

Getting Started With the R Commander*

John Fox

Version 1.9-1 (4 September 2012)

1 Starting the R Commander

This document directly describes the use of the **R Commander** under the **Windows** version of **R**. There are small differences in the appearance and use of the **R Commander** under **Mac OS X** and on **Linux** and **Unix** systems. Information about installing the **R Commander** on these platforms is available by following the link to the installation notes at the **R Commander** web page <<http://socserv.socsci.mcmaster.ca/jfox/Misc/Rcmdr/index.html>>, or directly at <tinyurl.com/Rcmdr>.

Once **R** is running, simply loading the **Rcmdr** package by typing the command `library(Rcmdr)` into the *R Console* starts the **R Commander** graphical user interface (“GUI”). To function optimally under **Windows**, the **R Commander** requires the single-document interface (SDI) to **R**.¹ After loading the package, *R Console* and *R Commander* windows should appear more or less as in Figures 1 and 2. These and other screen images in this document were created under **Windows 7**; if you use another version of **Windows** (or, of course, another computing platform), then the appearance of the screen may differ.²

The *R Commander* and *R Console* windows float freely on the desktop. You will normally use the menus and dialog boxes of the **R Commander** to read, manipulate, and analyze data.

- **R** commands generated by the **R Commander** GUI appear in the upper text window (labelled *Script Window*) within the main *R Commander* window. You can also type **R** commands directly into the script window or at the `>` (greater-than) prompt in the *R Console*; the main purpose of the **R Commander**, however, is to avoid having to type commands.
- Printed output appears by default in the second text window (labelled *Output Window*).
- The lower, gray window (labelled *Messages*) displays error messages, warnings, and some other information (“notes”), such as the start-up message in Figure 2.
- When you create graphs, these will appear in a separate *Graphics Device* window.

*This manual is adapted and updated from Fox (2005). Please address correspondence to jfox@mcmaster.ca.

¹The **Windows** version of **R** is normally run from a multiple-document interface (“MDI”), which contains the *R Console* window, *Graphical Device* windows created during the session, and any other windows related to the **R** process. In contrast, under the single-document interface (“SDI”), the *R Console* and *Graphical Device* windows are not contained within a master window. There are several ways to run **R** in SDI mode — for example, by selecting the SDI when **R** is installed, by editing the `Rconsole` file in **R**’s `etc` subdirectory, or by adding `--sdi` to the *Target* field in the *Shortcut* tab of the **R** desktop icon’s *Properties*. This limitation of the **Rcmdr** package is inherited from the `tcltk` package, on which **Rcmdr** depends.

²Notice that **Rcmdr** requires some packages in addition to several of the “recommended” packages that are normally distributed with **R**. **Rcmdr**, the required packages, and many other contributed packages are available for download from the Comprehensive R Archive Network (CRAN) at <<http://cran.r-project.org/>>.

If these packages are not installed, the **Rcmdr** will offer to install them from the Internet or from local files (e.g., on a CD/ROM). If you install the **Rcmdr** package via the **Windows** “R GUI,” not all of the packages on which the **Rcmdr** depends will be installed. You can install the **Rcmdr** package and all of the packages on which it depends via the `install.packages` function, setting the argument `dependencies = TRUE`.

Thanks to Dirk Eddelbuettel, **Debian Linux** users need only issue the command `$ apt-get install r-cran-rcmdr` to install the **Rcmdr** package along with all of the packages that it requires. In any event, building and installing the **Rcmdr** package on **Linux** systems is typically straightforward. The task can be more formidable under **Mac OS X**, since the `tcltk` package on which the **Rcmdr** depends requires that **Tcl/Tk** be installed and that **R** is running under **X-Windows**.

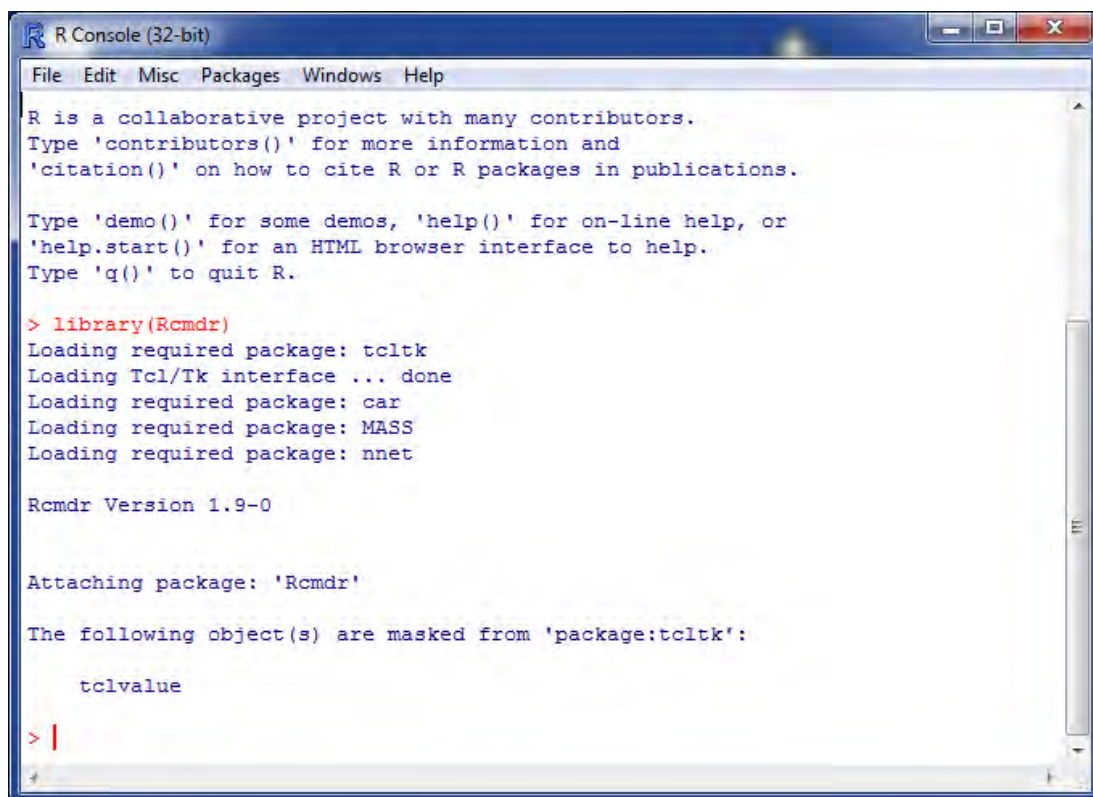


Figure 1: The *R Console* window after loading the **Rcmdr** package.



Figure 2: The *R Commander* window at start-up.

There are several menus along the top of the *R Commander* window:

- File** Menu items for loading and saving script files; for saving output and the **R** workspace; and for exiting.
- Edit** Menu items (*Cut*, *Copy*, *Paste*, etc.) for editing the contents of the script and output windows. Right clicking in the script or output window also brings up an edit “context” menu.
- Data** Submenus containing menu items for reading and manipulating data.
- Statistics** Submenus containing menu items for a variety of basic statistical analyses.
- Graphs** Menu items for creating simple statistical graphs.
- Models** Menu items and submenus for obtaining numerical summaries, confidence intervals, hypothesis tests, diagnostics, and graphs for a statistical model, and for adding diagnostic quantities, such as residuals, to the data set.
- Distributions** Cumulative probabilities, probability densities or masses, quantiles, and graphs of standard statistical distributions (to be used, for example, as a substitute for statistical tables) and samples from these distributions.
- Tools** Menu items for loading **R** packages unrelated to the **Rcmdr** package (e.g., to access data saved in another package), for loading **Rcmdr** plug-in packages (see Fox, 2007, and Fox and Carvalho, 2012), and for setting some options.
- Help** Menu items to obtain information about the **R Commander** (including this manual). As well, each **R Commander** dialog box has a *Help* button (see below).

The complete menu “tree” for the **R Commander** (version 1.9-0) is shown below. Most menu items lead to dialog boxes, as illustrated later in this manual. Menu items are inactive (“grayed out”) if they are inapplicable to the current context.

```
File - Change working directory
      |- Open script file
      |- Save script
      |- Save script as
      |- Save output
      |- Save output as
      |- Save R workspace
      |- Save R workspace as
      |- Exit - from Commander
              |- from Commander and R

Edit - Cut
      |- Copy
      |- Paste
      |- Delete
      |- Find
      |- Select all
      |- Undo
      |- Redo
      |- Clear Window

Data - New data set
      |- Load data set
      |- Merge data sets
      |- Import data - from text file, clipboard, or URL
              |- from SPSS data set
              |- from SAS xport file
              |- from Minitab data set
              |- from STATA data set
              |- from Excel, Access, or dBase data set [32-bit Windows only]
              |- from Excel file [currently 64-bit Windows only]
      |- Data in packages - List data sets in packages
              |- Read data set from attached package
      |- Active data set - Select active data set
              |- Refresh active data set
              |- Help on active data set (if available)
              |- Variables in active data set
              |- Set case names
              |- Subset active data set
              |- Aggregate variables in active data set
              |- Remove row(s) from active data set
              |- Stack variables in active data set
              |- Remove cases with missing data
              |- Save active data set
              |- Export active data set
      |- Manage variables in active data set - Recode variable
              |- Compute new variable
              |- Add observation numbers to data set
              |- Standardize variables
              |- Convert numeric variables to factors
              |- Bin numeric variable
              |- Reorder factor levels
              |- Define contrasts for a factor
```

- |- Rename variables
 - |- Delete variables from data set
- Statistics - Summaries - Active data set
 - |- Numerical summaries
 - |- Frequency distributions
 - |- Count missing observations
 - |- Table of statistics
 - |- Correlation matrix
 - |- Correlation test
 - |- Shapiro-Wilk test of normality
 - |- Contingency Tables - Two-way table
 - |- Multi-way table
 - |- Enter and analyze two-way table
 - |- Means - Single-sample t-test
 - |- Independent-samples t-test
 - |- Paired t-test
 - |- One-way ANOVA
 - |- Multi-way ANOVA
 - |- Proportions - Single-sample proportion test
 - |- Two-sample proportions test
 - |- Variances - Two-variances F-test
 - |- Bartlett's test
 - |- Levene's test
 - |- Nonparametric tests - Two-sample Wilcoxon test
 - |- Paired-samples Wilcoxon test
 - |- Kruskal-Wallis test
 - |- Friedman rank-sum test
 - |- Dimensional analysis - Scale reliability
 - |- Principal-components analysis
 - |- Factor analysis
 - |- Confirmatory factor analysis
 - |- Cluster analysis - k-means cluster analysis
 - |- Hierarchical cluster analysis
 - |- Summarize hierarchical clustering
 - |- Add hierarchical clustering to data set
 - |- Fit models - Linear regression
 - |- Linear model
 - |- Generalized linear model
 - |- Multinomial logit model
 - |- Ordinal regression model
- Graphs - Color palette
 - |- Index plot
 - |- Histogram
 - |- Stem-and-leaf display
 - |- Boxplot
 - |- Quantile-comparison plot
 - |- Scatterplot
 - |- Scatterplot matrix
 - |- Line graph
 - |- XY conditioning plot
 - |- Plot of means
 - |- Strip chart
 - |- Bar graph
 - |- Pie chart

```

|- 3D graph - 3D scatterplot
|   |- Identify observations with mouse
|   |- Save graph to file
|- Save graph to file - as bitmap
|   |- as PDF/Postscript/EPS
|   |- 3D RGL graph
Models - Select active model
|- Summarize model
|- Add observation statistics to data
|- Confidence intervals
|- Akaike Information Criterion (AIC)
|- Bayesian Information Criterion (BIC)
|- Stepwise model selection
|- Subset model selection
|- Hypothesis tests - ANOVA table
|   |- Compare two models
|   |- Linear hypothesis
|- Numerical diagnostics - Variance-inflation factors
|   |- Breusch-Pagan test for heteroscedasticity
|   |- Durbin-Watson test for autocorrelation
|   |- RESET test for nonlinearity
|   |- Bonferroni outlier test
|- Graphs - Basic diagnostic plots
|   |- Residual quantile-comparison plot
|   |- Component+residual plots
|   |- Added-variable plots
|   |- Influence plot
|   |- Effect plots
Distributions - Continuous distributions - Normal distribution - Normal quantiles
|   |- Normal probabilities
|   |- Plot normal distribution
|   |- Sample from normal distribution
|   |- t distribution - t quantiles
|   |- t probabilities
|   |- Plot t distribution
|   |- Sample from t distribution
|   |- Chi-squared distribution - Chi-squared quantiles
|   |- Chi-squared probabilities
|   |- Plot chi-squared distribution
|   |- Sample from chi-squared distribution
|   |- F distribution - F quantiles
|   |- F probabilities
|   |- Plot F distribution
|   |- Sample from F distribution
|   |- Exponential distribution - Exponential quantiles
|   |- Exponential probabilities
|   |- Plot exponential distribution
|   |- Sample from exponential distribution
|   |- Uniform distribution - Uniform quantiles
|   |- Uniform probabilities
|   |- Plot uniform distribution
|   |- Sample from uniform distribution
|   |- Beta distribution - Beta quantiles
|   |- Beta probabilities

```


- | - Introduction to the R Commander
- | - Help on active data set (if available)
- | - About Rcmdr
- | - Start R help system

The **R Commander** interface includes a few elements in addition to the menus and dialogs:

- Below the menus is a “toolbar” with a row of buttons.
 - The left-most (flat) button shows the name of the active data set. Initially there is no active data set. If you press this button, you will be able to choose among data sets currently in memory (if there is more than one). Most of the menus and dialogs in the **R Commander** reference the active data set. (The *File*, *Edit*, and *Distributions* menus are exceptions.)
 - Two buttons allow you to open the **R** data editor to modify the active data set or a viewer to examine it. The data-set viewer can remain open while other operations are performed.³
 - A flat button indicates the name of the active statistical model — a linear model (such as a linear-regression model), a generalized linear model, a multinomial logit model, or an ordinal regression model.⁴ Initially there is no active model. If there is more than one model in memory, you can choose among them by pressing the button.
- Immediately below the toolbar is the script window (so labelled), a large scrollable text window. As mentioned, commands generated by the GUI are copied into this window. You can edit the text in the script window or even type your own **R** commands into the window. Pressing the *Submit* button, which is at the right below the script window (or, alternatively, the key combination *Ctrl-r*, for “run,” or *Ctrl-Tab*), causes the line containing the cursor to be submitted (or resubmitted) for execution. If several lines are selected (e.g., by left-clicking and dragging the mouse over them), then pressing *Submit* will cause all of them to be executed. Commands entered into the script window can extend over more than one line, but all lines must be submitted simultaneously. The key combination *Ctrl-a* selects all of the text in the script window, and *Ctrl-s* brings up a dialog box to save the contents of the window.
- Below the script window is a large scrollable and editable text window for output. Commands echoed to this window appear in red, output in dark blue (as in the *R Console*).
- At the bottom is a small gray text window for messages. Error messages are displayed in red text, warnings in green, and other messages in dark blue. Errors and warnings also provide an audible cue by ringing a bell.

Once you have loaded the **Rcmdr** package, you can minimize the *R Console*. The *R Commander* window can also be resized or maximized in the normal manner. If you resize the **R Commander**, the width of subsequent **R** output is automatically adjusted to fit the output window.

The **R Commander** is highly configurable: I have described the default configuration here. Changes to the configuration can be made via the *Tools* → *Options...* menu, or — much more extensively — by setting options in **R**.⁵ See the **Rcmdr** help files for details.

³The data viewer, provided by the `showData` function from David Firth’s **relimp** package, can be slow for data sets with large numbers of variables. When the number of variables exceeds a threshold (initially set to 100), the less aesthetically pleasing **R View** command is used instead to display the data set. To use **View** regardless of the number of variables, set the threshold to 0. See the **R Commander** help file for details.

⁴**R Commander** plug-in packages (Fox, 2007; Fox and Carvalho, 2012) may provide additional classes of models.

⁵A menu item that terminates in ellipses (i.e., three dots, ...) leads to a dialog box; this is a standard GUI convention. In this document, → represents selecting a menu item or submenu from a menu.

2 Data Input

Most of the procedures in the **R Commander** assume that there is an active data set.⁶ If there are several data sets in memory, you can choose among them, but only one is active. When the **R Commander** starts up, there is no active data set.

The **R Commander** provides several ways to get data into **R**.⁷

- On platforms other than **Mac OS X**, you can enter data directly via *Data* → *New data set...*. This is a reasonable choice for a very small data set.
- You can import data from a plain-text (“ascii”) file or the clipboard, over the Internet from a URL, from another statistical package (**Minitab**, **SPSS**, **SAS**, or **Stata**), or (under **Windows**) from an **Excel**, **Access**, or **dBase** data set.
- You can read a data set that is included in an **R** package, either typing the name of the data set (if you know it), or selecting the data set in a dialog box.

2.1 Reading Data From a Text File

For example, consider the data file `Nations.txt`.⁸ The first few lines of the file are as follows:

```
TFR contraception  infant.mortality  GDP region
Afghanistan        6.90    NA  154  2848    Asia
Albania            2.60    NA   32   863    Europe
Algeria            3.81    52   44  1531    Africa
American-Samoa     NA      NA   11   NA     Oceania
Andorra            NA      NA   NA   NA     Europe
Angola             6.69    NA  124   355    Africa
Antigua            NA      53   24  6966    Americas
Argentina          2.62    NA   22  8055    Americas
Armenia            1.70    22   25   354    Europe
Australia          1.89    76    6 20046    Oceania
. . .
```

- The first line of the file contains variable names: **TFR** (the total fertility rate, expressed as number of children per woman), **contraception** (the rate of contraceptive use among married women, in percent), **infant.mortality** (the infant-mortality rate per 1000 live births), **GDP** (gross domestic product per capita, in U.S. dollars), and **region**.
- Subsequent lines contain the data values themselves, one line per country. The data values are separated by “white space” — one or more blanks or tabs. Although it is helpful to make the data values line up vertically, it is not necessary to do so. Notice that the data lines begin with the country names. Because we want these to be the “row names” for the data set, there is no corresponding variable name: That is, there are five variable names but six data values on each line. When this happens, **R** will interpret the first value on each line as the row name.
- Some of the data values are missing. In **R**, it is most convenient to use **NA** (representing “not available”) to encode missing data, as I have done here.
- The variables **TFR**, **contraception**, **infant.mortality**, and **GDP** are numeric (quantitative) variables; in contrast, **region** contains region names. When the data are read, **R** will treat **region** as a “factor” — that is, as a categorical variable. In most contexts, the **R Commander** distinguishes between numerical variables and factors.

⁶Procedures selected under via the *Distributions* menu are exceptions, as is *Enter and analyze two-way table...* under the *Statistics* → *Contingency tables* menu.

⁷Not all of these data sources may be available on all platforms.

⁸This file resides in the `etc` subdirectory of the **Rcmdr** package.

To read the data file into **R**, select *Data* → *Import data* → *from text file, clipboard, or URL...* from the *R Commander* menus. This operation brings up a *Read Text Data* dialog, as shown in Figure 3. The default name of the data set is **Dataset**. I have changed the name to **Nations**.

Valid **R** names begin with an upper- or lower-case letter (or a period, `.`) and consist entirely of letters, periods, underscores (`_`), and numerals (i.e., 0–9); in particular, do not include any embedded blanks in a data-set name. You should also know that **R** is case-sensitive, and so, for example, **nations**, **Nations**, and **NATIONS** are distinguished, and could be used to represent different data sets.

Clicking the *OK* button in the *Read Text Data* dialog brings up an *Open* file dialog, shown in Figure 4. Here I navigated to the file **Nations.txt**. Clicking the *Open* button in the dialog will cause the data file to be read. Once the data file is read, it becomes the active data set in the **R Commander**. As a consequence, in Figure 5, the name of the data set appears in the data set button near the top left of the *R Commander* window.

I clicked the *View data set* button to bring up the data viewer window, also shown in Figure 5. Notice that the commands to read and view the **Nations** data set (the **R** `read.table` and `showData` commands) appear, partially obscured by the display of the data set, in the script and output windows. When the data set is read and becomes the active data set, a note appears in the messages window. The **R Commander** also issued a `library` command to load the **relimp** package, which was used to display the data set; here, as in general, packages are loaded automatically by the **R Commander** as they are needed.

The `read.table` command creates an **R** “data frame,” which is an object containing a rectangular cases-by-variables data set: The rows of the data set represent cases or observations and the columns represent variables. Data sets in the **R Commander** are **R** data frames.

2.2 Entering Data Directly

To enter data directly into the **R** basic spreadsheet-like data editor you can proceed as follows.⁹ As an example, I use a very small data set from Problem 2.44 in Moore (2000):

- Select *Data* → *New data set...* from the *R Commander* menus. Optionally enter a name for the data set, such as **Problem2.44**, in the resulting dialog box, and click the *OK* button. (Remember that **R** names cannot include intervening blanks.) This will bring up a *Data Editor* window with an empty data set.
- Enter the data from the problem into the first two columns of the data editor. You can move from one cell to another by using the arrow keys on your keyboard, by tabbing, by pressing the *Enter* key, or by pointing with the mouse and left-clicking. When you are finished entering the data, the window should look like Figure 6.
- Next, click on the name **var1** above the first column. This will bring up a *Variable editor* dialog box, as in Figure 7.
- Type the variable name **age** in the box, just as I have, and click the **X** button at the upper-right corner of the *Variable editor* window, or press the *Enter* key, to close the window. Repeat this procedure to name the second column **height**. The *Data Editor* should now look like Figure 8.
- Select *File* → *Close* from the *Data Editor* menus or click the **X** at the upper-right of the *Data Editor* window. The data set that you entered is now the active data set in the **R Commander**.

⁹Because of limitations in the **R** data editor for that platform, data sets cannot be entered directly under **Mac OS X**, and the corresponding menu item consequently is suppressed.

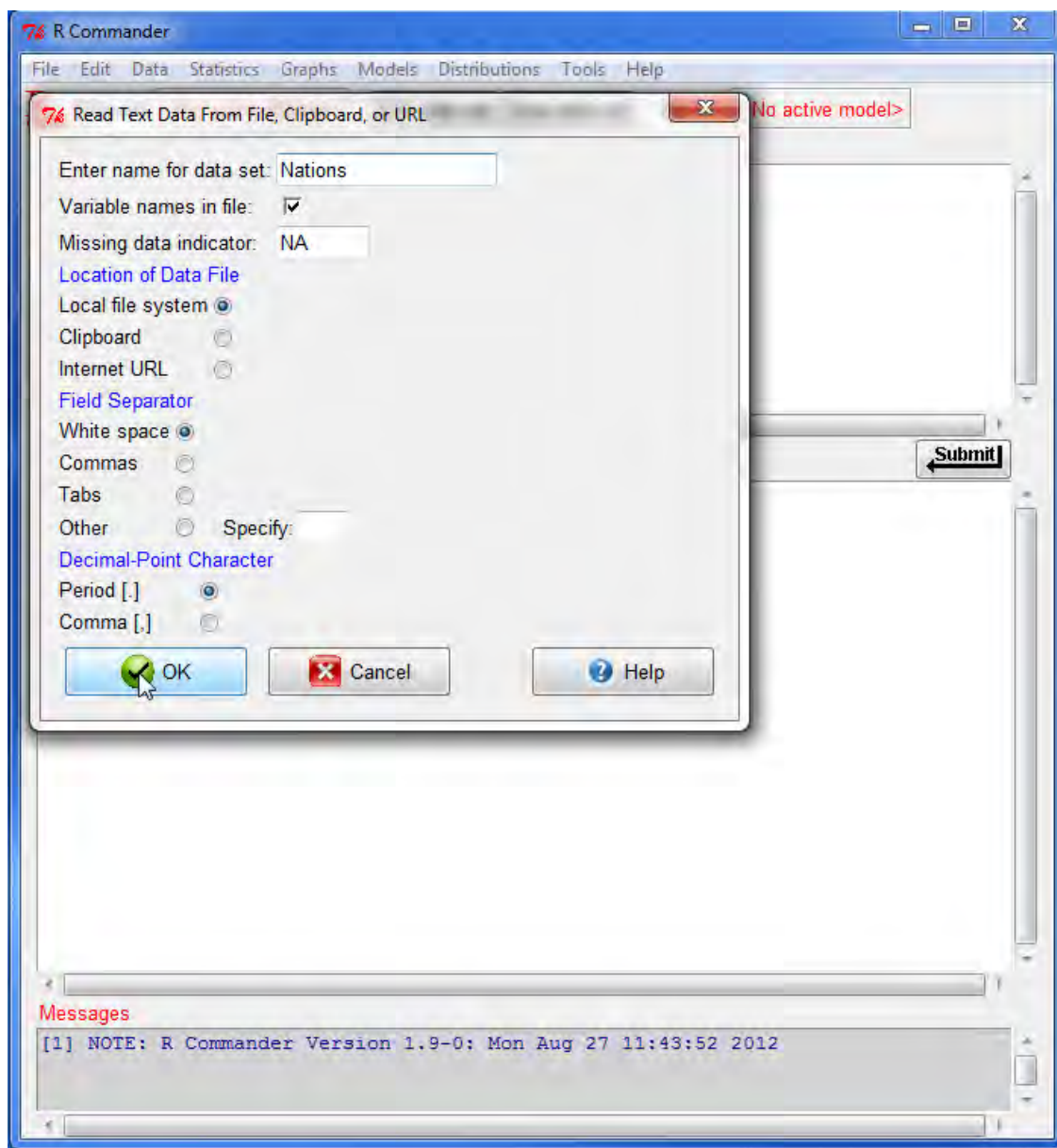


Figure 3: Reading data from a text file.

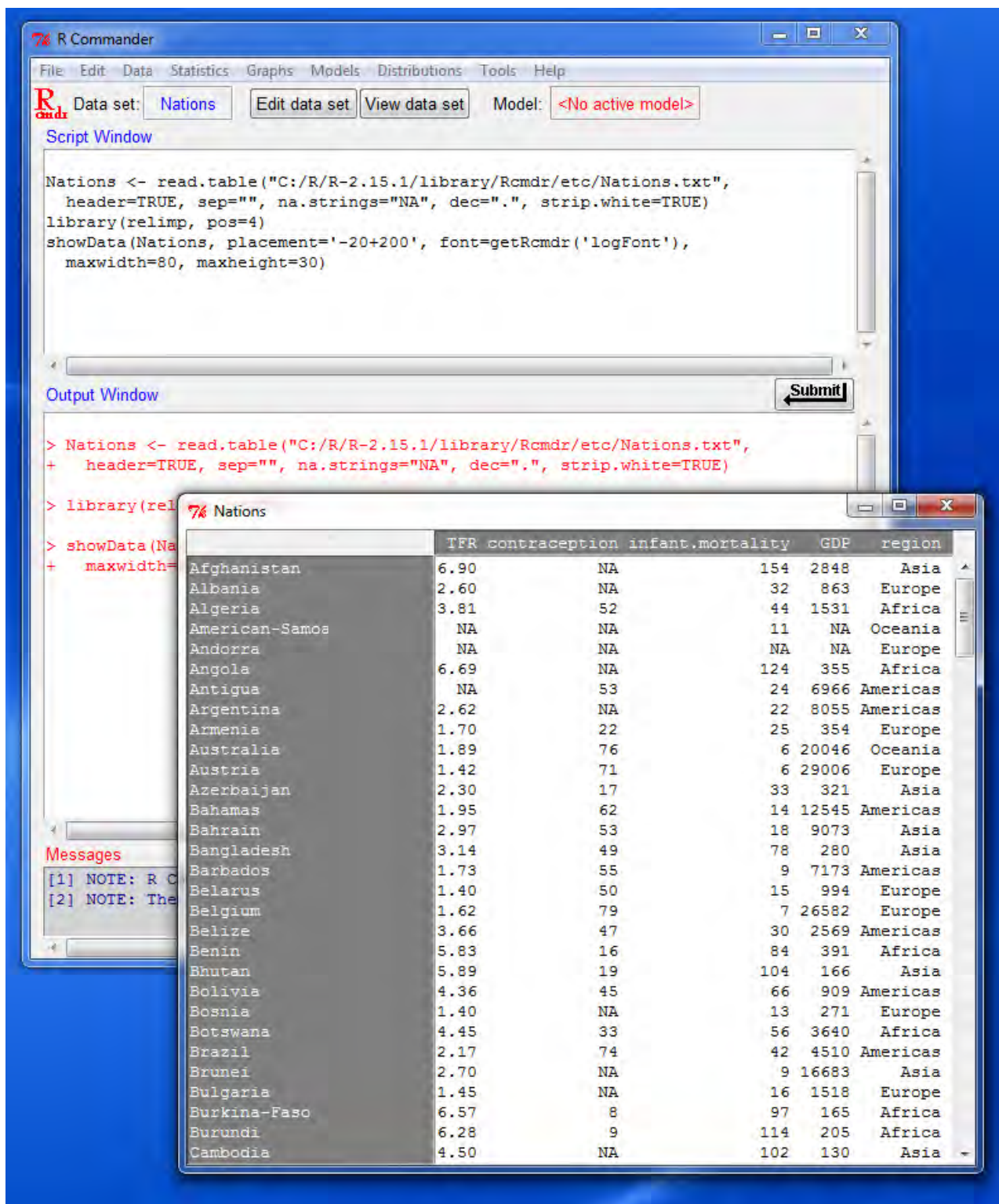
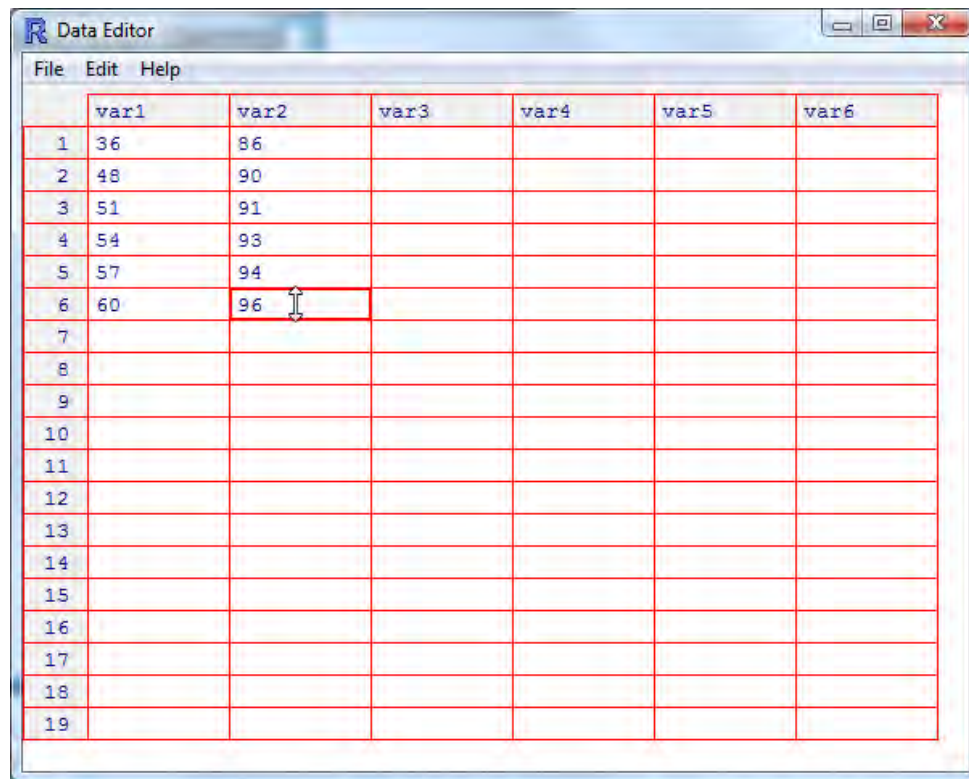
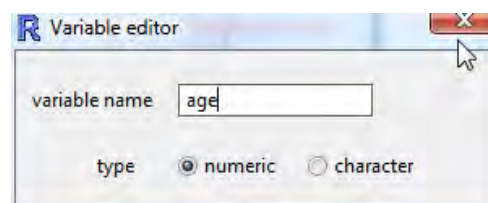


Figure 5: Displaying the active data set.



	var1	var2	var3	var4	var5	var6
1	36	86				
2	48	90				
3	51	91				
4	54	93				
5	57	94				
6	60	96				
7						
8						
9						
10						
11						
12						
13						
14						
15						
16						
17						
18						
19						

Figure 6: Data editor after the data are entered.

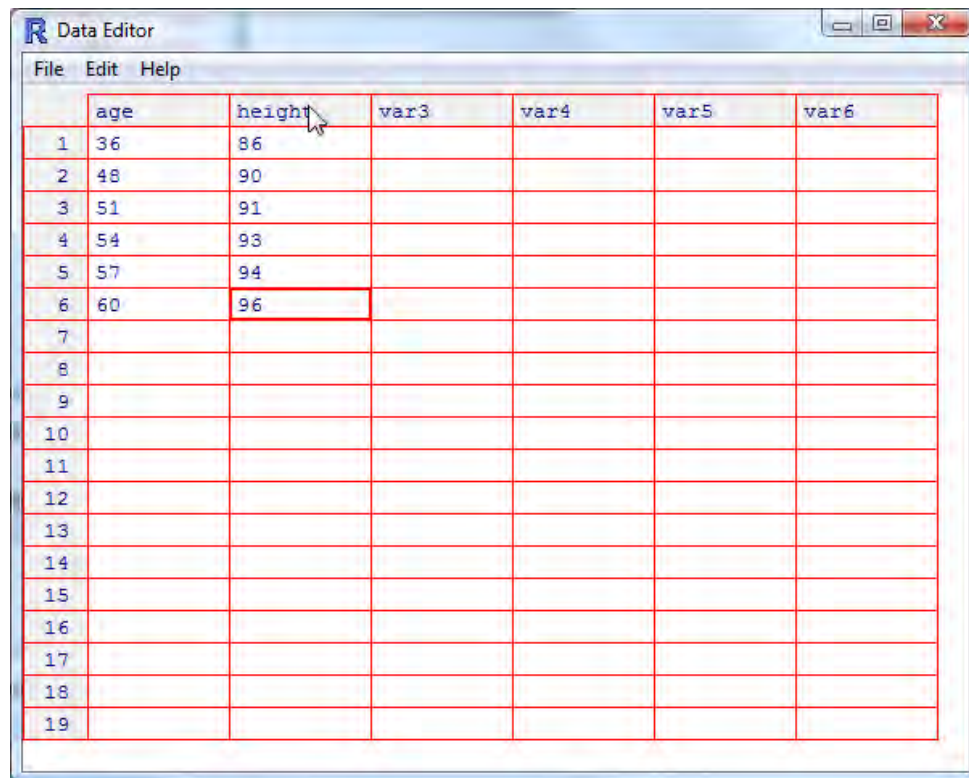


Variable editor

variable name

type ☒ numeric ☐ character

Figure 7: Dialog box for changing the name of a variable in the data editor.



	age	height	var3	var4	var5	var6
1	36	86				
2	48	90				
3	51	91				
4	54	93				
5	57	94				
6	60	96				
7						
8						
9						
10						
11						
12						
13						
14						
15						
16						
17						
18						
19						

Figure 8: The *Data Editor* window after both variable names have been changed.

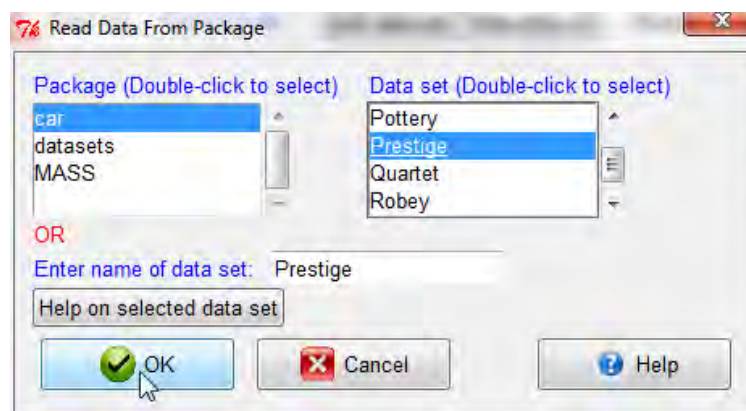


Figure 9: Reading data from an attached package — in this case the **Prestige** data set from the **car** package.

flat button with the active data set's name near the top left of the *R Commander* window, selecting from among a list of data sets currently resident in memory.

Selecting *Statistics* → *Summaries* → *Active data set* produces the results shown in Figure 10. For each numerical variable in the data set (*TFR*, *contraception*, *infant.mortality*, and *GDP*), **R** reports the minimum and maximum values, the first and third quartiles, the median, and the mean, along with the number of missing values. For the categorical variable *region*, we get the number of observations at each “level” of the factor. Had the data set included more than ten variables, the **R Commander** would have asked us whether we really want to proceed — potentially protecting us from producing unwanted voluminous output.

Similarly, selecting *Statistics* → *Summaries* → *Numerical summaries...* brings up the dialog box shown in Figure 11. Only numerical variables are shown in the variable list in this dialog; the factor *region* is missing, because it is not sensible to compute numerical summaries for a factor. Clicking on *infant.mortality*, and then clicking *OK*, produces the following output (in the output window):¹²

```
> numSummary(Nations[, "infant.mortality"], statistics=c("mean", "sd", "IQR",
+ "quantiles"), quantiles=c(0,.25,.5,.75,1))
      mean      sd IQR 0% 25% 50% 75% 100%   n NA
43.47761 38.75604  54  2  12  30  66  169 201  6
```

By default, the **R** command that is executed prints out the mean, standard deviation (*sd*), and interquartile range (*IQR*) of the variable, along with quantiles (percentiles) corresponding to the minimum, the first quartile, the median, the third quartile, and the maximum; *n* is the number of valid observations, and *NA* the number of missing values.

As is typical of **R Commander** dialogs, the *Numerical Summaries* dialog box in Figure 11 includes *OK*, *Cancel*, *Reset*, and *Help* buttons. The *Help* button leads to a help page (which appears in your web browser) either for the dialog itself or (as here) for an **R** function that the dialog invokes. The *Reset* button, which is present in most **R Commander** dialogs, resets the dialog to its original state; otherwise, the dialog retains selections from the previous invocation. Dialog state is also reset when the active data set changes.

The *Numerical Summaries* dialog box also makes provision for computing summaries within groups defined by the levels of a factor. Clicking on the *Summarize by groups...* button brings up the *Groups* dialog, as shown in Figure 12. Because there is only one factor in the *Nations* data set, only the variable *region* appears in the variable list, and it is pre-selected; clicking *OK* changes the *Summarize by groups...* button to *Summarize by region* (see Figure 13); clicking *OK* again produces the following results:

```
> numSummary(Nations[,c("GDP", "infant.mortality")], groups=Nations$region,
+ statistics=c("mean", "sd", "IQR", "quantiles"), quantiles=c(0,.25,.5,.75,1))
```

Variable: GDP

	mean	sd	IQR	0%	25%	50%	75%	100%	n	NA
Africa	1196.000	2089.614	795.50	36	209.00	389.5	1004.50	11854	54	1
Americas	5398.000	6083.311	5268.50	386	1749.25	2765.5	7017.75	26037	40	1
Asia	4505.051	6277.738	6062.50	122	345.00	1079.0	6407.50	22898	39	2
Europe	13698.909	13165.412	24582.25	271	1643.75	9222.5	26226.00	42416	44	1
Oceania	8732.600	11328.708	16409.25	654	1102.75	2348.5	17512.00	41718	20	5

Variable: infant.mortality

	mean	sd	IQR	0%	25%	50%	75%	100%	n	NA
Africa	85.27273	35.188095	50.0	7	61.00	85.0	111.00	169	55	0
Americas	25.60000	17.439713	24.0	6	12.00	21.5	36.00	82	40	1
Asia	45.65854	32.980001	50.0	5	22.00	37.0	72.00	154	41	0
Europe	11.85366	7.122363	10.0	5	6.00	8.0	16.00	32	41	4
Oceania	27.79167	29.622229	26.5	2	9.25	20.0	35.75	135	24	1

¹²To select a single variable in a variable-list box, simply left-click on its name. In some contexts, you will have to (or want to) select more than one variable. In these cases, the usual **Windows** conventions apply: Left-clicking on a variable selects it and de-selects any variables that have previously been selected; *Shift-left-click* extends the selection; and *Ctrl-left-click* toggles the selection for an individual variable.

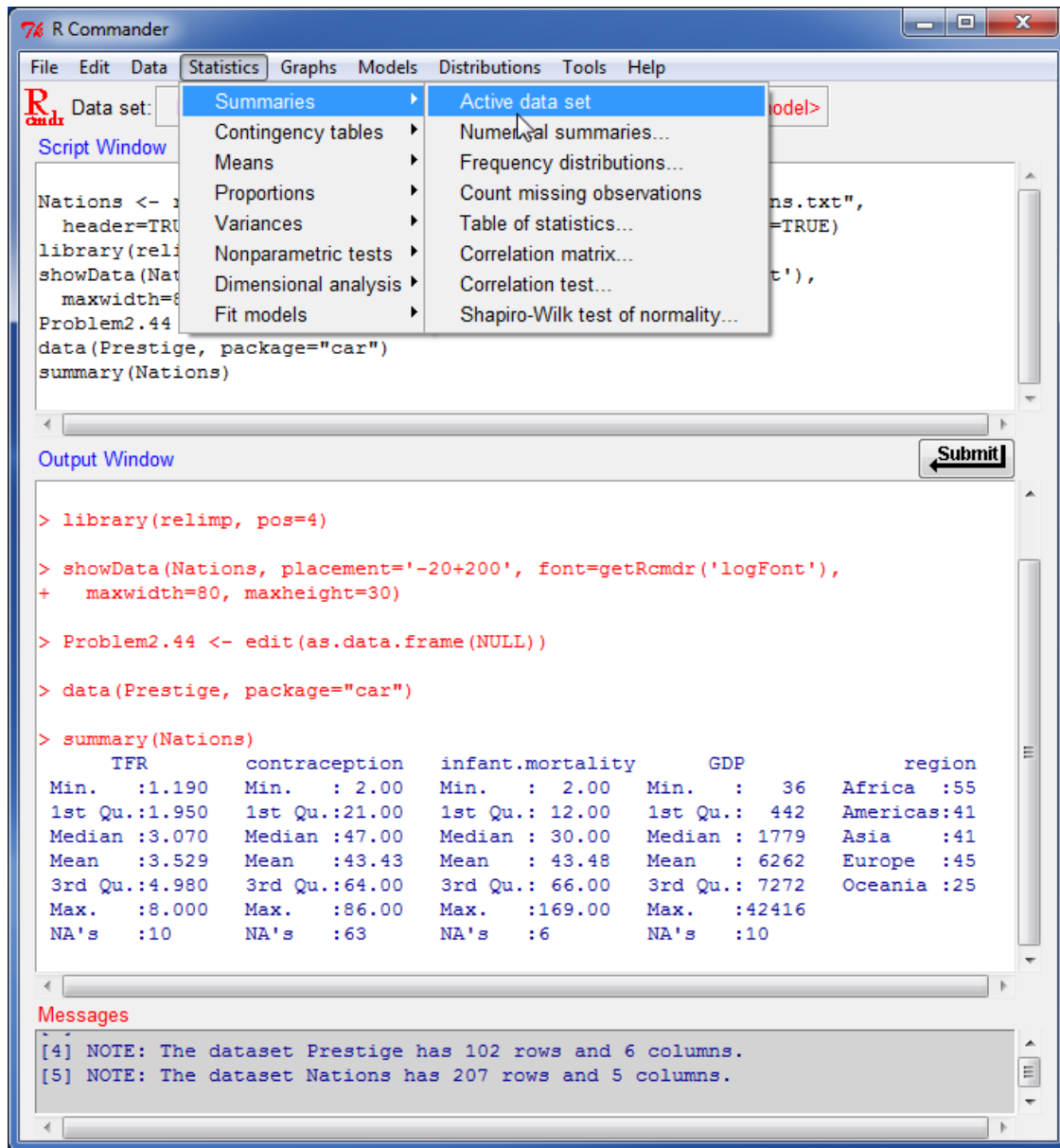


Figure 10: Getting variable summaries for the active data set.

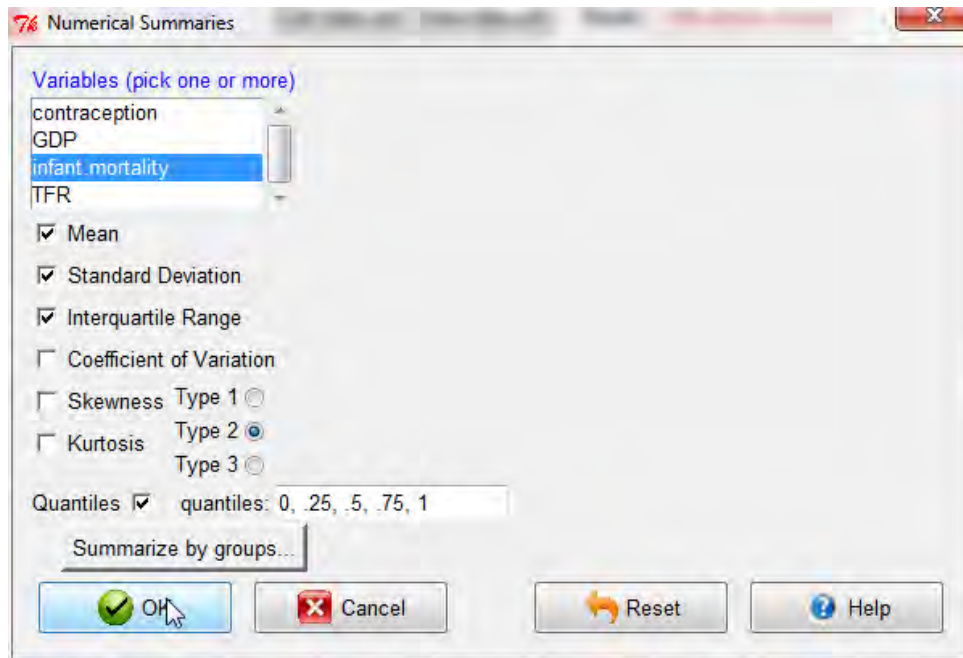


Figure 11: The *Numerical Summaries* dialog box.

Several other **R Commander** dialogs allow you to select a grouping variable in this manner.

Making graphs with the **R Commander** is also straightforward. For example, selecting *Graphs* → *Histogram...* from the *R Commander* menus brings up the *Histogram* dialog box in Figure 14; and clicking on `infant.mortality` followed by *OK*, opens a *Graphics Device* window with the histogram shown in Figure 15.

If you make several graphs in a session, then only the most recent normally appears in the *Graphics Device* window. You can recall previous graphs using the *Page Up* and *Page Down* keys on your keyboard.¹³

¹³ At start-up, the **R Commander** turns on the graph history mechanism; this feature is available only on **Windows** systems. Dynamic three-dimensional scatterplots created by *Graphs* → *3D graph* → *3D scatterplot...* appear in a special *RGL device* window; likewise, effect displays created for statistical models (Fox, 2003; Fox and Hong, 2009) via *Models* → *Graphs* → *Effect plots* appear in individual graphics-device windows.

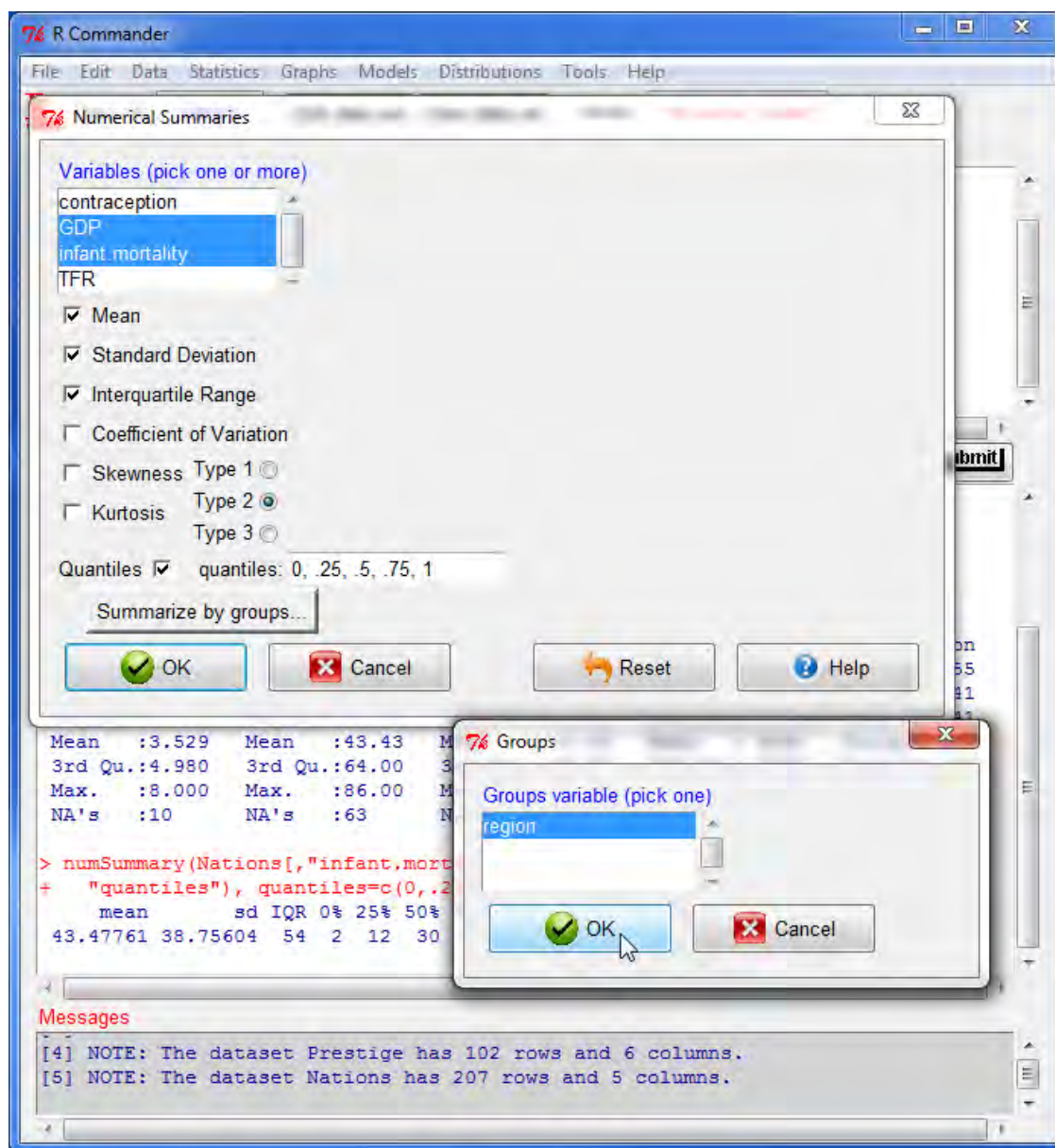


Figure 12: Selecting a grouping variable in the *Groups* dialog box.

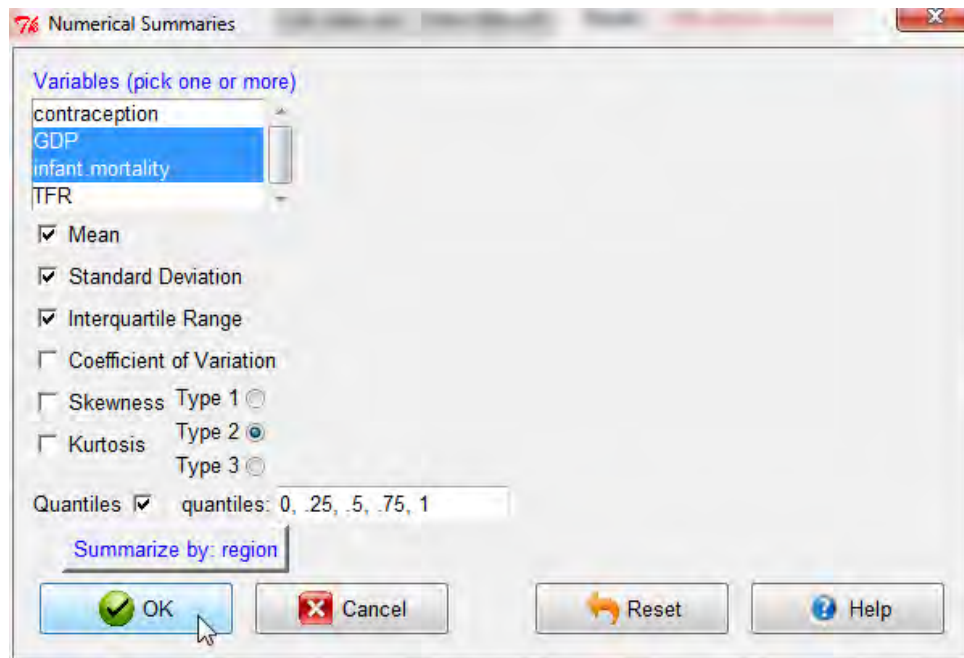


Figure 13: The *Numerical Summaries* dialog box after the grouping variable **region** has been selected and with two numeric variables selected.

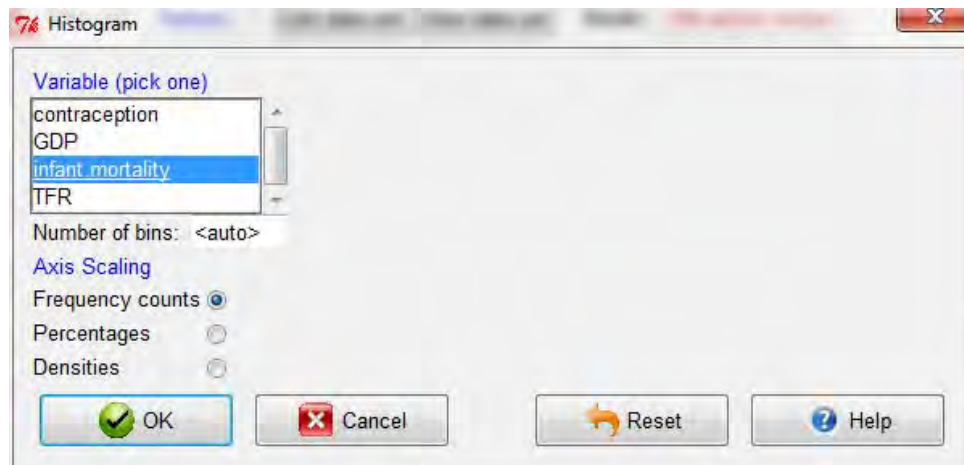


Figure 14: The *Histogram* dialog.

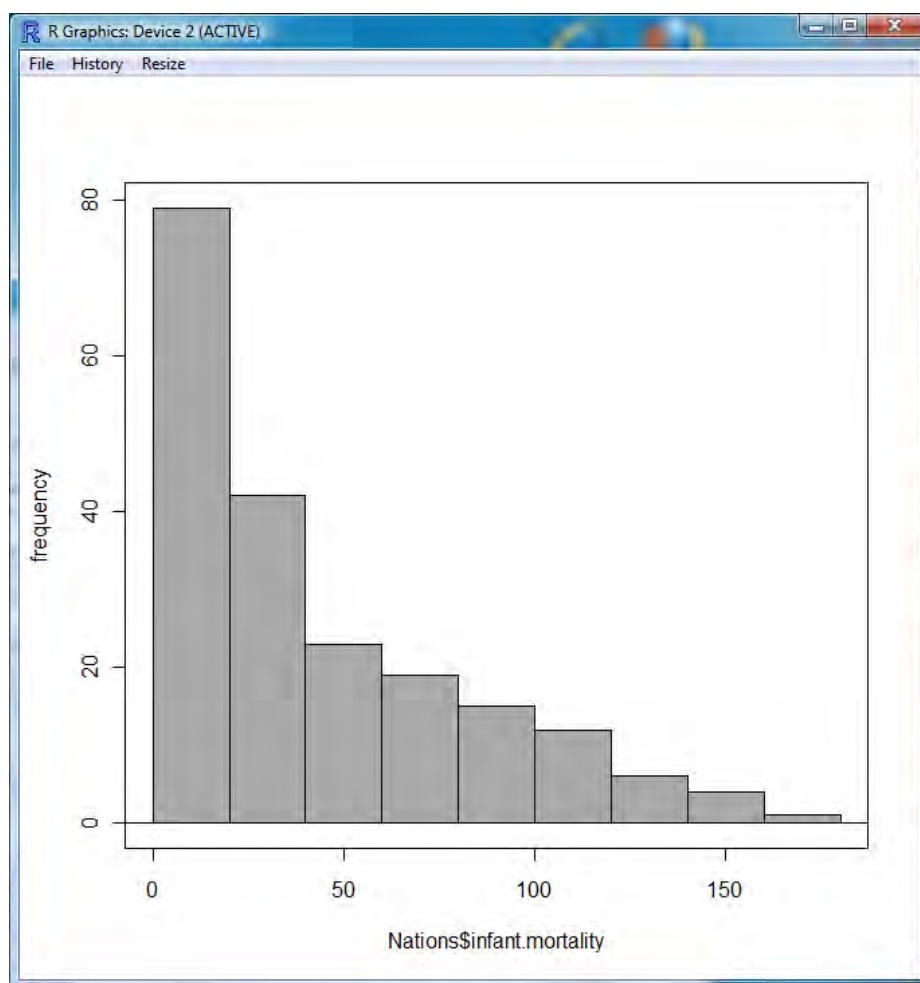


Figure 15: A graphics window containing the histogram for infant mortality in the `Nations` data set.

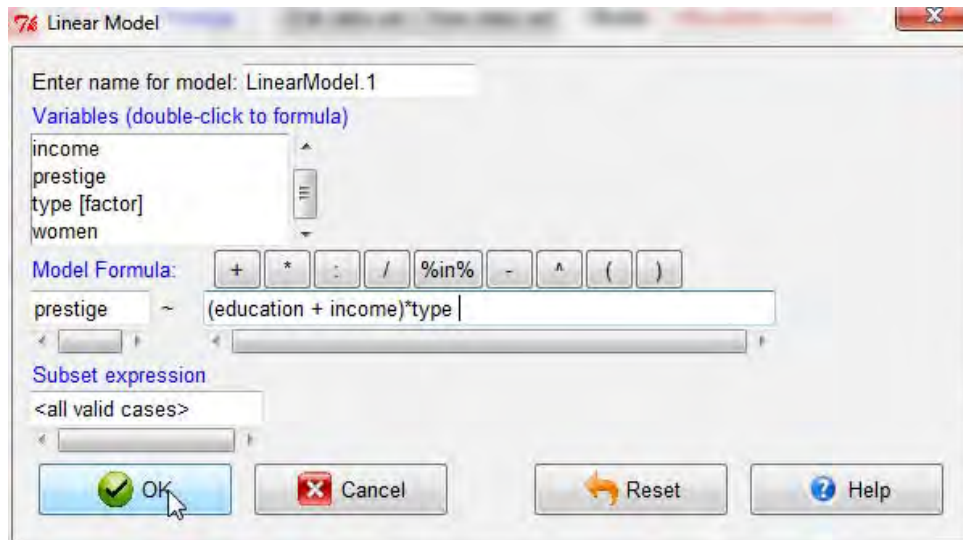


Figure 16: The *Linear Model* dialog box, with **Prestige** from the **car** package as the active data set.

4 Statistical Models

Several kinds of statistical models can be fit in the **R Commander** using menu items under *Statistics* → *Fit models*: linear models (by both *Linear regression* and *Linear model*), generalized linear models, multinomial logit models, and ordinal regression models such as the proportional-odds model [the latter two from Venables and Ripley’s (2002) **nnet** and **MASS** packages, respectively]. Although the resulting dialog boxes differ in certain details (for example, the generalized linear model dialog makes provision for selecting a distributional family and corresponding link function), they share a common general structure, as illustrated in the *Linear Model* dialog in Figure 16.¹⁴

- Double-clicking on a variable in the variable-list box copies it to the model formula — to the left-hand side of the formula, if it is empty, otherwise to the right-hand side (with a preceding + sign if the context requires it). Note that factors (categorical variables) are parenthetically labelled as such in the variable list.
- The row of buttons above the formula can be used to enter operators and parentheses into the right-hand side of the formula.
- You can also type directly into the formula fields, and indeed may have to do so, for example, to put a term such as `log(income)` into the formula.
- The name of the model, here **LinearModel.1**, is automatically generated, but you can substitute any valid **R** name.
- You can type an **R** expression into the box labelled *Subset expression*; if supplied, this is passed to the **subset** argument of the **lm** function, and is used to fit the model to a subset of the observations in the data set. One form of subset expression is a logical expression that evaluates to **TRUE** or **FALSE** for each observation, such as `type != "prof"` (which would select all non-professional occupations from the **Prestige** data set).

¹⁴An exception is the *Linear Regression* dialog in which the response variable and explanatory variables are simply selected by name from list boxes containing the numeric variables in the current data set. The explanation below assumes familiarity with **R** model formulas; see, for example, the *Introduction to R* manual that comes with **R**, which may be accessed from the *Help* menu in the *R Console*.

Clicking the *OK* button produces the following output (in the output window), and makes `LinearModel.1` the active model, with its name displayed in the *Model* button:

```
> LinearModel.1 <- lm(prestige ~ (education + income)*type, data=Prestige)

> summary(LinearModel.1)
```

Call:

```
lm(formula = prestige ~ (education + income) * type, data = Prestige)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-13.462	-4.225	1.346	3.826	19.631

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	2.276e+00	7.057e+00	0.323	0.7478
education	1.713e+00	9.572e-01	1.790	0.0769 .
income	3.522e-03	5.563e-04	6.332	9.62e-09 ***
type[T.prof]	1.535e+01	1.372e+01	1.119	0.2660
type[T.wc]	-3.354e+01	1.765e+01	-1.900	0.0607 .
education:type[T.prof]	1.388e+00	1.289e+00	1.077	0.2844
education:type[T.wc]	4.291e+00	1.757e+00	2.442	0.0166 *
income:type[T.prof]	-2.903e-03	5.989e-04	-4.847	5.28e-06 ***
income:type[T.wc]	-2.072e-03	8.940e-04	-2.318	0.0228 *

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.318 on 89 degrees of freedom

(4 observations deleted due to missingness)

Multiple R-squared: 0.8747, Adjusted R-squared: 0.8634

F-statistic: 77.64 on 8 and 89 DF, p-value: < 2.2e-16

Operations on the active model may be selected from the *Models* menu. For example, *Models* → *Hypothesis tests* → *Anova table...*, followed by selecting the default “Type-II” tests, produces the following output:

```
> Anova(LinearModel.1, type="II")
Anova Table (Type II tests)
```

Response: prestige

	Sum Sq	Df	F value	Pr(>F)
education	1068.0	1	26.7532	1.413e-06 ***
income	1131.9	1	28.3544	7.511e-07 ***
type	591.2	2	7.4044	0.001060 **
education:type	238.4	2	2.9859	0.055574 .
income:type	951.8	2	11.9210	2.588e-05 ***
Residuals	3552.9	89		

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

5 Odds and Ends

5.1 Saving and Printing Output

You can save text output directly from the *File* menu in the *R Commander*; likewise you can save or print a graph from the *File* menu in an *R Graphics Device* window. It is generally more convenient, however, to collect the text output and graphs that you want to keep in a word-processor document. In this manner, you can intersperse **R** output with your typed notes and explanations.

Open a word processor such as **Word**, **OpenOffice Writer**, or even **Windows WordPad**. To copy text from the output window, block the text with the mouse, select *Copy* from the *Edit* menu (or press the key combination *Ctrl-c*, or right-click in the window and select *Copy* from the context menu), and then paste the text into the word-processor window via *Edit* → *Paste* (or *Ctrl-v*), as you would for any **Windows** application. One point worth mentioning is that you should use a mono-spaced (“**typewriter**”) font, such as *Courier New*, for text output from **R**; otherwise the output will not line up neatly.

Likewise to copy a graph, select *File* → *Copy to the clipboard* → *as a Metafile* from the *R Graphics Device* menus; then paste the graph into the word-processor document via *Edit* → *Paste* (or *Ctrl-v*). Alternatively, you can use *Ctrl-w* to copy the graph from the *R Graphics Device*, or right-click on the graph to bring up a context menu, from which you can select *Copy as metafile*.¹⁵ At the end of your **R** session, you can save or print the document that you have created, providing an annotated record of your work.

Alternative routes to saving text and graphical output may be found respectively under the **R Commander** *File* and *Graphs* → *Save graph to file* menus. Saving the **R Commander** script window, via *File* → *Save script*, allows you to reproduce your work on a future occasion.

5.2 Terminating the R Session

There are several ways to terminate your session. For example, you can select *File* → *Exit* → *From Commander and R* from the *R Commander* menus. You will be asked to confirm, and then asked whether you want to save the contents of the script and output windows. Likewise, you can select *File* → *Exit* from the *R Console*; in this case, you will be asked whether you want to save the **R** workspace (i.e., the data that **R** keeps in memory); you would normally answer *No*.

5.3 Entering Commands in the Script Window

The script window provides a simple facility for editing, entering, and executing commands. Commands generated by the **R Commander** appear in the script window, and you can type and edit commands in the window more or less as in any editor. The **R Commander** does not provide a true “console” for **R**, however, and the script window has some limitations. For example, all lines of a multiline command must be submitted simultaneously for execution. For serious **R** programming, it is preferable to use the script editors provided by the **Windows** and **Mac OS X** versions of **R**, or — even better — a programming editor or interactive development environment, such as RStudio <www.rstudio.org>.¹⁶

References

- Fox, J. (2003). Effect displays in R for generalised linear models. *Journal of Statistical Software*, 8(15):1–27.
- Fox, J. (2005). The R Commander: A basic-statistics graphical user interface to R. *Journal of Statistical Software*, 19(9):1–42.
- Fox, J. (2007). Extending the R Commander by “plug in” packages. *R News*, 7(3):46–52.
- Fox, J. and Carvalho, Marilia S. (2012). The RcmdrPlugin.survival package: Extending the R Commander interface to survival analysis. *Journal of Statistical Software*, 49(7):1–32.

¹⁵As you will see when you examine these menus, you can save graphs in a variety of formats, and to files as well as to the clipboard. The procedure suggested here is straightforward, however, and generally results in high-quality graphs. Once again, this description applies to **Windows** systems.

¹⁶The **R Commander** will run under **RStudio**, in which case by default **R Commander** output and messages are directed to the **R** console within **RStudio**, but there are some issues, such as instability and incompatibility with the **RStudio** graphics device.

- Fox, J. and Hong, J. (2009). Effect displays in R for multinomial and proportional-odds logit models: Extensions to the effects package. *Journal of Statistical Software*, 32(1):1–24.
- Moore, D. S. (2000). *The Basic Practice of Statistics, Second Edition*. Freeman, New York.
- Venables, W. N. and Ripley, B. D. (2002). *Modern Applied Statistics with S, Fourth Edition*. Springer, New York.