Introduction to R and R Commander

Denise Gutermuth

Department of Mathematics
Wilfrid Laurier University

August 2010

Contents

1. What is R? What is R Commander? What do they do?

R is a statistical software package

- a suite of methods for data manipulation and calculation that includes many useful tools for statistical modelling and graphics

R is an interpreted language

- what you type is immediately executed

R is object-oriented

- everything you work with is some sort of object
- objects are created using the assignment operator "<-"
- objects can be scalars, vectors, matrices, characters, lists, data frames, etc
- objects can be class-specific; i.e. a linear modelling object

R is command-driven

- it accepts instructions in the form of special words or letters

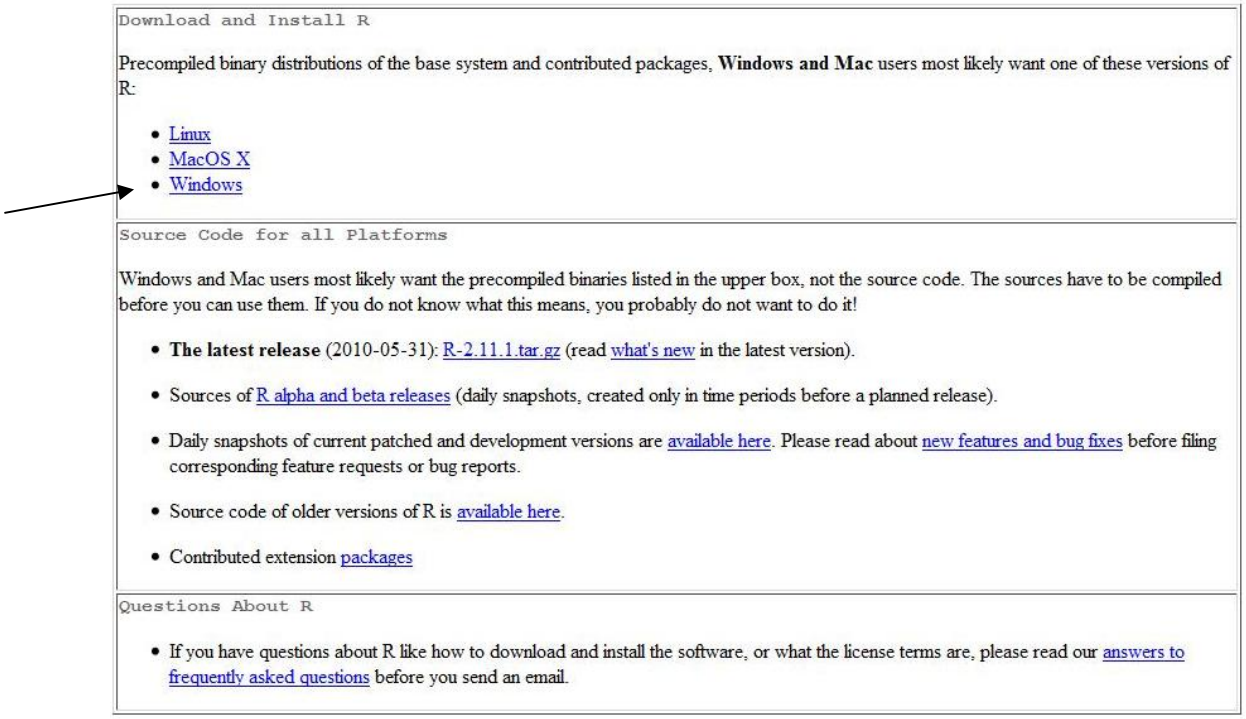
R Commander is a GUI for R

- a "graphical user interface" with menus (menu-driven) to use in R
- it is an R package that must be loaded in R to use
- R Commander is menu-driven
    - instructions are sent by choosing options from lists (menus)
- R Commander was developed by John Fox (McMaster University) to make it easier for students in introductory stats courses to see how software can be used to perform data analysis without the hindrance of learning commands
- R Commander is not appropriate for complex statistical analyses

2. Installing and opening R and R Commander

2.1 Installing and opening R

The R software must be installed on your computer in order to use R Commander. R can be downloaded from the CRAN (Comprehensive R Archive Network) accessible from the "CRAN" link on the R project website (www.r-project.org). A list of URL associated with various locations will be given. Click on the URL that is associated with the location closest to you. Click on the link for the operating system that you are using.



Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- Linux
- MacOS X
- Windows

Source Code for all Platforms

Windows and Mac users most likely want the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- **The latest release** (2010-05-31): R-2.11.1.tar.gz (read what's new in the latest version).
- Sources of R alpha and beta releases (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are available here. Please read about new features and bug fixes before filing corresponding feature requests or bug reports.
- Source code of older versions of R is available here.
- Contributed extension packages

Questions About R

- If you have questions about R like how to download and install the software, or what the license terms are, please read our answers to frequently asked questions before you send an email.

On the page following, click on “base.”

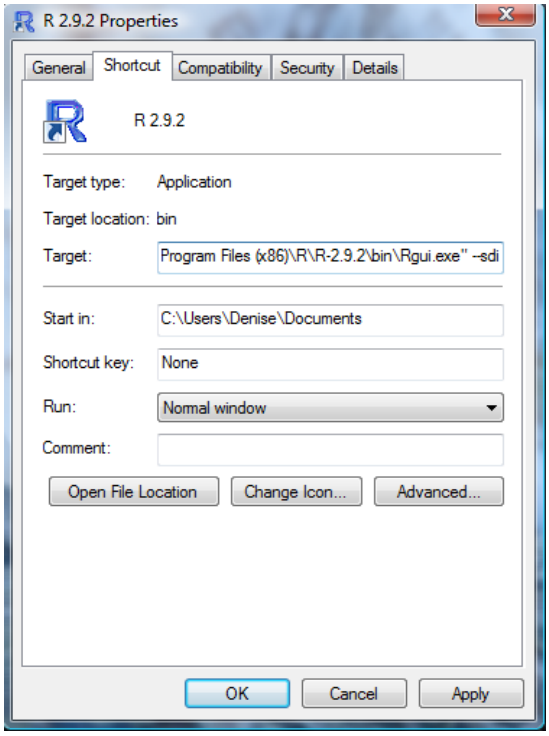
A link to the executable file to install R and the notes on installing R will be on the following page.

Once R is installed it can be opened by double-clicking the R icon shortcut on the desktop of your computer or via the start menu.

2.2 Installing R Commander

For R Commander to operate properly on Windows systems it must be run as an SDI (Single Document Interface) with R.

- copy a shortcut for R to the desktop.
- right-click on the R icon on the desktop; select “Properties”
- click on the “Shortcut” tab in the properties window
- edit the Target field by adding at the end of the target address a space then “--sdi”



NOTE:

- there should only be one space before the first dash and this is the only space
- the expression in the target field may vary as the location of the program may vary for different users

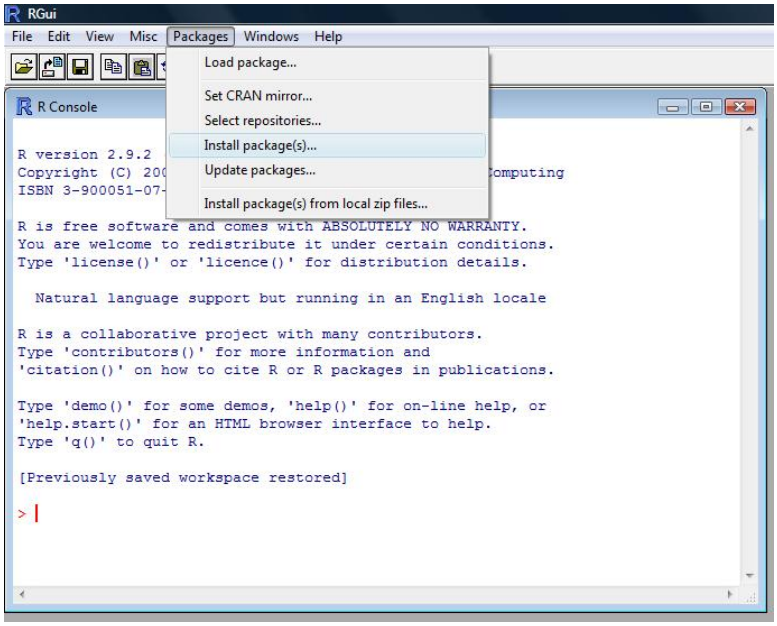
To change the name of the shortcut (to reflect that R opened from this shortcut will run in SDI format)

- select the “General” tab
- Modify the name of the program in the first field to something that reflects that fact that it has been modified to run in SDI, say R–SDI (or something similar).

To use R Commander, you must open R from this shortcut. (Opening from another shortcut or from the start menu will run R in a MDI format rather than an SDI format.)

Open R. R Commander is a package that must be installed in R to run the R Commander interface. To install the package click on “Packages” in the menu bar in R.

Packages→Install package(s)



A CRAN mirror window will appear. Select the location closest to you (Canada (ON)) and click OK. A window listing all packages that can be downloaded will appear. The R Commander package is listed as "Rcmdr". Locate Rcmdr in the list and click okay. The package will be installed into R.

Note: The Rcmdr package has a number of packages on which it depends. If you choose to install the package via the menu in R, the Rcmdr dependencies will not be automatically installed. After installing Rcmdr and loading the package it will offer to download and install missing dependencies.

An alternate method:

At the prompt (">") in the R window, enter the commands:

```
install.packages(Rcmdr,dependencies=TRUE)
```

Note: All dependencies will be installed with this method.

2.3 Starting R Commander

- open R by double-clicking the R icon (running in SDI) on the desktop
- enter `library(Rcmdr)` at the prompt (>) in R and press Enter
- the R Commander window will appear in a separate window
- minimize the R console window
- NOTE: if you close the R Commander window but not the R console, you can restart R Commander by entering Commander() at the prompt in R then pressing enter

Script Window

R commands generated by R Commander will appear here.

Commands can be entered directly into this window and executed by clicking *Submit* button or by typing the combination *ctrl-r* on the line to be executed.

Commands that extend over more than one line should have the second and subsequent lines indented by one or more spaces or tabs. Multi-line commands can be executed by highlighting the lines and clicking *Submit* or by typing the combination *ctrl-r*

Commands in R Commander must not be preceded by ">."

Output Window

Output will appear in blue

R commands used to obtain the output will appear in red

Message Window

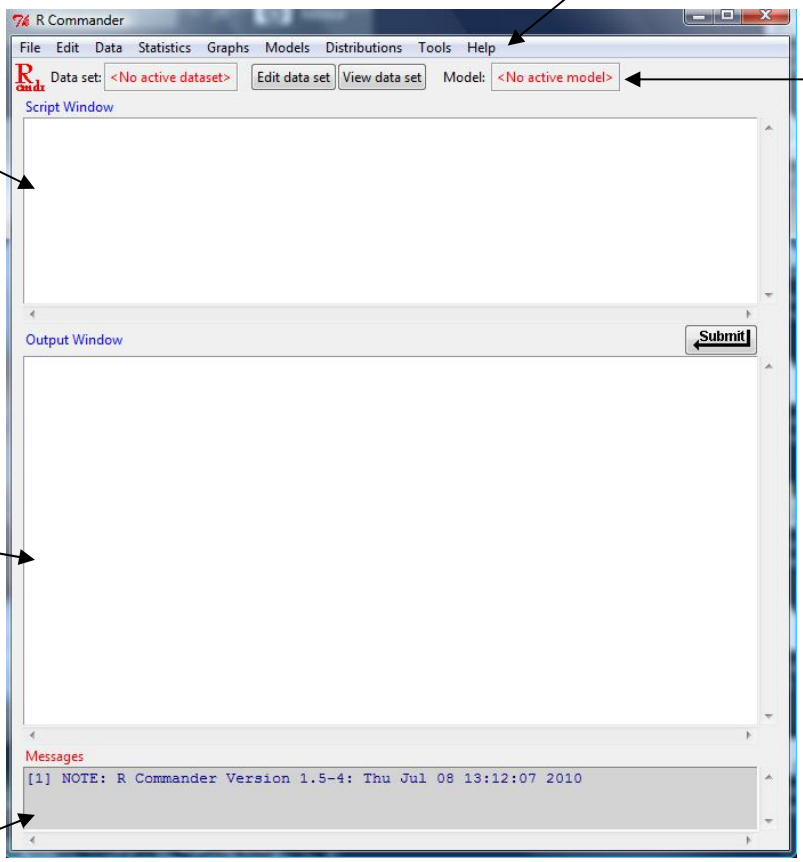
Error messages will appear in red

Notes will appear in blue (note the blue start-message in the screenshot)

Warnings will appear in green

Menus

Menu items will be greyed out if they are not applicable to the current context



Toolbar

Graphics will appear in a separate graphics window and only the most recent graphics will appear be shown. To view other graphics generated in the current session, use *page up* or *page down* keys.

3. R Objects

3.1 Scalars and vectors

scalars:

```
> x <- 7
> y <- x*2 + 3
> x
[1] 7
> y
[1] 17
```

vectors:

```
> x <- c(1,2,3,4,5)
> y <- x-1
> x
[1] 1 2 3 4 5
> y
[1] 0 1 2 3 4
> y[3]
[1] 2
> y[3] <- 7
> y
[1] 0 1 7 3 4
```

3.2 Data frames

- type of object
- type of table where the rows are usually observations and columns are variables
- columns can be different types of data
- constructed using the data.frame() function in R or in the script window of R commander or using the menus in R commander

In R:

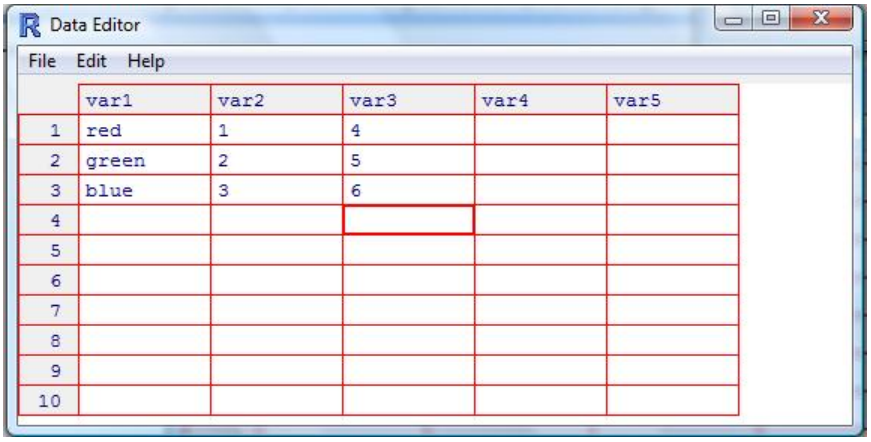
Using the data.frame() function

```
> x <- c('red','green','blue')
> y <- c(1,2,3)
> d <- data.frame(x,y,z=c(4,5,6))
> d
   x    y   z
1 red   1   4
2 green 2   5
3 blue  3   6
```

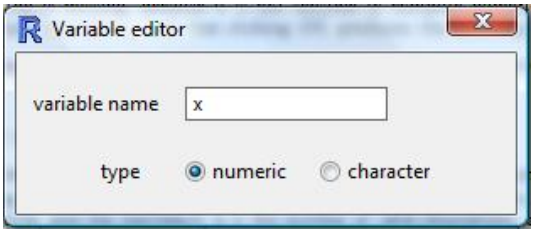
In R Commander:

Data→New data set…

- enter the new data set's name in the dialog box (or leave the default name "Dataset")
- CAUTION: R is case-sensitive and spaces are not permissible!
- click OK
- this will bring up a data editor window
- enter values for each variable in the columns
- you can move from cell to cell using the mouse, arrow keys, or the tab key

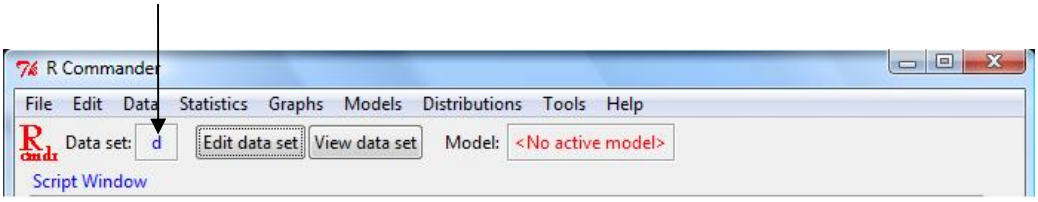


- enter the variable names:
    - left-click on “var1” to open the variable editor dialog box, enter the name and type and close the dialog box



    - repeat for the other variable
    - close the data editor dialog box using the *File* menu (*File→Close)* or by clicking the “X” button in the top-right corner.

This data set will be the active data set in R commander



To view the contents of an object or data frame that has been defined in R or R Commander

In R:

- type the name assigned to the data frame or object in R at the prompt (>) and press *enter*

```
> d

x y z

1 red 1 4

2 green 2 5

3 blue 3 6
```

In R Commander:

- type the name assigned to the data frame in R Commander in the script window and press *ctrl-r* → the data frame will appear in the output window
  OR
- click *View data set* to view the active data set in a separate window

Note: In R Commander, the active data set can be edited by clicking the flat button *Edit data set* appearing on the toolbar.

Note: the names of the columns, particular columns, rows or elements of a data frame can be extracted using R commands.

For the data frame "d" defined above:

```
> names(d)
[1] "x" "y" "z"
> d$y
[1] 1 2 3
> d[,2]
[1] 1 2 3
> d[2,]
x y z
2 green 2 5
> d[2,2]
[1] 2
```

3.3 Lists

- an object that can contain any other object
- many functions return results in this format
- e.g., when a linear regression model is assigned to an object that object is a list containing the estimated parameters, the fitted values, the residuals, etc.

- the **names()** function will give you the names of the objects contained in a list

```
> names(d)
[1] "x" "y" "z"
```

- you can also extract objects from a list using $:

```
> d$y
[1] 1 2 3
```

4. Importing Data

4.1 Importing data from a text file

Consider the text file melanoma.txt containing data on survival rates of individuals with malignant melanoma taken from a data set in the MASS package. The first few lines of data are as follows:

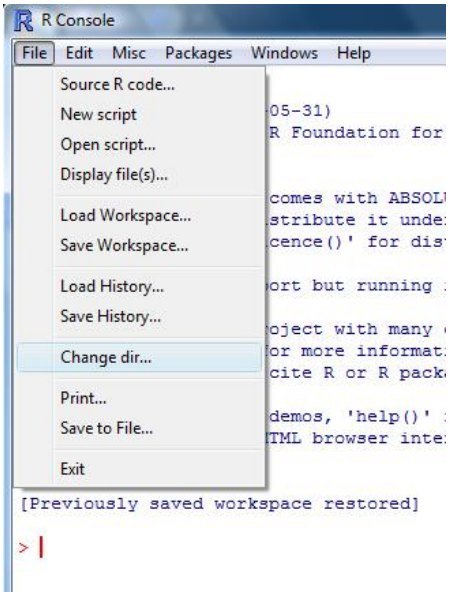
```
time status sex age year thickness ulcer
1     10      3   1  76 1972      6.76     1
2     30      3   1  56 1968      0.65     0
3     35      2   1  41 1977      1.34     0
4     99      3   0  71 1968      2.90     0
5    185      1   1  52 1965     12.08     1
6    204      1   1  28 1971      4.84     1
7    210      1   1  77 1972      5.16     1
8    232      3   0  60 1974      3.22     1
9    232      1   1  49 1968     12.88     1
10   279      1   0  68 1971      7.41     1
```

Note:

- the first line contains the variable names
- the remaining lines contains the variable values for each individual in the study
- the data values are separated by “white space”
- the indices for the rows will be the row names in the data frame in R (Note: there is no name for the column of indices)
- it is not necessary for the data to line up vertically in the text file

In R:

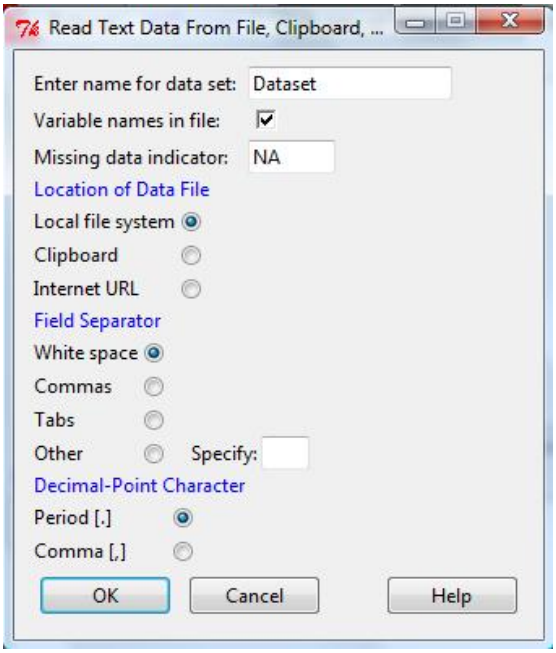
Change the directory to the location of the file:




```
read.table("melanoma.txt",header=T)
```

In R Commander:

Data→Import data→from text file, clipboard or URL…



Enter the name for the data set (no spaces)
Specify the characteristics of the text file (variable names in file, location of the file, Field Separator, etc)
Click OK
Browse, select the file and click Open

4.2 Importing data from a csv file

- "comma-separated-variable" file (e.g., *.csv format) is a "comma-delimited" text file
- e.g.

```
x,     y,   z
red,   1,   4
green, 2,   5
blue,  3,   6
```

In R:

Then, `read.csv()` can be used to read this table into R

```
> mydata <- read.csv("C:/…/Book1.csv",header=TRUE)
```

OR

```
mydata <- read.csv("Book1.csv",header=TRUE)
```

In R Commander:

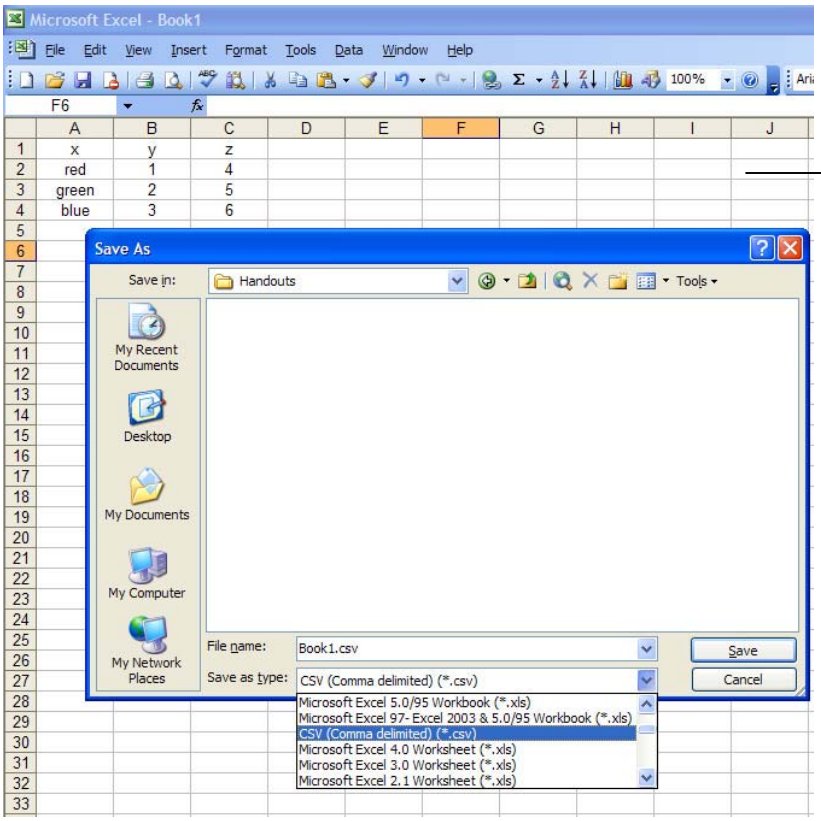
Data→Import data→from text file, clipboard or URL…

Specify the name of the data set and the characteristics, selecting "Comma" for the Field Separator

4.3 Importing data from an Excel file

In R:

- Avoid doing it!
- if you have a spreadsheet of data you want to import, you can save it as a "comma-separated-variable" file (e.g., *.csv format)




```
x,     y,   z
red,   1,   4
green, 2,   5
blue,  3,   6
```

- import the newly created csv file as in 4.2

In R Commander:
Data→Import data→from Excel, Access or dBase data set…
Browse, select the file and click open

4.4 Importing data from a package

In R:

`>install.packages("name of package")`	<- installs the package
`>library(name of package)`	<- loads the package
`>data(name of data set)`	<- loads the data set

Load the package:

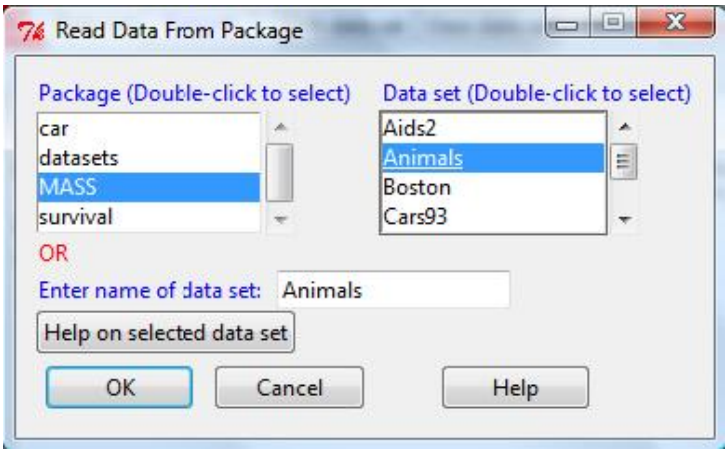
Tools→Load package…
Select the package from the list and click OK.

Note: if the package has not been installed on your computer then you must first open R and install the package in R using the command `install.packages()`, open R Commander and load the package using the *Tools* menu

Load the data set from the package:

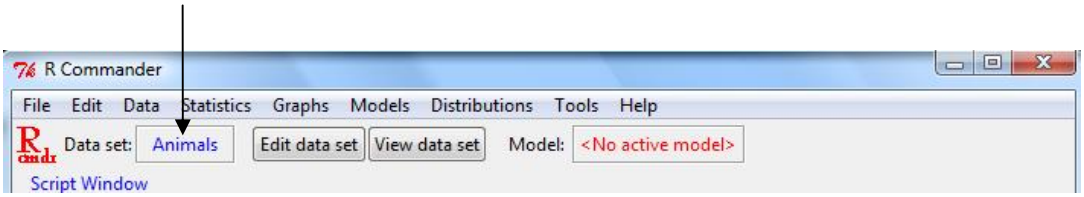
Data→Data in packages→Read data set from an attached package…

A dialog box will appear



- select the package by double-clicking (data sets in the package will appear in a list on the right)
- select the data set
- alternatively, enter the name of the data set in the appropriate space
- click OK

NOTE: In R Commander, the most recently imported/created data set will appear as the active data set. The name of the data set will appear in blue on a "flat" button in the toolbar



To change the active data set to a previously loaded data set, click the flat button with the name of the currently active data set. A dialog box with a list of loaded data sets will appear.

5. Calculations, R functions and new variables

5.1 Calculations

- R can do all of your usual calculations, as well as vector and matrix arithmetic
- most operations are done on whole vectors at once

- operations:

Addition	x+y
Subtraction	x-y
Multiplication	x*y
Division	x/y
Exponentiation	x^y
Logarithmic transformation	log10(x) or log(x,base)

- example:

```
> x <- c(1,2,3,4,1)
> y <- c(1,1,3,-1,5)
> x+y
[1] 2 3 6 3 6
> x-y
[1]  0  1  0  5 -4
> x*y
[1]  1  2  9 -4  5
> x/y
[1]  1.0  2.0  1.0 -4.0  0.2
> x^y
[1]  1.00  2.00 27.00  0.25  1.00
> log10(x)
[1] 0.0000000 0.3010300 0.4771213 0.6020600 0.0000000
> log(x,2)
[1] 0.000000 1.000000 1.584963 2.000000 0.000000
```

5.2 Some R Functions

In R:

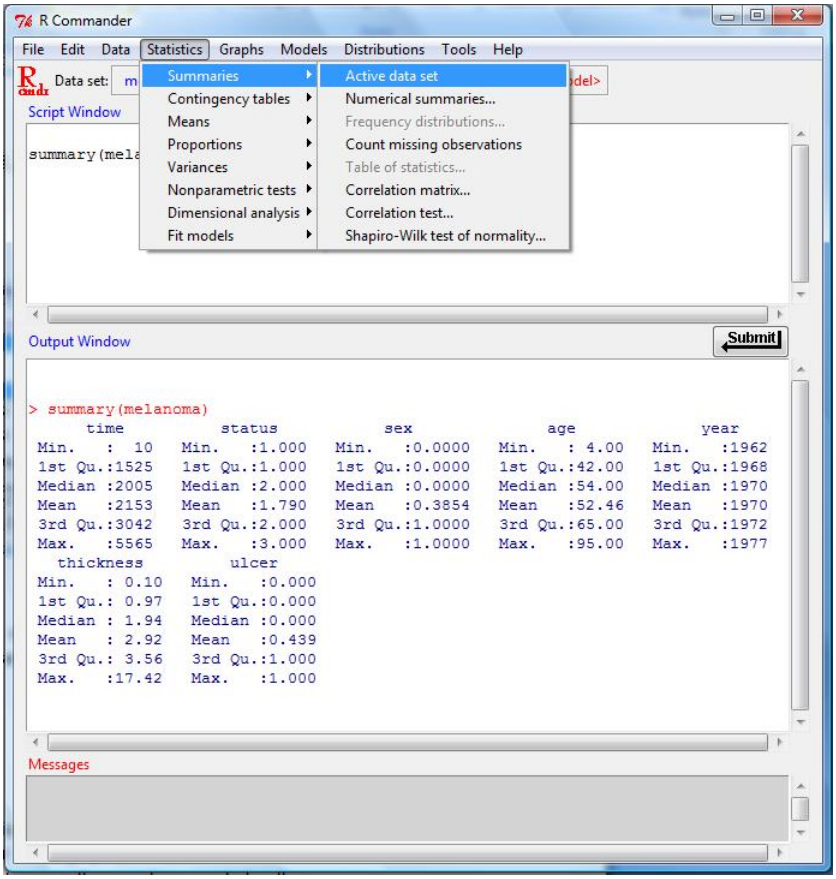
```
> x <- 1:10        # The numbers 1 to 10
> mean(x)          # The sample mean
> var(x)           # The sample variance
> sd(x)            # The sample standard deviation
> y <- 11:20       # The numbers 11 to 20
> var(x,y)         # The sample covariance
> cor(x,y)         # The sample correlation
> median(x)        # The sample median
> rnorm(30,0,15)   # simulates 30 Normal random variables with mean
                   of 0 and standard deviation of 15
> summary(x)       # Several useful statistics
```

- Note: the functions in the above table can be used in R Commander by entering and executing the commands in the script window of R Commander

In R Commander:

- R Commander does not have individual menu options that will allow us to compute the value of each statistic and variable listed in the table above, however there are two ways of obtaining basic summary statistics (yielding output obtained using the summary() function in R)

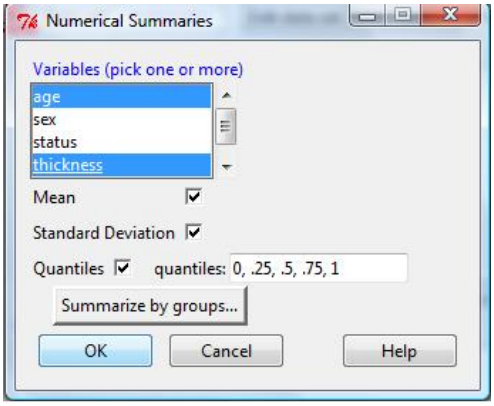
Statistics→Summaries→Active data set…



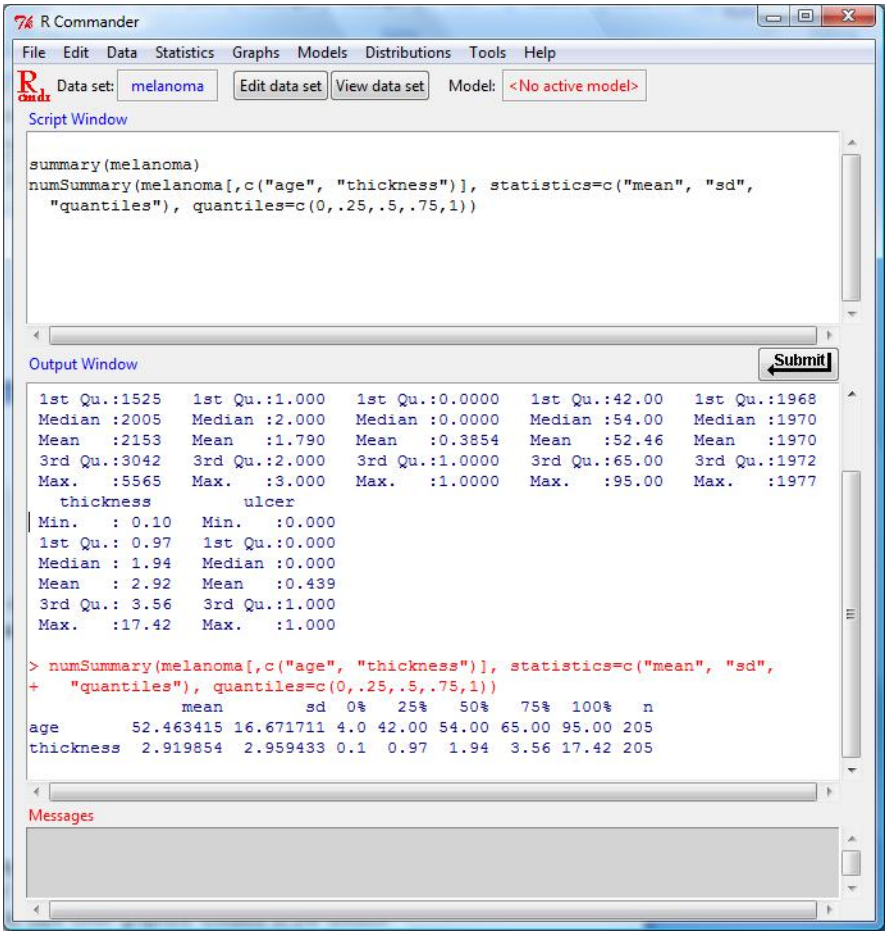
To obtain summary statistics for <u>specific variables</u> in a data set in R Commander:

Statistics→Summaries→Numerical summaries…

This will open the dialog box shown below



- select a variable by clicking on the name appearing in the list
- select multiples variables by holding ctrl and clicking on the variable names
- specify the statistics to compute and click OK



5.3 Computing new variables

In R:

- New variables can be created in R using the operators in 6.1 and R functions, including those in 5.2 and others including cbind() and rbind()
- Examples:

```
> w<-x+y
> w
[1] 2 3 6 3 6
```

```
> z<-cbind(x,y)
> z
     x  y
[1,] 1  1
[2,] 2  1
[3,] 3  3
```

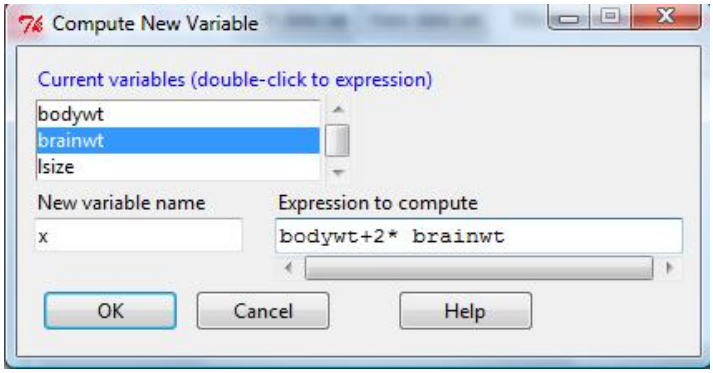
```
> u<-rbind(x,y)
> u
  [,1] [,2] [,3] [,4] [,5]
x    1    2    3    4    1
y    1    1    3   -1    5
```

```
     [4,] 4 -1
     [5,] 1  5
```

In R Commander:

- R Commander can compute new variables using the variables in an active data set
- Suppose the “litters” data set from the DAAG package is the current data set in R

  Data→Manage variables in active data set→Compute new variable…



  - enter the new variable name in the appropriate space and the expression to compute in the box on the right-hand-side
  - NOTE: the expression for the new variable can contain operators or functions and the current variables in the data set
  - to include a current variable in the expression, double-click on the variable in the current variables list (the variable will be inserted into the expression), or type the variable name into the expression directly
  - click OK
  - the new variable will be added to the active data set in a new column

litters

	lsize	bodywt	brainwt	x
1	3	9.447	0.444	10.335
2	3	9.780	0.436	10.652
3	4	9.155	0.417	9.989
4	4	9.613	0.429	10.471
5	5	8.850	0.425	9.700
6	5	9.610	0.434	10.478
7	6	8.298	0.404	9.106
8	6	8.543	0.439	9.421
9	7	7.400	0.409	8.218
10	7	8.335	0.429	9.193
11	8	7.040	0.414	7.868
12	8	7.253	0.409	8.071
13	9	6.600	0.387	7.374
14	9	7.260	0.433	8.126
15	10	6.305	0.410	7.125
16	10	6.655	0.405	7.465
17	11	7.183	0.435	8.053
18	11	6.133	0.407	6.947
19	12	5.450	0.368	6.186
20	12	6.050	0.401	6.852

6. Graphics

- Graphics will appear in a separate graphics window
- Only the most recent graphics will appear in the graphics window
    - use *page up* or *page down* keys to view other graphics created in the session

Consider the data set below:

```
> x <- 1:30
> y <- 3 + 2*x + rnorm(30, 0, 15)
> data.frame(x,y)
```

simulates 30 Normal random variables with a mean of 0 and a standard deviation of 15

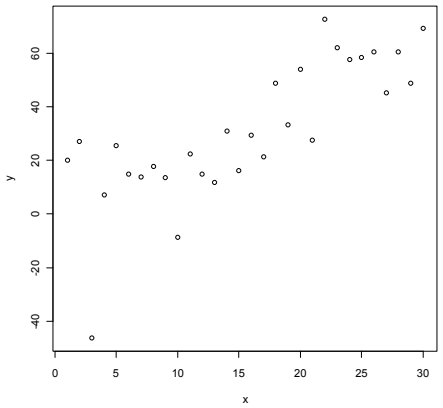
```
            x     y

            7  19.999753

            2  27.169996
            3 -46.345493
             …
            28  60.635952
            29  48.875499
            30  69.324089
```

CAUTION: To generate graphics for a particular data set in R Commander, the data set of interest must be the selected as the active data set

6.1 Scatterplots

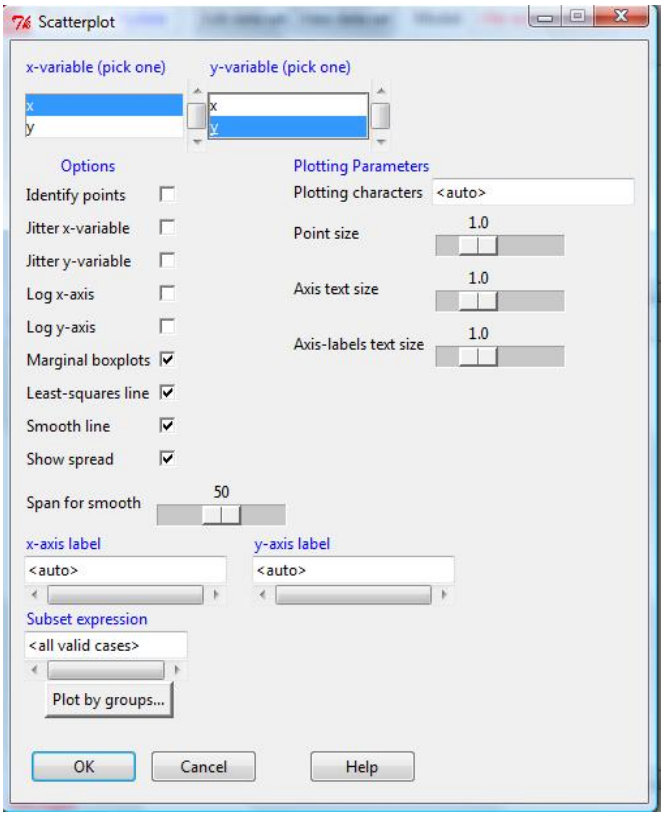
In R:

```
plot(x,y)
```



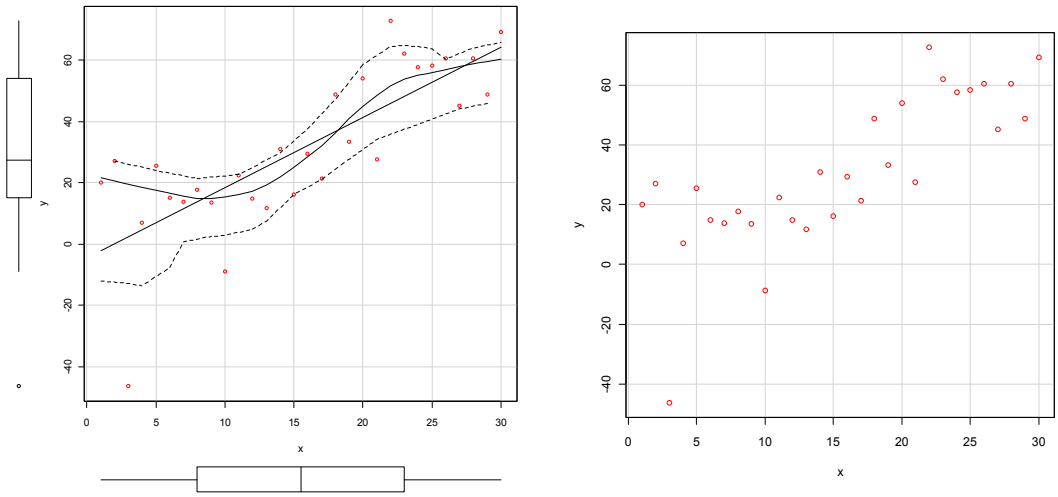
In R Commander:

Graphs→Scatterplot…



- select the x-variable (horizontal axis) and y-variable (vertical axis)
- enter labels for each axis if desired

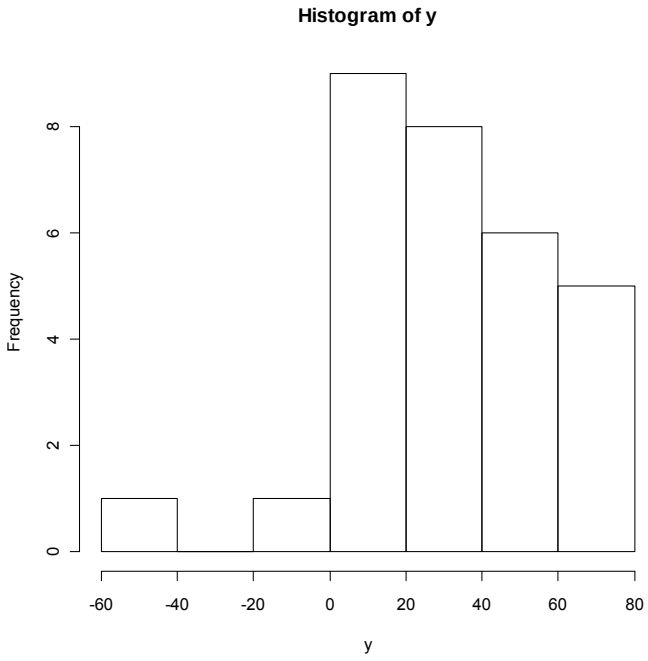
- the checked options shown above are automatically “checked” in R Commander (you must “uncheck” those options if you do not want the corresponding plots to appear with the scatterplot)
- click OK



6.2 Histograms

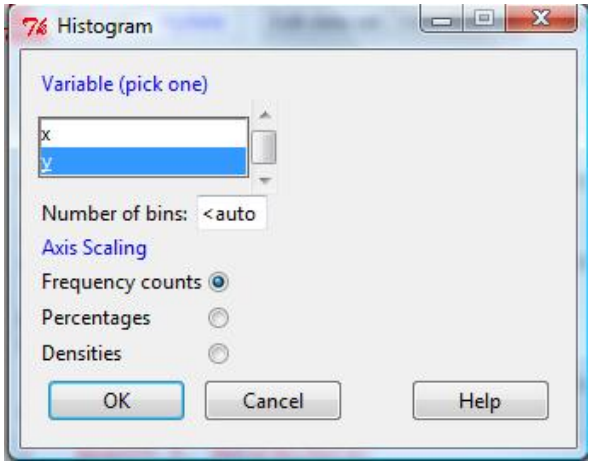
In R:

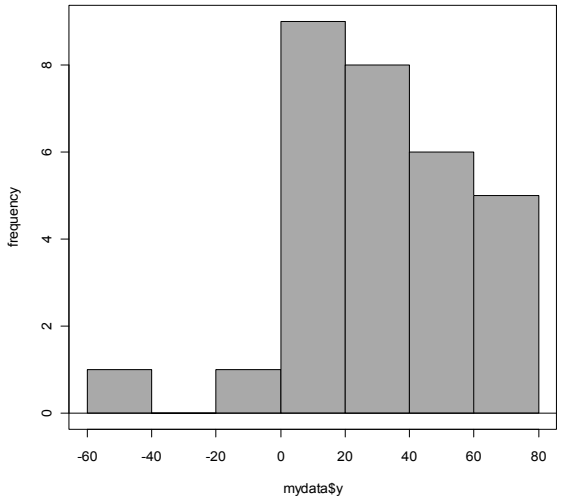
```
hist(y)
```



In R Commander:

Graphs→Histogram…

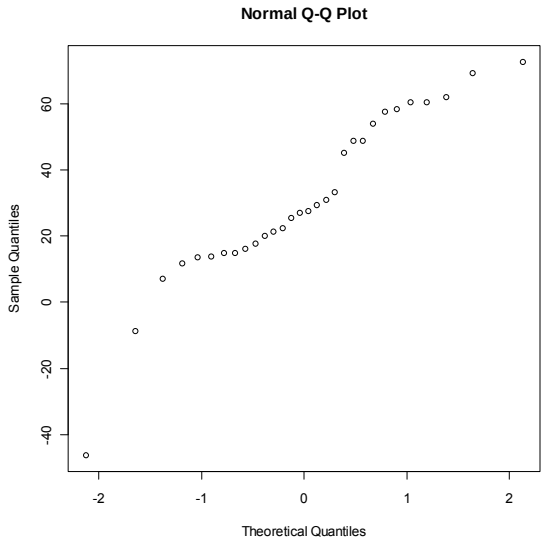




6.3 Normal Q-Q Plots

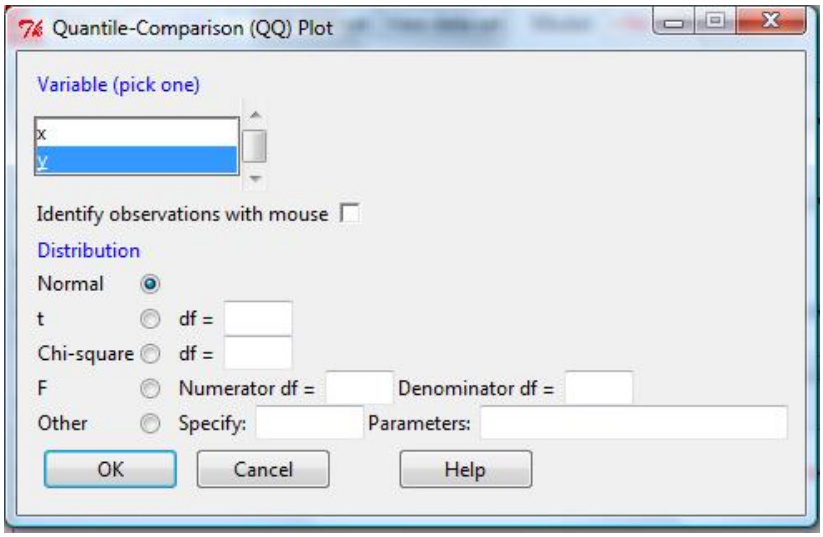
In R:

```
qqnorm(y)
```

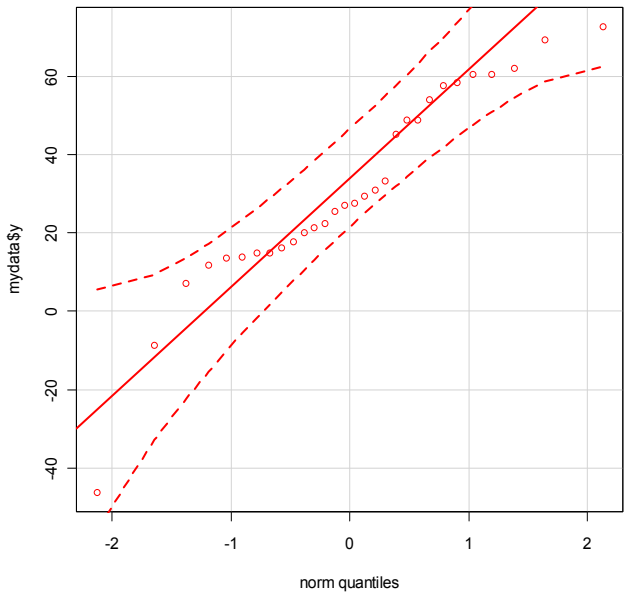


In R Commander:

Graphs→Quantile-comparison plot…



- select the variable of interest
- select “Normal” for the distribution
- click OK



6. 4 Amending Graphical Output

- the graphics options in R Commander are limited and often are not ideal for presentations or reports
- on occasion you may want to add lines to graphs, change the axis labels or modify the colours
- R commands must be used in R Commander to make amendments
- the commands listed below can also be used in R to amend graphical output

Add the line y=a+bx to a plot

```
abline(a=3,b=4)
```

Add lines joining the data points:

```
lines(x,y)
```

Add points to the plot:

```
points(x,y)
```

Add text to a plot:

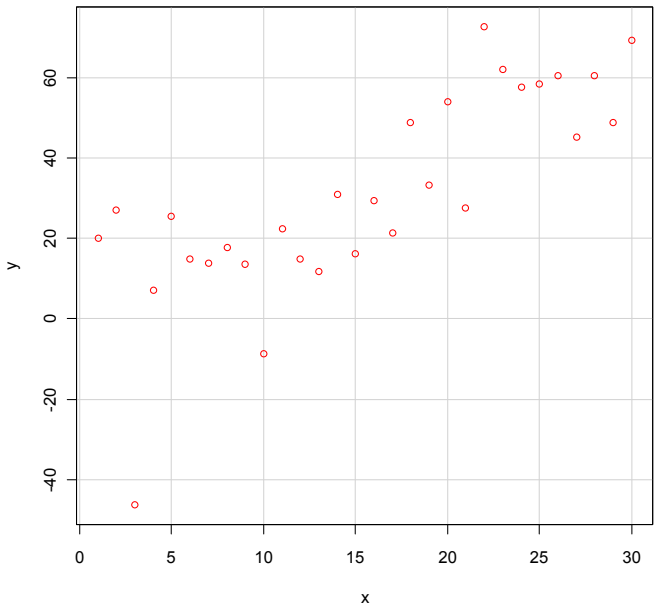
```
text(x,y,labels=y)
```

In R:

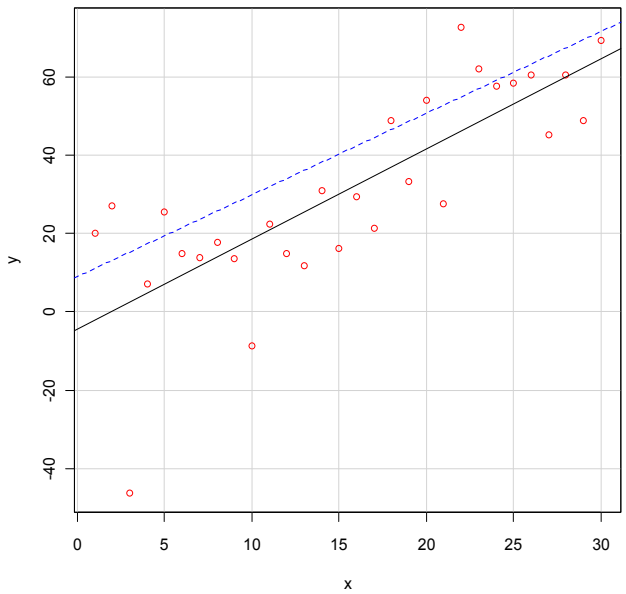
- enter the commands at the prompt (>) and press enter

In R Commander:

- enter the commands above in the script window and use the key combination *ctrl-r*
- CAUTION: commands entered into the script window of R Commander should not be preceded by ">"




```
abline(-4.4,2.3)

abline(a=8.893, b=2.091, col="blue", lty=2)
```



7. Fitting Linear Regression Models

Examples:
– suppose **y, x, x0, x1 and x2** are numeric variables
… **y** is the response
… the **x**'s are the predictors

7.1 Simple linear regression of y on x

In R:

Usually the data to be analyzed will be stored in a dataframe, say `mydata`.

```
lm(y ~ 1 + x,data=mydata)
```
or
```
lm(y ~ x,data=mydata)
```

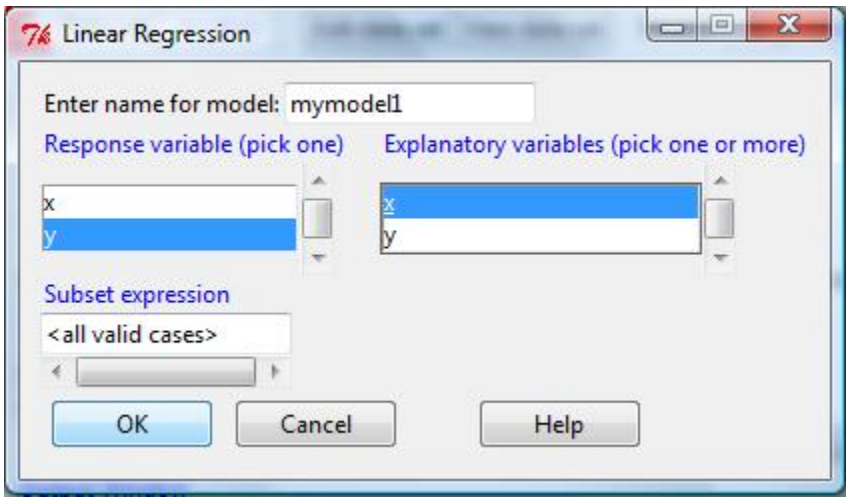
(i.e., an intercept is always included in the model by default)

If we want to give the model a name in R, say "mymodel"

```
mymodel <- lm(formula, data=mydata)
```

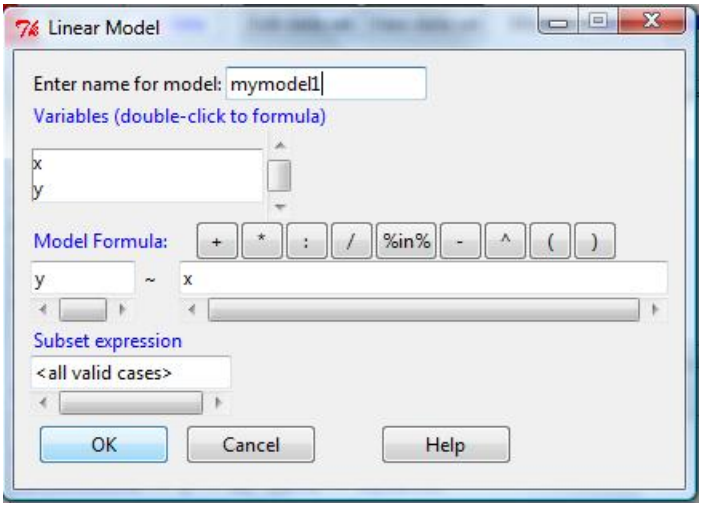
In R Commander:

Statistics→Fit models→Linear regression…



- enter the model name
- select the response variable and explanatory variable
- click OK

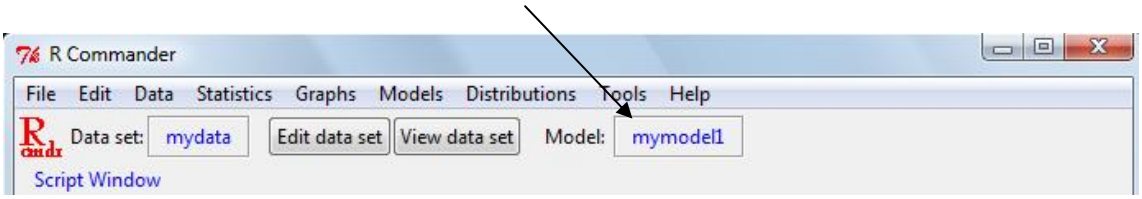
OR *Statistics→Fit models→Linear model*



- enter the model name
- select the variable for the left-hand side (response variable) by double-clicking on the variable in the list
    - this will copy the variable to the response variable space if empty (otherwise it will copy to the right-hand-side of the model formula)
- enter the formula for the right-hand-side (in terms of the predictor variable) by double-clicking on the variable in the variable list and use the buttons for the operations OR by tying the formula directly into the box
- click OK

- NOTE: for simple linear regression of y on x with an intercept term it is enough to enter "x" for the right-hand-side of the model formula

- the linear model will become the active model



Notes:
- You can type an R expression into the Subset expression box
- If supplied, the model will be fit to a subset of the data
- The subset expression can be left blank if the model must be fitted to the entire data set

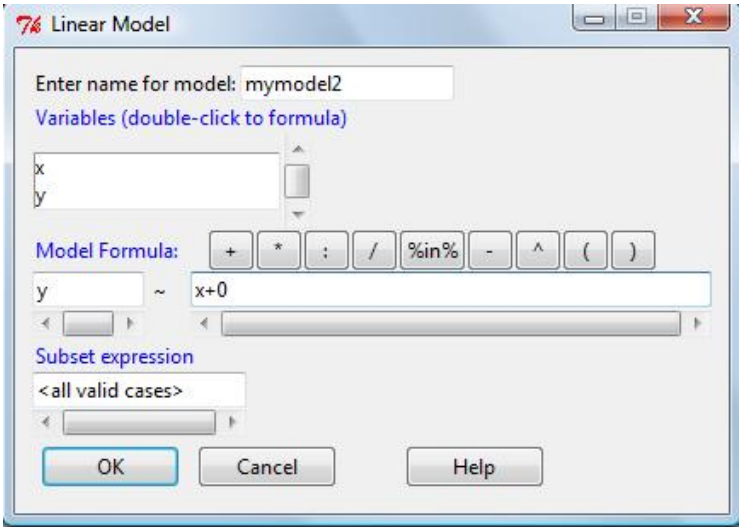
7.2 Simple linear regression of y on x through the origin

In R:

```
lm(y ~ 0 + x,mydata)
or
lm(y ~ -1 + x,mydata)
```

In R Commander:

Statistics→Fit model→Linear model…



- Note: the model formula x-1 can also be used for the right-hand-side

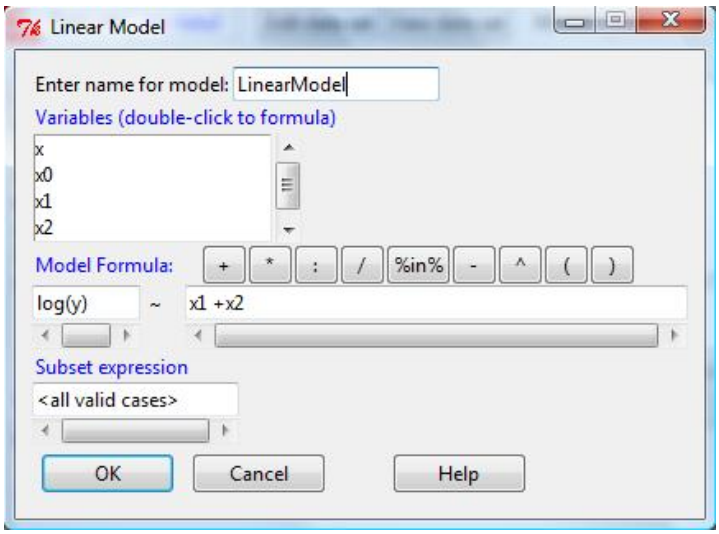
7.3 Multiple regression of a transformed variable

- example: multiple regression of a transformed variable, log(y) on x1 and x2 (with an implicit intercept term)

In R:

```
lm(log(y) ~ x1 + x2,data=mydata)
```

In R Commander:



- enter the expression for the transformed response variable manually
- enter the expression for the right-hand-side
- click OK

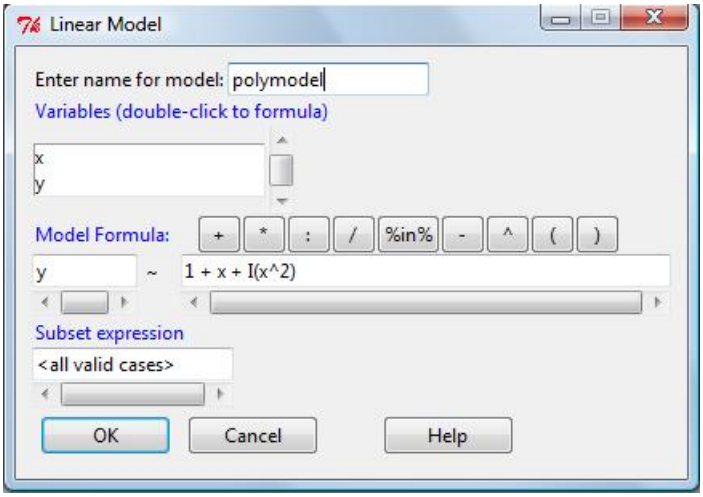
7.4 Polynomial regression of y on x of degree 2

In R:

```
lm(y ~ poly(x,2),data=mydata)
or
lm(y ~ 1 + x + I(x^2,data=mydata)
```

In R Commander:

- enter the model name
- enter the model formula 1+x+I(x^2) on the right-hand-side
- click OK



7.5 Working with fitted linear regression objects

Suppose you assigned a linear model fit in R, naming the object "mymodel"

`mymodel <- lm(`*`formula`*`, data=mydata)`

or in R Commander (*Statistics→Fit models→…etc)*

- **mymodel** is now a list of results of class "**lm**" with many related objects (coefficients, residuals, etc.)

- There are also many built in functions that "orient" themselves to objects of this class. They can be used to display and extract information about the fitted model in R:

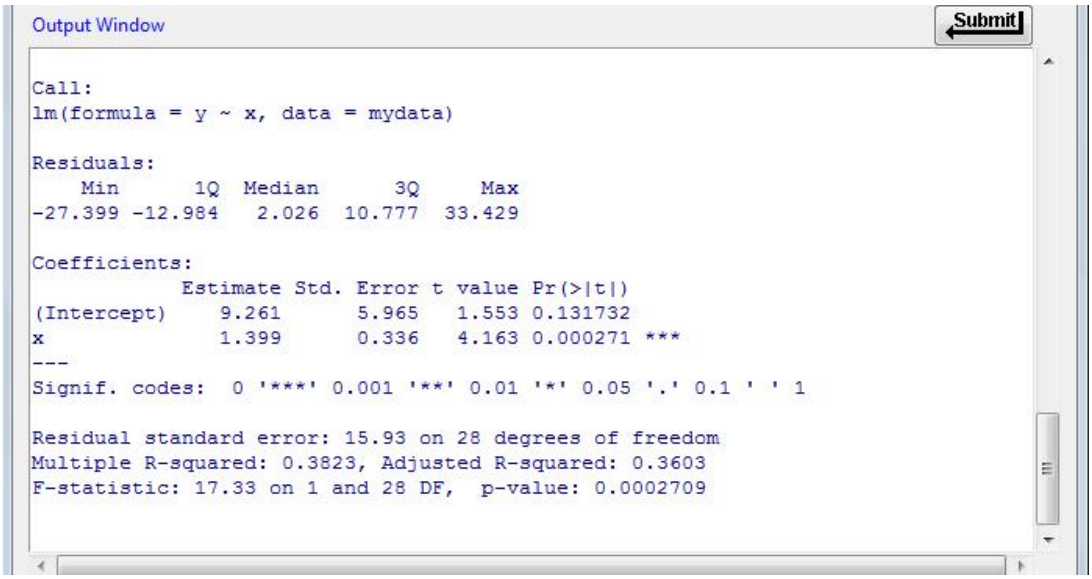
  … `summary(mymodel)`
  … `coef(mymodel)`
  … `plot(mymodel)`
  … `residuals(mymodel)`
  … `anova(mymodel.1, mymodel.2)`
  … `etc.`

- Note: the commands above can be used in R Commander (enter into the script window and click "Submit")

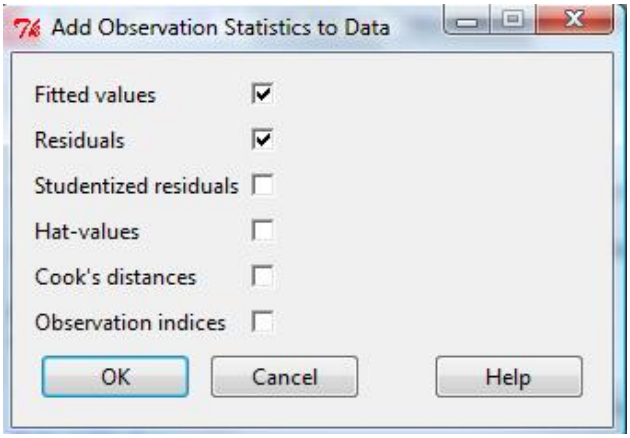
- When a linear model is fit in R Commander, the summary information will be shown automatically in the output window

- To obtain the summary information in R Commander,

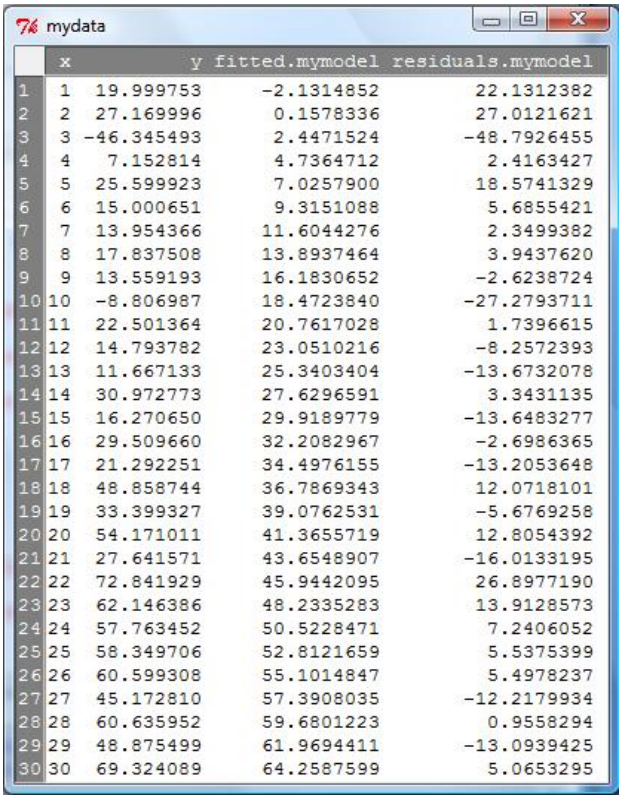
  Models→Summarize model



- To view the fitted values and residuals we can add the observation statistics to the data set: *Models→Add observation statistics to data set…*



- Variables will be created and added to the data set



mydata

	x	y	fitted.mymodel	residuals.mymodel
1	1	19.999753	-2.1314852	22.1312382
2	2	27.169996	0.1578336	27.0121621
3	3	-46.345493	2.4471524	-48.7926455
4	4	7.152814	4.7364712	2.4163427
5	5	25.599923	7.0257900	18.5741329
6	6	15.000651	9.3151088	5.6855421
7	7	13.954366	11.6044276	2.3499382
8	8	17.837508	13.8937464	3.9437620
9	9	13.559193	16.1830652	-2.6238724
10	10	-8.806987	18.4723840	-27.2793711
11	11	22.501364	20.7617028	1.7396615
12	12	14.793782	23.0510216	-8.2572393
13	13	11.667133	25.3403404	-13.6732078
14	14	30.972773	27.6296591	3.3431135
15	15	16.270650	29.9189779	-13.6483277
16	16	29.509660	32.2082967	-2.6986365
17	17	21.292251	34.4976155	-13.2053648
18	18	48.858744	36.7869343	12.0718101
19	19	33.399327	39.0762531	-5.6769258
20	20	54.171011	41.3655719	12.8054392
21	21	27.641571	43.6548907	-16.0133195
22	22	72.841929	45.9442095	26.8977190
23	23	62.146386	48.2335283	13.9128573
24	24	57.763452	50.5228471	7.2406052
25	25	58.349706	52.8121659	5.5375399
26	26	60.599308	55.1014847	5.4978237
27	27	45.172810	57.3908035	-12.2179934
28	28	60.635952	59.6801223	0.9558294
29	29	48.875499	61.9694411	-13.0939425
30	30	69.324089	64.2587599	5.0653295

- Once the fitted values and residuals have been added to the data set, plots using the fitted values and the residuals can be generated (*Graphs→Scatterplot…*)

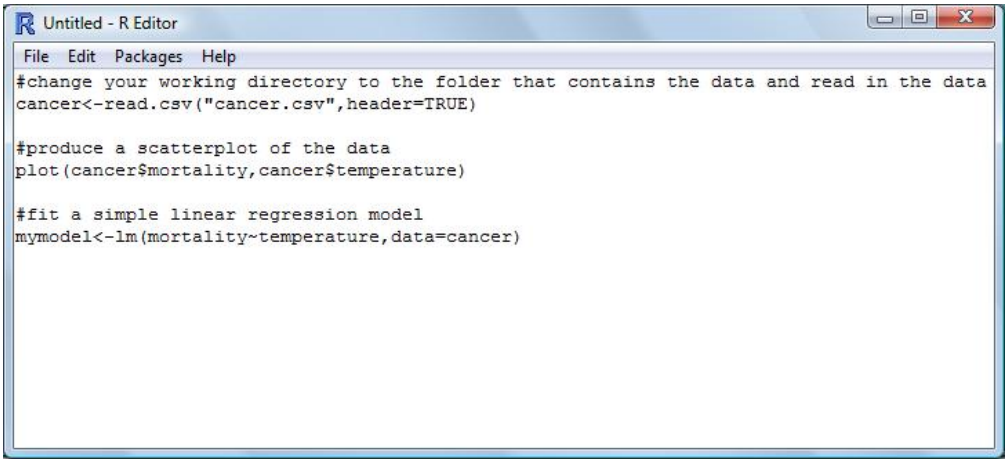
8. Comments in the script file

- Comments in a programming language are short notes that can be added to the code in the script file to make it easier for another person to read and understand what the program is being asked to do
- Comments are not meant to be read by the program
- In R and R Commander comments must be preceded by "#" on each line to ensure that the program ignores the comments

In R:

- When using R, code should be written in the script editor, with comments to explain what the code has been written to do

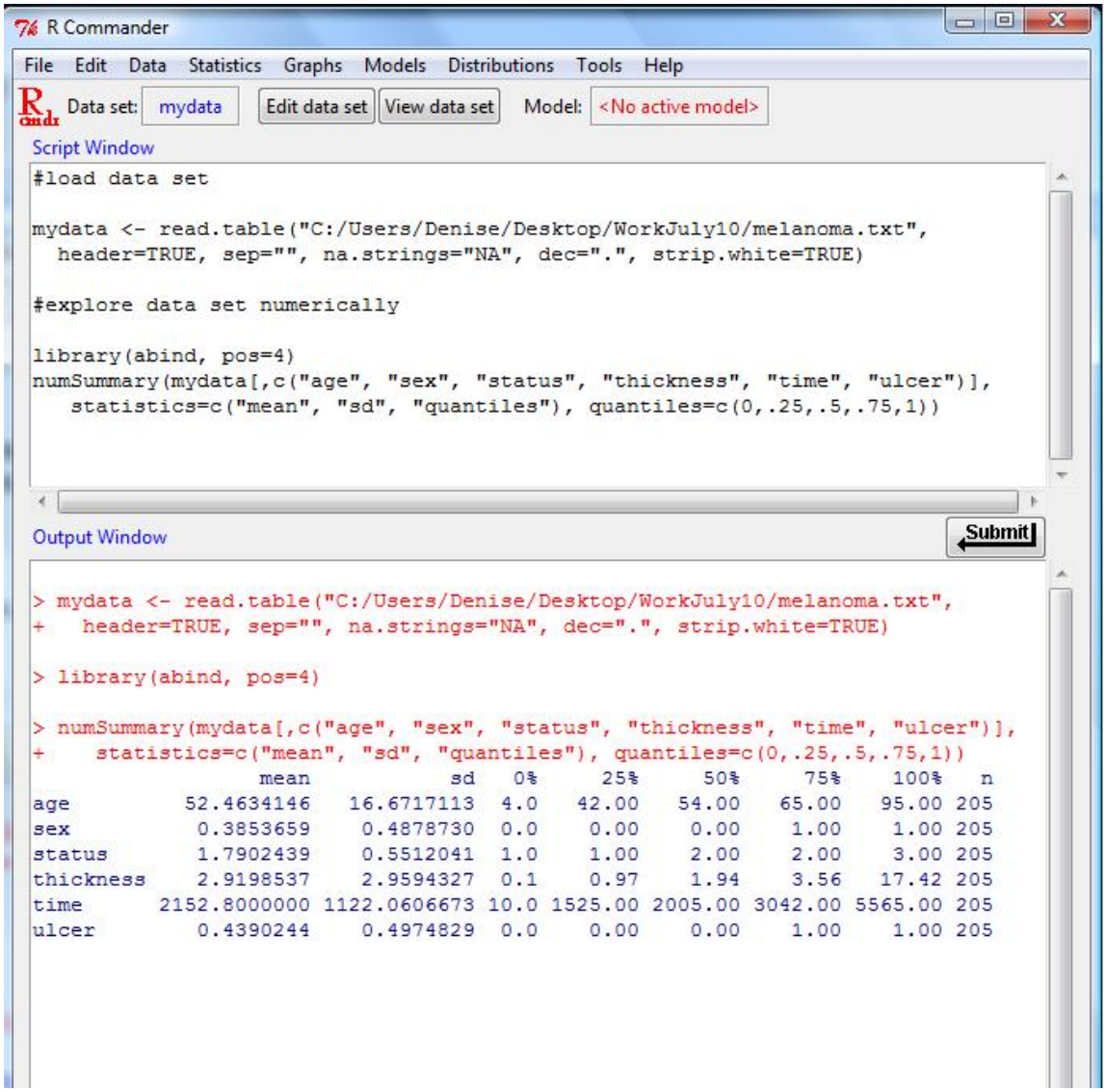
  File→New script



- Once you have entered the code with comments you can run the code in R by copying and pasting the code into R
- R will ignore those lines preceded by "#"
- Once the code has been written and edited it can be copied and pasted into the R console window

In R Commander:

- Add comments in the script window, beginning with "#" on each line, between sections of work to explain what the program is being asked to do



9. Saving your script, output and graphs

9.1 Saving your script

In R:

File→Save as…
Enter the script file name ending with ".R" extension

In R Commander:

File→Save script as…
The ".R" extension need not be added at the end of the name

9.2 Saving your output

- Output generated in R and in R Commander can be saved in a text file

In R:

File→Save to file…

In R Commander:

File→Save output as…

9.3 Saving your graphs

In R and R Commander:

In the graphics window,

File→Save as…
select the type of file (metafile, pdf, jpeg etc)

Note: It may be more useful to copy and paste the output and graphics into a document (such as a word document), where notes and explanations can be added.
To copy graphics to a word document,

- right-click on the graph, *Copy as metafile,* and paste into the document (right-click *Paste* or use the combination *ctrl-v)*

OR

- *File→Copy to clipboard…*

  Select the file type (bitmap, metafile) then paste into the document

10. Closing R and R Commander

In R:

File→Exit

You will be asked if you would like to save your workspace. In most cases, select no.

In R Commander:

File→Exit…

Choose whether you would like exit from R Commander alone or R Commander and R

You will be asked whether you would like to save your script file and output.

11. Plug-ins for R and R Commander

- Plug-ins are packages that extend the range of applications for R and R Commander
- Some plug-ins have been created for R Commander to support R packages and plug-ins
- For example, the R Commander plug-in RcmdrPlugin.HH provides support for the HH package
- RcmdrPlugin.HH can be useful for some calculations related to linear regression models
- To use a plug-in:

Install the plug-in in R

```
install.packages("name")
```

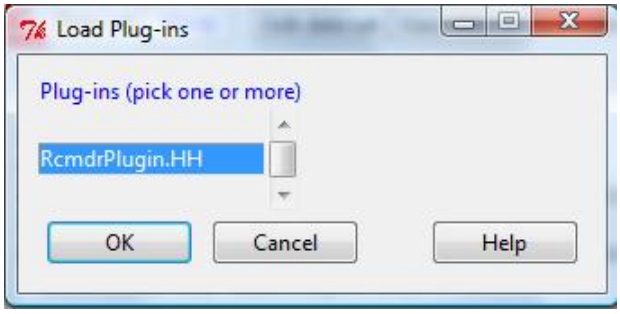
Load the plug-in in R Commander

In R:

```
library("name")
```

In R Commander:

Tools→Load Rcmdr plug-in(s)…



CAUTION: R Commander must be restarted for the plug-in to take effect and the program will prompt you to do so. If you require the use of a plug-in load the plug-in before you have completed any work or be sure to save your script and output files before loading the plug-in.

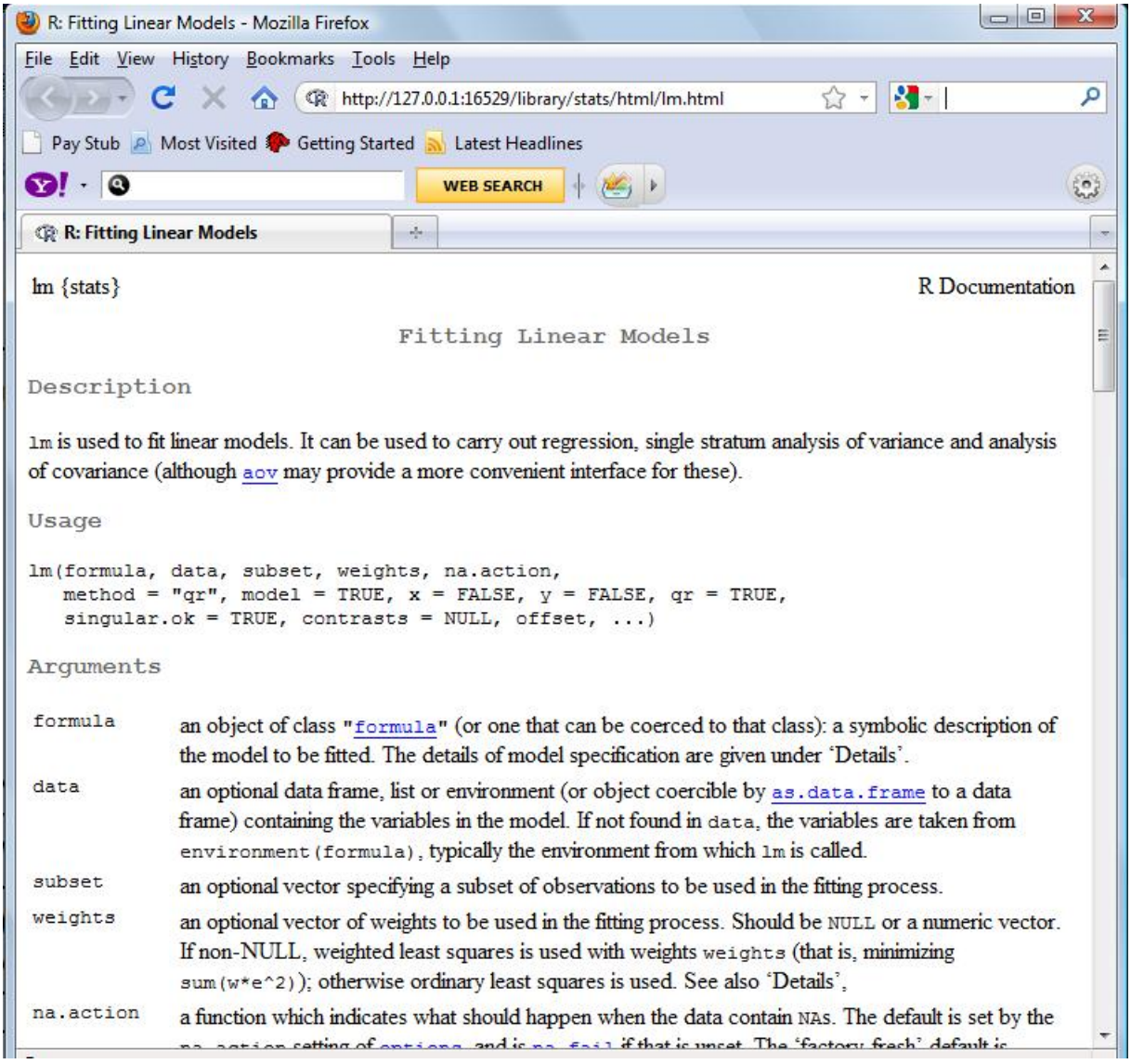
12. Getting help in R & R Commander

- Say you want help on the function **lm()** which fits linear models. You can get help on it through the Help menu, or by typing the following at the prompt (">")

```
?lm
```

or

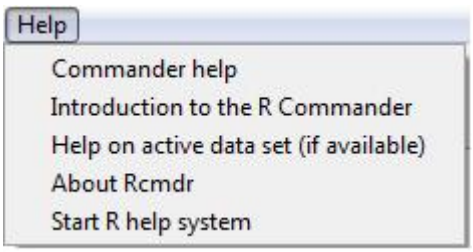
```
help(lm)
```



- Warning: it takes a while to get used to R help files; they are very technical. Try looking (and running) the examples at the end of the help file and/or consulting other suggested references on this topic if you're struggling to understand the help file.

In R Commander:

- help files for R Commander, including a document on the basics of R Commander, can be accessed via the "Help" menu on the menu bar in R Commander



13. References

- Fox, John A., *Getting Started with R Commander.*
- Fox, John A. (2005) *The R Commander: A basic-statistics graphical user interface to R*. Journal of Statistical Software, 19(9):1—42.
- Karp, N. A. (2010) *R Commander an Introduction*
- Murdoch, D.J. (2002) *Introduction to Using R.*
- Venebles, W.N., Smith, D.M. and the R Development Core Team (2009) *An introduction to R: Notes on R: A Programming Environment for Data Analysis and Graphics, Version 2.9.2.*