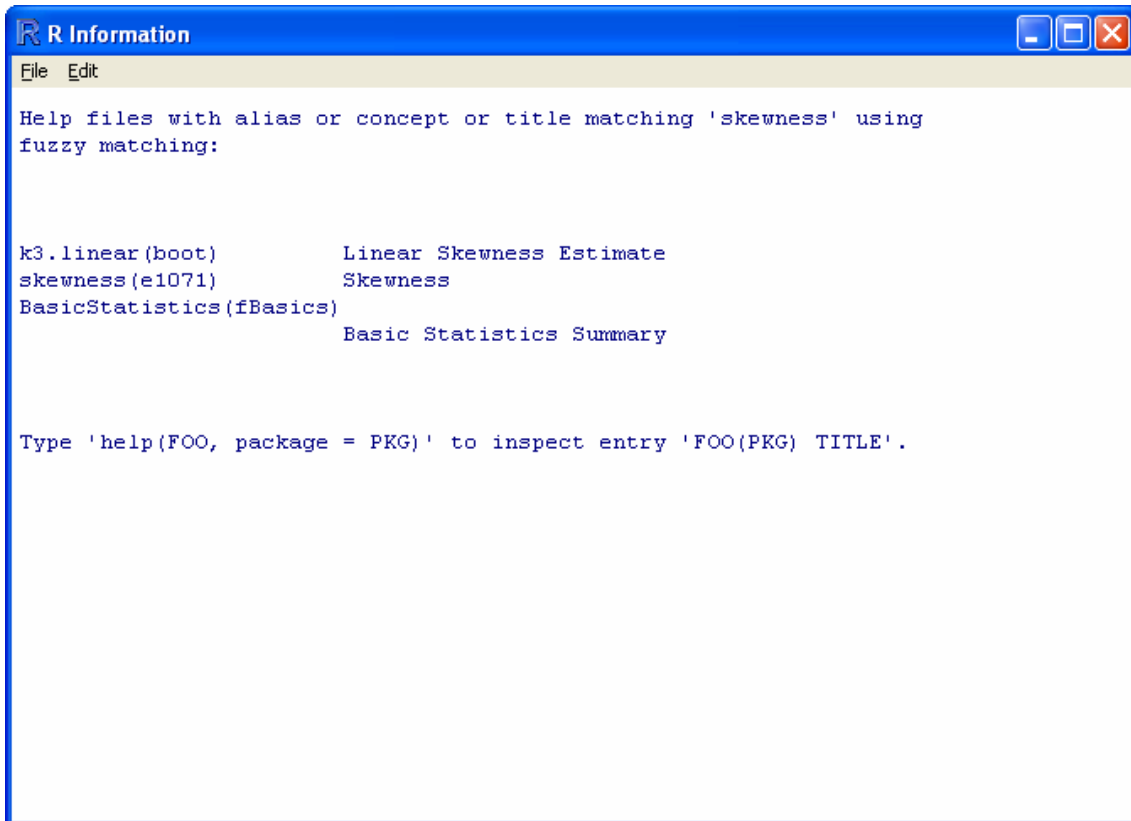
```
> help.start("skewness")
```

y pulsamos la tecla *enter*.

2. En la línea de menú superior de la consola picamos en **Help** y en el submenú que se despliega elegimos la opción `Search help` tal como se muestra en la siguiente pantalla. En la ventana que se nos abre a continuación escribimos `skewness` (en este caso no hacen falta las comillas):



Usando cualquiera de los dos métodos, R nos devuelve la siguiente pantalla a modo de respuesta:



The screenshot shows a window titled "R Information" with a menu bar containing "File" and "Edit". The main content area displays the following text:

```

Help files with alias or concept or title matching 'skewness' using
fuzzy matching:

k3.linear(boot)          Linear Skewness Estimate
skewness(e1071)         Skewness
BasicStatistics(fBasics)
                        Basic Statistics Summary

Type 'help(FOO, package = PKG)' to inspect entry 'FOO(PKG) TITLE'.

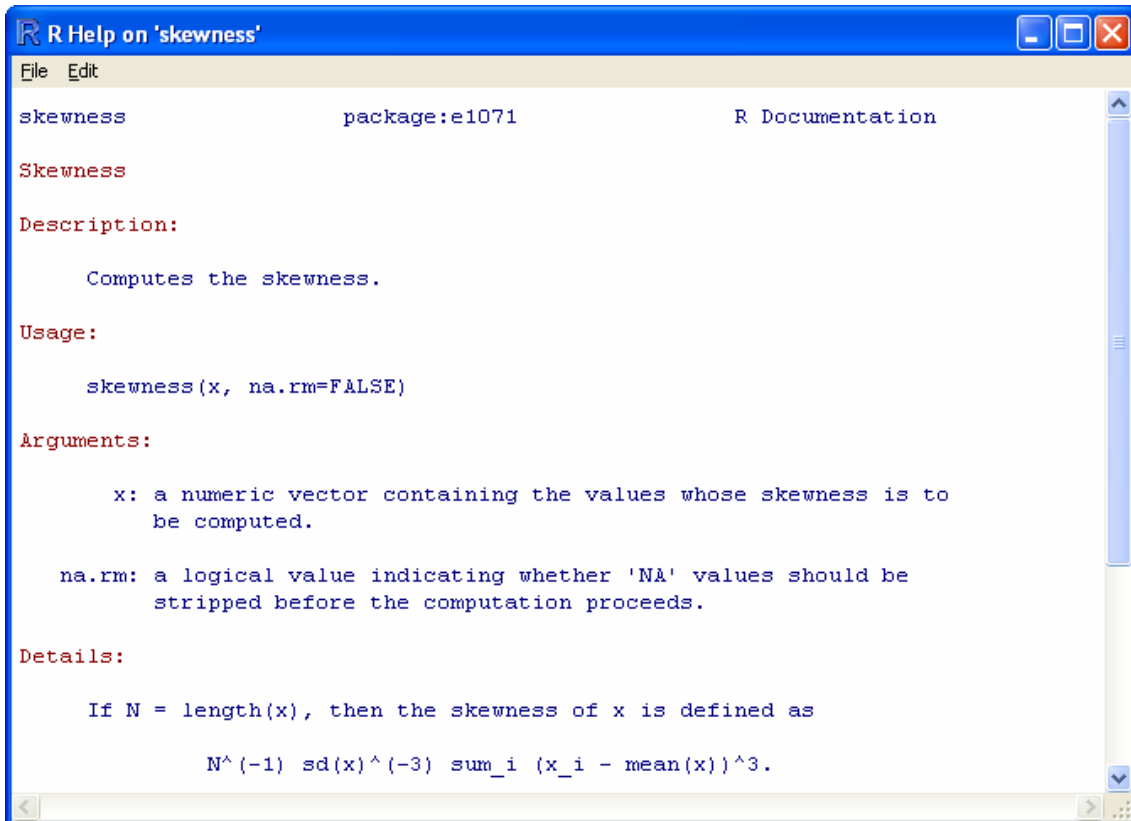
```

Esta pantalla nos indica que la función *skewness* está definida en el paquete `e1071`. Asimismo nos informa también que el paquete `fBasics` contiene funciones para la realización de resúmenes estadísticos básicos entre los cuales probablemente se encuentre la asimetría.

Si volvemos a la consola y tecleamos

```
> help(skewness, package=e1071)
```

se nos muestra la siguiente ventana de ayuda, con la sintaxis del comando `skewness` dentro del paquete `e1071`:

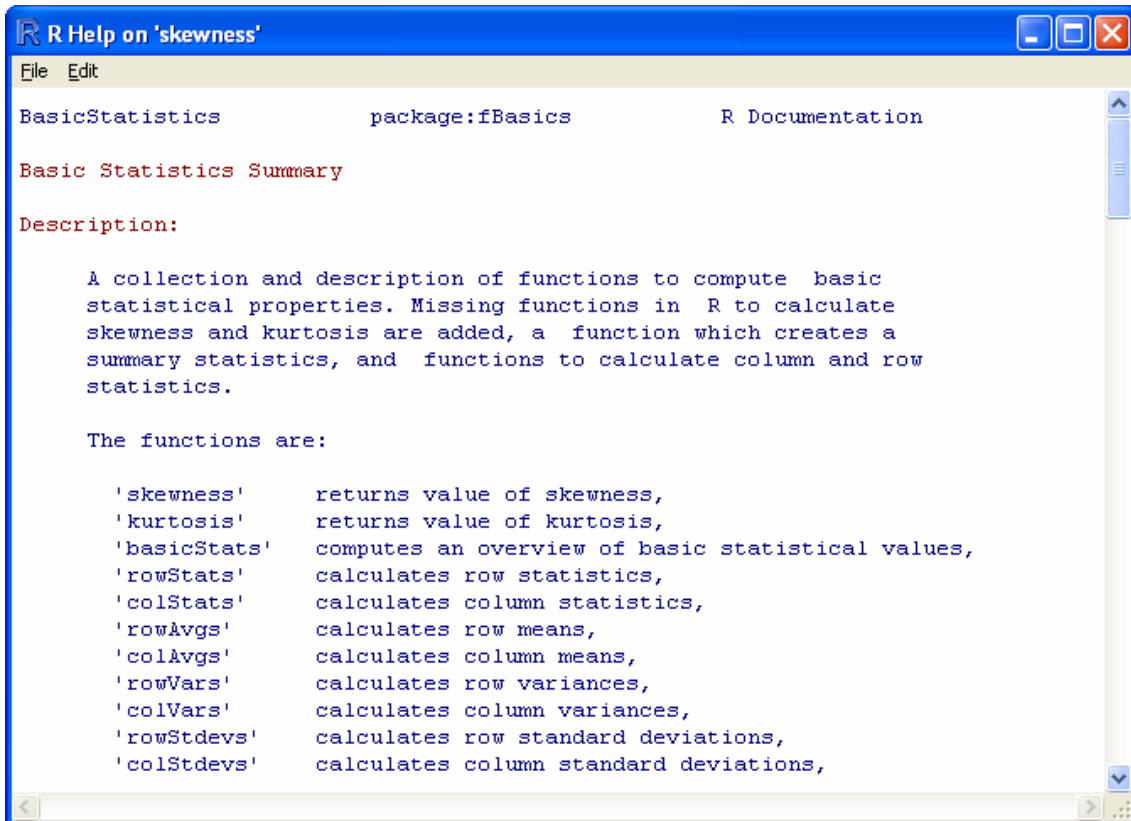


Asimismo, si tecleamos:

```
> help(skewness, package=fBasics)
```

(OJO CON LAS MAYÚSCULAS Y LAS MINÚSCULAS. LOS NOMBRES DE FUNCIONES Y PAQUETES DEBEN ESCRIBIRSE TAL COMO SE NOS INDICA, YA QUE R DISINGUE ENTRE MAYÚSCULAS Y MINÚSCULAS)

se nos muestra la siguiente ventana de ayuda, con la sintaxis del comando `skewness` dentro del paquete `fBasics`: (notemos que este paquete incluye no sólo la asimetría sino también la curtosis; y si nos fijamos en la pantalla de ayuda, vemos que además en `fBasics` está definida la función `basicStats` que según parece, calcula una colección de valores estadísticos básicos)



Ya sabemos, por tanto, donde encontrar funciones para calcular la asimetría y la curtosis. ¿Cómo conseguir ahora que R-commander las calcule sobre nuestros datos? Simplemente, volvemos a la ventana de R-commander y en la ventana superior (*script*) escribimos el comando:

```
library(e1071)
```

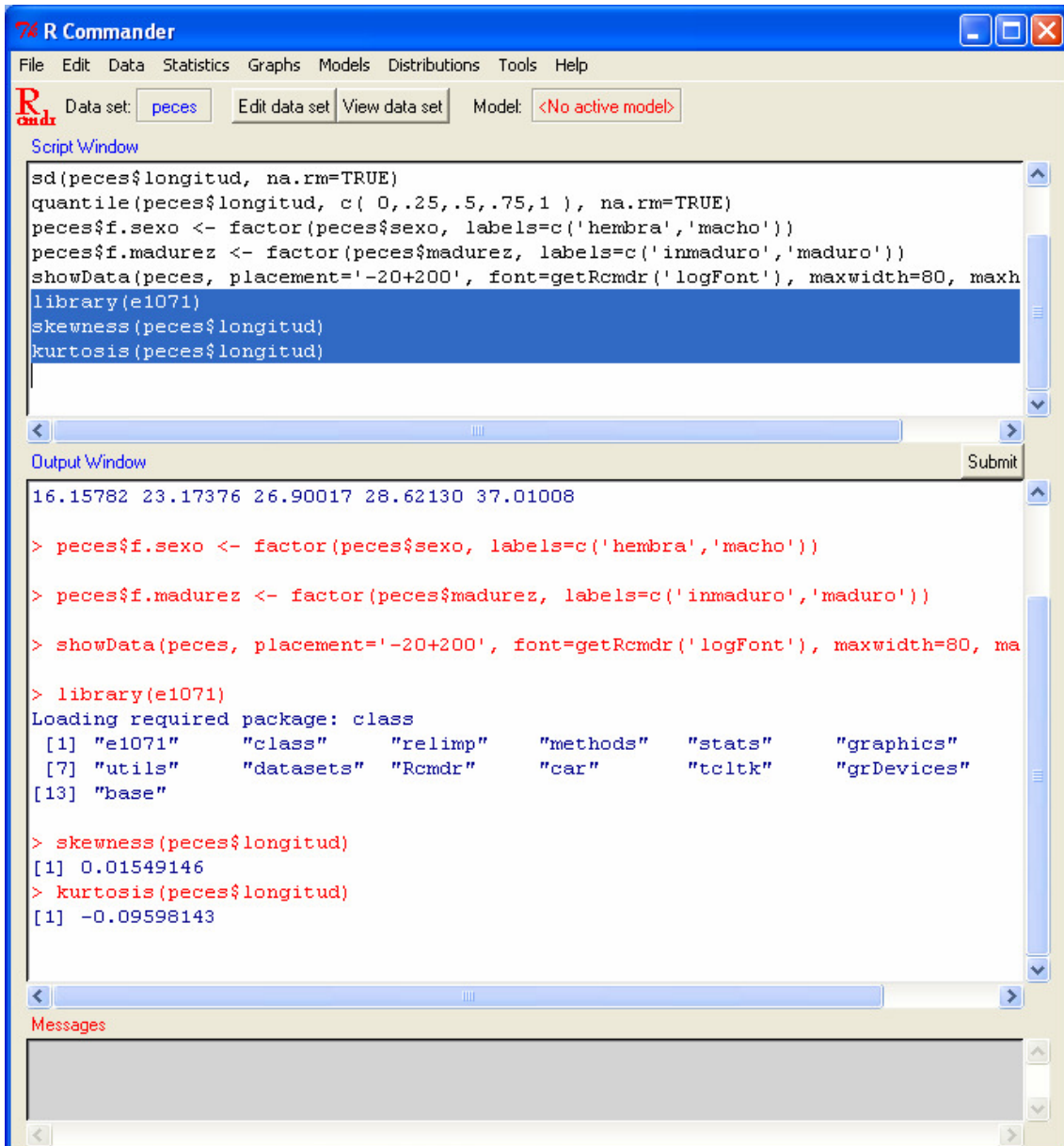
y a continuación picamos en **Submit** (en el centro a la derecha). En la ventana *output* aparece el siguiente resultado:

```
> library(e1071)
Loading required package: class
 [1] "e1071" "class" "relimp" "methods" "stats" "graphics"
 [7] "utils" "datasets" "Rcmdr" "car" "tcltk" "grDevices"
[13] "base"
```

que indica que el paquete `e1071` ha sido leído correctamente. Si nuevamente en la ventana superior (*script*) tecleamos:

```
skewness(peces$longitud)
kurtosis(peces$longitud)
```

y picamos en **Submit** obtenemos el siguiente resultado:



The screenshot shows the R Commander window with the following content:

Script Window

```
sd(peces$longitud, na.rm=TRUE)
quantile(peces$longitud, c( 0,.25,.5,.75,1 ), na.rm=TRUE)
peces$f.sexo <- factor(peces$sexo, labels=c('hembra','macho'))
peces$f.madurez <- factor(peces$madurez, labels=c('inmaduro','maduro'))
showData(peces, placement='-20+200', font=getRcmdr('logFont'), maxwidth=80, maxh
library(e1071)
skewness(peces$longitud)
kurtosis(peces$longitud)
```

Output Window

```
16.15782 23.17376 26.90017 28.62130 37.01008

> peces$f.sexo <- factor(peces$sexo, labels=c('hembra','macho'))
> peces$f.madurez <- factor(peces$madurez, labels=c('inmaduro','maduro'))
> showData(peces, placement='-20+200', font=getRcmdr('logFont'), maxwidth=80, ma
> library(e1071)
Loading required package: class
 [1] "e1071"      "class"      "relimp"     "methods"    "stats"      "graphics"
 [7] "utils"     "datasets"  "Rcmdr"     "car"        "tcltk"     "grDevices"
[13] "base"

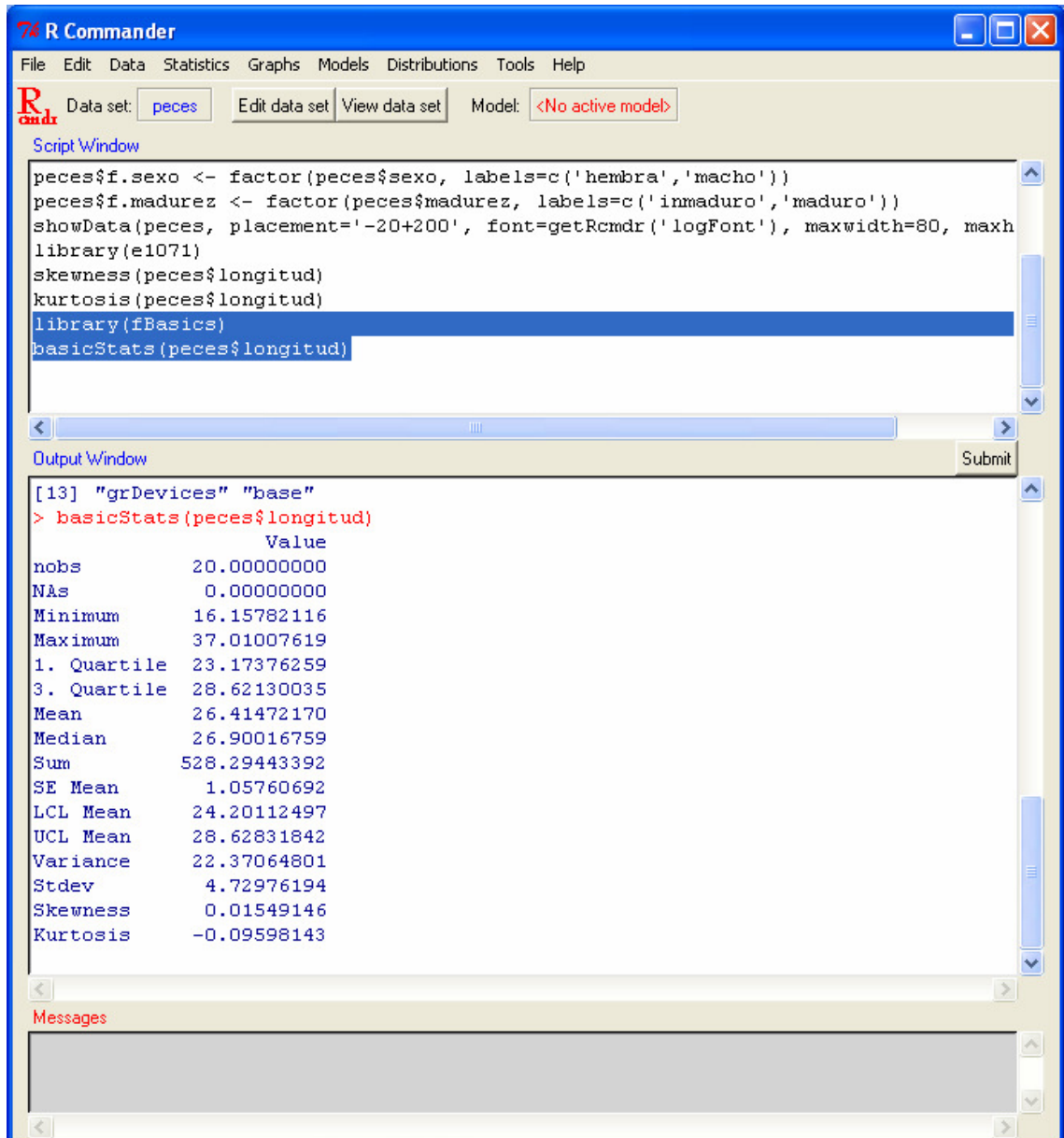
> skewness(peces$longitud)
[1] 0.01549146
> kurtosis(peces$longitud)
[1] -0.09598143
```

Messages

También podríamos usar la librería `fBasics`, con la función `basicStats`. Para ello en la ventana *script* tecleamos:

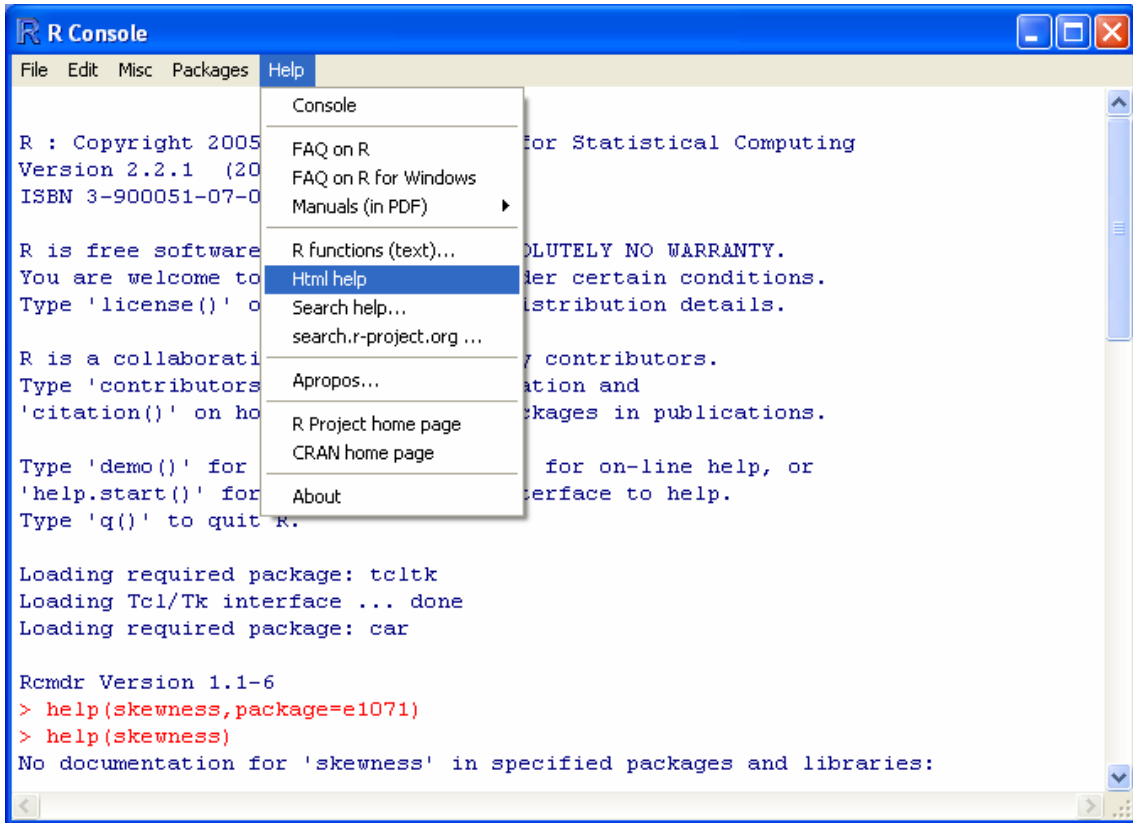
```
library(fBasics)
basicStats(peces$longitud)
```

marcamos ambas líneas con el ratón y picamos en **Submit**. El resultado obtenido es:

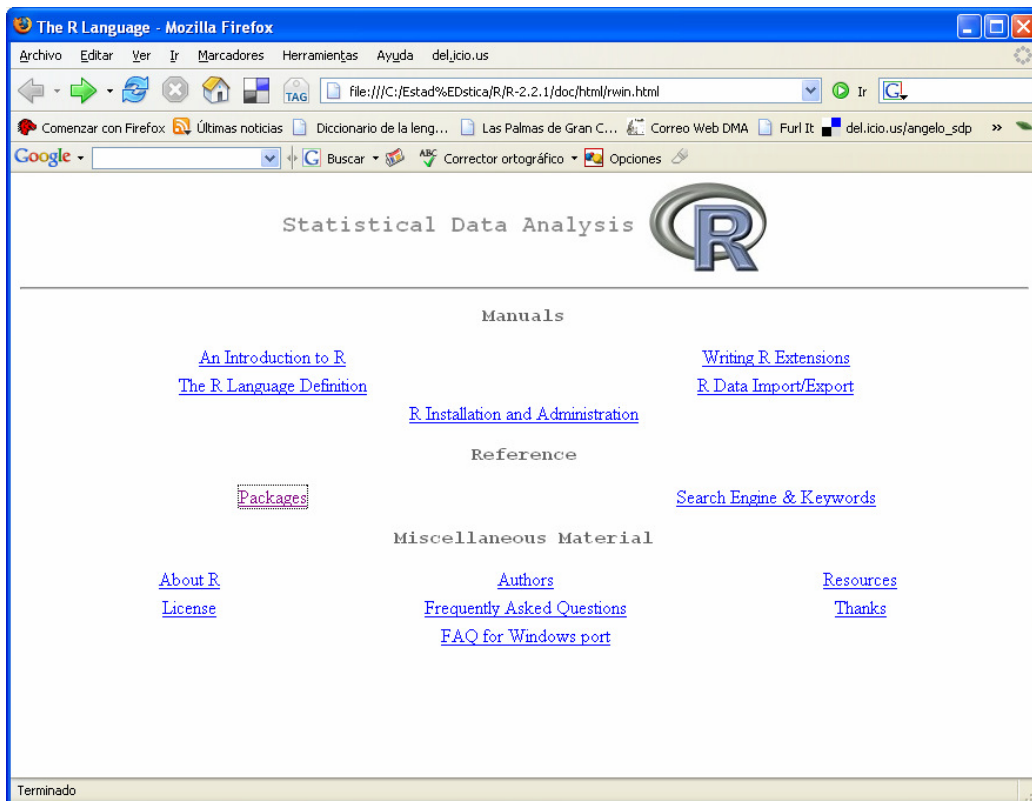


Otras fuentes de ayuda en R:

Podemos acceder a la ayuda de todos los paquetes de R que tengamos disponibles localmente a través de cualquier explorador (Mozilla, Netscape, Internet Explorer, etc.). Para ello, desde la ventana de *consola* de R picamos en Help y elegimos Html Help en el menú que se despliega (o directamente en el *prompt* tecleamos `help.start()` y le damos a enter):

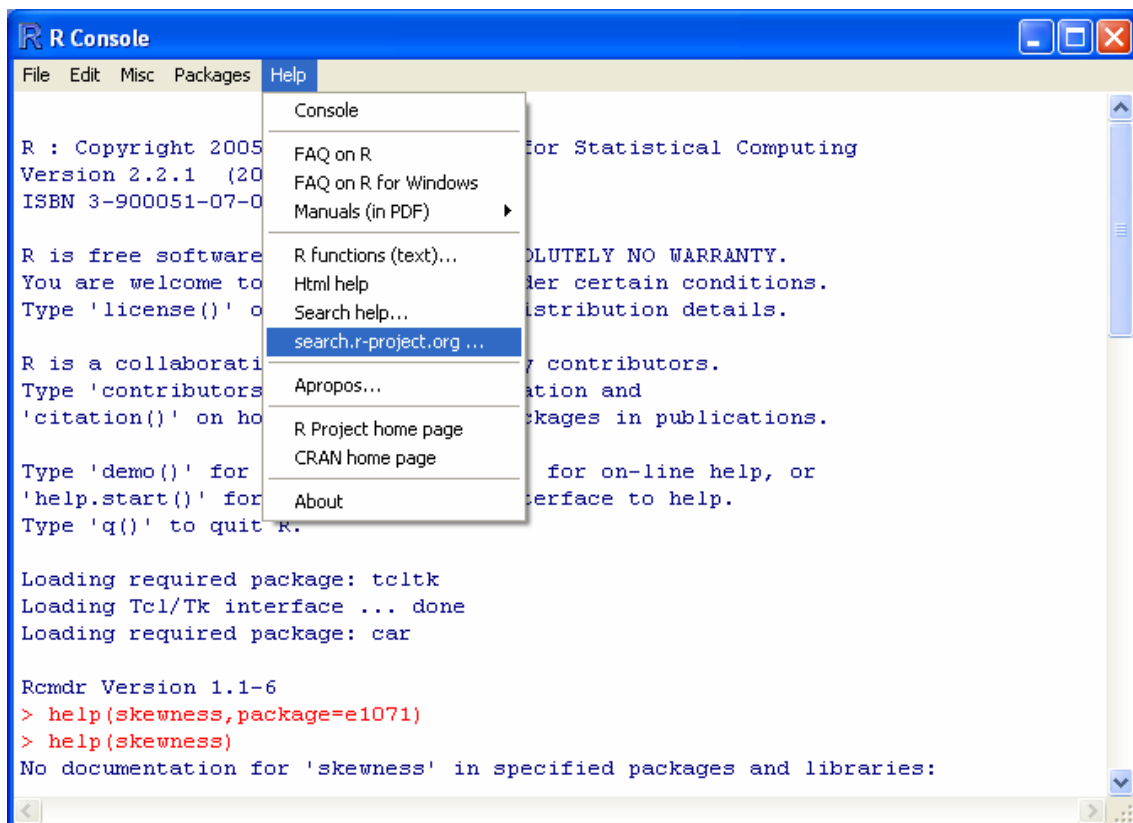


Esto arrancará el explorador que tengamos definido por defecto en nuestro PC, que nos llevará al siguiente menú de ayuda:



A través de este menú podemos acceder a ayuda sobre las funciones en los paquetes instalados, así como a los manuales básicos de R y a otro material.

Asimismo, como se ha dicho más arriba, a veces la ayuda que ofrece R localmente (en nuestro propio PC) puede ser insuficiente, por no encontrar la función o procedimiento que nos hace falta. En ese caso, si disponemos de conexión a internet, de nuevo desde la consola de R, picamos en `Help` y a continuación en `search r-project.org` y en la ventana que se abre escribimos la palabra sobre la que buscamos ayuda (o bien escribimos directamente en la consola `RSitesearch("foo")`, donde `foo` es la palabra sobre la que buscamos ayuda):



Esto nos conecta a la página web de R, en la que se muestran distintos enlaces, dentro del sitio de R, en los que aparece el término que estamos buscando. Con un poco de suerte podemos encontrar algún paquete o procedimiento que nos permita resolver nuestro problema. Y en el peor de los casos, como R es también un potente lenguaje de programación ¡siempre podremos programar nuestra propia función! (y si resulta útil enviarla al sitio de R para que la puedan utilizar otros usuarios. Así es como está creciendo este sistema).

Un ejemplo de implementación de rutinas de cálculo en R

Supongamos que deseamos calcular la desviación media absoluta de un conjunto de datos, y no hemos encontrado ninguna función que lo haga. La desviación media absoluta se define como:

$$D = \frac{\sum_{i=1}^n |x_i - \bar{x}|}{n-1}$$

Si queremos calcular este valor solamente para la variable `peces$longitud` podríamos introducir el siguiente código en la ventana *script* de R-commander:

```
x = peces$longitud
n = length(x)
sum(abs(x-mean(x)))/(n-1)
```

La primera línea simplemente crea un nuevo vector `x` con los mismos valores que `peces$longitud`³ (la única razón de hacer esto es escribir menos en las líneas siguientes, ya que el nombre de la variable resulta muy largo). La segunda línea determina la longitud del vector `x`, que coincide precisamente con el tamaño de la muestra. Por último la tercera línea calcula la desviación media absoluta: suma los valores absolutos de las diferencias entre los valores observados y la media de todos ellos y divide por el tamaño de la muestra menos uno. Obsérvese que la notación es muy clara.

Si marcamos estas tres líneas con el ratón y picamos en **Submit** obtenemos el resultado siguiente:

```
> x = peces$longitud (las 20 longitudes de peces observados en la muestra)
[1] 26.66707 25.60623 16.15782 20.45869 28.49850 27.24857 22.21253
27.21955
[9] 30.45803 32.60189 37.01008 23.36160 20.78470 31.59311 25.09649
22.61024
[17] 26.88931 26.91103 28.98970 27.91929
> n = length(x) (tamaño de la muestra)
[1] 20
> sum(abs(x-mean(x)))/(n-1) (desviación media absoluta)
[1] 3.687313
```

³ Señalemos aquí que, en lo que atañe a R, la variable `peces$longitud` es un vector cuyos valores son las 20 longitudes observadas en los 20 peces de la muestra.

Si copiamos y pegamos otra vez estas tres líneas cambiando el nombre de la variable, podemos repetir la tarea para la variable peso:

The screenshot shows the R Commander interface. The 'Data set' is 'peces'. The 'Script Window' contains the following R code:

```
x = peces$longitud
n = length(x)
sum(abs(x-mean(x)))/(n-1)

x = peces$peso[is.na(peces$peso)==F]
n = length(x)
sum(abs(x-mean(x)))/(n-1)
```

The 'Output Window' shows the results of running this code:

```
> x = peces$longitud
[1] 26.66707 25.60623 16.15782 20.45869 28.49850 27.24857 22.21253 27.21955
[9] 30.45803 32.60189 37.01008 23.36160 20.78470 31.59311 25.09649 22.61024
[17] 26.88931 26.91103 28.98970 27.91929
> n = length(x)
[1] 20
> sum(abs(x-mean(x)))/(n-1)
[1] 3.687313

> x = peces$peso[is.na(peces$peso)==F]
[1] 440.3066 419.8517 312.1246 351.3037 486.2833 455.8813 387.4544 435.5840
[9] 511.8794 485.3602 588.6873 436.2024 378.7381 498.4533 414.1054 422.2532
[17] 416.9785 440.7724 431.3175
> n = length(x)
[1] 19
> sum(abs(x-mean(x)))/(n-1)
[1] 45.243
```

Obsérvese que en este caso hemos añadido `[is.na(peces$peso)==F]` en la definición de x. La línea completa:

```
x=peces$peso[is.na(peces$peso)==F]
```

significa que en x se incluyen sólo aquellos valores de `peces$peso` que **no** sean valores perdidos⁴. De no hacerse así se producirían errores ya que R no calcula sumas ni valores absolutos de variables que tengan valores perdidos.

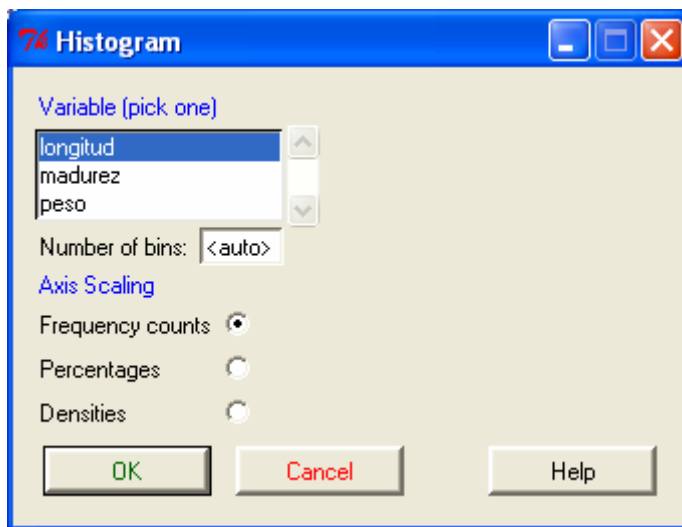
⁴ La función `is.na(v)` puede leerse como ¿es v un valor perdido?, y devuelve T (TRUE) si v es efectivamente un valor perdido y F (FALSE) si no lo es. Por tanto `is.na(v)` será igual a F solo para aquellos valores que **no** sean valores perdidos.

Gráficos en R-Commander

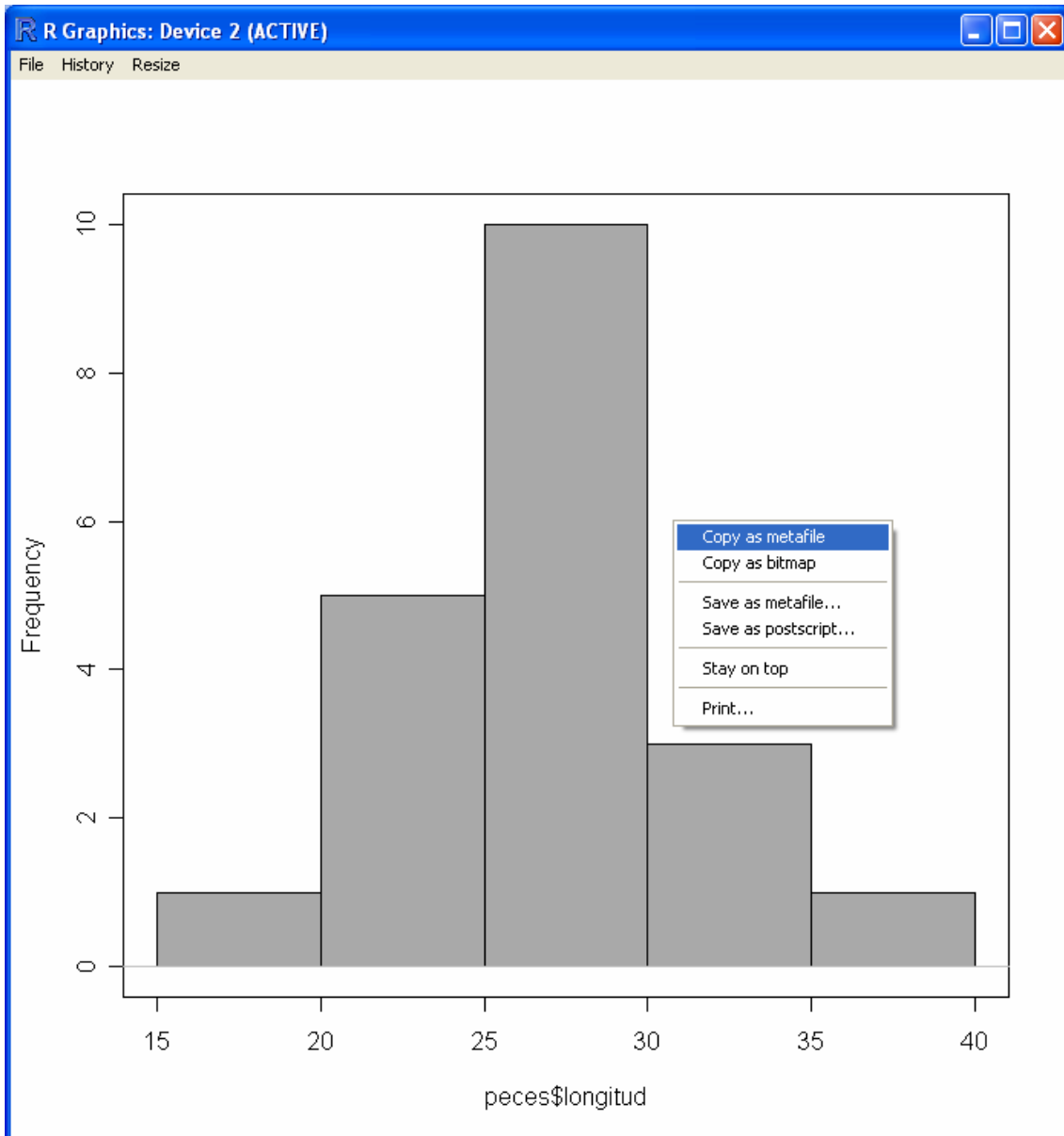
La realización de gráficos en R-Commander es bastante intuitiva y se lleva a cabo muy fácilmente a través de los menús. Por ejemplo, para hacer un histograma de la variable `peces$longitud` simplemente hay que picar en:

Graphs > Histogram ...

Marcamos longitud:



picamos OK, y obtenemos el histograma en una nueva ventana:



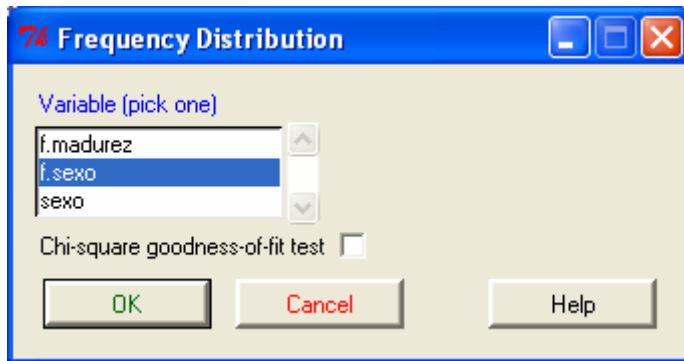
Si picamos con botón derecho del ratón sobre el gráfico y elegimos la opción "Copy as metafile", luego lo podremos pegar (botón derecho del ratón, pegar) en el fichero word (o equivalente) en que estemos redactando nuestro informe.

Tablas de Frecuencias: Variables discretas

Por defecto, R-commander sólo hace tablas de frecuencias para variables discretas que hayan sido definidas como factores. Por ejemplo, si queremos contar el número de machos y hembras de nuestra muestra, simplemente picaremos en:

Statistics > Summaries > Frequency distributions ...

y en la ventana que aparece seleccionamos la variable, en ese caso `f.sexo`:



Como resultado obtenemos:

```
> .Table <- table(peces$f.sexo)
> .Table # counts
```

```
hembra macho (Tabla de frecuencias absolutas)
  10      8
```

```
> 100*.Table/sum(.Table) # percentages
```

```
hembra macho (Tabla de frecuencias relativas)
55.55556 44.44444
```

Tablas de Frecuencias: Variables continuas

Si deseamos obtener una tabla de frecuencias para una variable continua hay que hacer una pequeña “trampa”. En primer lugar pedimos a R que haga un histograma de la variable de interés:

Graphs > Histogram ...

Tras seleccionar la variable que nos interesa y dibujar el histograma, en la ventana *script* se muestra el comando que ha generado el histograma:

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

Nos situamos en la ventana *script* sobre este comando, escribimos `hh =` al principio del mismo, y sustituimos la `H` mayúscula de `Hist` por una `h` minúscula:

```
hh = hist(peces$longitud, scale="frequency",
         breaks="Sturges", col="darkgray")
```

Como resultado, la ventana inferior (*output*) nos muestra lo siguiente:

```
> hh=hist(peces$longitud, scale="frequency", breaks="Sturges",
col="darkgray")

$breaks
[1] 15 20 25 30 35 40

$counts
[1] 1 5 10 3 1

$intensities
[1] 0.009999998 0.050000000 0.100000000 0.030000000 0.010000000

$density
[1] 0.009999998 0.050000000 0.100000000 0.030000000 0.010000000

$mids
[1] 17.5 22.5 27.5 32.5 37.5

$xname
[1] "peces$longitud"

$equidist
[1] TRUE

attr(,"class")
[1] "histogram"
```

Estos valores son los que ha usado R para dibujar el histograma. Los valores que figuran bajo `$breaks` son los extremos de los intervalos; los que figuran bajo `$counts`, las frecuencias absolutas en esos intervalos; los que figuran en `$intensities` son las frecuencias relativas, y los que figuran en `$mids` son las marcas de clase (puntos medios) de los intervalos. De esta forma, estos datos se pueden ordenar en la tabla de frecuencias:

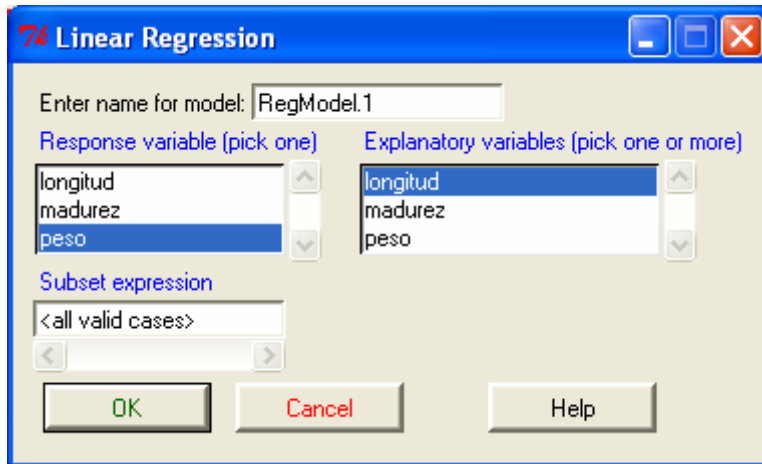
Intervalo	Marca de clase	Frec. Absoluta	Frec. Relativa
(15, 20]	17.5	1	0.0099
(20, 25]	22.5	5	0.050
(25, 30]	27.5	10	0.100
(30, 35]	32.5	3	0.300
(35, 40]	37.5	1	0.010

[Cálculo de la recta de regresión](#)

Para calcular la recta de regresión entre dos variables picamos en

Statistics > Fit models ... > Linear regression

y elegimos la variable respuesta y la variable explicativa:



En este caso la variable respuesta es el peso y la explicativa es la longitud. Tras picar en OK, en la ventana de resultados (*output*) obtenemos lo siguiente:

```
> RegModel.1 <- lm(peso~longitud, data=peces)
> summary(RegModel.1)

Call:
lm(formula = peso ~ longitud, data = peces)

Residuals:
    Min       1Q   Median       3Q      Max
-27.794 -13.687  -1.167   15.829   34.767

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  122.779     27.056   4.538 0.000291 ***
longitud     11.928      1.009  11.820 1.27e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 20.8 on 17 degrees of freedom
Multiple R-Squared:  0.8915,    Adjusted R-squared:  0.8851
F-statistic: 139.7 on 1 and 17 DF,  p-value: 1.268e-09
```

Los valores marcados en negrita corresponden a la ordenada en el origen (122.729) y a la pendiente (11.928) de la recta de regresión, que sería por tanto de la forma:

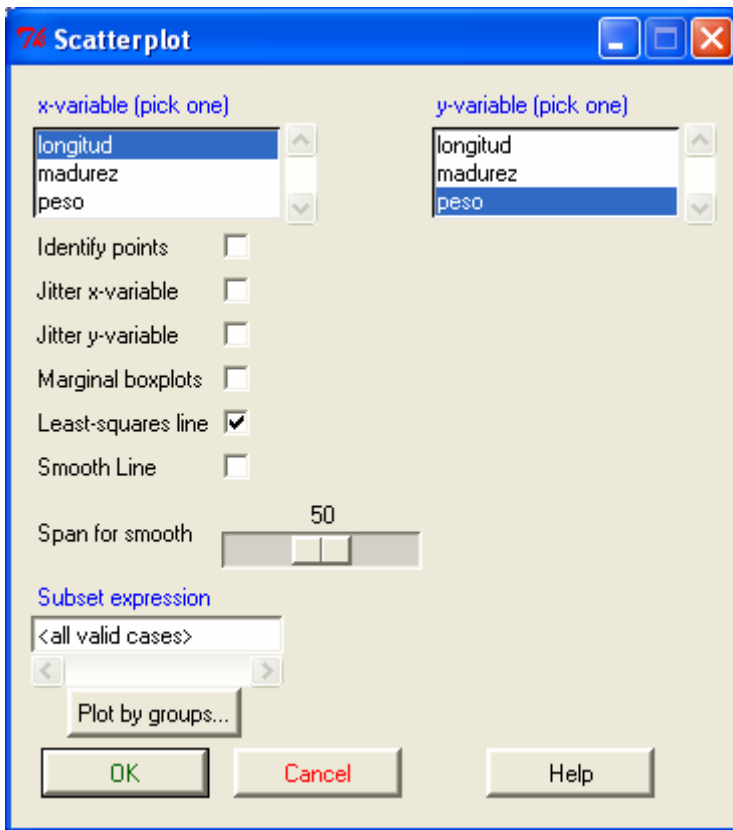
$$\text{Peso} = 122.729 + 11.928 \text{ longitud}$$

También se muestran el coeficiente de determinación $R^2 = 0.8914$ y la desviación típica residual $s_e^2 = 20.8$.

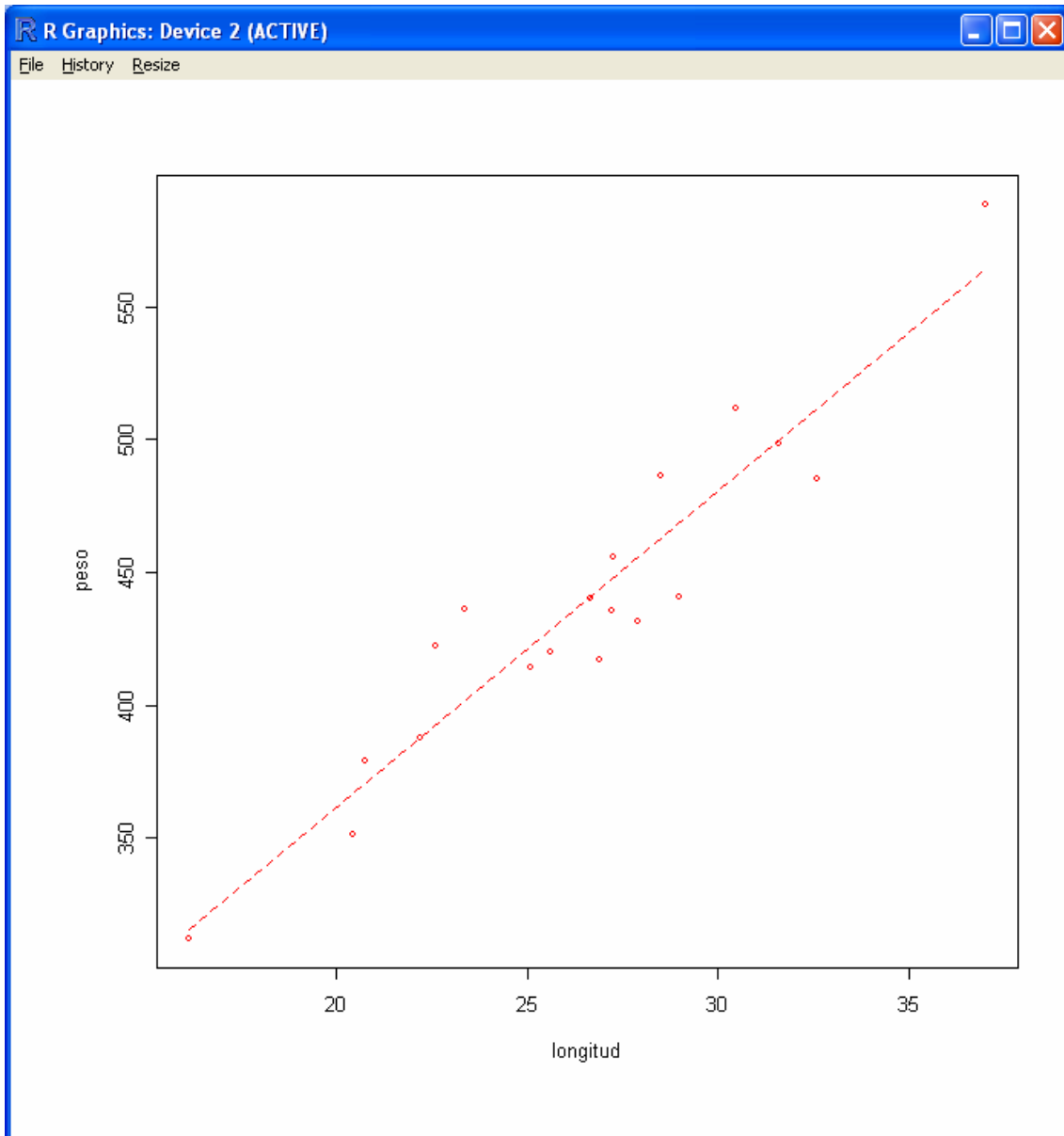
Si queremos ver el gráfico de la nube de puntos con la recta de regresión superpuesta picamos en:

Graphs > Scatterplot ...

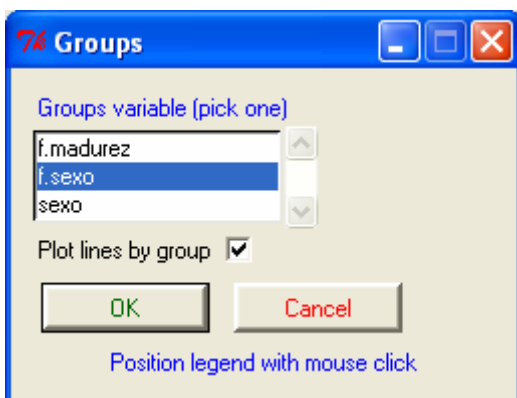
En la ventana que nos aparece a continuación señalamos cuál es la variable respuesta y cuál la explicativa, y marcamos solamente la opción “Least-squares line” para que nos muestre la recta de regresión de mínimos cuadrados:



El resultado:



También podemos obtener una recta para cada sexo, picando en el cuadro anterior en **Plot by groups** y eligiendo el factor sexo:



Ahora obtenemos:

