

S-PLUS® GUIDE
FOR
MOORE AND McCABE'S
**INTRODUCTION TO THE
PRACTICE OF STATISTICS**
FIFTH EDITION

GREGORY L. SNOW
Intermountain Health Care

LAURA M. CHIHARA
Carleton College

TIM HESTERBERG
Insightful Corp.

W. H. FREEMAN AND COMPANY
NEW YORK

Teachers: Chapter Order We encourage teachers to consider using Chapter 14 after Chapter 7. David Moore writes:

I'd like users to consider resampling (Chapter 14) right after Chapter 7 in their course. That's the way *Introduction to the Practice of Statistics* flows: first the traditional procedures (and a strong dose of practical concerns), then resampling methods as an answer to some (not all) of the practical issues. I initially wanted Chapter 14 to be Chapter 8, but bowed to market reality. In some future world, resampling will replace Chapters 6 and 7, but not yet.

In addition to the practical issues that Moore notes, there are sound pedagogical reasons for considering Chapter 14 early. Resampling methods provide concrete graphical illustrations of some of the concepts that students find most difficult, including sampling distributions, standard errors, and P -values. And resampling provides a way for students to check their answers obtained by formula methods.

Students: Chapter 14 Whether or not your course includes Chapter 14, we encourage you to use this chapter in connection with other chapters. Bootstrap methods and permutation tests provide concrete visual representations for some of the more abstract and difficult concepts treated in Chapter 6 and later. These methods may help you better understand the material in other chapters. The approach is based on computer simulation and visualization, rather than the traditional mathematical approach.

For Best Viewing This document is intended to be viewed online. Screen shots are taken at 96 dpi, so should be clear when viewing the document at 100% of actual size on a screen with 96 dpi. If your resolution is different, then another magnification may be better.

Online Version The current version of this document will be available online through links at www.whfreeman.com/ips or www.whfreeman.com/ipsresample.

S-PLUS® is a registered trademark of Insightful Corp.

© 2005 by W. H. Freeman and Company

This document may be freely copied by students and faculty for use as a supplement to Moore and McCabe, *Introduction to the Practice of Statistics*.

Contents

0	Introduction to S-PLUS	1
0.1	Obtaining S-PLUS and Two Libraries	1
0.2	Getting Started	1
0.3	Getting Help	2
0.4	Data in S-PLUS	3
0.5	The Object Explorer	11
0.6	Exiting S-PLUS	11
0.7	Typing Commands	12
1	Looking at Data—Distributions	20
1.1	Displaying Distributions with Graphs	20
1.2	Describing Distributions with Numbers	31
1.3	The Normal Distributions	34
1.4	Editing Graphs	40
1.5	Multiple Plots in Command Line Graphics	42
1.6	Exercises	42
1.7	Solutions	43
2	Looking at Data—Relationships	45
2.1	Scatterplots	45
2.2	Correlation	53
2.3	Least-Squares Regression	55
2.4	Cautions About Correlation and Regression	60
2.5	Exercises	63
2.6	Solutions	64
3	Producing Data	66
3.1	Exercise	69
3.2	Solution	69
4	Probability	70
4.1	Randomness	70
4.2	Exercise	75
4.3	Solution	75
5	Sampling Distributions	76
5.1	Sampling Distributions	79
6	Introduction to Inference	82
6.1	Confidence Intervals and Hypothesis Tests	82
6.2	Power	84
6.3	Exercises	86
6.4	Solutions	87

7	Inference for Distributions	88
7.1	Inference for the Mean of a Population	88
7.2	Comparing Two Means	93
7.3	Exercises	96
7.4	Solutions	97
8	Inference for Proportions	99
8.1	Inference for a Single Proportion	99
8.2	Two Sample Proportions	101
8.3	Exercises	103
8.4	Solutions	104
9	Inference for Two-Way Tables	106
9.1	Two-Way Tables and the Chi-Square Test	106
9.2	Exercises	113
9.3	Solutions	114
10	Inference for Regression	116
10.1	Graphing the Data	116
10.2	Inference About the Model	118
10.3	Inference About Prediction	120
10.4	Checking the Regression Assumptions	123
10.5	More Detail About Simple Linear Regression	129
10.6	Exercise	130
10.7	Solution	131
11	Multiple Regression	135
11.1	Inference for Multiple Regression	135
11.2	Exercise	142
11.3	Solution	143
12	One-Way Analysis of Variance	145
12.1	The Analysis of Variance F Test	145
12.2	Comparing the Means	148
12.3	Exercises	152
12.4	Solutions	153
13	Two-Way Analysis of Variance	155
13.1	Preliminary Data Analysis	155
13.2	Two-Way ANOVA	160
13.3	Exercise	161
13.4	Solution	162
14	Bootstrap Methods and Permutation Tests	165
14.1	One and Two Means (Chapter 7)	166
14.2	Ratios of Means, Standard Deviations, or Other Statistics	183
14.3	Proportions (Chapter 8)	187
14.4	Two-Way Tables and the Chi-Square Test (Chapter 9)	187
14.5	Regression (Chapter 10)	187
14.6	Multiple Regression (Chapter 11)	196
14.7	Analysis of Variance (Chapters 12 and 13)	196
14.8	Correlation (Chapter 2)	196

15 Nonparametric Tests	199
15.1 Wilcoxon Rank Sum Test	199
15.2 The Wilcoxon Signed Rank Test	201
15.3 The Kruskal-Wallis Test	203
15.4 Exercises	204
15.5 Solutions	205
16 Logistic Regression	208
16.1 The Logistic Regression Model	208
16.2 Multiple Logistic Regression	210
16.3 Exercise	212
16.4 Solution	212
Index	213

Chapter 0

Introduction to S-PLUS

This manual is a supplement to Moore and McCabe's *Introduction to the Practice of Statistics* (which we abbreviate **IPS** throughout this manual). It is intended to guide the student through the use of S-PLUS for Windows for the data analyses and statistical procedures detailed in the text.

S-PLUS has both a **Graphical User Interface** as well as a command line interface. We describe both, focusing primarily on the GUI. We assume that you are using S-PLUS 6.1 or later. Chapter 14 requires S-PLUS 6.1 or later; earlier versions of S-PLUS may be used for other chapters, although there are minor differences in the GUI.

0.1 Obtaining S-PLUS and Two Libraries

To use this book you need a copy of S-PLUS, and two supplemental libraries: the IPSdata library containing the data sets, and (for Chapter 14) the Resample library containing additional software. If you are not using a lab with this software already installed, then see

www.whfreeman.com/ipsresample

for up-to-date information on obtaining and installing this software.

The following information is current at the time this manual was written (but the above site may have more current information): there is a free student version of S-PLUS available at

<http://elms03.e-academy.com/splus>,

instructors can obtain evaluation or academic copies of S-PLUS from

www.insightful.com,

and the supplemental libraries can be downloaded from links at

www.whfreeman.com/ipsresample.

0.2 Getting Started

If S-PLUS is installed on your PC, then you normally start it using Windows Start menu: **Start > Program Files > S-PLUS**. Otherwise, there may be an icon for S-PLUS on the Windows desktop. If not, you will need to find out where it is located on your system or network, or install it.

The following screen shot shows the main S-PLUS program window, containing other windows that may be present in a typical session. We will learn more about these windows in the next few sections.

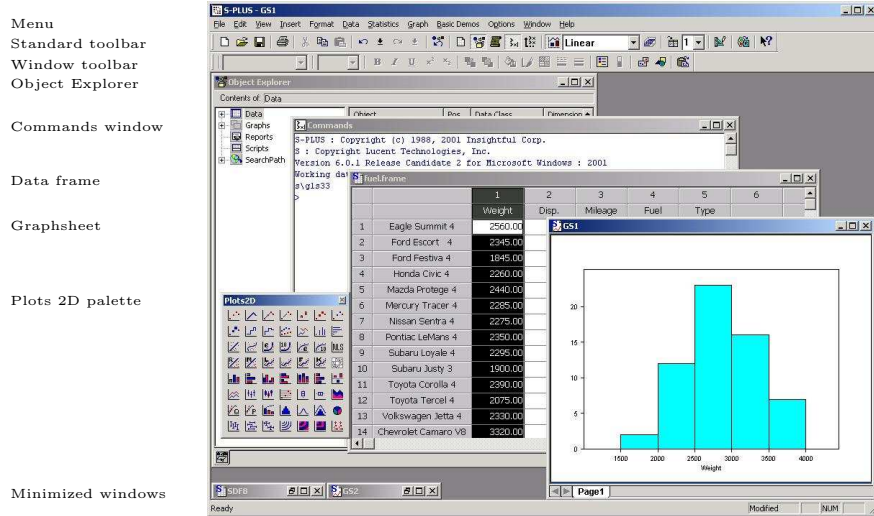


Figure 0.1

0.3 Getting Help

There is extensive online help available in S-PLUS. You can access this information from the menu. For example, to get help on GUI topics, select [Help > Available Help > S-PLUS Help](#).

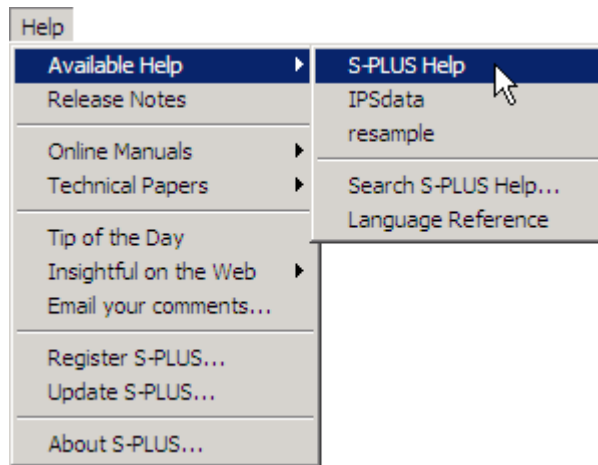


Figure 0.2

This brings up a standard Windows help application:

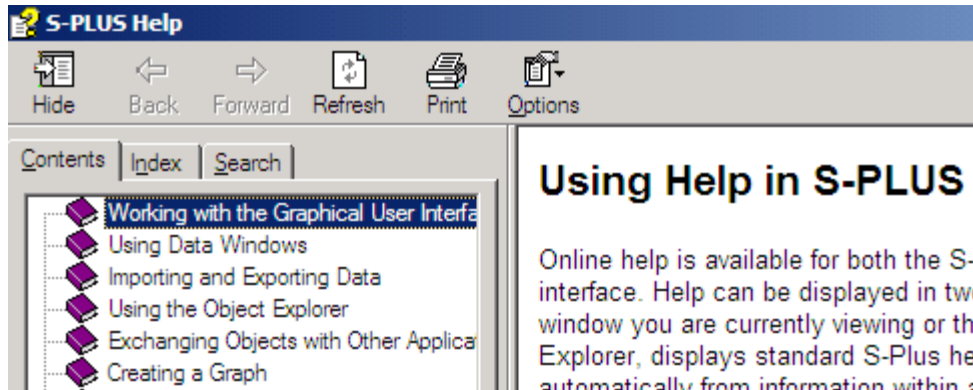


Figure 0.3

The [Contents](#) tab groups help files by topic, the [Index](#) tab has an alphabetical listing of help files, and the [Search](#) tab allows a search for words in a help page.

To get a listing of command line functions by topic, select [Help > Available Help > Language Reference](#). Finally, there are a number of manuals available under [Help > Online Manuals](#), including:

- *Getting Started Guide*. An overview of the S-PLUS GUI and Commands window.
- *User's Guide*. A manual for the S-PLUS GUI.

We recommend that you take a look at the brief tour of S-PLUS given in the *Getting Started* online manual.

0.4 Data in S-PLUS

0.4.1 Opening data in S-PLUS

You can bring up data in a window in S-PLUS using either the [Select Data](#) menu option or the Object Explorer. We describe the first method here and postpone discussion of the Object Explorer until later in this chapter (Section [0.5](#)).

S-PLUS comes supplied with many sample data sets. We begin with a look at [fuel.frame](#) which has information about automobiles from a 1990 issue of *Consumer Reports*.

1. At the menu, select [Data > Select Data...](#)
2. In the [Source](#) box, check the radio button for [Existing Data](#).
3. In the [Existing Data](#) box, for [Name:](#) type [fuel.frame](#).

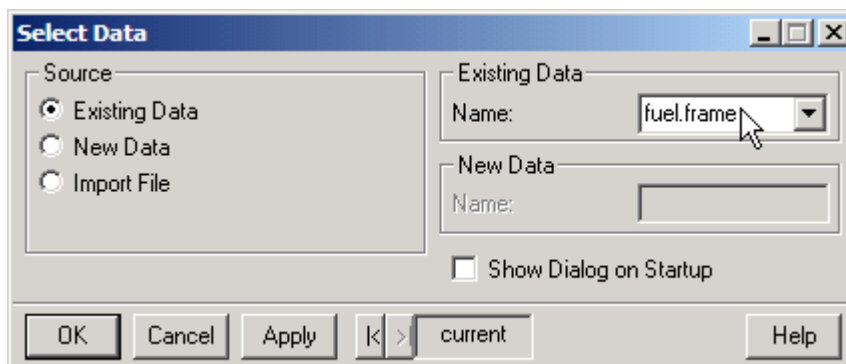


Figure 0.4

- Click [OK](#) or [Apply](#).

		1	2	3	4	5
		Weight	Disp.	Mileage	Fuel	Type
1	Eagle Summit 4	2560.00	97.00	33.00	3.03	Small
2	Ford Escort 4	2345.00	114.00	33.00	3.03	Small
3	Ford Festiva 4	1845.00	81.00	37.00	2.70	Small

Figure 0.5

The data appear in a *data window*, a tool for viewing and editing data.

Remark: The [OK](#) and [Apply](#) buttons in the [Select Data](#) dialog box are similar, except that [Apply](#) keeps the dialog box open after executing the command. This is useful if you wish to try things out, or execute several instances of a command (for example, to open multiple data sets).

0.4.2 Opening data in the IPSdata library

The data sets for *Introduction to the Practice of Statistics, 5th Edition* are included in a special library, the IPSdata library. To use these data, you must first load the library; if it is installed in the standard location, then you can load the library using menus:

- At the menu, select [File > Load Library](#)
- For [Library Name](#), select [IPSdata](#).

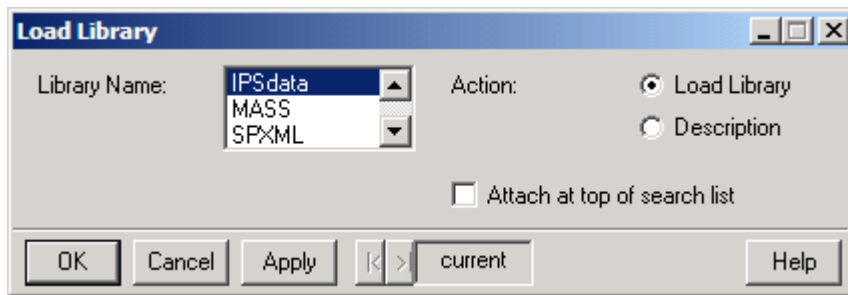


Figure 0.6

- Click [OK](#).

This adds a new menu item [IPSdata](#) on the right of the main S-PLUS menu bar.

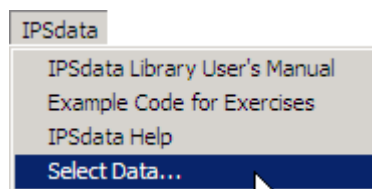


Figure 0.7

If the [IPSdata](#) library is installed in a nonstandard location on your system, you will need to type a command to load it into the Commands window or a Script window; see [Section 0.7](#) for an introduction and [Section 0.7.1](#) for the command.

Once the library is loaded, you can open a data set using the procedure described above in [Section 0.4.1](#). Or you can use menus designed to make it easier to find data in this library more easily:

1. At the menu, select [IPSdata > Select Data...](#) (or equivalently, [Data > Select IPS Data...](#)).
2. Select the type of data set you want, for example, for an exercise in Chapter 1 select either [Chapter 1](#) or [Exercises](#).
3. Select the data set from the drop-down menu, for example, [Exercise1.29](#).

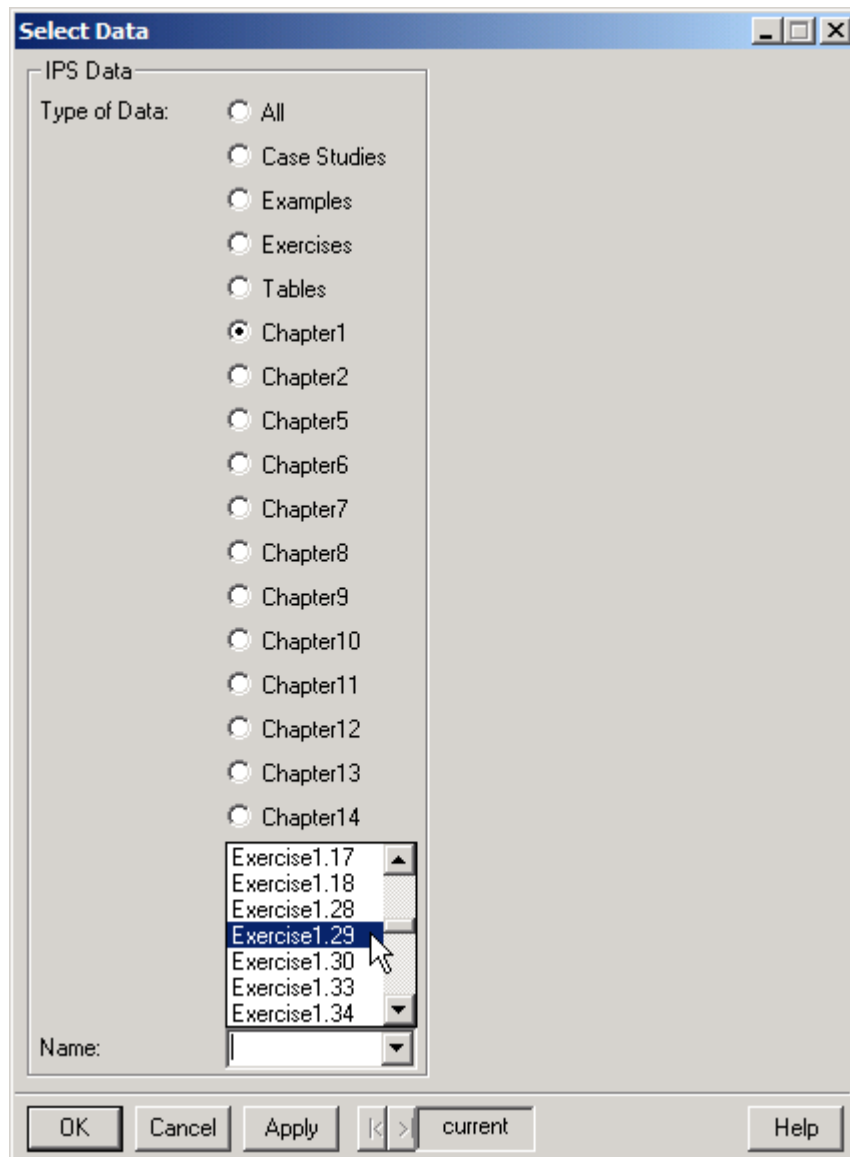


Figure 0.8

0.4.3 Loading the Resample library

Loading the [IPSdata](#) library automatically loads the [resample](#) library, containing bootstrap and permutation test software. If you want to load the library separately, in order to analyze data that is outside the [IPSdata](#) library, the procedure is the same except you must also select the button to [Attach at top of search list](#) so that when S-PLUS searches for certain functions it finds the versions in the [resample](#) library instead of its built-in versions.

0.4.4 Importing data

S-PLUS can read data files stored in many different formats, such as text (.txt), Excel spreadsheets (.xls), as well as S-PLUS's own format (.sdd).

To import data from a file:

1. At the menu, select [File > Import Data > From File...](#) This opens the [Import From File](#) dialog box.
2. Click on the [Browse](#) button and navigate to the location of the data file you wish to load. As an example, we use the file [csdata.txt](#) located in the [IPSdata/data](#) folder.
3. Select the data file.

The [File Format:](#) field will fill automatically.

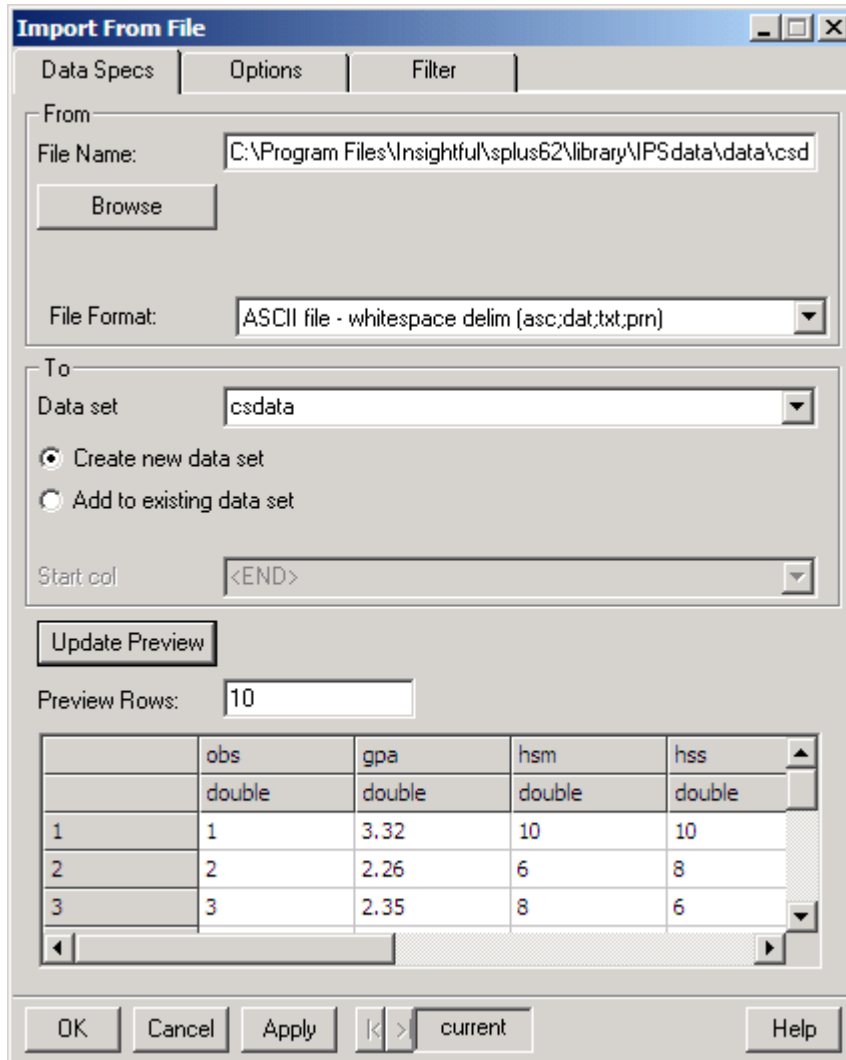


Figure 0.9

By default, the new data set in S-PLUS is given the same name as the original file. If you wish to change the name, type the new name in the [Data Set](#) field in the [To](#) box.

4. Click [Update Preview](#), to be sure that data will be read in the format you expect.
5. Click [OK](#).

0.4.5 Data types

The data that we opened or imported in the previous sections appear in a data window; the data window is a mechanism for viewing and manipulating the values that make up the data.

Each data set in the `IPSdata` library is stored in a *data frame*, a type of matrix in which each column corresponds to a variable and each row to an observation. The variables may be numeric, character, logical (TRUE or FALSE), factor, or other less-common types. For example, in `fuel.frame`, the first four columns are numeric and the fifth is a *factor* variable.

Factor variables contain categorical data such as `gender`; the groups (for example, male and female) are called the *levels* of the factor variable. In `fuel.frame`, the factor variable `Type` has six levels: `Small`, `Compact`, `Medium`, `Large`, `Sporty`, and `Van`.

To check the type of column, hover your cursor over the column number to see a data tip; **double** indicates numerical data stored in double-precision.

4	5	
Fuel	Type	factor
3.03	Small	
3.03	Small	
2.70	Small	

Figure 0.10

An entry of **NA** indicates a *missing value*.

0.4.6 Creating a data set

In addition to importing already prepared data, you can use S-PLUS to create your own data sets. To create a new data frame:

1. Click on the **New Data Set** button on the Standard toolbar.

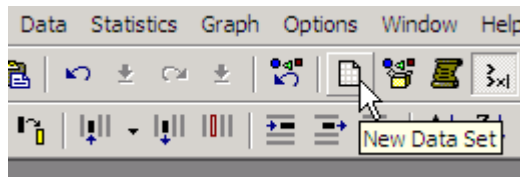


Figure 0.11

A blank window resembling a spreadsheet opens. You can type data directly into this window.

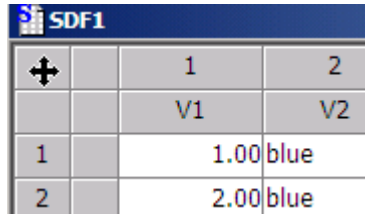
2. Type the numbers **1, 2, 3, 4, 5** into the first five cells of column 1.
Pressing the **Enter** key moves the active cell in the same direction as the last arrow key.
3. In the next column, enter **blue, blue, red, red, green**.
When entering character data, the column is stored as a factor variable.

SDF1		1	2	
		V1	V2	factor
1		1.00	blue	
2		2.00	blue	
3		3.00	green	
4		4.00	green	
5		5.00	red	

Figure 0.12

0.4.7 Making changes to data

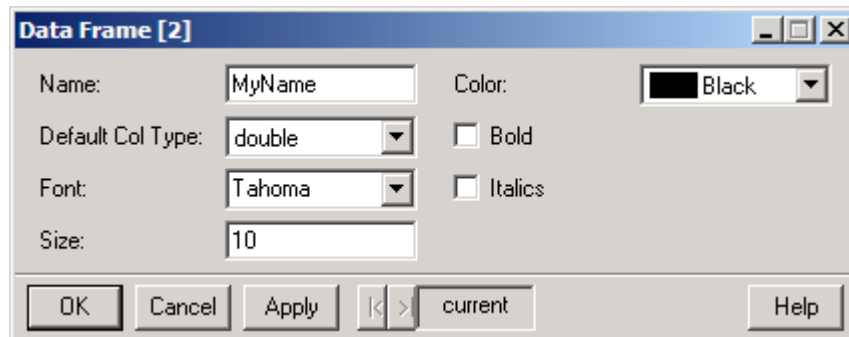
You may need to make changes to data. These changes might involve the entire data set (for example, rename the data set), a column or columns (for example, rename a column or change the number of significant digits displayed), or a single cell (for example, change the value of an entry).



	1	2
	V1	V2
1	1.00	blue
2	2.00	blue

Figure 0.13

To change the name of the whole data frame, place the cursor in the top left cell, right-click and select [Properties...](#) from the shortcut menu. In the [Data Frame](#) dialog box, type a new name in the [Name:](#) field. Click [OK](#).



Data Frame [2]

Name: Color:

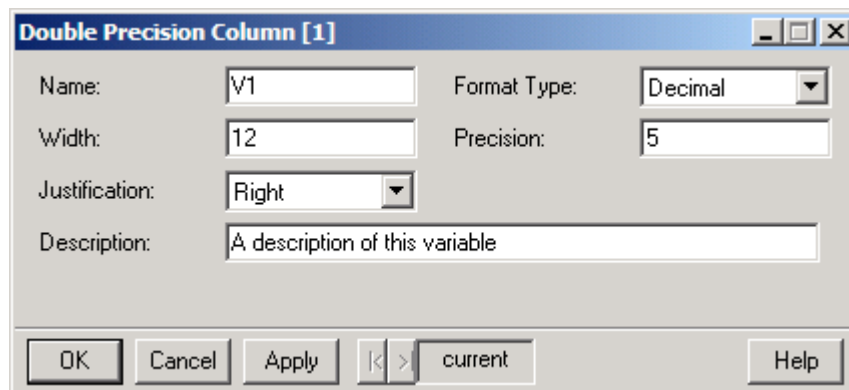
Default Col Type: Bold

Font: Italics

Size:

Figure 0.14

To make changes to a column, right-click on the *column header* (column number or name) and select [Properties...](#) from the shortcut menu. In the [Double Precision Column](#) dialog box, you can, for example, change the name of the column, add a description to be used for labeling graphs, and so on, or change the number of significant digits ([Precision:](#)) that are displayed.



Double Precision Column [1]

Name: Format Type:

Width: Precision:

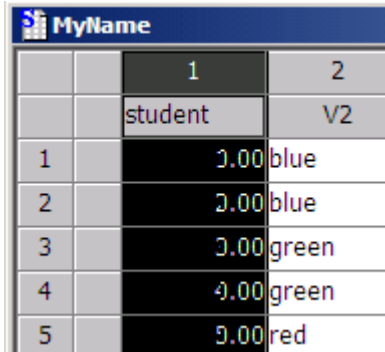
Justification:

Description:

Figure 0.15

Remark: In S-PLUS, names can be any length and can consist of letters, digits, or the period symbol. Names must begin with a letter and no spaces are allowed. Thus, `weight`, `Weight1994`, `weight.94` are all legitimate names while `94Weight`, `Weight 94`, or `weight-94` are not. S-PLUS is case-sensitive.

If the only change you wish to make is to the column name, then you can also do this by double-clicking on the current column name and then typing in the new name.



		1	2
		student	V2
1		3.00	blue
2		3.00	blue
3		3.00	green
4		4.00	green
5		5.00	red

Figure 0.16

In addition to making changes from a dialog box, many common operations can be done quickly via the toolbar, including changing precision, adding or removing columns, and sorting. Select the column of interest from the data window, then click on the relevant button on the toolbar.

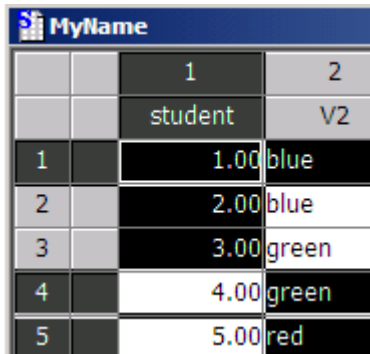


Figure 0.17

For a complete description of the data set buttons, see the **Data Set Toolbar** help file.

0.4.8 Selecting columns or rows

For many analyses, you may need to select more than one column or row. Click on the first column or *row header* (either number or name) in your selection. While holding down the CTRL key, click on the next column or row header. Continue until all desired columns and/or rows have been selected.



		1	2
		student	V2
1		1.00	blue
2		2.00	blue
3		3.00	green
4		4.00	green
5		5.00	red

Figure 0.18

To select several contiguous columns (or rows), click on the header of the first column (or row), hold down the SHIFT key, and then click on the last column or row.

0.5 The Object Explorer

S-PLUS is an object-oriented environment; everything in S-PLUS is an object—data sets, functions, graphs. The Object Explorer is a tool for organizing and managing these objects. New objects that are imported or created are saved automatically to your *working database*, typically located in

`C:/Program Files/Insightful/splus62/users/YourName/.Data`

The Object Explorer displays the objects in this database.

To open the Object Explorer, click on the [Object Explorer](#) button on the Standard toolbar.

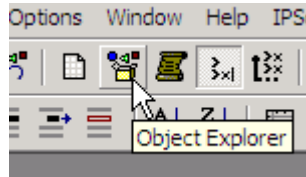


Figure 0.19

The Object Explorer consists of two panes: the left pane consists of folders that hold shortcuts to S-PLUS objects (data, graphsheets, and so on), while the right pane consists of properties of the objects selected in the left pane. For example, if the **Data** folder is selected in the left pane, then all objects in the working database are displayed in the right pane. If a specific data frame in the **Data** folder is selected in the left pane, then all columns of this data frame are displayed in the right pane. Thus, for example, you can make your variable selection for graphs or statistical analyses from the Object Explorer rather than from the data window.

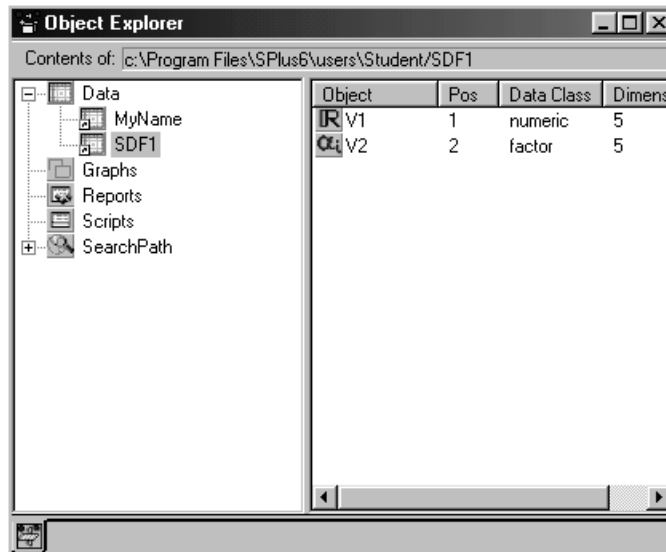


Figure 0.20

To open a data frame into a data window, you can double-click on its icon in the Object Explorer. To delete an object from the working database, right-click on its icon in the Object Explorer and select [Delete](#) from the shortcut menu.

0.6 Exiting S-PLUS

To quit S-PLUS, select [File > Exit](#) from the menu. You will be presented with the [Save Database Changes](#) dialog box.

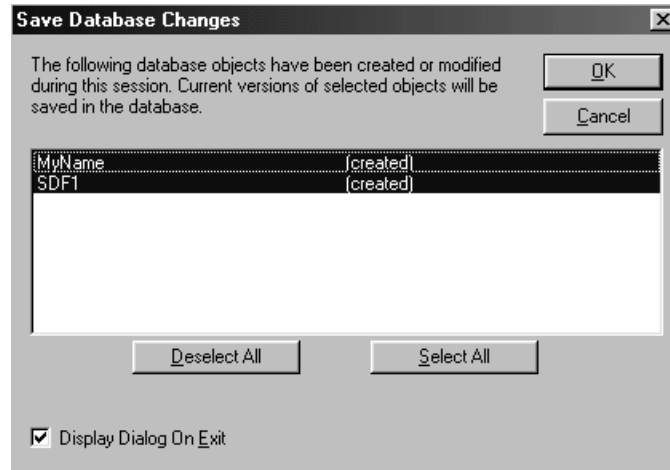


Figure 0.21

By default, S-PLUS saves all your newly created or modified data sets to the working database. If, however, you decide that you do not want to keep any of the current versions (for example, you wish to revert to the version of a data set that you had at the beginning of the session), you may de-select the data set or sets, and then click **OK**.

0.7 Typing Commands

Although all of the graphs and analyses in IPS can be performed through the GUI, you may prefer typing commands rather than using the point-and-click interface.

If you have never used S-PLUS before, we recommend that you begin with the point-and-click interface. You may come back later and try typing commands.

Commands may be given in two different ways, using the *Commands Window* or a *Script Window*. In the Commands window you type commands and run them immediately, while in a Script Window you may write a script with one or more commands, then run the commands when desired. Contents of a Script Window may be saved in a file. We begin with the Commands Window, then describe Script Windows.

0.7.1 The Commands Window

To open the Commands Window, use the menu:

1. At the menu, select **Window > Commands Window**.

or click on the **Commands Window** button , located on the Standard toolbar.

In the Commands window, you should see a *prompt*, the “greater than” sign **>**. Type an expression at this prompt, then press the **ENTER** key to evaluate the expression. For example:

```
> 3*(9-4) <ENTER>
[1] 15
```

If you press **ENTER** before typing a complete expression, then S-PLUS prints the *continuation prompt*, a “+” sign. At this prompt, finish the expression and then press **ENTER**.

```
> 3*(9-
+ 4) <ENTER>
[1] 15
```

From now on, we will not indicate the **ENTER** key.

In general, S-PLUS ignores spaces. We recommend using spaces to make what you write more readable.

```
> 3 * (9 - 4)
[1] 15
```

The pound symbol `#` is used for comments—anything typed after `#` is not evaluated.

```
> (5 <= 8) # a logical expression; either TRUE (T) or FALSE (F)
[1] T
```

Missing values are denoted by `NA`.

```
> log(-1)
[1] NA
```

To access help files, use either `help` or `?`; for example

```
> help() # bring up the help system, help on help
> help(mean) # help for the mean command
> ?mean # help for the mean command (equivalent to previous)
```

There are several useful keys for editing commands. The Up (\uparrow) and Down (\downarrow) keys scroll backward or forward through previously typed commands; this is particularly useful for fixing errors in long commands without retyping everything. The Home and End keys bring the cursor to the beginning or end of the current line.

Load the `IPSdata` library

To load the `IPSdata` library, give the command

```
> library(IPSdata)
```

However, if the library is not loaded in the standard location (inside the `library` folder inside the S-PLUS folder), then you also need to specify where the library can be found; for example, if the library is located at `c:/Stat101/IPSdata` then the command would be

```
> library(IPSdata, lib.loc = "c:/Stat101")
```

Note—a backslash is used to include special characters in quoted strings (for example, `"\t"` gives a tab), so if you want a real backslash you need to type it twice, for example,

```
> library(IPSdata, lib.loc = "c:\\Stat101")
```

Load the `resample` library

Loading the `IPSdata` library automatically loads the `resample` library, which contains bootstrap and permutation test software.

You can also load the `resample` library separately, using almost the same procedure; you should also load the library “first”, so that when S-PLUS searches for certain functions it finds the versions in the `resample` library instead of its built-in versions.

```
> library(resample, first = T)
```

0.7.2 Script Windows

The second way to give commands is to use a Script Window. This has the advantage of making it easier to save your work, and to rerun all or parts of your previous work.

To open a Script Window, use the menu:

1. At the menu, select `File > New`.
2. Select `Script File`.

3. Click **OK**.

In a Script Window, you type the same commands as in the Commands Window (without prompts), then execute them. For example,

```
3 * (9 - 4)
(5 <= 8) # a logical expression; either TRUE (T) or FALSE (F)
log(-1)
help()
?mean # help for the mean command
```

To execute one or more lines, select them using the mouse (click at the beginning of a line, or drag across multiple lines), and hit **F10** or click the black triangle (this only appears when a script window is active).



You may use multiple Script Windows. You may save the contents of a script window to a file, when the Script Window is active (usually, when it is not behind another window).

1. At the menu, select **File > Save As...**
2. Type the name of the file.
3. Click **OK**.

In the following sections we give commands as if you are using the Commands Window; however you can give the same commands using a Script Window.

0.7.3 Assigning names to objects

S-PLUS evaluates each expression you type and outputs the result to the screen. Occasionally, you may wish to save the results of a calculation for later use. You do this by *assigning* (or *saving*) the value to a name. You can then obtain the value later by typing the name. For example:

```
> IQ = 120
> IQ
[1] 120
```

In this manual we use **=** to show assignment, but many of the other S-PLUS manuals use **<-**, a “less than” symbol followed immediately (that is, no space) by a “hyphen”; think of this as an arrow, with the value on the right being put into the name on the left.

When assigning names, it is important *not* to use a name already used by S-PLUS. For example, **c**, **C**, **T**, **F**, **cat**, and **date** are reserved by S-PLUS. If you are not sure if a name is already in use, type the potential name and check the response.

```
> dog
Problem: Object "dog" not found
Use traceback() to see the call stack
> cat
function(..., file="", sep=" ", fill=F, labels=NULL, append=F)
.Internal(cat(..., file, sep, fill, labels, append), "S_cat", T, 0)
```

We see that **dog** is safe to use, but not **cat**.

To see a listing of the objects in the working database, use the **objects** command.

```
> objects()
[1] ".Last.value" "IQ"
```

To remove an object from the working database, use the `rm` command.

```
> IQ
[1] 120
> rm(IQ)
> IQ
Problem: Object "IQ" not found
Use traceback() to see the call stack
```

0.7.4 Vectors

The most basic data object in S-PLUS is the vector, a sequence of values. A single number is a vector of length one. We give examples of some of the ways you can create a vector.

To create a sequence of numbers differing by 1, use the `:` command.

```
> 4:10
[1] 4 5 6 7 8 9 10
> 4:-5
[1] 4 3 2 1 0 -1 -2 -3 -4 -5
```

To increment a sequence of numbers by values other than 1, use `seq`.

```
> seq(0, 20, by=2)
[1] 0 2 4 6 8 10 12 14 16 18 20
> seq(0, 20, length=25)
[1] 0.0000000 0.8333333 1.6666667 2.5000000
[5] 3.3333333 4.1666667 5.0000000 5.8333333
[9] 6.6666667 7.5000000 8.3333333 9.1666667
[13] 10.0000000 10.8333333 11.6666667 12.5000000
[17] 13.3333333 14.1666667 15.0000000 15.8333333
[21] 16.6666667 17.5000000 18.3333333 19.1666667
[25] 20.0000000
```

The bracketed integers in the output are indices for the first value on a line. For example, the 21st value in the last example is 16.6666667.

To create a vector of values that has no particular pattern, or a vector of character or boolean values, use the `c` command (`c` for concatenate or combine).

```
> c(3, -1, 0, 5, 2)
[1] 3 -1 0 5 2
> c("A", "A", "B", "B")
[1] "A" "A" "B" "B"
```

The `rep` command is a useful function for creating vectors with values that repeat in a regular pattern.

```
> rep(c("A", "B"), 5)
[1] "A" "B" "A" "B" "A" "B" "A" "B" "A" "B"
> rep(c("A", "B"), c(5, 3))
[1] "A" "A" "A" "A" "A" "B" "B" "B"
> rep(c("A", "B"), each = 5)
[1] "A" "A" "A" "A" "A" "B" "B" "B" "B" "B"
```

Here we show three different ways of calling `rep`. For more information, see `help(rep)`.

0.7.5 Interrupting output

You may interrupt any printout using the `ESC` key

```
> 1:1000000
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
 [17] 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
... <ESC>
Problem in print.atomic(x, quote): User interrupt requested.
```

0.7.6 Working with vectors

Basic arithmetic operations work component-wise on vectors.

```
> c(2, 4, 6) * c(10, 20, 30) # multiply 2 vectors
 [1] 20 80 180
> c(2, 4, 6)/2                # divide by 2
 [1] 1 2 3
> y = c(8, 2, 9, 7, 5, 2, -3, -4, 0, 2, 1)
> y
 [1] 8 2 9 7 5 2 -3 -4 0 2 1
> y + 2
 [1] 10 4 11 9 7 4 -1 -2 2 4 3
> y * 5
 [1] 40 10 45 35 25 10 -15 -20 0 10 5
```

In many cases, you need to work with only a portion of a vector. The basic syntax for subscripting a vector is `vector[indices]`. The indices may be a vector of integers, or a logical vector.

```
> y
 [1] 8 2 9 7 5 2 -3 -4 0 2 1
> y[3] # extract the 3rd value
 [1] 9
> y[c(3, 8)] # extract the 3rd and 8th values
 [1] 9 -4
> y[-c(3, 8)] # extract all but the 3rd and 8th values
 [1] 8 2 7 5 2 -3 0 2 1
> y[length(y):1] # reverse the order
 [1] 1 2 0 -4 -3 2 5 7 9 2 8
> y < 2 # test whether a value is less than 2
 [1] F F F F F F T T T F T
> y[y < 2] # extract those values of y that are less than 2
 [1] -3 -4 0 1
> y[0 < y & y < 5] # extract those values of y between 0 and 5
 [1] 2 2 2 1
```

To replace the value of one entry with another, use the assignment operator.

```
> y[3]
 [1] 9
> y[3] = 12 # replace the 3rd entry of y with 12
> y[3]
 [1] 12
> y
 [1] 8 2 12 7 5 2 -3 -4 0 2 1
```

Symbol	Function
==	Equal to
!=	Not equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
	Element-wise or
&	Element-wise and

Table 0.1: Logical and Comparison Operators

A vector can store only one kind of information—for example, numerical, character, or logical, but not all three. The technical term is *mode*. If you try to store data of different modes together in a single vector, values are changed as needed to a common mode, whichever mode is most flexible—character is most flexible, then numeric, then logical.

```
> c(T, F, 3.35)
[1] 1.00 0.00 3.35
> c(T, F, 3.35, "Seattle")
[1] "TRUE" "FALSE" "3.35" "Seattle"
```

0.7.7 Data frames

Vectors are relatively simple data structures that can only contain values of one mode. However, most data is more complex and may consist of numeric information, as well as categorical information such as gender or country of birth.

A *data frame* is a matrix whose columns are vectors. Different columns may be of different modes. The built-in data set `fuel.frame` is a data frame with four columns of numeric values and one factor column.

```
> fuel.frame
      Weight Disp. Mileage   Fuel   Type
Eagle Summit 4  2560   97    33 3.030303 Small
Ford Escort 4  2345  114    33 3.030303 Small
Ford Festiva 4  1845   81    37 2.702703 Small
Honda Civic 4  2260   91    32 3.125000 Small
...

```

There are several methods for accessing the columns of a data frame.

- Using the `$` sign

The basic syntax is `dataframe$column.name`

```
> fuel.frame$Weight
[1] 2560 2345 1845 2260 2440 2285 2275 2350 2295 1900
[11] 2390 2075 2330 3320 2885 3310 2695 2170 2710 2775
...

```

- Using brackets `[,]`

The basic syntax is `dataframe[row indices, column indices]`.

```

> fuel.frame[4, 3] # 4th row, 3rd column
[1] 32
> fuel.frame[1:5, c(1,3)] # rows 1 through 5, columns 1 and 3.
      Weight Mileage
Eagle Summit 4  2560    33
Ford Escort 4  2345    33
Ford Festiva 4  1845    37
Honda Civic 4  2260    32
Mazda Protege 4 2440    32

```

To obtain all the rows, leave the row specification blank. The following are equivalent to `fuel.frame$Weight`:

```

> fuel.frame[, 1]
> fuel.frame[, "Weight"]

```

Similarly, for all columns, leave the row specification blank.

```

> fuel.frame[1:3, ]

```

- The `attach` command

The columns of a data frame are not accessible directly from the command line.

```

> Weight
Problem: Object "Weight" not found
Use traceback() to see the call stack

```

We can access them using `$` or subscripting,

```

> fuel.frame$Weight

```

but if we plan to do a substantial amount of work with a single data frame, it is nice to access the columns directly. We can *attach* a data frame to the S-PLUS *search path*, a list of databases that S-PLUS refers to when a function or data set is called.

```

> attach(fuel.frame)
> Weight
Eagle Summit 4   Ford Escort 4 Ford Festiva 4
          2560           2345           1845

Honda Civic 4 Mazda Protege 4 Mercury Tracer 4
          2260           2440           2285
...

```

When done working with the data frame, we remove it from the search list:

```

> detach("fuel.frame")
> Weight
Problem: Object "Weight" not found
Use traceback() to see the call stack

```

The `detach` command removes `fuel.frame` from the S-PLUS search path. Notice that this command requires double quotes around the name of the data frame whereas the `attach` command does not.

Incidentally, when you load the `IPSDdata` library, both it and the `resample` library are attached to the search path. You can see the current search path using

```

> search()

```

0.7.8 Creating a data frame

Use the `data.frame` command to create a data frame. The arguments to the function are the columns.

```
> x = 1:5
> y = c("blue", "blue", "red", "red", "green")
> df = data.frame(x, y)
> df
  x    y
1 1 blue
2 2 blue
3 3  red
4 4  red
5 5 green
```

0.7.9 Functions

In S-PLUS, function calls are always followed by a pair of parentheses, which may or may not enclose arguments.

```
> objects()          # function call with no argument
> names(fuel.frame) # function call with one argument
[1] "Weight" "Disp." "Mileage" "Fuel"  "Type"
> length(fuel.frame$Weight)
[1] 60
> c(3, 4, 9)        # function call with 3 arguments
[1] 3 4 9
```

To quickly check options for a function, use the `args` command:

```
> args(mean)
function(x, trim=0, na.rm=F)
NULL
```

(You may ignore that `NULL`.)

You can abbreviate the arguments to command line functions, provided the abbreviation is unambiguous:

```
> mean(fuel.frame$Weight, trim=.1) # 10% trimmed mean (10% each side)
[1] 2895.729
> mean(fuel.frame$Weight, tr=.1)
[1] 2895.729
```

There is no other argument which begins “tr”, so “trim” may be abbreviated. In contrast, when calling the `plot` function, there are two optional arguments `xlim` and `xlabel`, so the abbreviation `xl` would be ambiguous.

To exit S-PLUS from the command line, type

```
> q()
```


Chapter 1

Looking at Data—Distributions

In this chapter, we perform exploratory data analysis—creating graphs and looking at basic numeric summaries—to get an overview of the data.

1.1 Displaying Distributions with Graphs

1.1.1 Categorical variables: Bar graphs and pie charts

Categorical data places an individual into one of several groups or categories. In S-PLUS, a categorical variable is called a factor and the groups are called levels (*see also* Section 0.4.5).

Example 1.1 Sizes of cars

The data set `cu.summary` contains information from a 1990 issue of *Consumer Reports* on the properties of several cars.

1. At the menu, select `Data > Select Data...`
2. In the `Select Data` dialog box, check the radio button next to `Existing Data`.
3. For `Name:` type `cu.summary`.

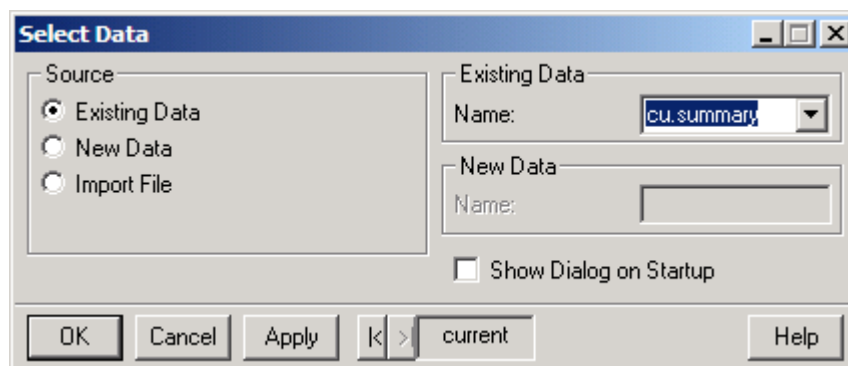


Figure 1.1

4. Click `OK`.

Verify that the data tip for `Type` shows it is indeed a factor variable.

Next, we create a bar or pie chart of `Type`.

5. Select the **Type** column.
6. Bring up the **Plots 2D** palette by clicking on the **Plots 2D** button on the Standard toolbar.

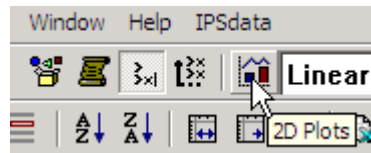


Figure 1.2

7. For a bar graph, click on the **Bar Zero Base** button.

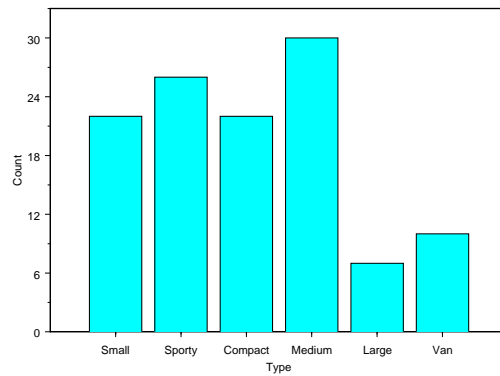
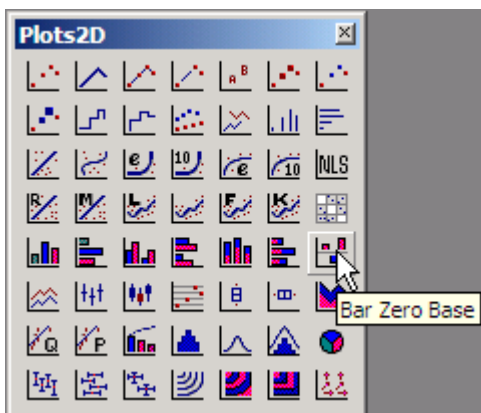


Figure 1.3

8. For a pie chart, click on the **Pie Chart** button.

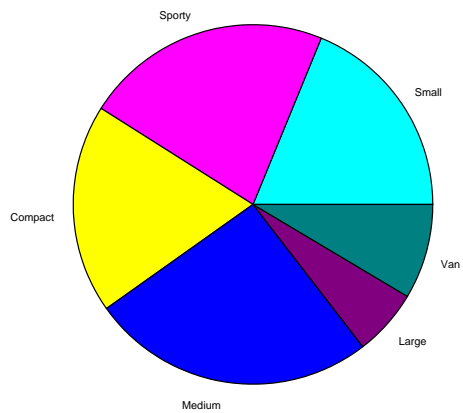
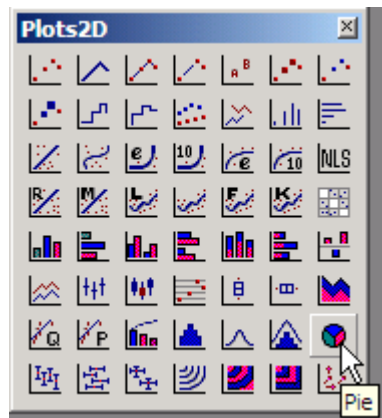


Figure 1.4

End of Example 1.1

Remarks:

- To add grid lines to the bar graph, right-click on the y axis and select [Grids/Ticks...](#) from the shortcut menu. In the **Y Axis** dialog box under **Major Grids**, change the **State:** to **On** and click on **OK**.

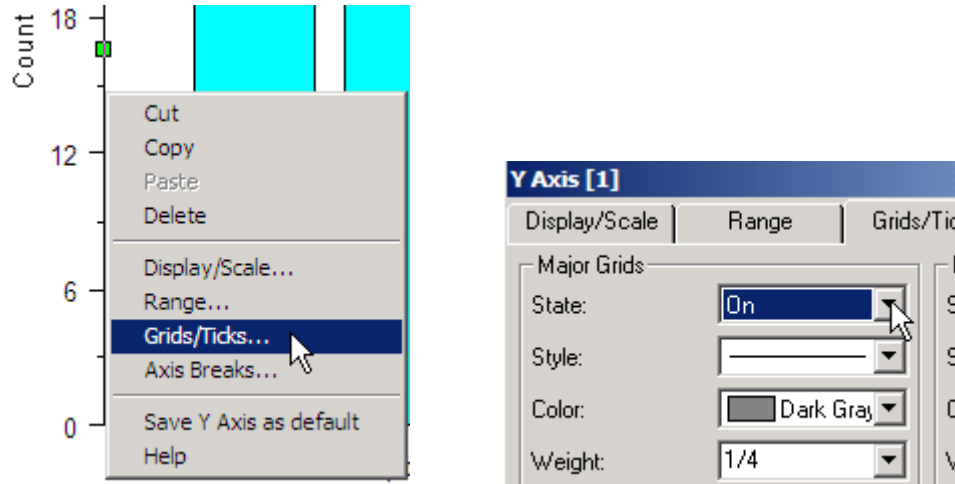


Figure 1.5

- To make changes to the bars or slices (such as colors, labels, and so forth), right-click on a bar and select [By Bar...](#) from the shortcut menu, or right-click on a pie slice and select [By Slice...](#) from the shortcut menu. This opens the appropriate dialog box.

In the previous example, we worked with a factor variable (**Type**) which contained all the information in raw form—we have a listing of each observation and the level to which it belongs. In many instances, we instead have a summary of a factor variable, a list of the levels and the number of observations (counts) for each level. S-PLUS can handle these summaries too.

Example 1.2 Education

The textbook (IPS) provides data on the education level of 30-something young adults. We do not have the information on each of the over 35 million people in the data set, but rather a summary of the number of observations for each level of the variable **Education**.


Education	Count (millions)	Percent
Less than high school	4.6	11.8
High school graduate	11.6	30.6
Some college	7.4	19.5
Associate degree	3.3	8.8
Bachelor's degree	8.6	22.7
Advanced degree	2.5	6.6

To analyze this information in S-PLUS,


1. Create a new data set by clicking on the [New Data Set](#) button  and enter the data above.

SDF1				
		1	2	3
		Education	Count	Percent
1		Less than high school	4.6	11.80
2		High school graduate	11.6	30.60
3		Some college	7.4	19.50
4		Associate degree	3.3	8.80
5		Bachelor's degree	8.6	22.70
6		Advanced degree	2.5	6.60

Figure 1.6

2. Select **Education**, then CTRL-click on the **Count** column.
3. Click on the **Bar Zero Base** button  on the **Plots 2D** palette.


For a pie chart,

4. Select the **Percent** column, then CTRL-click on the **Education** column.
5. Click on the **Pie Chart** button  on the **Plots 2D** palette.

The two plots should look similar to the plots in IPS. If either one looks funny, right-click on a bar or a pie slice and select **Data to Plot...** from the shortcut menu. Check that the correct information is listed in the appropriate fields: for the bar graph, **x Columns:** should be the categorical variable **Education** and **y Columns:** should be **Counts** or **Percents**; for the pie chart, **x Columns:** should be **Counts** or **Percents** and the **y Columns:** should be **Education**.

End of Example 1.2

Remark:

- The dotchart is another informative plot.
 1. Select the **Percent** column, then CTRL-click on **Education**.
 2. Click on the **Dot** (dotplot) button  on the **Plots 2D** palette.
 3. In the resulting graph, right-click on the x axis and select **Range...** Enter **100** in the **Axis Maximum:** and **Last Tick:** fields, enter **0** in the **Axis Minimum:** field, and click on **OK**.

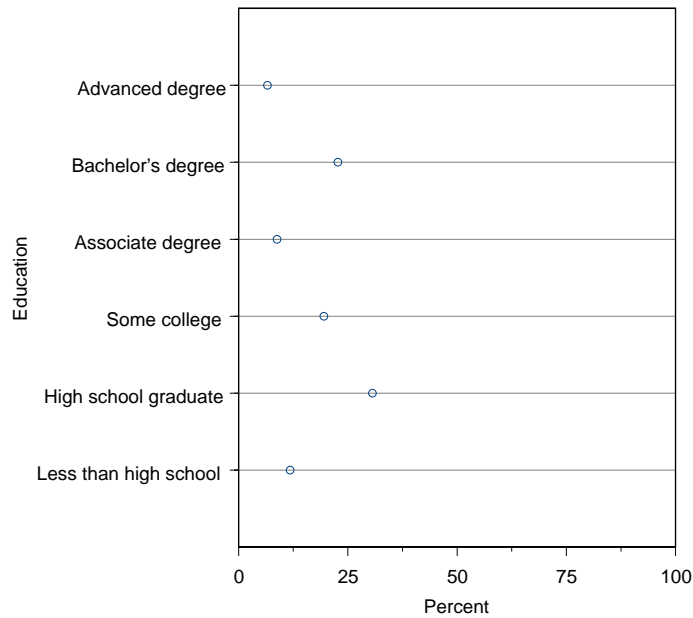



Figure 1.7

Like the bar chart, you can compare the sizes of each level by noting how far each symbol is from an axis (in this case, the left vertical axis); since the horizontal axis is scaled to 100, you can compare relative sizes like the pie chart.

- A Pareto chart  is a bar graph with the bars sorted in decreasing height and a line graph showing the cumulative percent in the bars. Since the final cumulative percent is always 100%, we can easily see what proportion each bar is of the whole and still easily compare heights of bars next to each other.

In addition to graphical representations of factor variables, S-PLUS can calculate numeric summaries for factor variables.

Example 1.3 Counts and percentages

We will continue to use the `cu.summary` data.

1. At the menu, select **Statistics > Data Summaries > Crosstabulations...**
2. For **Data Set:** select `cu.summary`, and for **Variables:** select `Type`.

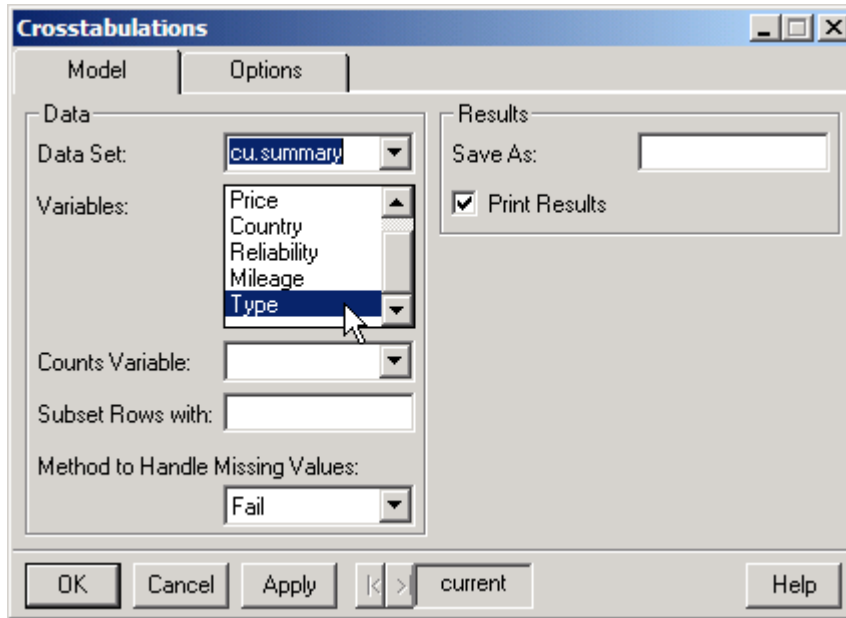


Figure 1.8

3. Click [OK](#).

```

*** Crosstabulations ***
Call:
crosstabs(formula = ~ Type, data = cu.summary,
          na.action = na.fail, drop.unused.levels = T)
117 cases in table
+-----+
|N      |
|N/Total|
+-----+
Type   |
      |      |RowTotl|
+-----+-----+
Compact|22    |22    |
      |0.19  |0.19  |
+-----+-----+
      |
      |
+-----+-----+
Van    |10    |10    |
      |0.085 |0.085 |
+-----+-----+
ColTotl|117   |117   |
      |1     |      |
+-----+-----+

```

This gives the counts and percentages for each car type and the percentages. For example, there are ten vans representing 8.5% of the total.

End of Example 1.3

Command Line Notes

The `table` command returns the counts of a factor variable and the functions `barplot`, `pie`, and `dotchart` create plots of a factor variable. The table of counts and proportions can be created with `crosstabs`. When there is one line per subject, the first argument to `crosstabs` is a formula of the form `~factor`.

```
> attach(cu.summary)
> table(Type)
  Compact Large Medium Small Sporty Van
      22     7     30     22     26    10
> barplot(table(Type), names = levels(Type))
> pie(table(Type), names = levels(Type))
> dotchart(table(Type))
> crosstabs( ~ Type)
> detach()
```

When instead there is a variable containing counts, the first argument is of the form `counts~factor`. Here we multiply by 10^6 to convert from millions to ones:


```
> SDF1 = data.frame(
+   Education = c("Less than high school",
+   "High school graduate", "Some college", "Associate degree",
+   "Bachelor's degree", "Advanced degree"),
+   Count = c(4.6, 11.6, 7.4, 3.3, 8.6, 2.5),
+   Percent = c(11.8, 30.6, 19.5, 8.8, 22.7, 6.6))
> crosstabs(10^6 * Count ~ Education, data = SDF1)
```

1.1.2 Quantitative variables: Histograms

Histograms provide a graphical view of quantitative or continuous data.

Example 1.4 How to make a histogram

Table 1.3 of IPS gives IQ test scores for 60 randomly chosen fifth-grade students. To make a histogram of these data:

1. Load the `IPSdata` library, from menu `File > Load Library...` Instructions for obtaining the library are in Section 0.1.
2. Open the data using `IPSdata > Select Data...`, and select `Table1.3`.
3. Select the `iq` column
4. Click on the `Histogram` button  on the `Plots 2D` palette.

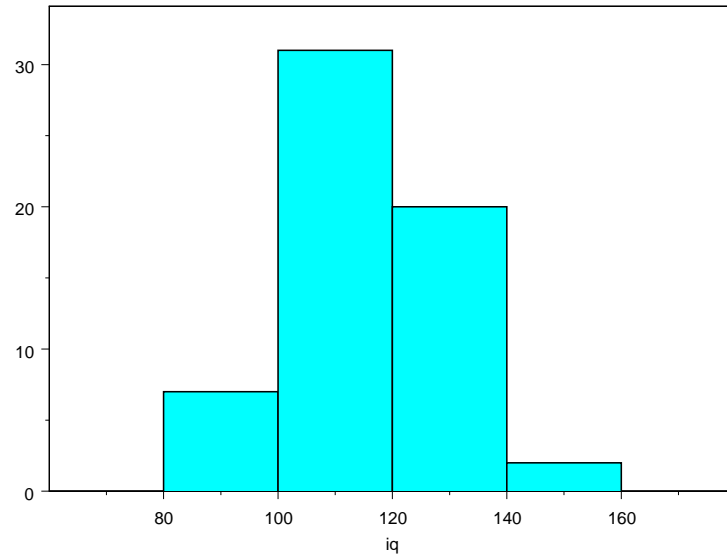


Figure 1.9

Histograms can look very different depending on just where you place the edges of the bins. To change the number or location of bins,

5. Right-click on the histogram and choose [Options...](#) from the shortcut menu.
6. In the [Histogram/Density](#) dialog, for [Lower Bound:](#) enter [80](#), and for [Interval Width](#) enter [10](#).

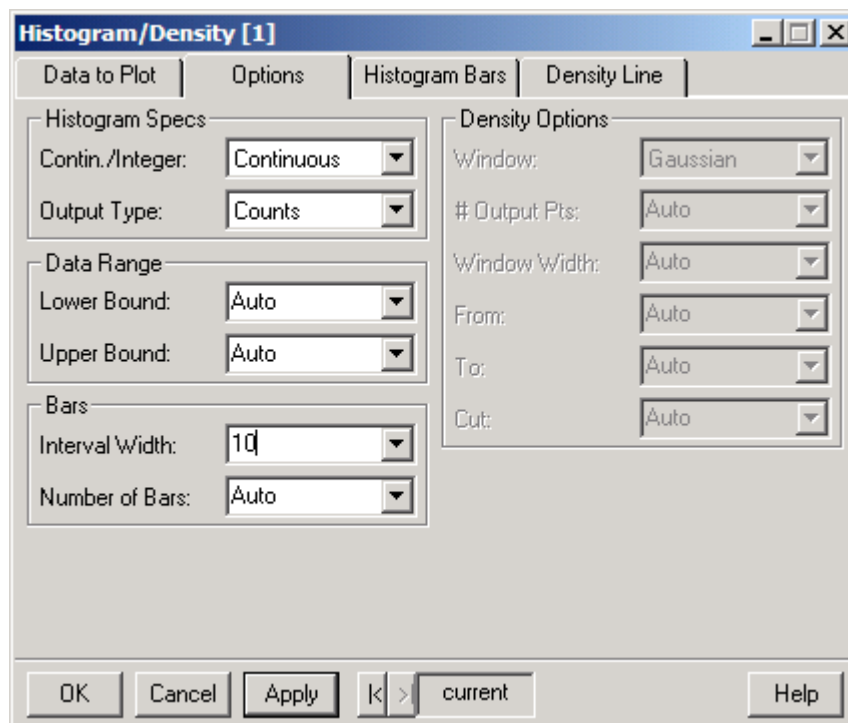


Figure 1.10

7. Click [Apply](#).
8. Try some other bar widths, or change other parameters; click [Apply](#) to see each.
9. To add a density curve, for [Output Type](#): select [Density](#), select the [Density Line](#) tab, and select the checkbox for [Draw Line](#).
10. Click [OK](#) when you are done.

Here are histograms with bars of width 10 and 5, with a lower bound of 80:

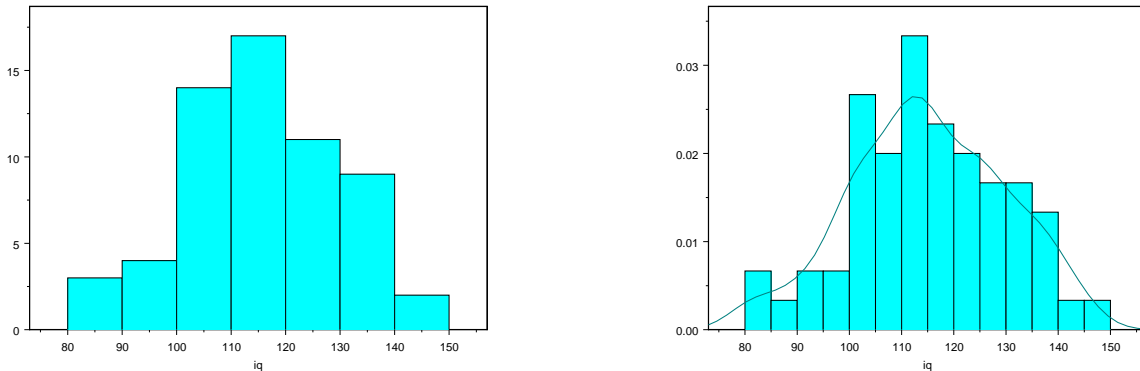




Figure 1.11

End of Example 1.4

Remarks:

- You can add a title and explanatory labels along the axes using the [Insert > Titles > Axis](#) or [Insert > Text](#) menus.
- You can change other aspects of the histogram (color, shading, and so on) by returning to the [Histogram/Density](#) dialog box (right-click on the histogram and select [Options](#) from the shortcut menu).
- A density plot  is an alternative to a histogram, generally superior because it is not dependent on bar edges. However, you may work with people not familiar with them. A good compromise is to combine a histogram and density curve . An example is shown above.

Command Line Notes

The `hist` function creates histograms and the `stem` function creates stem and leaf plots.

```
> library(IPSdata)
> names(Table1.3)
[1] "iq"
> hist(Table1.3$iq)
> hist(Table1.3$iq, nclass = 15)
> hist(Table1.3$iq, breaks = seq(80, 160, by = 10))
> hist(Table1.3$iq, probability = T) # probability scale
> lines(density(Table1.3$iq)) # add density curve
> plot(density(Table1.3$iq), type = "l") # density only
> stem(Table1.3$iq)
```

```
N = 60   Median = 114
Quartiles = 104, 125.5
```

Decimal point is 1 place to the right of the colon

```
 8 : 129
 9 : 0467
10 : 01112223568999
11 : 00022334445677788
12 : 223444456778
13 : 0134446799
14 : 25
```


1.1.3 Time plots

Time plots are used to measure change over time.

Example 1.5 Mississippi river discharge

Table 1.4 in IPS presents the yearly discharge of the Mississippi River from 1954 to 2001.

1. Load the data using `IPSdata > Select Data...` and select `Table1.4`.
2. Select the `year` column.
3. CTRL-click on the `discharge` column. Select the first column in the data set (the prices).

4. Click on any of the three line plot buttons from the top row of the `Plots 2D` palette .

To change the labeling,

5. Right-click on the x axis and choose `Range...` from the shortcut menu.
6. Set `Axis Minimum:` to 1950, and `Axis Maximum` to 2005.
7. You can try other options, and see their results by clicking `Apply`.
8. Click `OK`.

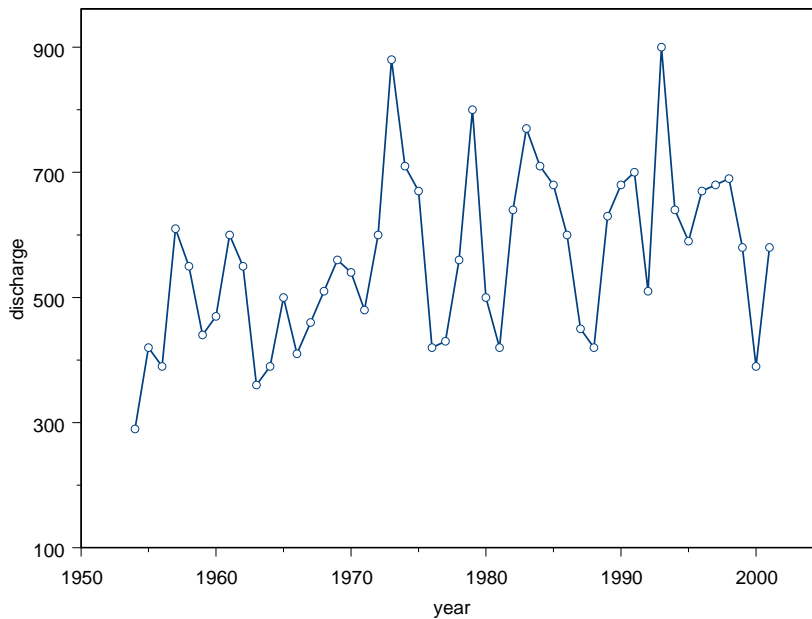
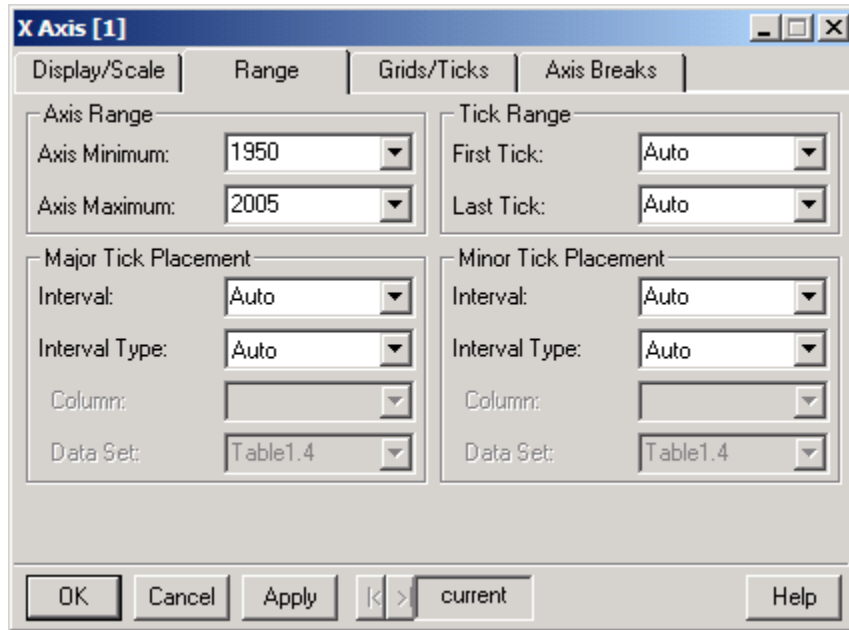


Figure 1.12

End of Example 1.5

The previous example had one column of dates and another containing data. If your data set has only a single column, containing data, then you need select only that data; S-PLUS then assumes the times are equally spaced.

If your data was not entered in time order, then right-click on the line in the plot and choose [Smooth/Sort](#) from the shortcut menu. In the dialog box, change [Pre-Sort Data:](#) to [X, Y on X](#) and click [OK](#).

Command Line Notes

Time plots can be made using the `plot` command.

```
> attach(Table1.4)
> plot(year, discharge)
> plot(year, discharge, type="l")
> plot(year, discharge, type="b")
> detach()
```

The `type` option to `plot` includes "p" for points, "l" for line, "b" for a line and points. See the online help files for `plot` and `lines` for more options.

1.2 Describing Distributions with Numbers

Summary statistics of numeric data including the mean, median, quartiles, and standard deviation can be computed using [Statistics > Data Summaries > Summary Statistics...](#)

Example 1.6 Car mileage

We return to the vehicle data from *Consumer Reports*, 1990.

1. Open the `fuel.frame` data set.
2. Select menu [Statistics > Data Summaries > Summary Statistics...](#)
3. In the [Summary Statistics](#) dialog box, for **Variables:** choose `<ALL>`, and for **Group Variables:** select `(None)`.

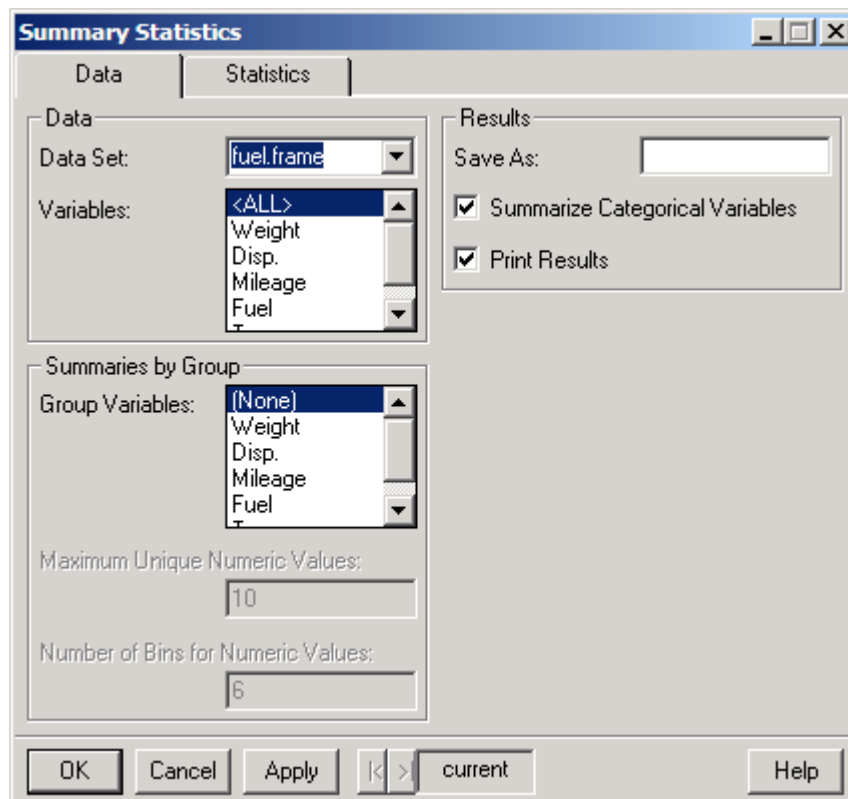


Figure 1.13

- Click on the [Statistics](#) tab at the top of the dialog box to go to the next page. S-PLUS computes the statistics with check marks next to them; the most common summary statistics are selected by default.
- Click [OK](#).

```

*** Summary Statistics for data in: fuel.frame ***

    $$$"Factor Summaries":
      Type
Compact:15
  Large: 3
Medium:13
  Small:13
  Sporty: 9
    Van: 7

    $$$"Numeric Summaries":
      Weight   Disp.   Mileage   Fuel
Min: 1845.0000  73.00000  18.000000  2.7027027
1st Qu.: 2571.2500 113.75000  21.000000  3.7037037
Mean: 2900.8333 152.05000  24.583333  4.2100326
Median: 2885.0000 144.50000  23.000000  4.3478261
3rd Qu.: 3231.2500 180.00000  27.000000  4.7619048
Max: 3855.0000 305.00000  37.000000  5.5555556
Total N:   60.0000  60.00000  60.000000  60.0000000
NA's :     0.0000   0.00000  0.000000  0.0000000
Std Dev.: 495.8661  54.16091  4.791559  0.7575258

```

The output gives counts for the categorical variable and the five-number summary as well as the mean, standard deviation, and a five-number summary for each quantitative variable.

End of Example 1.6

Remarks:

- The method used by S-PLUS to compute the quartiles is a bit different from the process described in IPS.
- To see the summary statistics by car type, return to the [Summary Statistics](#) dialog box and change the [Group Variables:](#) field to [Type](#). Click [OK](#).
- If you select a continuous variable for the [Group Variables:](#) field, then S-PLUS automatically breaks it into equal-length categories, and supplies summary statistics for the corresponding groups.

S-PLUS can also produce boxplots based on the five-number summary.

Example 1.7 Boxplots of car data

- Bring up the [fuel.frame](#) data set.
- Select [Type](#), then CTRL-click on [Mileage](#).

3. Click on the [boxplot](#) button  to create the boxplot.

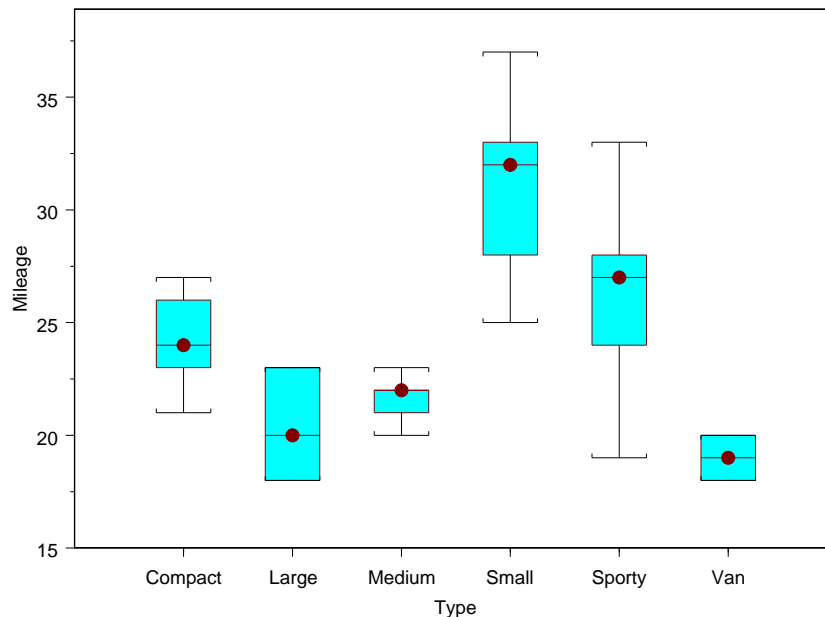


Figure 1.14

End of Example 1.7

Remarks:

- The order in which you select the variables is important. If your boxplot does not look correct, reselect the two variables (factor variable first, then CTRL-click the numeric variable).
- You can right click on any of the boxes and click on the shortcut menu to get the boxplot dialog box. This gives you options to change the colors of the boxes, the styles of lines and points used in the boxes, and other options.
- S-PLUS plots suspected outliers (using the $1.5 \times IQR$ criterion) as points above and below the boxplots. The [Range:](#) option on the [Box Specs](#) page of the dialog allows you to change the decision rule for plotting suspected outliers (entering 2 means to use $2 \times 1.5 \times IQR$).
- If the data for each group are in its own column, you can still create side-by-side boxplots: with no columns selected in the data frame, click on the [boxplot](#) button. The [Boxplot](#) dialog box opens with the [x Columns:](#) and [y Columns:](#) fields blank. Leave the [x Columns:](#) field blank and select the desired columns (use CTRL-click to make multiple selections) for the [y Columns:](#). Click [OK](#).

You can transform variables by using the [Data > Transform](#) menu. Select the data set, then enter the name you want the new (transformed) column to have in the [Target Column:](#) box. Then enter the function for the transformation in the [Expression:](#) box.

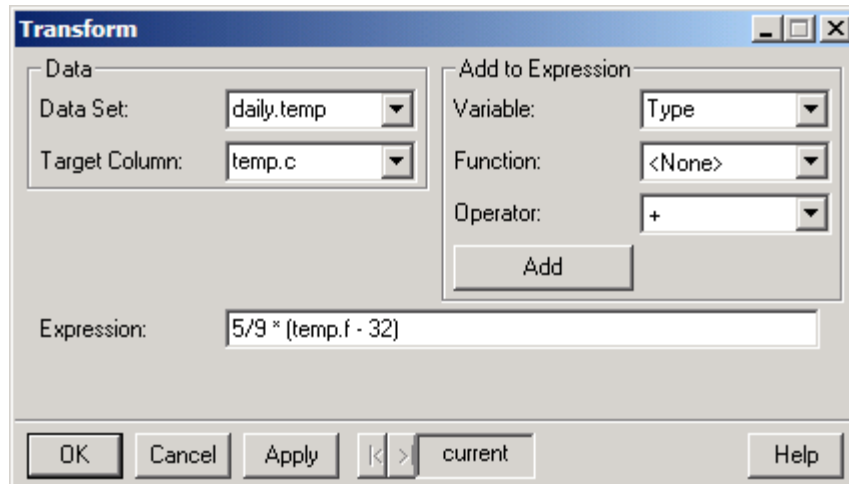


Figure 1.15

Command Line Notes

The `summary` command computes summaries of numeric and factor variables, `mean` computes the mean, `stdev` finds the standard deviation, and `boxplot` together with `split` creates boxplots.

```

> summary(fuel.frame)
  Weight      Disp.      Mileage  ...
  Min.:1845    Min.: 73.0    Min.:18.00  ...
  1st Qu.:2571  1st Qu.:113.8  1st Qu.:21.00  ...
  ...
> attach(fuel.frame)
> summary(Mileage)
  Min. 1st Qu. Median Mean 3rd Qu.  Max.
 18.00 21.00  23.00 24.58 27.00  37.00
> mean(Mileage)
[1] 24.58
> stdev(Mileage)
[1] 4.791559
> boxplot(split(Mileage,Type))
> detach()
> daily.temp$temp.c = 5/9 * (daily.temp$temp.f - 32)

```

1.3 The Normal Distributions

You can compute the normal probability values found in Table A of IPS.

Example 1.8 SAT scores (IPS Example 1.26)

We want to find the proportion of students who have SAT scores between 720 and 820. SAT scores approximately follow a normal distribution with $\mu = 1026$ and $\sigma = 209$.

1. Create a new data set .

- Enter 720 and 820 into the first column.
- At the menu, select **Data > Distribution Functions...**
- In the **Distribution Functions** dialog box, select the name of the data set for **Data Set:**.
- For **Source Column:** select the column that you just created.
- The output is sent to a new column. Designate the name of this new column in the **Target Column:** field (the default name is **Probability**).
- Under **Distribution Parameters**, For **Mean:** enter 1026, and for **Std. Deviation:** enter 209.

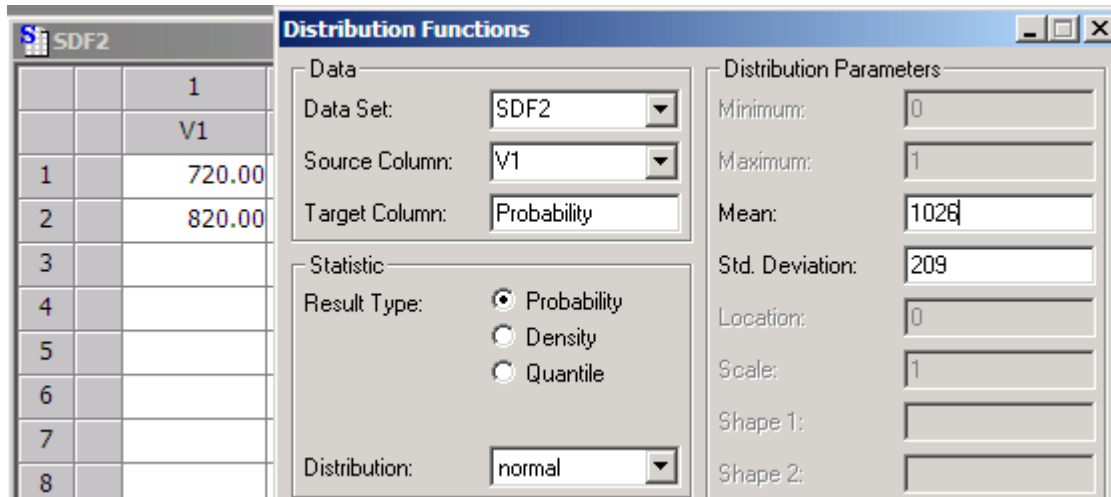



Figure 1.16

- Click **OK**.
- Select the new column and click on the **Increase precision**  button twice to view more significant digits.

If we let X denote the SAT score, then the S-PLUS output tells us that $P(X \leq 720) = 0.0716$ and $P(X \leq 820) = 0.1622$. Thus, we want $P(720 \leq X \leq 820) = 0.1622 - 0.0716 = 0.0906$.

End of Example 1.8

Example 1.9 Inverse normal calculations (IPS Example 1.30)

We want to find what score we need to earn on the SAT verbal section in order to be in the top 10%, given that the scores are approximately $N(505, 110)$.

- Create a new data frame and enter the value 0.9 into the first cell.
- At the menu, select **Data > Distribution Functions...**
- In the **Distribution Functions** dialog box, for **Data Set:** select the name of the data set.
- For **Source Column:** select the column that you just created. For **Target Column:** keep the default name.

- For **Result Type**:, check the radio button next to **Quantile**.
- Under **Distribution Parameters**, for **Mean**: enter 505, and for **Std. Deviation**: enter 110.

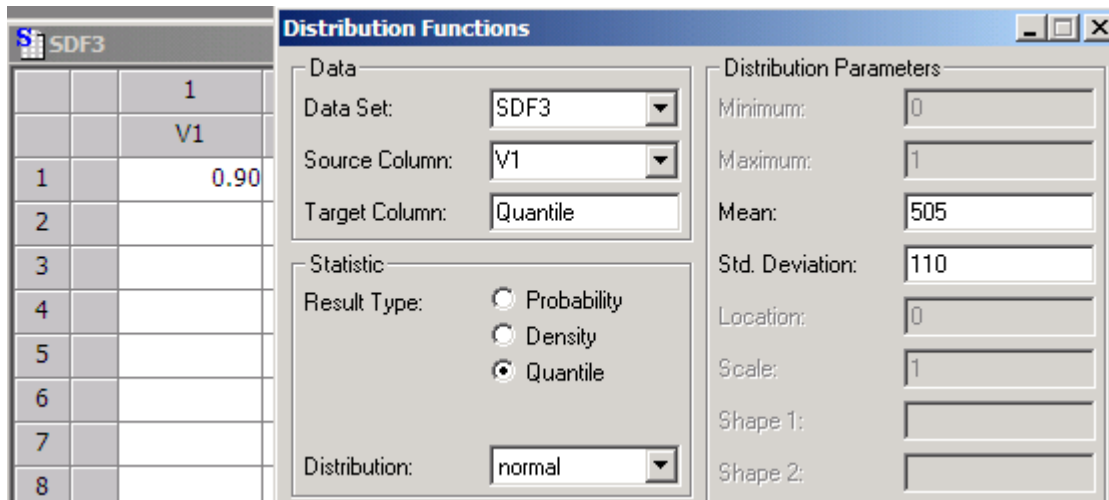


Figure 1.17

- Click **OK**.

We see that we would need to score 646 (rounded up from 645.97) to place in the highest 10%.

End of Example 1.9

Example 1.10 Superimposing a normal curve on a histogram

It is often informative to see how a histogram compares to a normal curve with the same mean and standard deviation.

- Bring up the **fuel.frame** data set and create a histogram of **Weight** (recall Section 1.1.2).
- Right-click on the histogram and choose **Options...** from the shortcut menu.
- In the **Histogram/Specs** dialog box, change **Output Type**: to **Density**.
This scales the histogram so that the total area is 1.
- Click **OK**.

We next generate a sequence of values, 1000 to 4400, which encompasses the range of x -values in the histogram above.

- Create a new data set.
- At the menu, select **Data > Fill...**
- In the **Fill Numeric Columns** dialog box, for **Data Set**: select the data set, and for **Columns**: select **<END>**.
- Under **Fill Options**, for **Length**: enter 341, for **Start**: enter 1000, and for **Increment**: enter 10.
- Click **OK**.

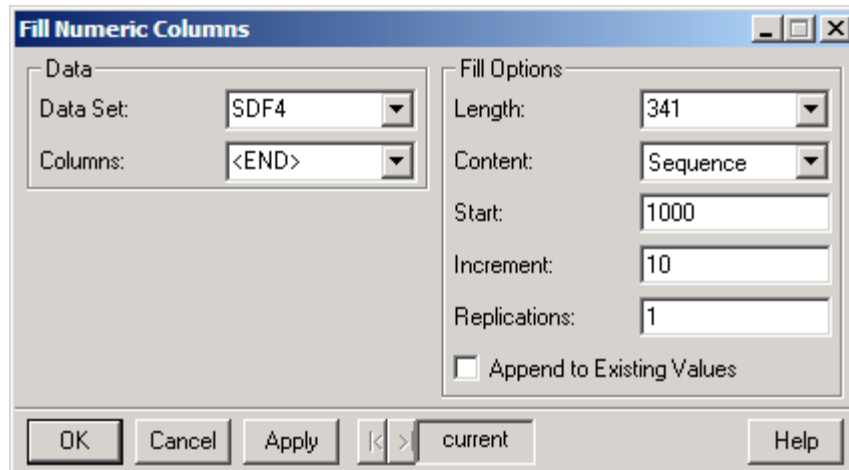


Figure 1.18

From Example 1.4, we know that the mean of **Weight** is 2900.83 and the standard deviation is 495.866. We wish to graph the normal density curve with these parameters over the range 1000 to 4400.

10. At the menu, select **Data > Distribution Functions...**
11. In the **Distribution Functions** dialog box, for **Data Set**: select the name of the data set created above.
12. For **Source Column**: select the column created by the **Fill** command. For **Target Column**: leave the default name.
13. Change **Result Type**: to **Density**.
14. Under **Distribution Parameters**, for **Mean**: enter **2900.83**, and for **Std. Deviation**: enter **495.866**.
15. Click **OK**.

The new column contains small numbers which appear to be zero, unless you increase the precision. Finally, we plot the normal curve and superimpose it onto the histogram of **Weight**.

16. Select the column created by the **Fill** command above, then CTRL-click on the **Density** column just created by the **Distribution Functions** command.
17. Click on the white background of the graphsheet with the histogram. There should be eight green knobs around the border of the histogram graph area.

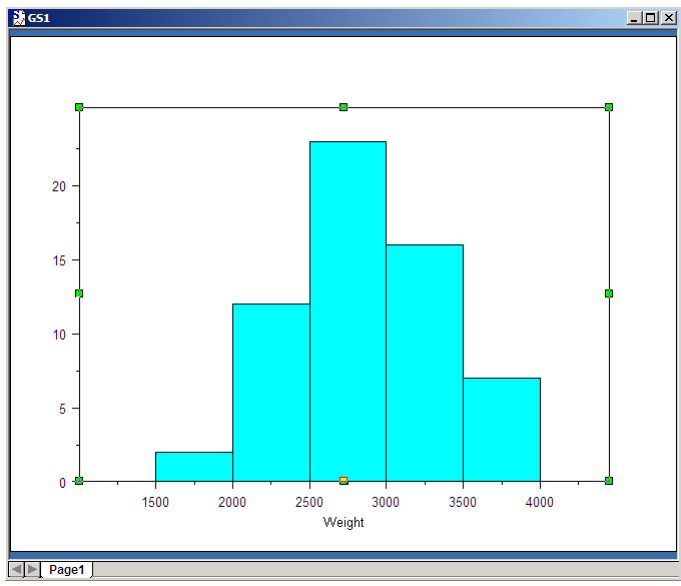



Figure 1.19

18. Hold down the SHIFT key and click on the [Line](#) button  on the [Plots 2D](#) palette.

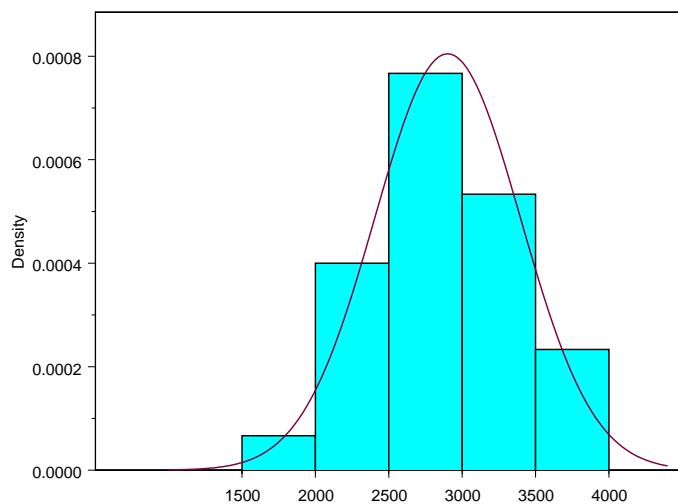


Figure 1.20


The density plot should be superimposed on the histogram. If instead, you have two plots adjacent to each other on the same graphsheet, or if you created a new plot on a separate graphsheet, repeat the above steps (from 16). The histogram graphsheet must be selected as noted above and the SHIFT key must be held down *simultaneously* while the [Line](#) plot is being clicked on.

End of Example 1.10

1.3.1 Normal quantile plot

A *normal quantile* plot is a graph used to see how well the normal distribution fits a data set. A normal quantile plot graphs the data against what would be expected from a normal distribution. If the data fall close to a straight line then the data are approximately normally distributed. Data points far from the straight line show where there are differences between the data and a normal distribution.

Example 1.11 Normal quantile plot of car weight

1. Bring up the `fuel.frame` data set.
2. Select the `Weight` column.
3. Click on the `QQ Normal` button  on the `Plots 2D` palette.

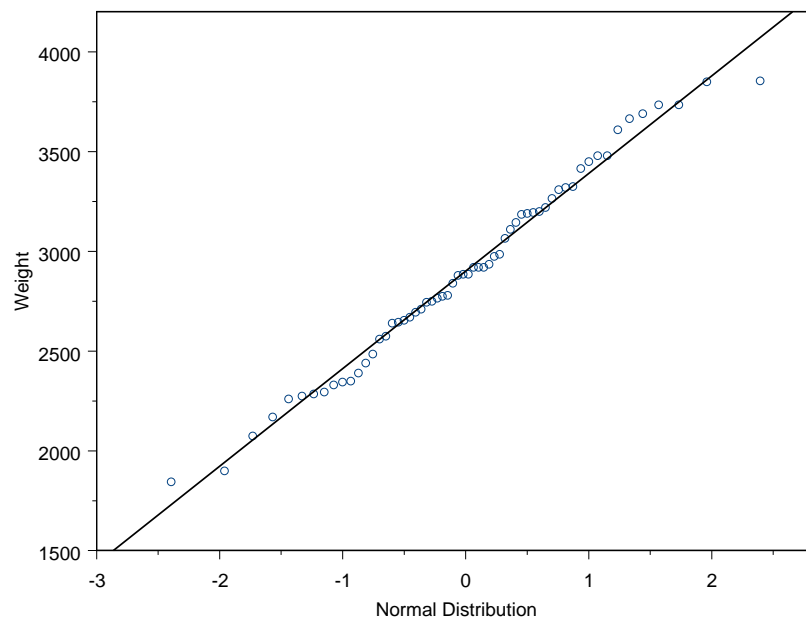


Figure 1.21

The weights of the cars are fairly normally distributed.

End of Example 1.11

Command Line Notes

The `pnorm` function computes the probabilities for the normal distribution, `qnorm` gives the x -value for a given probability, and `dnorm` gives the height of the normal curve—“p” for probability, “q” for quantile, and “d” for density. The function `qqnorm` creates a normal quantile plot and `qqline` adds the reference line to it.

```
> pnorm(c(720, 820), mean = 1026, sd = 209)
[1] 0.07158129 0.16215344
> 0.16215344 - 0.07158129
[1] 0.09057215
> qnorm(0.9, mean = 505, sd = 110)
[1] 645.9707
> attach(fuel.frame)
> hist(Weight, prob = T, col = 6, xlim = c(1500, 4500))
> lines(1000:4400,
+       dnorm(1000:4400, mean = mean(Weight),
+           sd = stdev(Weight)))
> qqnorm(Weight)
> qqline(Weight)
> detach()
```

1.4 Editing Graphs

In this section, we summarize the editing features that are possible with S-PLUS graphs created through the GUI.

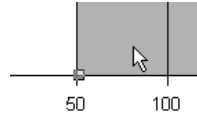
Creating a good-looking graph in S-PLUS is both quick and easy. Sometimes the graph can be improved by making small changes such as showing a larger range of data values, adding reference lines, adding tick marks, adding a comment, or changing the symbol used in the plot.

1.4.1 Editing existing portions of a graph

Any plot created through the GUI can be customized by right-clicking on different features of the graph and choosing the desired option from the shortcut menu. Below, we show where the cursor should be placed to access the appropriate dialog boxes.

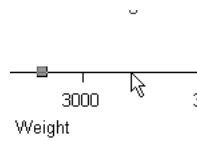
Bars, Slices, and Points

Right-click on a bar, slice, or point to access the dialog box with options for changing colors, fill patterns, line styles, and plotting symbols and their size.



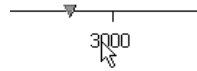
X and Y Axes

Right-click on either axis to access the dialog box with options for changing the range of the data that is plotted, the placement or appearance of tick marks, the scaling of the axis (for example, to a log scale), or for adding grid lines.



Tick Labels

Right-click on the labels corresponding to the tick marks to access the dialog box with options for changing the appearance of tick labels (font, rotation angle, and so on).




Axis Titles

Right-click on the axis title to access the dialog box with options to change fonts, edit content, add a date stamp, and so forth.



1.4.2 Annotating or adding to the graph

The [Annotations](#) button  brings up the graph [Annotations](#) palette.

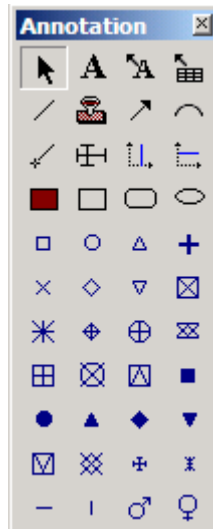












Figure 1.22

This palette has buttons for adding objects such as text (for labels or notes) , a time and date stamp , reference lines and arrows    , or various reference symbols   .

Any of the annotations can be modified by choosing the [Select Tool](#) button  then right-clicking on the annotation to be changed to access the shortcut menu.


The [Graph Tools](#) palette  provides additional tools for labeling points, cropping, and so forth:



Figure 1.23

1.5 Multiple Plots in Command Line Graphics

Graphs created at the command line in S-PLUS are written to a *graphics device*. When you create a plot, a graphics device is opened automatically. Subsequent plots are written to the same device, overwriting earlier plots.

```
> hist(fuel.frame$Weight)
> boxplot(fuel.frame$Weight)
```

You can have two plots visible by manually opening another graphics device.

```
> graphsheet()
> hist(fuel.frame$Mileage)
> dev.list() # Show list of current devices
  graphsheet  graphsheet
      2         3
> dev.cur() # Show currently active device
  graphsheet
      3
> help(dev.list) # Help, including commands for closing
> # or switching devices
```



Graphsheet 3 (GSD3) is the active device so subsequent graphs are displayed in this window.


1.6 Exercises

These exercises use the data set [cu.summary](#).

1. Summarize graphically the variables [Reliability](#) and [Price](#).
2. Summarize numerically the variable [Price](#).
3. Is the [Price](#) variable normally distributed?

1.7 Solutions

1. From the GUI, select the **Reliability** column and click on either the **Bar Zero Base** button  or the **Pie Chart** button .

Next, select the **Price** column and click on the **Histogram** button .

From the command line,

```
> attach(cu.summary)
> barplot(table(Reliability), names = levels(Reliability))
> hist(Price)
> detach()
```

2. From the GUI,

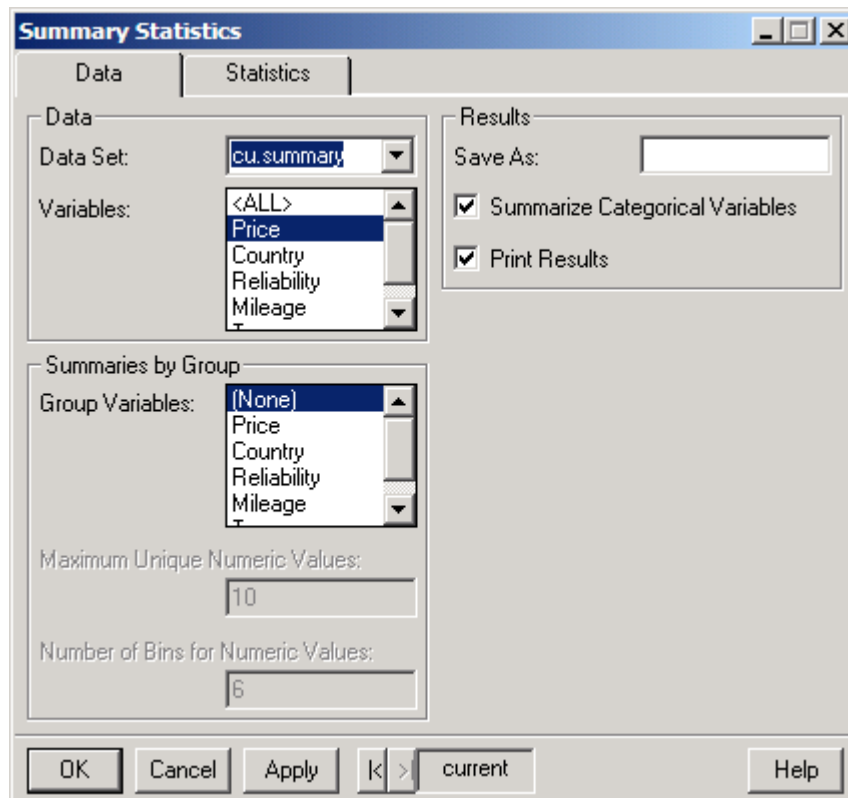



Figure 1.24

From the command line,

```
> summary(cu.summary$Price)
```

3. Create a QQ normal plot, .

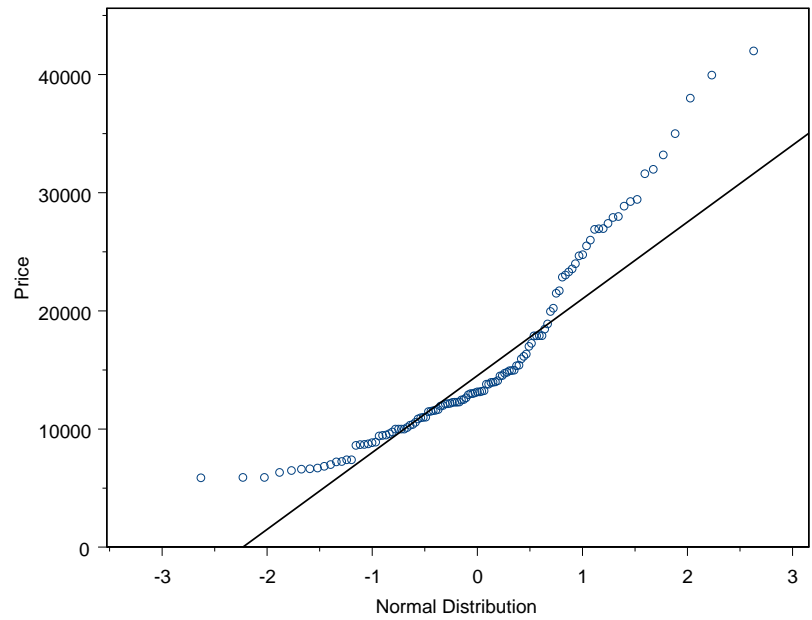


Figure 1.25

It is clear that `Price` does not follow a normal distribution; the largest values in `Price` are much greater than we would expect from a normal distribution.

Chapter 2


Looking at Data—Relationships

2.1 Scatterplots

Scatterplots are the easiest and most effective way to see relationships between two numeric variables. S-PLUS has several variations on scatterplots to allow you to explore these relationships.

Example 2.1 Gas mileage

We will explore data on the weight of cars and their gas mileage. The `fuel.frame` data set contains information on gas mileage for several cars as reported in *Consumer Reports* (April 1990).

1. Open the `fuel.frame` data set by clicking on the `Data > Select Data` menu, and typing `fuel.frame` in the `Name:` field and clicking on `OK`.
2. Click on the column header for `Weight`, then CTRL-click on the column header for `Mileage`. The first column selected becomes the x axis (horizontal axis), and the second column selected becomes the y axis (vertical axis).
3. Bring up the `Plots 2D` palette and click on the `Scatter` button in the upper left corner .

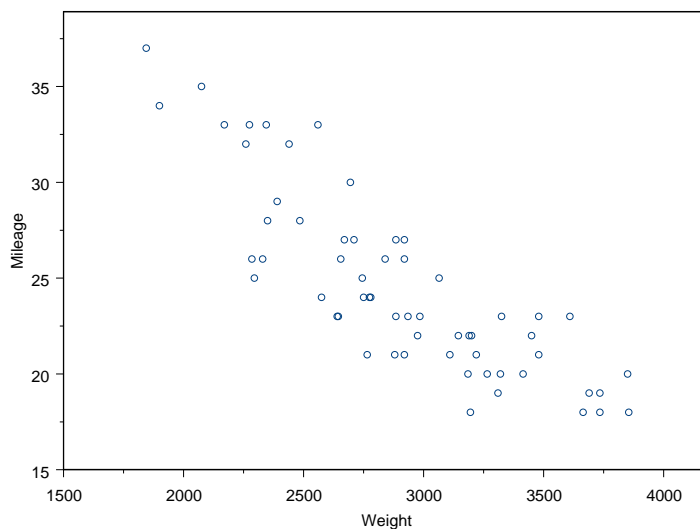


Figure 2.1

We continue with some additional features of S-PLUS graphs.

4. If you hover the cursor over a data point, a data tip appears giving you information about that point (row number, x-value, y-value).

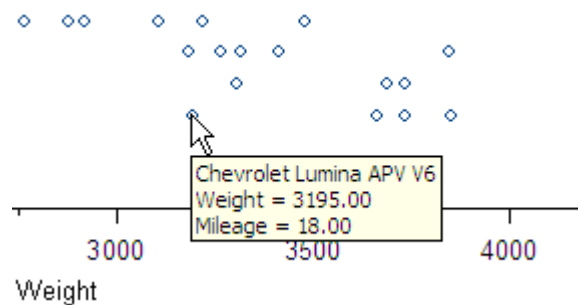



Figure 2.2

To change what appears in the data tip, right-click on a point and choose [Data to Plot...](#) from the shortcut menu. Under [Data Tips and Point Labels](#), select the column(s) that you want to use for the data tip.

5. To find the point that corresponds to a given car, bring the data window to the front and click on the row header of the observation that you want to highlight (you can CTRL-click on additional row headers to highlight multiple points, Section 0.4.8). Now return the graphs sheet to the front. The car(s) that you selected is indicated by a bold red symbol.
6. To change the symbol style, color or size, right-click on any data point and choose [Symbol...](#) from the shortcut menu.
7. Return to the `fuel.frame` data window and click on [Weight](#), CTRL-click on [Mileage](#), CTRL-click on [Type](#), then click on the [Text as Symbol](#) button  on the [Plots 2D](#) palette.. The resulting scatterplot uses each car's type for symbols.

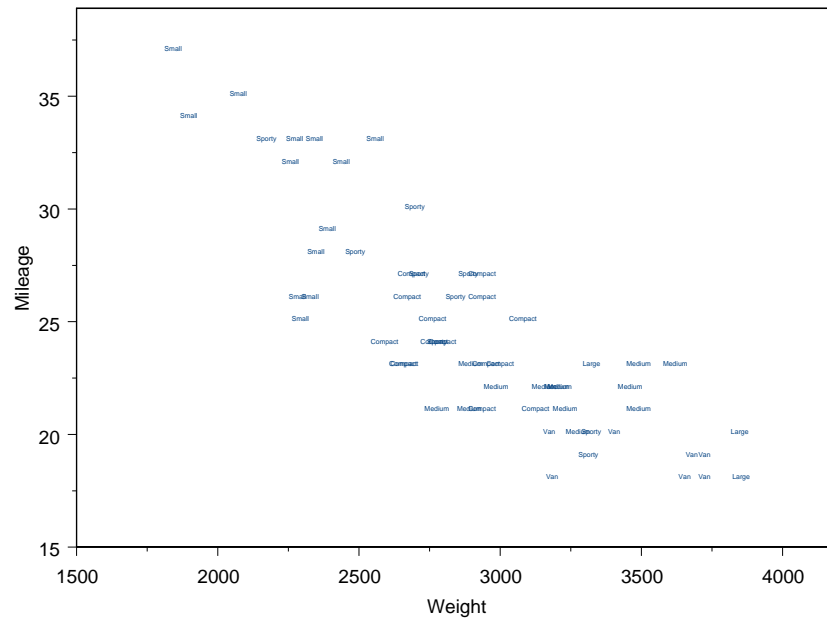




Figure 2.3

End of Example 2.1

Example 2.2 Are vans different?

It is often informative to use different symbols and colors for different groups in a scatter plot.

1. Select the columns **Weight**, **Mileage**, and **Type** in this order.
2. On the **Plots 2D** palette, click on the **Scatter** button . A scatterplot is drawn with different symbols and colors for each Type of car.
3. Add a legend to the plot by clicking on the **Auto Legend** button  on the toolbar below the standard toolbar.

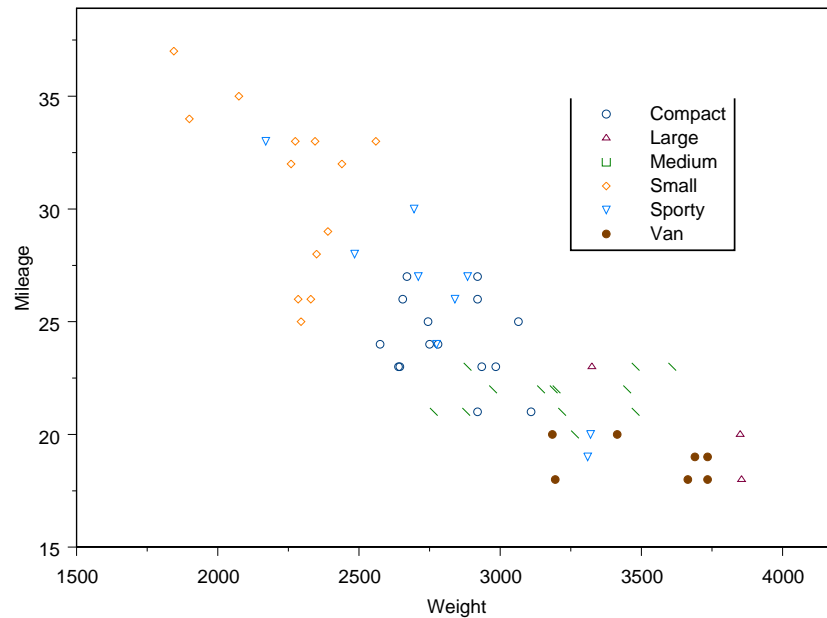


Figure 2.4

Now we can see that the vans are all heavier and get lower gas mileage, but do not fall outside of the overall pattern made by the other cars.


You can move the legend to a different location by clicking on the box around the legend (green knobs will appear) and then dragging the box.

To change the default symbol styles and colors for each group of symbols, right-click on a data point from a group and select [Symbols...](#) from the shortcut menu. Make your changes in the resulting dialog box.

End of Example 2.2

Example 2.3 Scatterplot matrices

When first exploring data we often make several different scatterplots. A scatterplot matrix is a way to look at all possible combinations of two variables in one step.

1. Bring up the `fuel.frame` data set.
2. Select `Weight`, hold down the SHIFT key and click on the header of the second to last column (`Fuel`). This selects *all* of the numeric columns.
3. Click on the [Scatter Matrix](#) button  on the [Plots 2D](#) palette.

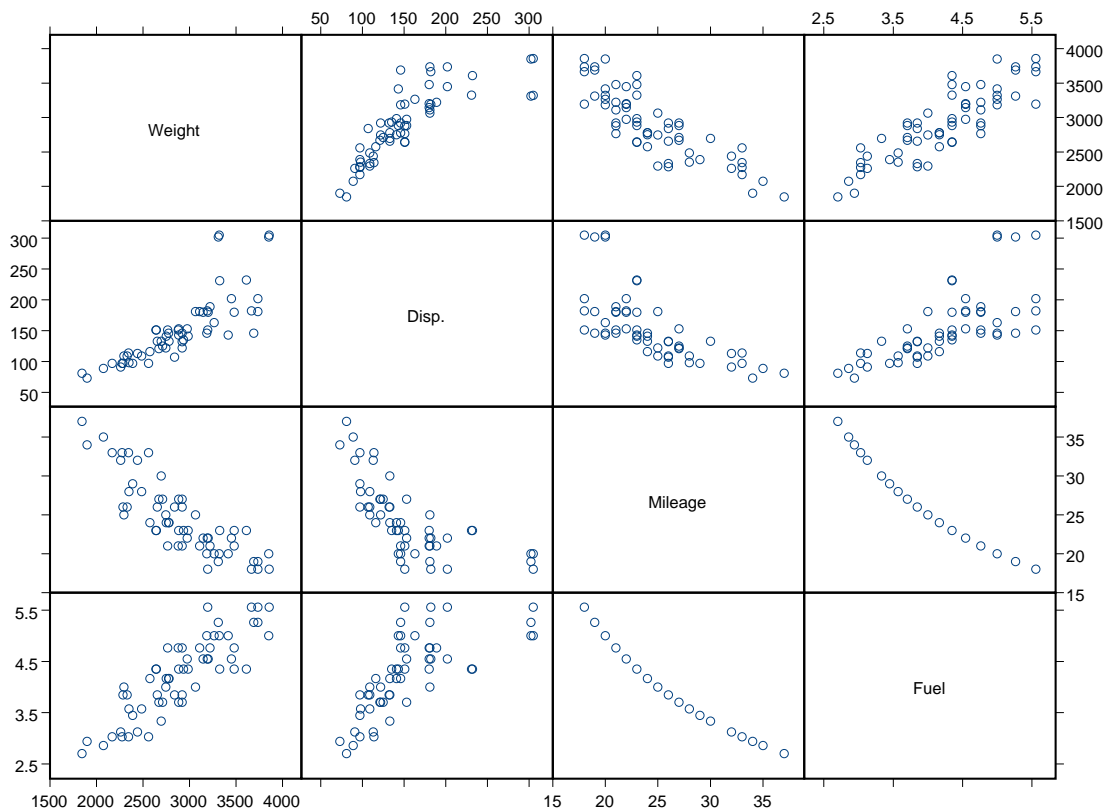


Figure 2.5

This produces scatterplots using all possible pairs of the variables selected.

To determine the y axis for any plot, scan horizontally until you reach the square with a variable name.

To determine the x axis for this plot, scan vertically until you reach the square with a variable name.

4. To add a trend line to the scatterplot matrix, right-click on any data point and select **Smooth...** from the shortcut menu.
5. This brings up the **Smooth** tab of the **Scatter Plot Matrix** dialog box; go to the **Smoothing Type:** field and select **Least Squares**.
6. Click **OK**.

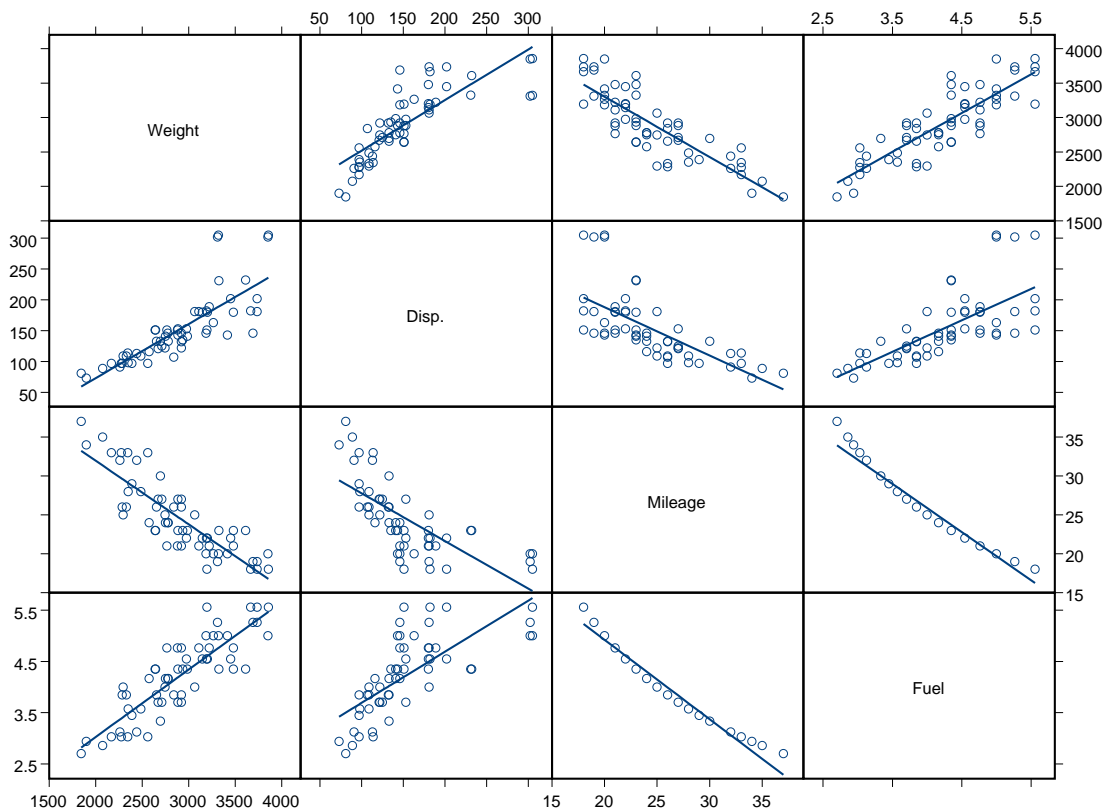


Figure 2.6

End of Example 2.3

2.1.1 Brush and spin

Brushing and *spinning* are dynamic graphics techniques for visualizing data in three or more dimensions. Brushing is a variation of a scatterplot matrix with dynamically linked plots; points selected in one plot are highlighted in other plots. Spinning is a three-dimensional scatterplot that may be spun around to be viewed from different angles. The brush and spin windows may be linked, so that points selected in either are highlighted in the other.

1. Bring up the `fuel.frame` data set.
2. Select `Weight`, hold down the SHIFT key and click on the header of the second to last column (`Fuel`). This selects *all* of the numeric columns.
3. At the menu, select `Graph > Brush and Spin...`
4. Click `OK`.
5. Use the arrows in the upper left to spin the three-dimensional scatterplot.
6. Brush across points in any of the windows to highlight them in all windows.

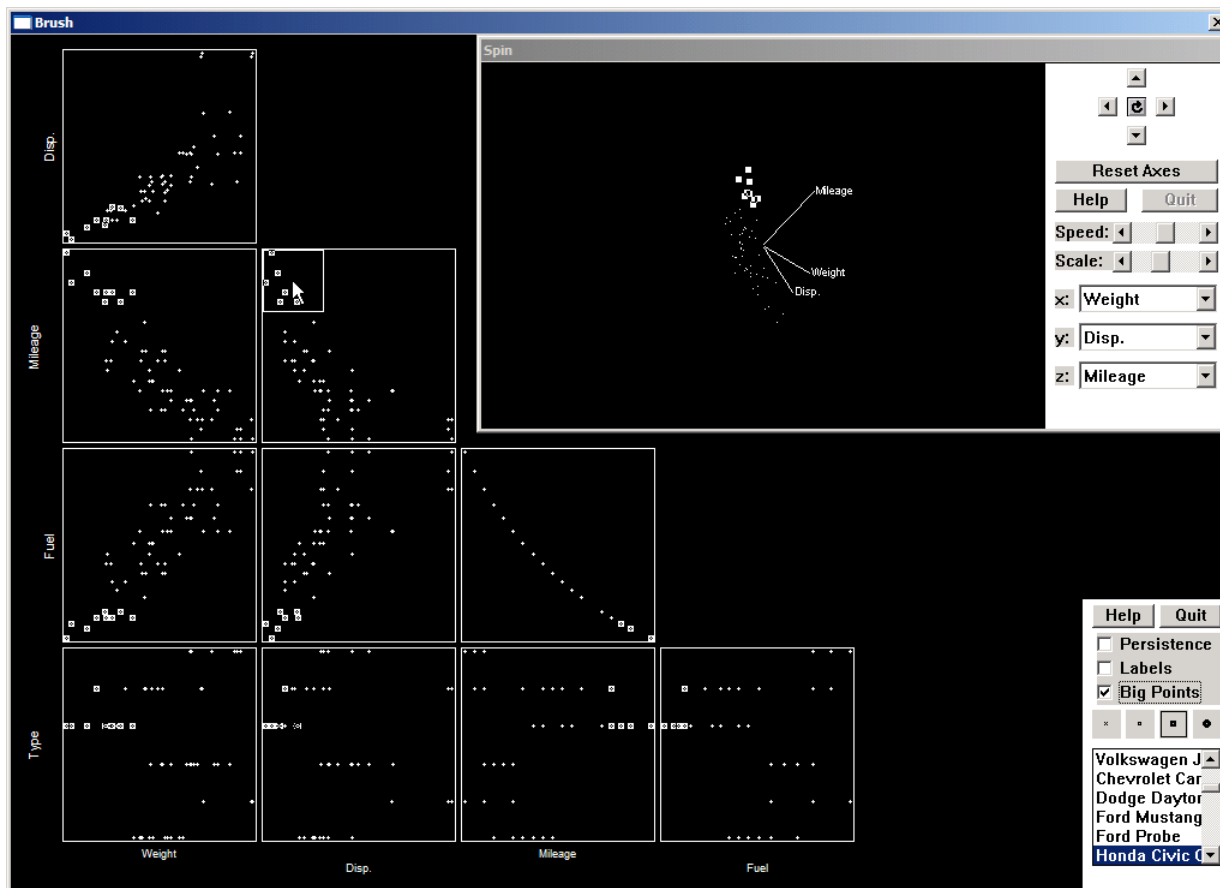


Figure 2.7

2.1.2 Smoothers

A *smoother* is a line that is added to a scatterplot to indicate curvature. There are many different algorithms for constructing smoothers, and we refer the interested reader to the online manuals (*Guide to Statistics, Vol. 1*) for further details.

Example 2.4 Scatterplot smoothers

We will continue with the Mileage data.

1. Bring up the `fuel.frame` data set and select `Weight`, then `Mileage`.
2. Click on any of the four scatterplot smoother buttons, `Loess`, `Spline`, `Supersmooth`, or `Kernel`



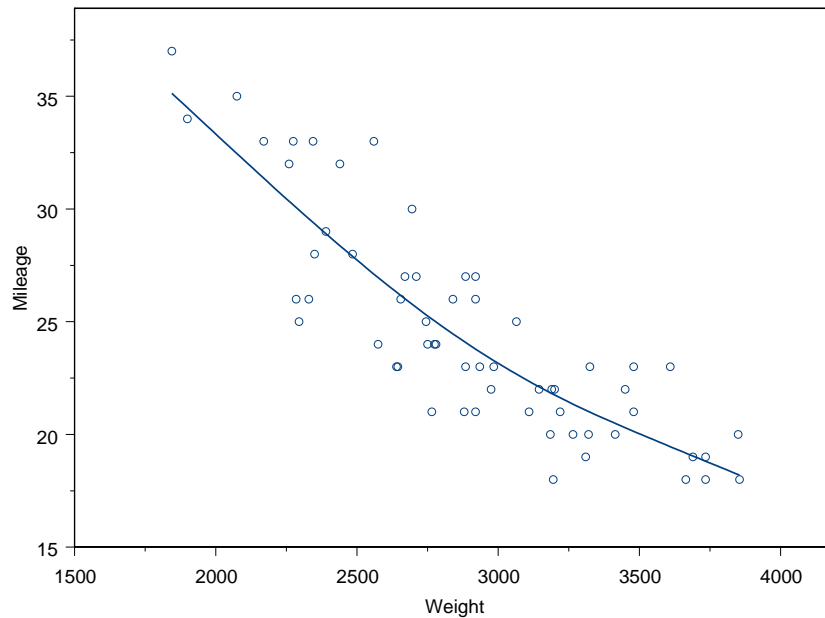


Figure 2.8

The added smooth line emphasizes that the relationship has some curvature and is probably not a straight line relationship (the plot in the above figure used the spline smooth).

3. Try the Kernel smoother. This is a case where the default smoothing parameter is poor (really poor), resulting in a smooth that is not smooth enough. We have often found this with the Loess smoother too, for small data sets. To adjust the parameter right-click on any data point and select [Smooth...](#) from the shortcut menu. You should be in the [Smooth/Sort](#) button. In the [Kernel Specs](#) field, increase the [Bandwidth](#) to 500 and change the [Kernel](#) to [Normal](#). You can try different values by clicking [Apply](#).
4. When you have a curve you are satisfied with click [OK](#).

End of Example 2.4

See the *S-PLUS User's Guide* chapters on creating plots, exploring data, and editing graphics for more details on the different types of graphs you can make, and various options for changing the appearance of your graphs.

Command Line Notes

Scatterplots are created using the `plot` command, scatterplot matrices are created with the `pairs` command. Brushing with or without spinning are done with `brush`, and spinning alone with `spin`. Creating scatterplots using different symbols for different groups is more complicated from the command line. See the chapters on traditional and trellis graphics in the *S-PLUS Programmer's Guide* (specifically “superposing”) if you are interested.

A scatterplot with a loess smoother can be created using `scatter.smooth` and the `panel.smooth` function can be used to add a loess smoother to a scatterplot matrix. Spline smooths are created using `smooth.spline`, kernel smooths using `ksmooth`, and Super Smooths using `supsmu`.

```
> attach(fuel.frame)
> plot(Weight, Mileage)
> plot(Weight, Mileage, pch = ".")
> plot(Weight, Mileage, pch = "*")
> pairs(fuel.frame)
> pairs(fuel.frame, panel = panel.smooth)
> brush(fuel.frame[,1:4])
> spin(fuel.frame[,1:4])
> scatter.smooth(Weight, Mileage)
> detach()
```

2.2 Correlation

Correlation measures the linear relationship between two numeric variables.

Example 2.5 Computing correlations

1. At the menu, select **Statistics > Data Summaries > Correlations...**
2. In the **Correlations and Covariances** dialog box, for **Data Set:** select `fuel.frame`.
3. Select the numeric variables in the **Variables:** box by clicking on `Weight`, then scrolling down and SHIFT-clicking on `Fuel`.

All the numeric variables are highlighted.

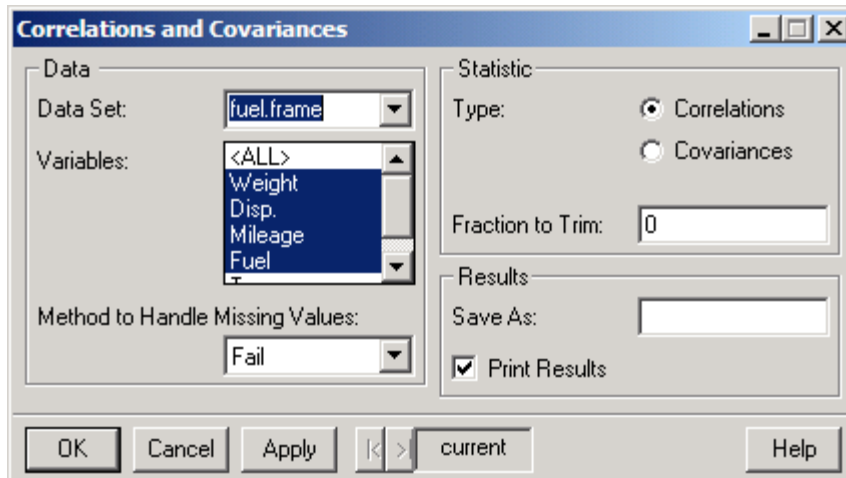


Figure 2.9

4. Click [OK](#).

```

*** Correlations for data in: fuel.frame ***

numeric matrix: 4 rows, 4 columns.
      Weight  Disp.  Mileage  Fuel
Weight 1.000000 0.8032804 -0.8478541 0.8616848
Disp.  0.8032804 1.0000000 -0.6931928 0.7138435
Mileage -0.8478541 -0.6931928 1.0000000 -0.9796598
Fuel    0.8616848 0.7138435 -0.9796598 1.0000000

```

The correlation between [Disp.](#) and [Weight](#) is 0.8032.

End of Example 2.5

Command Line Notes

The [cor](#) function computes the correlation of two variables or all possible pairs of correlations for multiple variables. The [rmvnorm](#) function generates random data with the specified correlation.

```

> attach(fuel.frame)
> cor(Disp., Mileage)
> cor(fuel.frame[,1:4]) # Correlations for columns 1 through 4
> plot(rmvnorm(100, rho = -0.75))
> detach()

```

Here is a script that produces 10 plots of variables with correlation 0.5. Paste this into a script window ([File > New](#)), modify as desired, and run it.

```

n = 1000
rho = .5
for(i in 1:10)
  xy = rmvnorm(n, rho = rho)
  plot(xy)


```

2.3 Least-Squares Regression

Correlation measures the linear relationship between two variables and least-squares regression finds the “best” linear equation $y = bx + a$ that describes this relationship. The x variable is also called the “explanatory variable,” “independent variable,” or “predictor variable.” The y variable is also called the “response variable” or “dependent variable.”

Example 2.6 Predicting a child’s growth

Table 2.5 in IPS contains data on two measurements of blood sugar in 18 diabetics. We will fit a least squares regression line to this data.

1. Load the data: [Table2.5](#).
2. Click on the explanatory variable [hba](#), then CTRL-click on the response variable [fpg](#).
3. On the [Plots 2D](#) palette, click on the [Linear Fit Line](#) button .
4. From the main menu, select [Insert > Curve Fit Equation](#).

This adds the equation of the line to the plot.

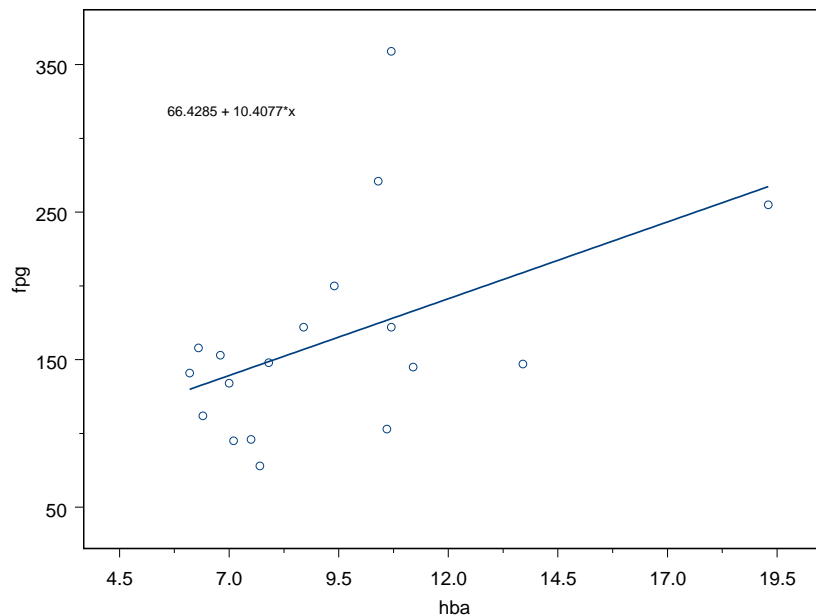


Figure 2.10

To move the equation, click on the equation (green knobs appear) and drag it to the desired location. You can also resize the equation by dragging on one of the green knobs.

We now continue with a more numerical approach.

5. Select [Table2.5](#) from menu [IPSdata > Select Data...](#)

6. Select **Statistics > Regression > Linear...** from the main menu.
7. In the **Linear Regression** dialog box, for **Data Set:** select the name of the data set (**Table2.5**).
8. Under **Variables**, for **Response:** select **fpg**, and for **Explanatory:** select **hba**.
(Note—some versions of S-PLUS use **Dependent:** and **Independent:**).
The **Formula:** field should read **fpg~hba**.

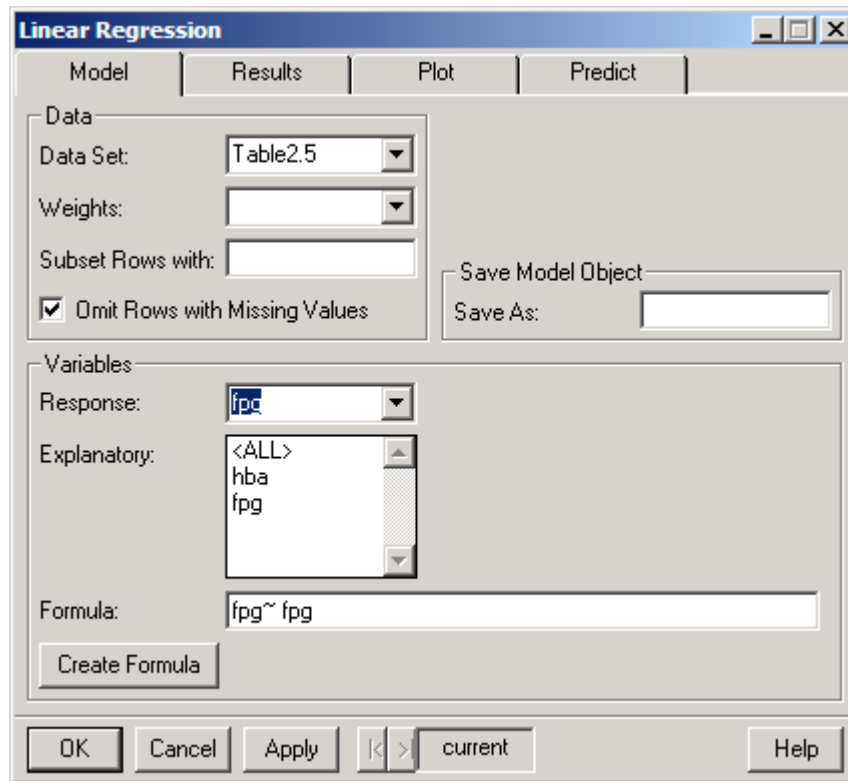


Figure 2.11

9. Click on the **Results** tab to go to the next page of the dialog box.
10. Under **Saved Results**, choose the same data set (**Table2.5**) for the **Save In:** field and check the boxes next to **Fitted Values** and **Residuals**.

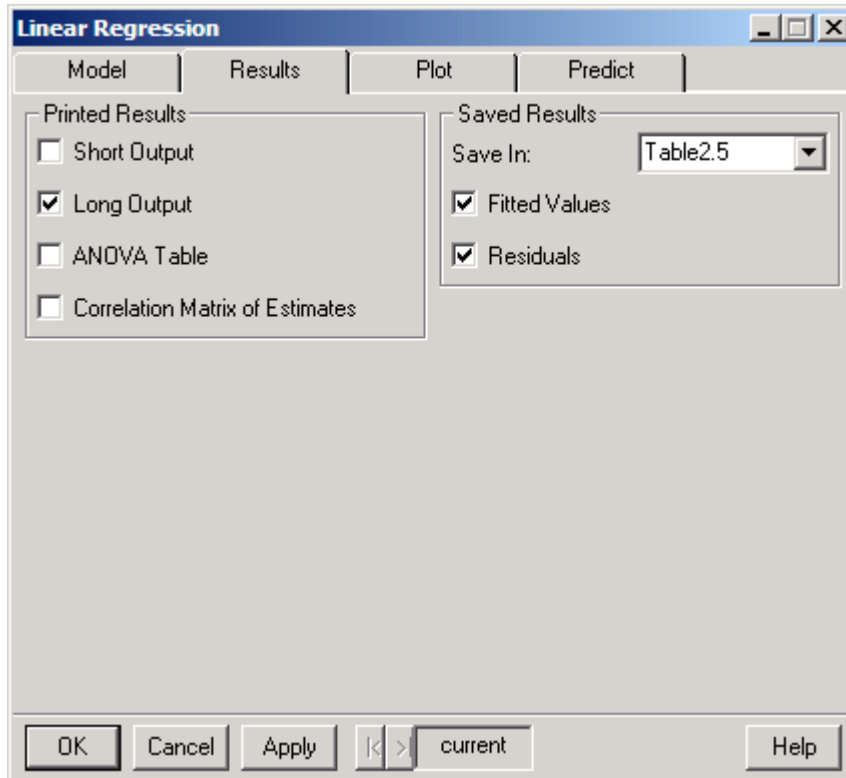


Figure 2.12

11. Click on the [Plot](#) tab.
12. Under [Plots](#), check the [Residuals vs Fit](#) box (and any of the other plots that you want to see).

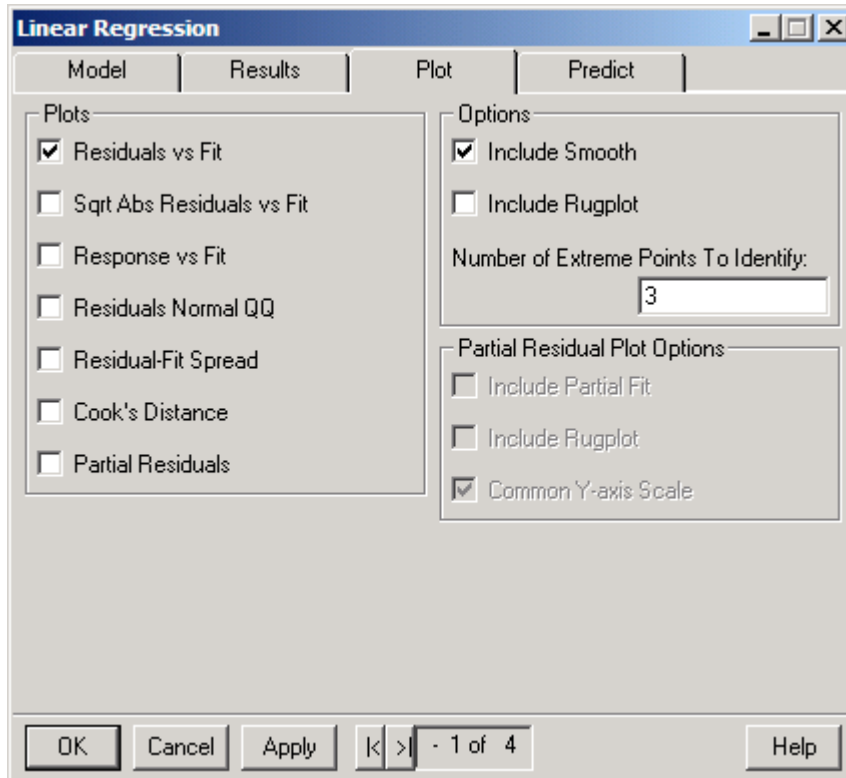


Figure 2.13

13. Click [OK](#).

The results of this command:

- A Report window contains the estimates of the slope (10.4077) and intercept (66.4285) along with other information about this regression.

```

*** Linear Model ***

Call: lm(formula = fpg ~ hba, data = Table2.5, na.action = na.exclude)
Residuals:
    Min     1Q  Median     3Q    Max
-73.75 -43.49  -5.536  15.61  181.2

Coefficients:
                Value Std. Error t value Pr(>|t|)
(Intercept)  66.4285  46.5231     1.4279  0.1726
          hba  10.4077   4.7310     2.1999  0.0429

Residual standard error: 63.82 on 16 degrees of freedom
Multiple R-Squared:  0.2322
F-statistic: 4.84 on 1 and 16 degrees of freedom, the p-value is 0.04285

```

- The fitted values (predictions) and residuals have been added to the data set.

Table2.5					
		1	2	3	4
		hba	fpg	fit	residuals
1		6.10	141.00	129.92	11.08
2		6.30	158.00	132.00	26.00
3		6.40	112.00	133.04	-21.04
4		6.80	153.00	137.20	15.80
5		7.00	134.00	139.28	-5.28
6		7.10	95.00	140.32	-45.32
7		7.50	96.00	144.49	-48.49
8		7.70	78.00	146.57	-68.57

Figure 2.14

- A graph of the residuals against the fitted values, with a scatterplot smooth added, and large residuals highlighted. The smooth used here is a Loess curve, which is resistant to outliers; hence it curves down in the middle even though there are large positive residuals in the middle.

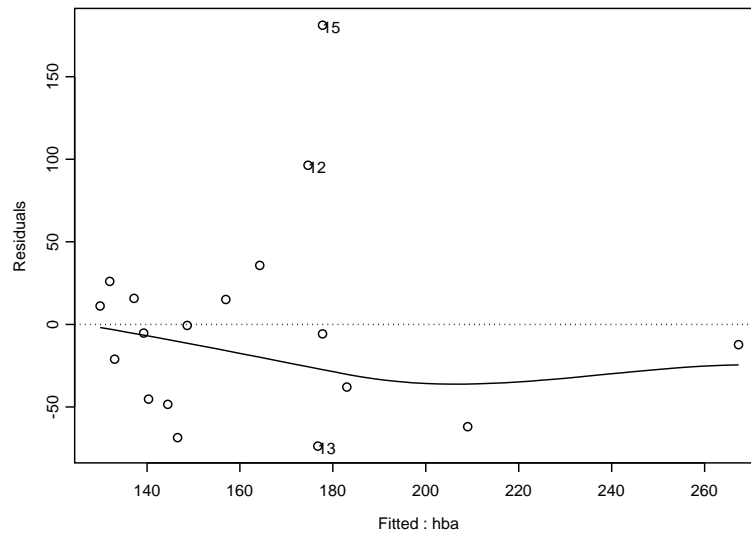


Figure 2.15

Note that you can create this plot by using the data added to the data set: Select [Age](#), then [residuals](#), then click on the [Scatter](#) button on the [Plots 2D](#) palette. Then use the methods described earlier in the chapter to add smooths and highlight points.

End of Example 2.6

Command Line Notes

Regression lines are fit with the `lm` function (Linear Models, another term for least squares regression), the detailed output produced with the `summary` function and the fitted values and residuals given with the `fitted` and `resid` functions, respectively.

The command `abline` can be used for adding reference lines to a plot.

```
> fit.diabetes = lm(fpg ~ hba, data = Table2.5)
> summary(fit.diabetes)
> attach(Table2.5)
> plot(hba, fpg)
> abline(fit.diabetes)
> plot(hba, resid(fit.diabetes))
> abline(h=0)
> detach()
```

Remarks:


- S-PLUS uses a special formula syntax of the form $y \sim x$ to specify the x and y variables. The response variable is written to the left of the tilde \sim , and the explanatory variable is written to the right of the tilde.
- Additional details on the regression output are given in Chapters 10 and 11.

2.4 Cautions About Correlation and Regression

Observing a relationship between two variables does not imply that the explanatory variable causes the response variable. S-PLUS has some tools that allow us to investigate relationships in greater detail.

Example 2.7 Mileage, price, and weight of cars

We continue to explore the data from *Consumer Reports*: what is the relationship between **Price** and **Mileage**? The data set `car.test.frame` combines data from `cu.summary` and `fuel.frame`.

1. Bring up the `car.test.frame` data set using `Data > Select Data`.
2. Create a **Linear Fit** plot  with **Mileage** on the horizontal (x) axis and **Price** on the vertical (y) axis.

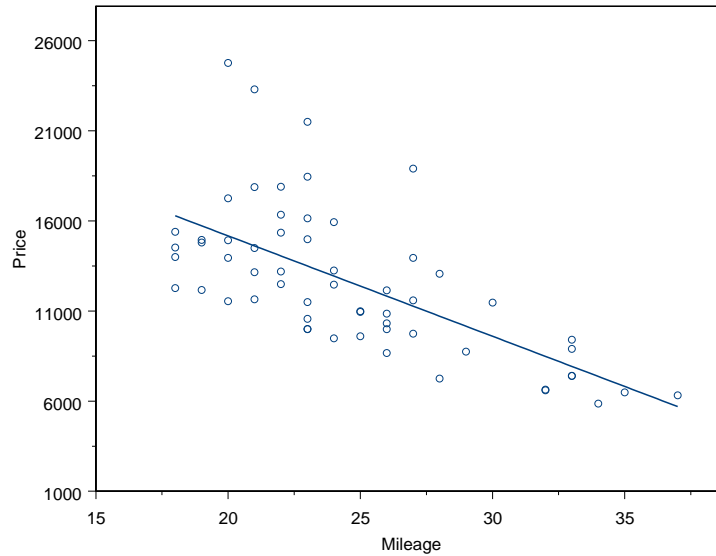



Figure 2.16

Someone ignorant of statistics may interpret this graph to say that improving your gas mileage may also lower the overall price of the car. We know that there could be lurking variables like the size of the car that affect both the price and the gas mileage.

3. CTRL-click on the **Weight** column. There should now be three columns selected: **Mileage**, **Price**, **Weight**.

4. Toggle on the **Set Conditioning Mode** button  on the Standard toolbar. The adjacent field should be a 1.

The buttons on the **Plots 2D** palette should now have yellow strips on the tops.

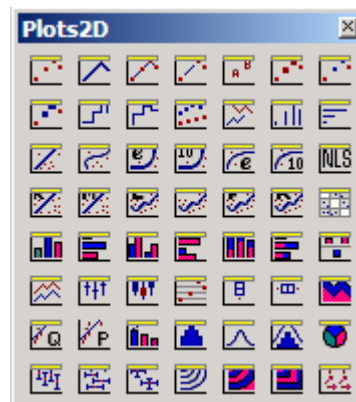



Figure 2.17

5. On the **Plots 2D** palette, click on the **Linear Fit**  button.

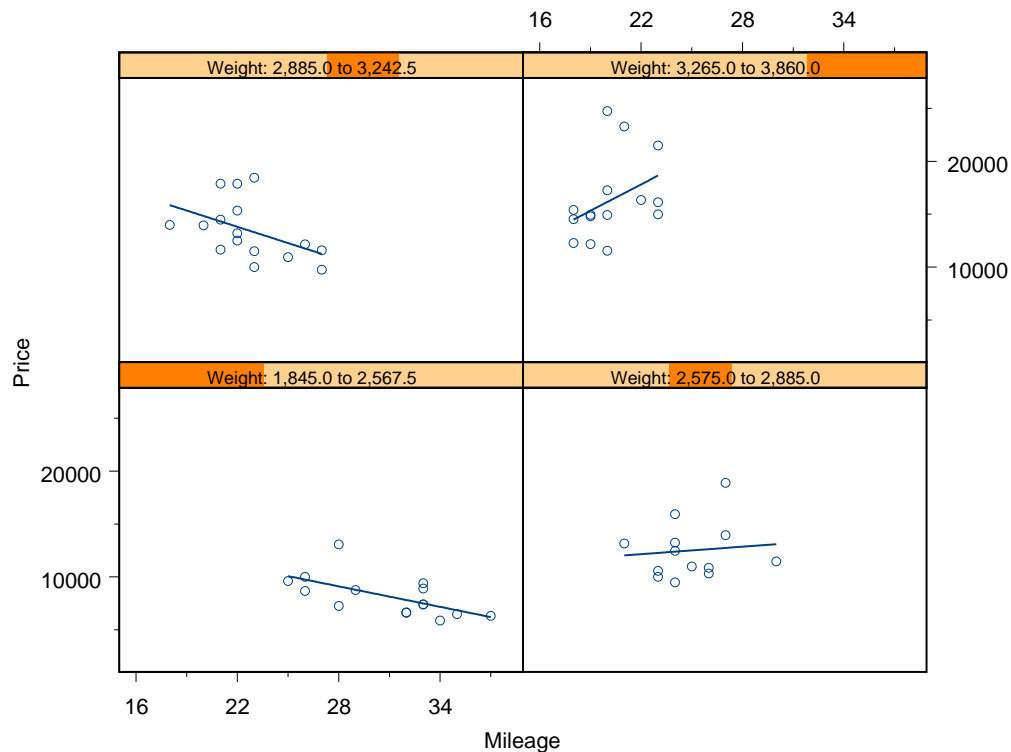


Figure 2.18

This is a *trellis plot*: it presents four graphs grouped by the size of the cars (measured by their weight). The variable `Weight` is split into quartiles. The plot in the lower left corner graphs `Mileage` against `Price` for those cars with weights in the lowest quartile; the plot in the upper right corner graphs `Mileage` against `Price` for those cars with weights in the top quartile.

All four plots are on the same scale, so we can see that the cars that are the biggest also cost the most and get the lowest mileage, and the smallest cars are also cheaper and have high gas mileage. The relationship between mileage and price shows up as much less important after taking the size of the cars into account.

See the section on “Trellis Graphs” in Chapter 3 of the *S-PLUS User’s Guide* for more detail.

Important: Toggle the `Set Conditioning Mode` button  to off.


End of Example 2.7

Command Line Notes

Trellis plots can be constructed using the `xyplot` command. The first argument is a formula: `y~x|cond`. The function `equal.count` breaks up a continuous variable into groups for the conditional plotting (the default is to use overlapping groups, unlike the GUI plot).

```
> xyplot(Price ~ Mileage | equal.count(Weight),
+       data = car.test.frame)
```

Remarks:




- To see another good example of correcting for lurking variables, download the SAT data mentioned at the beginning of Chapter 2 in IPS. First plot **SAT Math** (y) against **Teachers Pay** (x). Then redo the plot conditioning on **Percent Taking the SAT**.
- Other options for exploring two numeric variables in S-PLUS are bubble plots and color plots . See the online help for additional information.

2.5 Exercises

These problems use data sets supplied with S-PLUS. Recall the **Data > Select Data** command (Section 0.4.1).

1. Bring up **fuel.frame** and create a scatterplot matrix of all the numeric columns to look for relationships. Next, plot **Weight** on the x axis and **Mileage** on the y axis (**Mileage** against **Weight**) with different symbols for each **Type**.
2. What is the correlation between **Price** and **Mileage** in the S-PLUS supplied **cu.summary** data set? You should include the argument **na.method = "omit"** to omit the missing observations.
3. Find the regression line of **Mileage** (y) against **Weight** (x) from **fuel.frame**. Plot the residuals against **Weight** and include a smoothed line. Does the residual plot show any nonlinearity?

2.6 Solutions

1. From the GUI, select all columns except `Type` and click on the **Scatter Matrix** button . Select `Weight`, `Mileage`, and `Type` in that order and click on the **Scatter** button , then click on the **auto legend** button .

From the command line,

```
> pairs(fuel.frame[,1:4], panel = panel.smooth)
> xyplot(Mileage ~ Weight, data = fuel.frame,
+       groups = Type, panel = panel.superpose, pch = 1:6,
+       key = list(columns = 3, text = levels(fuel.frame$Type),
+       points = list(pch = 1:6)))
```

2. From the GUI,

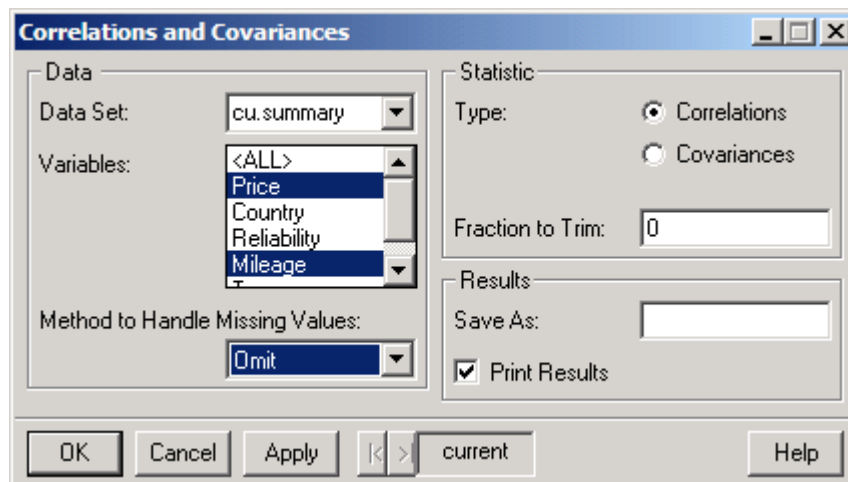


Figure 2.19

From the command line,

```
> attach(cu.summary)
> cor(Price, Mileage, na.method = "omit")
[1] -0.6537541
> detach()
```

3. From the GUI,

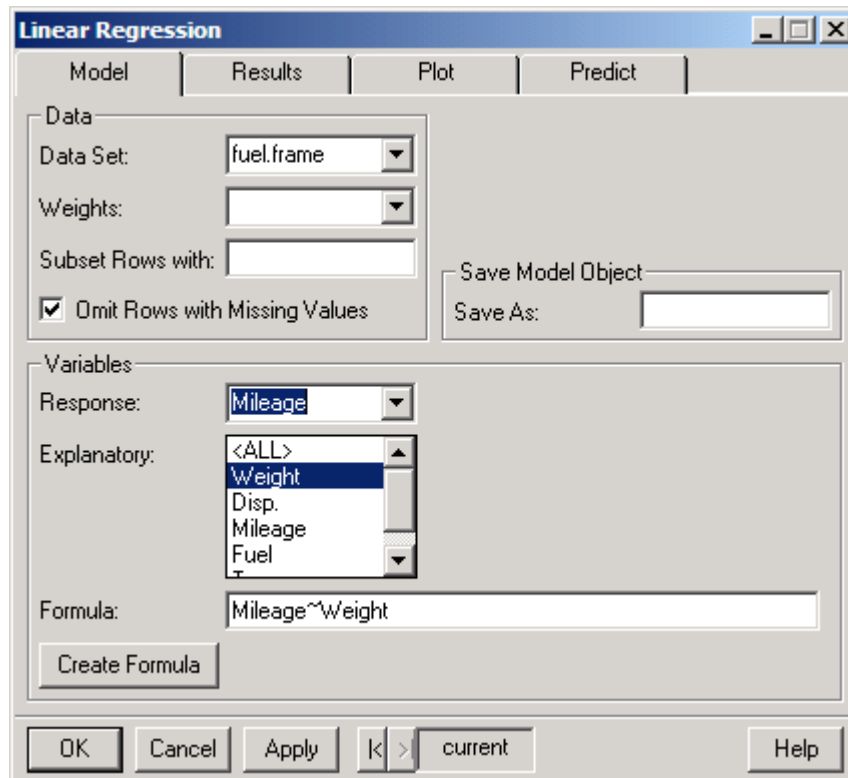


Figure 2.20

From the command line,

```
> fuel.fit = lm(Mileage ~ Weight, data = fuel.frame)
> summary(fuel.fit)
...
> scatter.smooth(fuel.frame$Weight, resid(fuel.fit))
> abline(h = 0)
```

There is definite curvature in the residuals.

Chapter 3

Producing Data

To obtain a **simple random sample (SRS)** of size n from a population, we need a method that gives every set of n individuals an equal chance of actually being selected.

Example 3.1 How to choose an SRS

We will use the built-in data set (`fuel.frame`) (Section 0.4.1) which has 60 observations. Suppose we wish to obtain an SRS of 10 cars.

1. From the main menu select `Data > Random Sample ...`.
2. In the `Random Sample of Rows` dialog box, for `Data Set`: select the name of the data set.
3. For `Sample Size`: type 10.
4. For `Save In`: type `fuel.sample`.

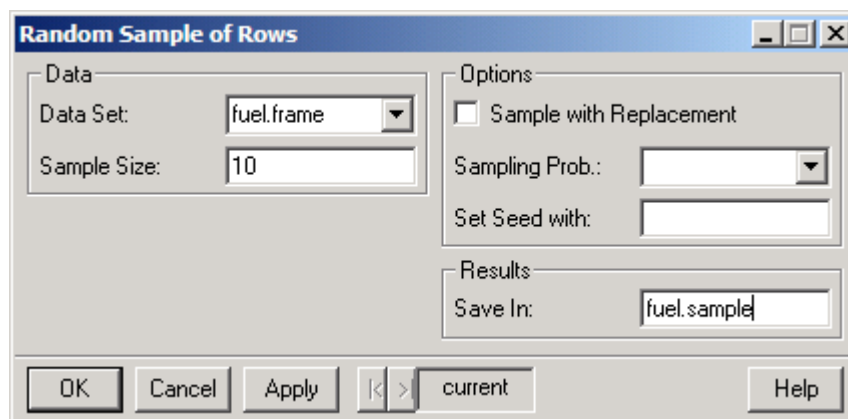


Figure 3.1

5. Click `OK`.

The result is a data frame with 10 rows from the original 60 rows and all columns of `fuel.frame`. Notice that the rows are not only a random sample, but that they are returned in random order. If you were doing an experiment, for example, trying one marketing strategy in five states and a different marketing strategy in another five states, then you could assign the first treatment to the first five states and the second treatment to the last five states. The states would be assigned randomly to the treatments.

End of Example 3.1

Remarks:

- Repeating this example will result in a different sample of size 10. To reproduce the same sample, you can specify a seed ([Set Seed:](#)), an integer between 0 and 1023 for the random number generator. Any time you specify the same sample size, population, and seed, the exact same sample results. This is useful so you (or someone else) can replicate your results.
- In the [Random Sample of Rows](#) dialog box, if you set the [Sample Size:](#) field equal to the number of rows in your data set (or equivalently, if you leave this field blank), then S-PLUS returns the entire data set but with the rows permuted in a random order.

In some cases, you may have a large data set created outside of S-PLUS and you would like to work on a random sample of this data in S-PLUS. Rather than importing the entire data set into S-PLUS first and then obtaining an SRS, you can import just a random sample of this data into S-PLUS.

For example, suppose you wish to survey a sample of students at your university, and your registrar has provided you with a file containing the names of the entire student body. You could import a random sample of this population into a data frame in S-PLUS and then enter the responses directly into the data frame as you interview this sample of students.

Example 3.2 Sampling an input file

The [workers](#) data set has 71076 rows. This data set is part of the IPSdata library, in two forms—already loaded into S-PLUS, and as a plain text file.

Here we import a random sample from the plain text file.

1. At the menu, select [File > Import Data...](#)
2. That brings up the [Import From File](#) dialog box, with the [Data Specs](#) tab active. Click on [Browse](#), and navigate to the location of [workers.txt](#). This is typically something like [c:/Program Files/Insightful/splus62/library/IPSdata/data/workers.txt](#)

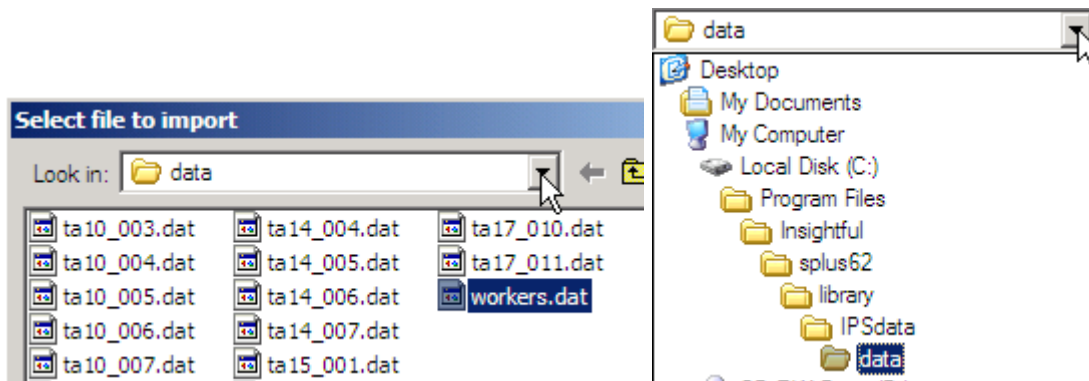


Figure 3.2

3. Under [To](#), change the [Data set:](#) field to [workers.sample](#).

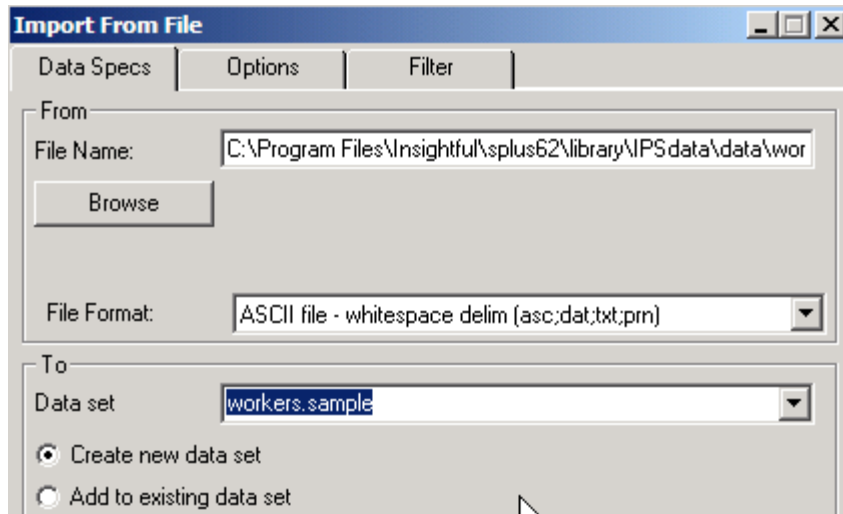


Figure 3.3

4. Click on the [Filter](#) tab.
5. Under [Filter columns](#), type `samp_fixed(100, 71076)` in the [Filter expression:](#) field.

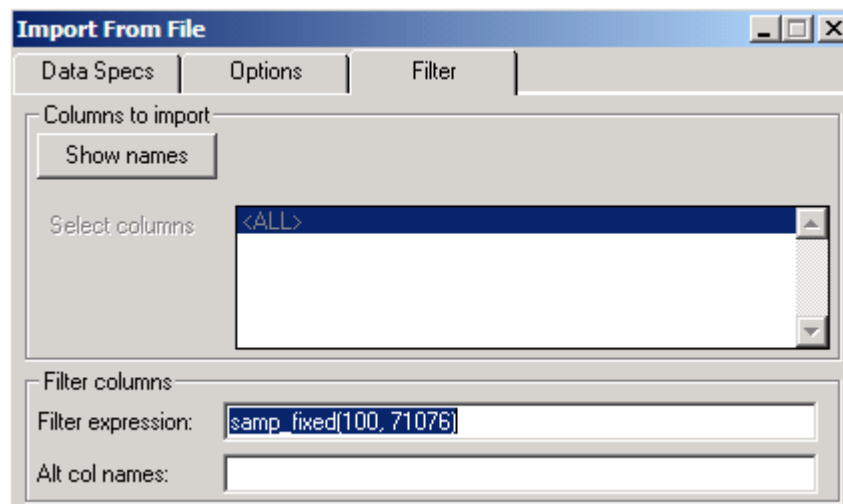


Figure 3.4

6. Click [OK](#).

End of Example 3.2

Command Line Notes

The function `sample` takes a vector for its argument as well as the sample size (unlike the GUI command which works on a data frame).

```
> attach(fuel.frame)
> sample(Weight, 10)
> sample(Weight, 10) # a different sample
> set.seed(4)
> sample(Weight, 10)
> set.seed(4)
> sample(Weight, 10) # the same sample
> detach()
> fuel.frame[sample(60, 10),]
```

3.1 Exercise

Create a new data set and enter the names of at least ten friends or classmates in the first column. Take a random sample of size five from the list.

3.2 Solution

From the GUI,

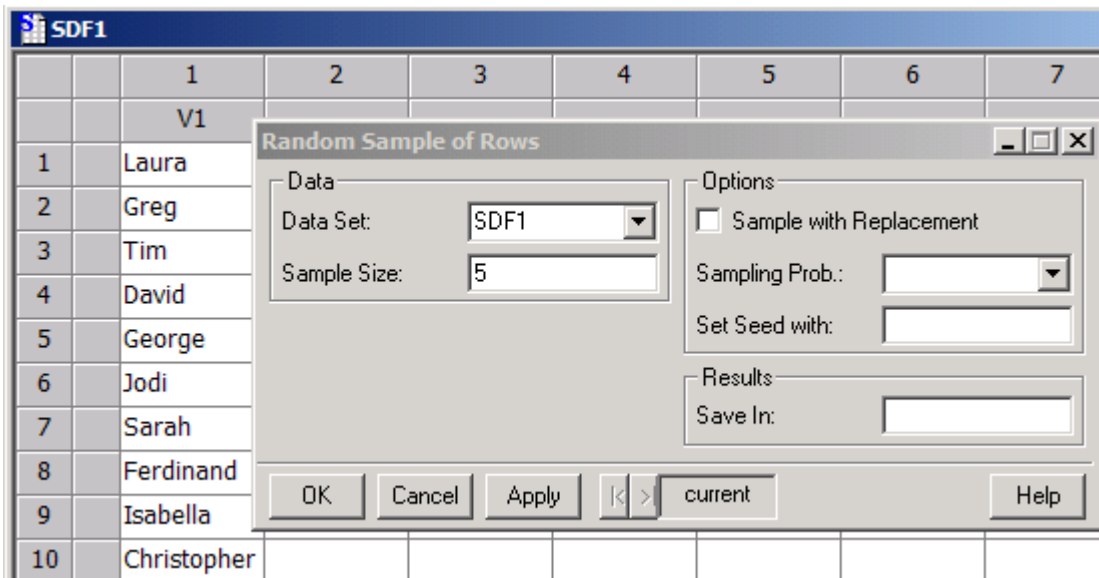


Figure 3.5

From the command line,

```
> temp = c("Laura", "Greg", "Tim", "David", "George", "Jodi",
+         "Sarah", "Ferdinand", "Isabella", "Christopher")
> sample(temp, 5)
[1] "Ferdinand" "Sarah"      "David"      "Jodi"       "Laura"
```

Chapter 4

Probability


4.1 Randomness

Chapter 4 in IPS introduces the axioms of probability and the Law of Large Numbers.

The Law of Large Numbers says roughly that as we take larger and larger samples from a fixed population, the sample means tend to get closer to the true population mean.

Example 4.1 Flipping a fair coin

To illustrate the Law of Large Numbers, we simulate the flipping of a fair coin a large number of times.

1. Create a new empty data set by clicking the  button on the main toolbar (Section 0.4.6).
2. From the main menu, select [Data > Random Numbers...](#)
3. For [Target Column](#): select [<END>](#).
4. Under [Sampling](#), for [Sample Size](#): type [100000](#), and for [Distribution](#): select [binomial](#).
For [Seed](#): type 12. (This step is optional; it should make your output match ours.)
5. Under [Distribution Parameters](#), for [Probability](#): type [.5](#), and for [Sample Size](#): type [1](#).

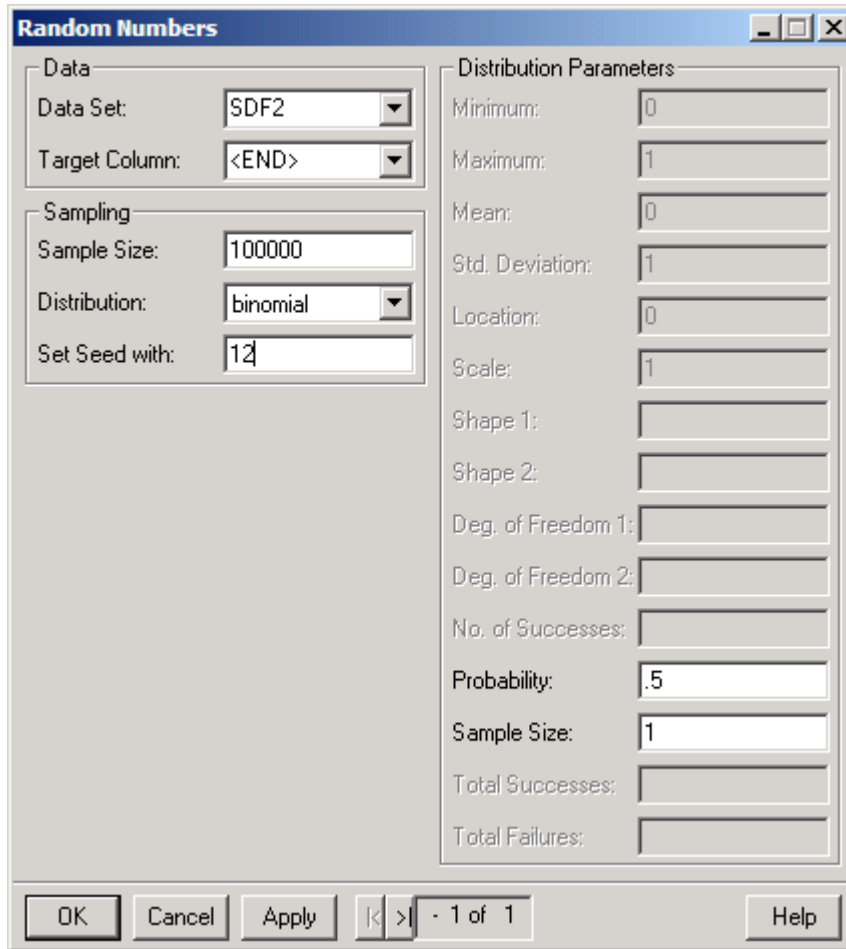


Figure 4.1

6. Click [OK](#).

SDF2		
		1
		V1
1		1.00
2		1.00
3		0.00
4		0.00
5		0.00
6		1.00
7		1.00

Figure 4.2

In the newly created column, each row gives the number of heads in one toss of a fair coin. This experiment was repeated 100000 times. For example, the first toss resulted in a head (1), the second toss resulted in a head (1), the third toss was not a head (0), and so on.

Now we want to convert those raw numbers into fractions. We start by creating a column that contains the numbers 1 to 100000.

- At the menu, select **Data > Fill...**
- In the **Fill Numeric Columns** dialog box, for **Data Set:** select the name of the data set, and for **Columns:** select **<END>**.
- Under **Fill Options**, for **Length:** type **100000**.

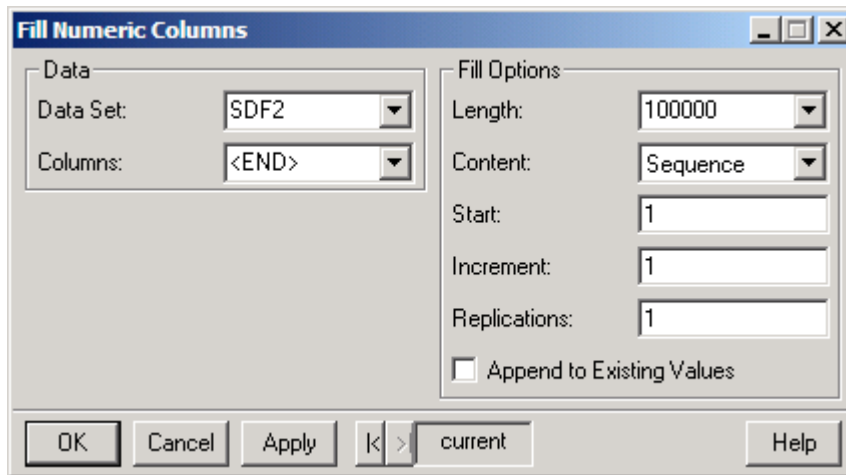


Figure 4.3

- Click **OK**.

We next create a column that contains, for each row, the proportion of heads up to that point.

- From the main menu, select **Data > Transform....** For **Data Set:** select the name of the data set. For **Target Column:** select **<END>**.
- For **Expression:** type **cumsum(V1)/V2**.

The command **cumsum(V1)** computes the cumulative sums of the entries in **V1**, that is, the number of heads so far.

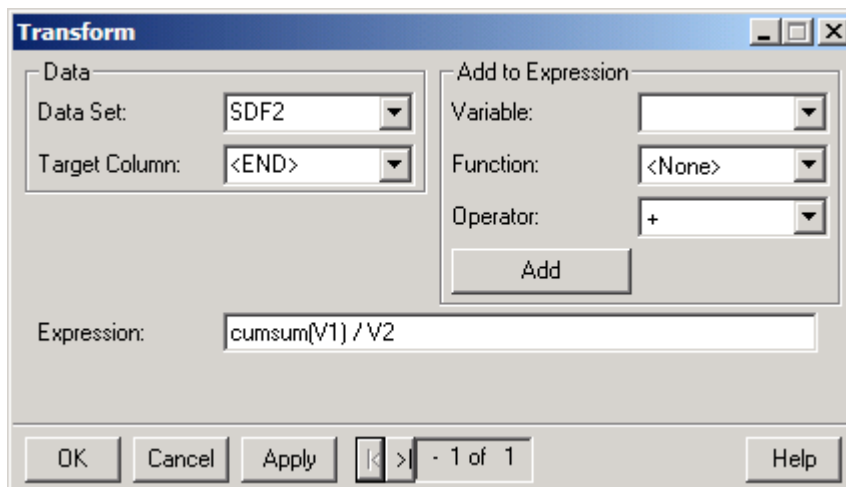


Figure 4.4

13. Click [OK](#).

SDF2				
		1	2	3
		V1	V2	V3
1		1.00	1	1.00
2		1.00	2	1.00
3		0.00	3	0.67
4		0.00	4	0.50
5		0.00	5	0.40
6		1.00	6	0.50
7		1.00	7	0.57

Figure 4.5

By the 7th flip (row 7), the proportion of heads up to this point is 0.57 (4 out of the 7 flips). We next plot this information.

14. Select [V2](#) for the x axis, CTRL-click on [V3](#) for the y axis.

15. On the [Plots 2D](#) palette, click on the [Line](#) plot button 

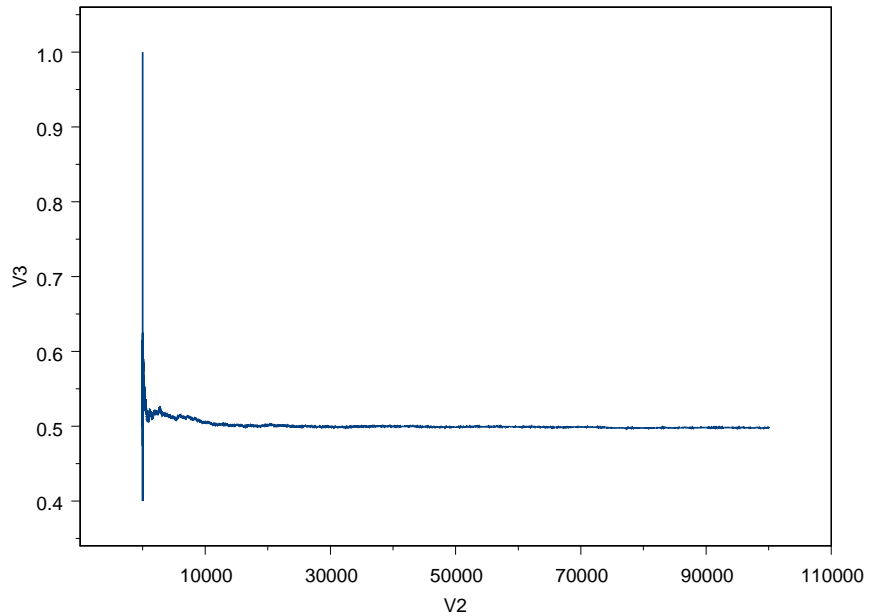




Figure 4.6

To get a better sense of what this plot is telling us, we change the x axis to a logarithmic scale.

16. Right-click on the x axis and select [Display/Scale](#) from the shortcut menu.
17. This brings up the [X Axis](#) dialog box, [Display/Scale](#) tab. In the [Axis Scaling](#) box, change [Scaling](#): to [Log](#).
18. Click [OK](#).
19. Open the [Annotations](#) palette using the  icon from the main toolbar (Section [1.4.2](#)) and click on the [Horizontal Reference Line](#) button .
20. Add the reference line by clicking at the $y = 0.5$ tick mark.
You can fine-tune by right-clicking on this reference line and choosing [Position](#) from the shortcut menu. Type [0.5](#) in the [Position](#): field.

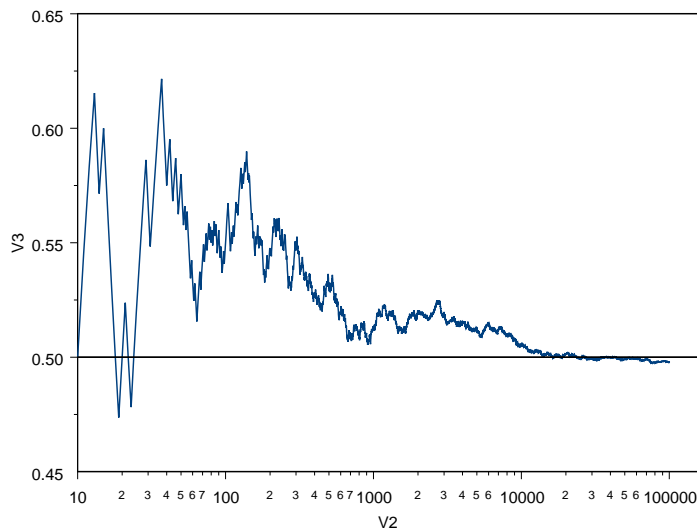
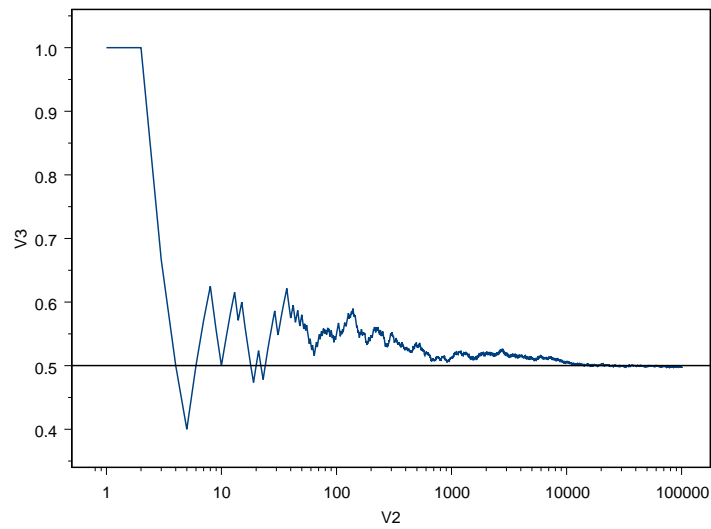


Figure 4.7

You can zoom in by right-clicking on the x axis and/or y axis and changing the axis ranges. Shown here is the result of omitting the first nine observations, and limiting the y axis range.

We see that as the number of coin flips increases, the proportion of heads approaches 0.5.

End of Example 4.1

4.2 Exercise

1. The procedures described in this chapter can be used for a variety of distributions other than the binomial. The F distribution comes up in the analysis of variance (IPS Chapter 12). The F distribution requires two parameters, the numerator degrees of freedom and the denominator degrees of freedom.

If the random variable X follows the F distribution with numerator degree of freedom equal to 41 and denominator degree of freedom equal to 32, find $P(1.5 \leq X \leq 2.4)$.

4.3 Solution

1. From the GUI,

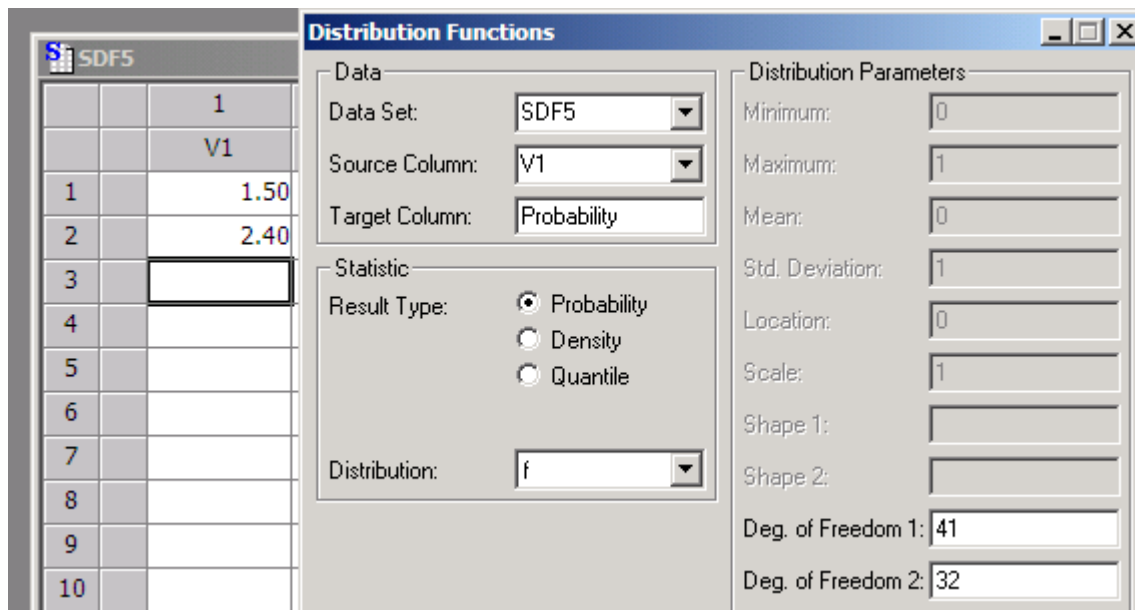


Figure 4.8

From the command line,

```
> pf(c(1.5, 2.4), 41, 32)
[1] 0.8809057 0.9939659
> .9939659 - .8809057
[1] 0.1130602
```



Chapter 5

Sampling Distributions

In Chapters 2 and 4 we saw how to compute values from the normal and other distributions. In Chapter 5 we learn about the *binomial* distribution. Distributions like the binomial, Poisson, and hypergeometric are called discrete distributions because they only apply to counts (0, 1, 2, 3, ...) instead of a continuous range of values like the normal distribution. This means that in addition to computing cumulative probabilities (for example, the probability of getting 5 or fewer heads out of 10 flips of a coin), we can also compute exact probabilities (for example, the probability of getting *exactly* 5 heads out of 10 flips).

Example 5.1 Flipping a biased coin

Suppose you have a biased coin that comes up heads 60% of the time instead of 50%. If you toss this coin 10 times, what is the probability of getting no heads? One head? Two heads, and so on?

1. Create a new data set using the  button and fill the first column with the integers from 0 to 10. Entries in this column represent the possible number of heads on 10 flips of the coin.
2. From the main menu, select [Data > Distribution Functions...](#)
3. For [Source Column](#): type the name of the first column. For [Target Column](#): leave the default name.
4. Under [Statistics](#), for [Result Type](#): select the [Probability](#) radio button. For [Distribution](#): select [binomial](#).
5. Under [Distribution Parameter](#), for [Probability](#): enter [0.6](#), and for [Sample Size](#): enter [10](#).

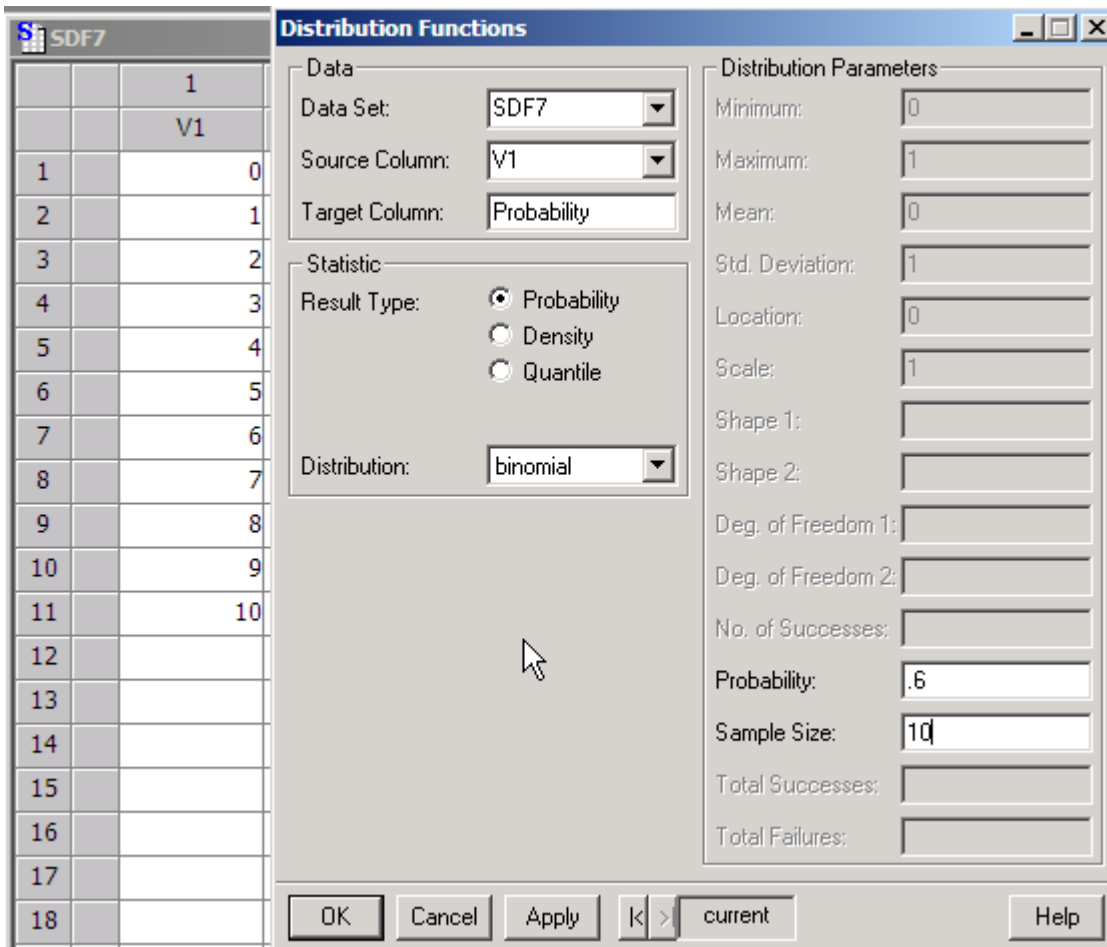


Figure 5.1


6. Click [Apply](#). (Recall that [Apply](#) executes the command but does not close the dialog box.)
7. Change the [Result Type](#): to [Density](#). Notice that the default name for [Target Column](#): becomes [Density](#).
8. Click [OK](#).

SDF7				
		1	2	3
		V1	Probability	Density
1		0	0.00	0.00
2		1	0.00	0.00
3		2	0.01	0.01
4		3	0.05	0.04
5		4	0.17	0.11
6		5	0.37	0.20
7		6	0.62	0.25
8		7	0.83	0.21
9		8	0.95	0.12
10		9	0.99	0.04
11		10	1.00	0.01

Figure 5.2

The **Probability** column gives cumulative probabilities and the **Density** column gives individual probabilities. For example, the probability of 7 heads or less is 0.83 and the probability of exactly 7 heads is 0.21

To avoid rounding the probabilities for no heads or 1 heads to zero, increase the precision (Section 0.4.7)

of the **Probability** and **Density** columns using  on the toolbar.

End of Example 5.1

Command Line Notes

The function `pbinom` gives the cumulative probabilities and `dbinom` gives the exact probabilities for the binomial distribution.

```
> pbinom(7, size = 10, prob = .6) # P(X <= 7)
[1] 0.8327102
> dbinom(7, size = 10, prob = .6) # P(X == 7)
[1] 0.2149908
```

To reproduce the output in the GUI example, type:

```
> cbind(V1 = 0:10, Probability = pbinom(0:10, 10, .6),
+       Density = dbinom(0:10, 10, .6))
      V1 Probability      Density
[1,]  0 0.0001048576 0.0001048576
[2,]  1 0.0016777216 0.0015728640
...
[10,]  9 0.9939533824 0.0403107840
[11,] 10 1.0000000000 0.0060466176
```

The column names are optional.

5.1 Sampling Distributions

It can be useful to see the effects of a sampling distribution by generating your own random numbers.

Example 5.2 Generating a sampling distribution

We will generate several columns of data from a known distribution, compute the sample means, and see how the distribution of the sample means compares to the population.

1. Create a new data set.
2. At the menu, select [Data > Random Numbers...](#)
3. In the [Random Numbers](#) dialog box, for [Data Set](#): select the name of the data set. For [Target Column](#): select [<END>](#).
4. Under [Sampling](#), for [Sample Size](#): type [1000](#), and for [Distribution](#): select [normal](#).
5. Under [Distribution Parameters](#), for [Mean](#): type [100](#), and for [Std. Deviation](#): type [10](#).
6. Click on [Apply](#) four times, then click on [Cancel](#).

This creates four columns of random numbers, each column coming from a normal distribution with mean 100, standard deviation 10.

Next, we create a fifth column whose entries are the average of the four columns.

7. At the menu, select [Data > Transform...](#)
8. In the [Data Transform](#) dialog box, for [Data Set](#): select the name of the data set. For [Target Column](#): select [<END>](#).
9. For [Expression](#): type $(V1+V2+V3+V4)/4$.

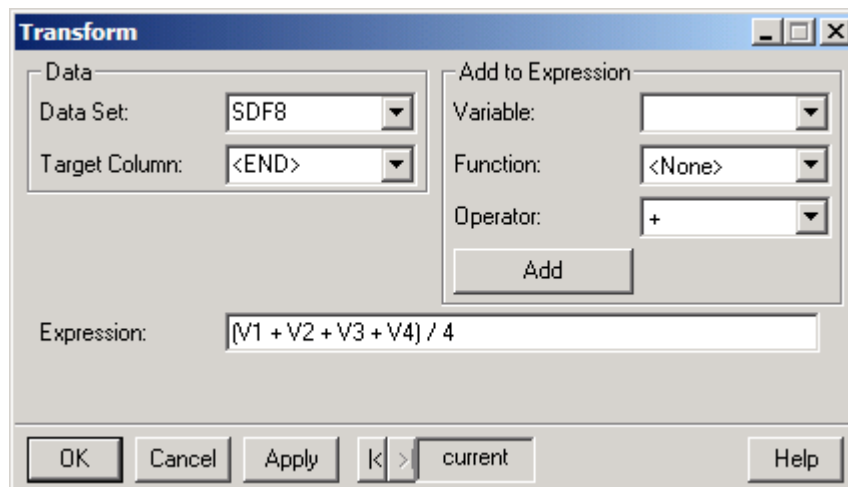



Figure 5.3

10. Click [OK](#).
11. In the data set, select all five columns by clicking on the first column header, then shift-clicking on the last column header.

12. On the [Plots 2D](#) palette, click on the [Density](#) plot button .

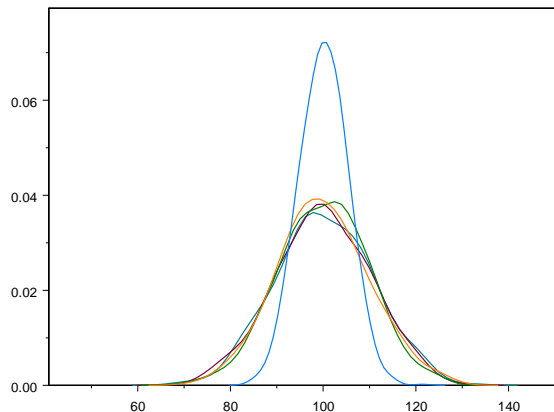


Figure 5.4

This draws five density plots, one for each column. The height (y -coordinate) for each plot is determined by the relative concentration of values from one column of data, so this plot provides information similar to the histogram. Notice that one plot is much narrower (smaller standard deviation) and taller than the others. This corresponds to the plot from [V5](#), the average of the first four columns. We know from the theory that averages have less variability than individuals.

Repeat this example but try some of the other distributions available in S-PLUS. For instance, the gamma distribution with shape parameter (mean) 3 will show the sampling distribution to be narrower and more symmetric.

End of Example 5.2

We have already shown ([Section 1.3](#)) how to use S-PLUS to compute normal probabilities. S-PLUS also computes probabilities for other distributions, including the binomial, t , chi-square, and F distributions (all introduced in IPS in later chapters).

Example 5.3 The t distribution

The t distribution is important in inference when the population follows a normal distribution with unknown mean μ and standard deviation σ .

Random variables that follow a t distribution must also have a parameter, the *degrees of freedom*, to complete its description.

If the random variable T follows a t distribution with 43 degrees of freedom, find the probability that T is less than 0.75, and find the probability that T is greater than 3.1; that is, find $P(T \leq 0.75)$ and $P(T > 3.1)$.

1. Create a data set with .75 and 3.1 entered into the first column.
2. At the menu, select [Data > Distribution Functions...](#)
3. In the [Distribution Functions](#) dialog box, for [Data Set](#): select the name of the data set. For [Source Column](#): select the name of the column.
4. For [Target Column](#): type a name for a new column (the default is [Probability](#)). The probabilities are saved to a newly created column.

- Under **Result Type**, check the radio button next to **Probability**. For **Distribution**: select **t**.
- For **Deg. of Freedom 1**: type **43**.

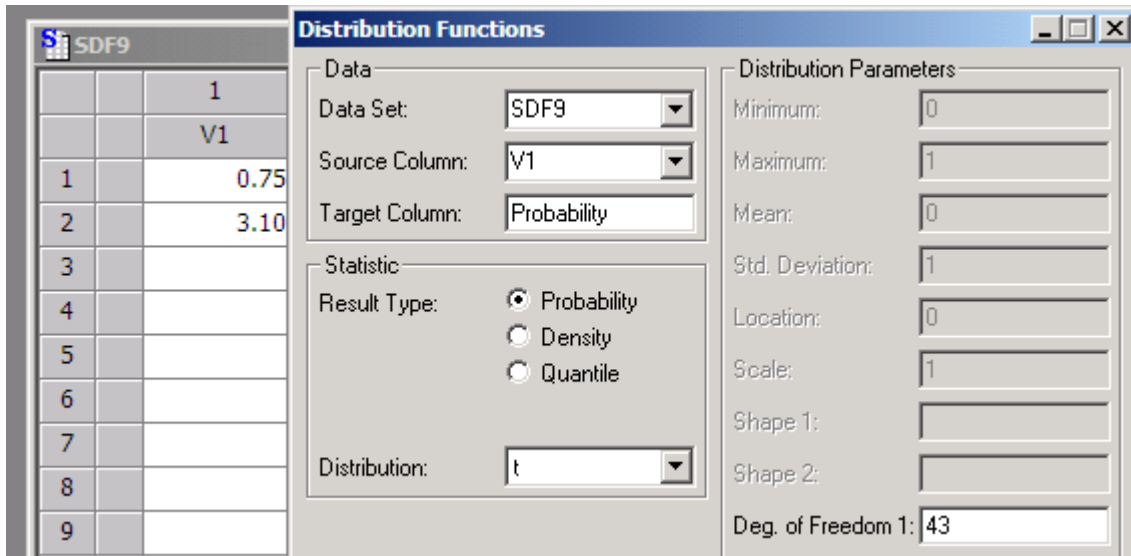



Figure 5.5

- Click **OK**.

S-PLUS gives us $P(T \leq .75) = 0.77$ and $P(T \leq 3.1) = 1$. Increase the precision of the **Probability** column to 4 by selecting the column and clicking on the **Increase Precision** button on the data set toolbar  (Section 0.4.7). We then see $P(T \leq .75) = 0.7713$ and $P(T \leq 3.1) = 0.9983$, so $P(T > 3.1) = 1 - 0.9983 = 0.0017$.

End of Example 5.3

Command Line Notes

Functions to generate random numbers start with the letter “r” and are followed by the (abbreviated) name of the distribution (**rnorm**, **rt**, **rbinom**, **rf**). The same names prefaced with “p” give the probabilities (**pnorm**, **pt**, **pbinom**, **pf**).

```
> coin = cumsum(rbinom(100000, 1, .5)) / (1:100000)
> plot(1:100000, coin, log="x", type="l")
> abline(h=.5, lty=4)
> samp.gamma = matrix(rgamma(9*10000, 3), ncol=9)
> m.gamma = rowMeans(samp.gamma)
> d1 = density(samp.gamma)
> d2 = density(m.gamma)
> plot(d1, ylim = range(d2$y), type="l")
> lines(d2, lty=4)
> abline(v=3, lty=2)
> pt(c(0.75, 3.1), 43)
[1] 0.7713306 0.9982958
```

Chapter 6

Introduction to Inference

Chapter 6 introduces confidence intervals and hypothesis tests. In this chapter, we make the assumption that the population follows a normal distribution with unknown mean μ but *known* standard deviation σ .

6.1 Confidence Intervals and Hypothesis Tests

If we obtain an SRS of size n with sample mean \bar{x} from a normal population, $N(\mu, \sigma)$, then a C level confidence interval is given by

$$\bar{x} \pm z^* \frac{\sigma}{\sqrt{n}},$$

where z^* is the z -critical value, the value such that $C \times 100\%$ of the area under the normal density curve occurs between $-z^*$ and z^* .

We give the critical values for the most common confidence levels.

Table 6.1 Critical values

$C \times 100\%$	$(1 - C)/2$ (upper tail area)	z^*
90	0.05	1.645
95	0.025	1.960
99	0.005	2.560

To perform a hypothesis test with $H_0 : \mu = \mu_0$, we form the z -test statistic,

$$z = \frac{\bar{x} - \mu_0}{\sigma/\sqrt{n}}$$

which will follow the standard normal distribution $N(0, 1)$ under the null hypothesis.

S-PLUS does not have a command for computing confidence intervals or the z -test statistic based on a given known population standard deviation. You can use S-PLUS to find the mean of a set of data and then calculate the confidence interval or test statistic by hand (see the Command Line Notes).

Example 6.1 Finding a confidence interval

Suppose you weigh yourself once a week for 10 weeks and record the following measurements:

190.5 189.0 195.5 187 189 190 190.5 192.5 191 189.5

Assume that your weight follows a normal distribution with standard deviation $\sigma = 3$. To compute a 90% confidence interval, we need to find the mean of these ten numbers.

1. Create a new data set using the  button and enter the data (Section 0.4.6).

- At the menu, select **Statistics > Data Summaries > Summary Statistics...**
- On the **Data** tab, for **Data Set:** enter select the name of the data set, and for **Variables:** select the column that holds the data.
- On the **Statistics** tab, select **Mean** and clear the other options.

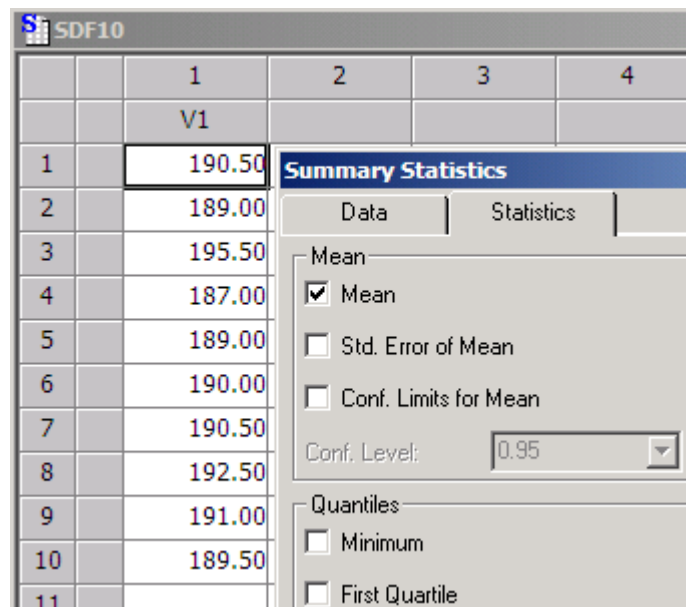


Figure 6.1

- Click **OK**.

```

*** Summary Statistics for data in: SDF1 ***

weight
Mean: 190.45

```

To compute a 90% confidence interval for your true mean weight, using Table 6.1 (or Section 1.3) to note the z-critical value 1.645; then calculate $190.45 \pm 1.645 \cdot \frac{3}{\sqrt{10}} = (188.9, 192.0)$.

End of Example 6.1

```

Command Line Notes

> weight = c(190.5, 189, 195.5, 187, 189, 190, 190.5
+           192.5, 191, 189.5)
> mean(weight)
[1] 190.45
> 190.45 + c(-1,1) * 1.645 * 3/sqrt(10) # CI
[1] 188.8894 192.0106
> round(.Last.value, 1) # Round to 1 decimal digit
[1] 188.9 192.0

```


Example 6.2 Two-sided hypothesis test

Continuing with the data from Example 6.1, suppose your drivers's license states that your weight is 192. We wish to test the hypothesis that these observations are compatible with a mean $\mu = 192$:

$$H_0 : \mu = 192 \text{ against } H_a : \mu \neq 192$$

Use a calculator or the S-PLUS command line to compute the z-statistic:

$$z = \frac{\bar{x} - \mu}{\sigma/\sqrt{n}} = \frac{190.45 - 192}{3/\sqrt{10}} = -1.633$$

Recalling Section 1.3 or Section 4.1, we can use S-PLUS to compute the probability $P(Z \leq -1.633)$ where Z has a $N(0, 1)$ distribution. This probability is 0.051 and thus, the P -value is $2 \cdot 0.051 = 0.102$, which does not give strong evidence that your true mean weight differs from 192 pounds.

End of Example 6.2

6.2 Power

The *power* of a significance test is a measure of its ability to detect an alternative hypothesis. It is the probability of rejecting the null hypothesis given that the alternative hypothesis is true.

In order to compute the power of a significance test, we need to know the hypothesized mean μ_0 , the alternative hypothesis, the standard deviation σ , the significance level α , and the sample size.

Example 6.3 Power

Researchers measure the percent change in total body bone mineral content of young women. They assume that $\sigma = 2$ and test the hypothesis $H_0 : \mu = 0$ against $H_a : \mu > 0$. Find the power of the test if the alternative mean is $\mu = 1$, 25 subjects are used, and $\alpha = 0.05$.

1. At the menu, select **Statistics > Power and Sample Size > Normal Mean...**
2. For **Compute:** check **Power** and for **Sample Type:** select **One Sample**.
3. Under **Probabilities**, for **Alpha(s):** type 0.05.
4. Under **Sample Sizes**, for **N1:** type 25.
5. Under **Standard Deviations**, for **Sigma1:** type 2.
6. Under **Null Hypothesis**, for **Mean:** type 0.
7. Under **Alternative Hypothesis**, for **Alt. Mean** type 1 and for **Test Type** select **greater**.

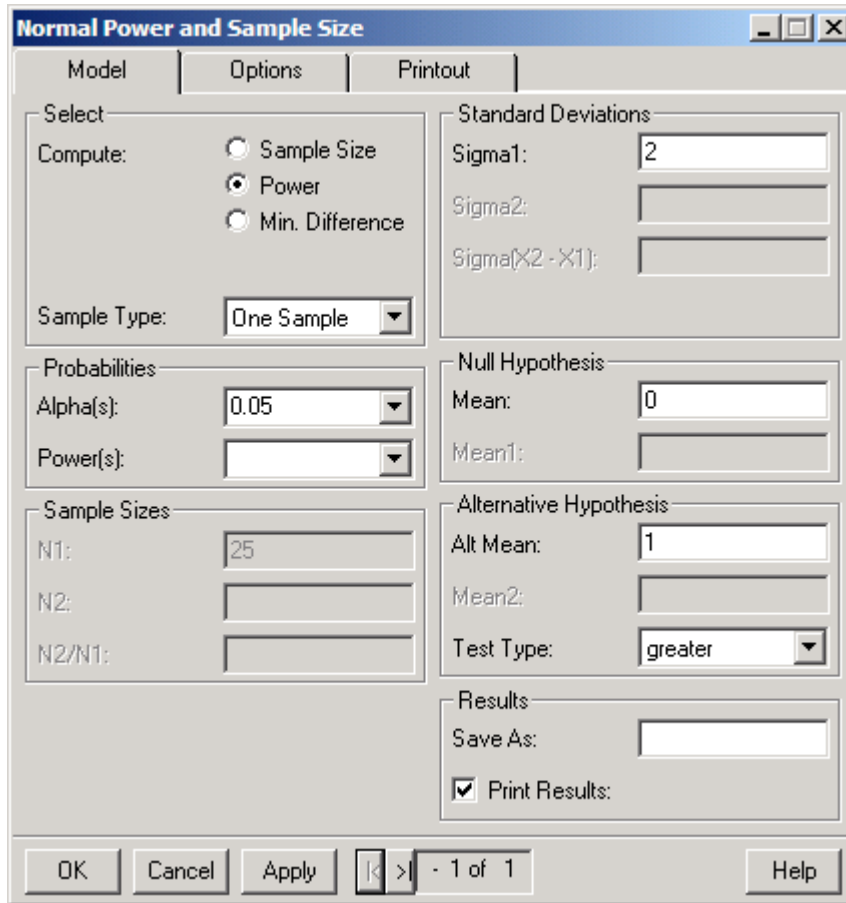


Figure 6.2

8. Click [OK](#).

```

*** Power Table ***
mean.null sd1 mean.alt delta alpha    power n1
1         0  2         1    1 0.05 0.8037649 25

```

The power of the test is 0.80.

End of Example 6.3

Command Line Notes

The `normal.sample.size` command computes the power of a test.

```

> normal.sample.size(mean = 0, mean.alt = 1, sd1 = 2,
+                    n1 = 25, alpha = .05)
+                    alternative = "greater")
mean.null sd1 mean.alt delta alpha    power n1
1         0  2         1    1 0.05 0.8037649 25

```

6.3 Exercises

1. A runner checks her pulse immediately upon awakening on seven randomly chosen mornings. She records

56 58 52 53 58 52 56

in beats per minute. Assume that the standard deviation is known to be $\sigma = 2$ beats per minute. Find a 90% confidence interval for the true mean pulse rate of this runner.

2. Suppose the runner in the above example wishes to test the hypothesis that $H_0 : \mu = 53$ against $H_0 : \mu > 53$. What is the power of the test if the alternative mean is 56 beats per minute? (Use $\alpha = .01$.)

6.4 Solutions

1. From the command line,

```
> pulse = c(56, 58, 52, 53, 58, 52, 56)
> mean(pulse) + c(-1,1) * 1.645 * (1/sqrt(7))
[1] 54.37825 55.62175
```

2. From the GUI,

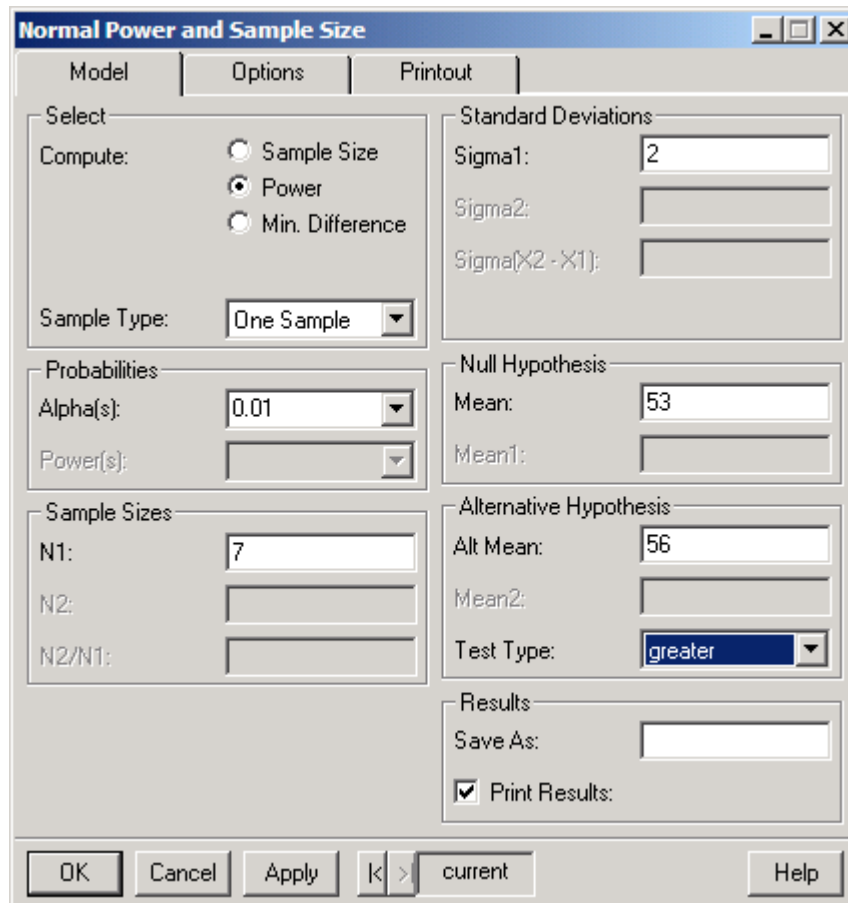


Figure 6.3

From the command line,

```
> normal.sample.size(mean = 53, mean.alt = 56,
+                   sd1 = 2, n1 = 7, alpha = .01,
+                   alternative = "greater")
*** Power Table ***
mean.null sd1 mean.alt delta alpha   power n1
1      53   2      56     3 0.01 0.9497339 7
```

Chapter 7

Inference for Distributions

This chapter introduces the t distribution and its use in finding confidence intervals and performing significance tests for one or two population means. The t distribution is necessary when samples arise from normal populations with unknown standard deviation σ .

We encourage you to use Chapter 14 together with this chapter, and in particular Section 14.1. The approach taken there, based on computer simulation and visualization, complements the more mathematical approach taken here, and may help you better understand the concepts in this chapter. It also provides a way for you to check your work.

7.1 Inference for the Mean of a Population

7.1.1 Confidence intervals

Suppose we have a normal population with unknown mean μ and standard deviation σ . Draw a simple random sample of size n and compute the sample mean \bar{x} and sample standard deviation s . A level C confidence interval for μ is

$$\bar{x} \pm t^* \frac{s}{\sqrt{n}}$$

where t^* is the upper $(1 - C)/2$ critical value for $t(n - 1)$, the t distribution with $n - 1$ degrees of freedom.

Example 7.1 Corn soy blend (CSB)—IPS Example 7.1

Researchers wished to evaluate appropriate vitamin C levels (mg/100gm) in a random sample of eight batches of CSB from a production run:

26 31 23 22 11 22 14 31

Find a 95% confidence interval for the mean vitamin C content of CSB produced during this run.

1. Open the [Example7.1](#) data set from the `IPSdata` library (Section 0.4.2).
2. At the menu, select [Statistics > Data Summaries > Summary Statistics...](#)
3. In [Data Set:](#) enter [Example7.1](#). In [Variables:](#) select `vitc`.
4. On the [Statistics](#) tab, check [Mean](#), [Conf. Limits for Mean](#), and [Std. Deviation](#).
5. The default confidence level is 95%. If you want a different confidence level, type this value in the [Conf. Level:](#) field.

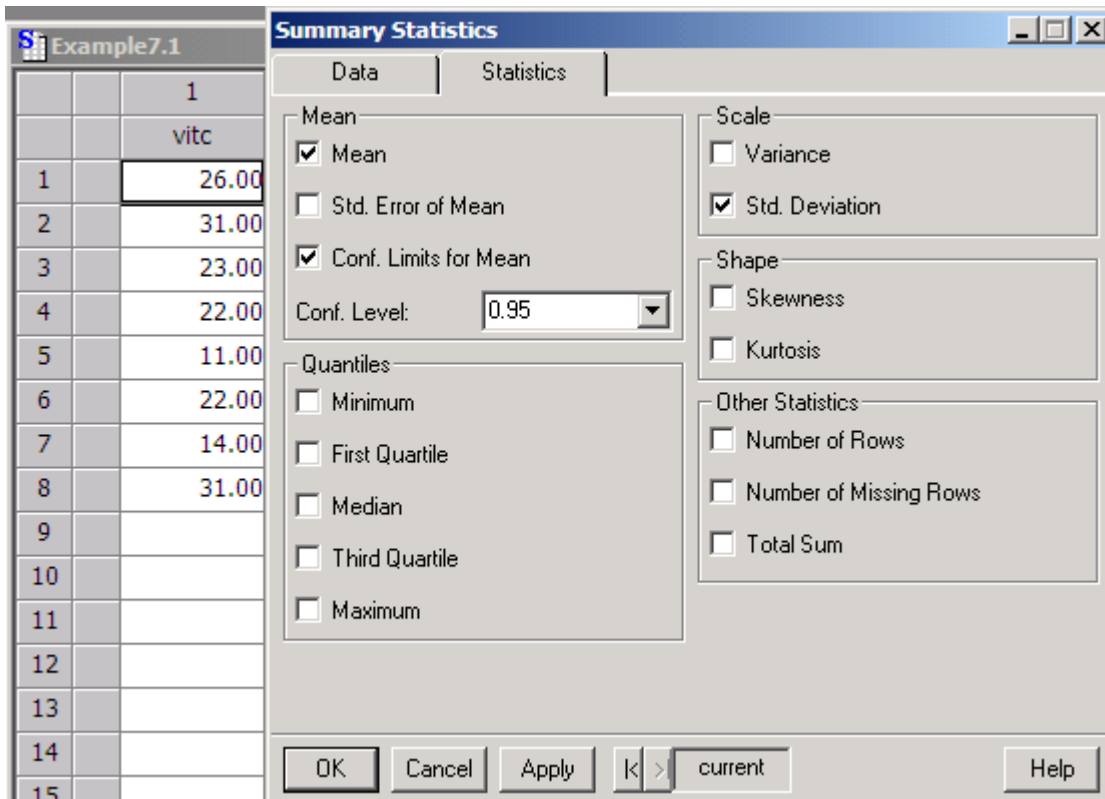


Figure 7.1

6. Click [OK](#).

```

*** Summary Statistics for data in: Example7.1 ***

          vitc
      Mean: 22.500000
  Std Dev.:  7.191265
  LCL Mean: 16.487952
  UCL Mean: 28.512048

```

Thus, we see a wide 95% confidence interval of (16.49, 28.51). The interval is found by computing $22.50 \pm t^* \frac{7.19}{\sqrt{8}}$, where t^* is the upper 0.025 critical value on seven degrees of freedom.

End of Example 7.1

We just obtained confidence intervals using the [Summary Statistics](#) menu. We can also obtain them from the same menu we use for hypothesis tests, the [One Sample > t Test...](#) menu.

7.1.2 Hypothesis tests

To test the hypothesis $H_0 : \mu = \mu_0$, compute the one-sample t statistic

$$t = \frac{\bar{x} - \mu_0}{s/\sqrt{n}}.$$

This test statistic has a $t(n - 1)$ distribution under the null hypothesis if the population is normal. In S-PLUS, the t test procedure performs a hypothesis test and computes a confidence interval.

Example 7.2 Corn soy blend, continued

Suppose we wish to test the hypothesis that the mean vitamin C content for the production run in Example 7.1 is 40mg/100g: $H_0 : \mu = 40$ against $H_a : \mu \neq 0$.

1. At the menu, select **Statistics > Compare Samples > One Sample > t Test...**
2. In the **Data** box, for **Data Set** enter **Example7.1**, and for **Variable:** select **vitc**.
3. In the **Hypotheses** box, for **Mean Under the Null Hypothesis:** enter **40** and for **Alternative Hypothesis:** select **two.sided**.

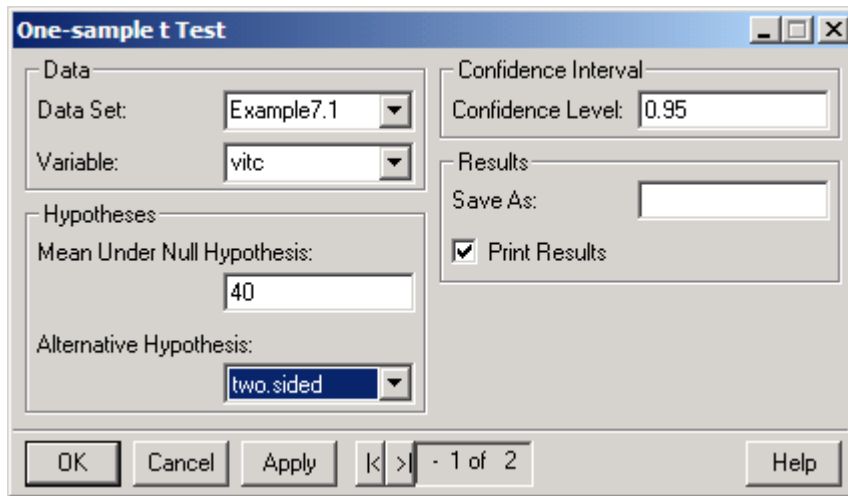


Figure 7.2

4. Click **OK**.

```

One-sample t-Test

data: vitc in Example7.1
t = -6.883, df = 7, p-value = 0.0002
alternative hypothesis: mean is not equal to 40
95 percent confidence interval:
 16.48795 28.51205
sample estimates:
mean of x
 22.5

```

With a P -value of 0.0002, we reject the null hypothesis at the $\alpha = 0.05$ significance level. Notice that in the t test report, S-PLUS states the alternative hypothesis associated with this particular test. It does not mean that this is the conclusion that S-PLUS is drawing.

To perform a one-sided test, repeat the above steps, but in step 3 select **less** for **Alternative Hypothesis:**.

```
One-sample t-Test

data: vitc in Example7.1
t = -6.883, df = 7, p-value = 0.0001
alternative hypothesis: true mean is less than 40
95 percent confidence interval:
    NA 27.31696
sample estimates:
mean of x
    22.5
```

In the case of a one-sided alternative hypothesis, S-PLUS returns a one-sided confidence interval.

End of Example 7.2

```
Command Line Notes

The function t.test performs the  $t$  test and computes a confidence interval. For a one sample test, it requires a vector for the first argument.

> t.test(Exercise7.1$vitc, mu = 40)

By default, t.test assumes a .95 confidence level, a null hypothesis of  $\mu = 0$ , and a two-sided test (the command t.test(Exercise7.1$vitc) would invoke these defaults). The options conf.level, mu, and alternative can be used to change these defaults.

For example, to compute a 99% confidence interval or to test  $H_0 : \mu = 35$  against  $H_a : \mu < 35$ , type

> t.test(Exercise7.1$vitc, mu = 35, conf.level = .99,
        alternative = "less")
```

7.1.3 Matched pairs t procedures

In a matched pairs design, subjects are matched in pairs, and each treatment is given to one subject in each pair. In many cases, the same person is used for each pair and the responses are before-and-after measurements. The one-sample t procedure is applied to the observed differences in the responses.

Example 7.3 French language test

Twenty French teachers took a language test before and after an intense four-week summer French course. Did the course improve the test scores of these teachers?

1. Open the `Exercise7.41` data set from the `IPSdata` library.

Let μ denote the true mean of the difference in response times (`pre-post`). We wish to test the hypothesis $H_0 : \mu = 0$ against $H_0 : \mu > 0$.

2. At the menu, select `Statistics > Compare Samples > Two Samples > t Test...`
3. For `Data Set:` enter `Exercise7.41`.
4. For `Variable 1:` select `post`, and for `Variable 2:` select `pre`.

- For **Type of t Test** select **Paired t**.
- In the **Hypotheses** box, for **Mean Under Null Hypothesis**: type **0** and for **Alternative Hypothesis**: select **greater**.

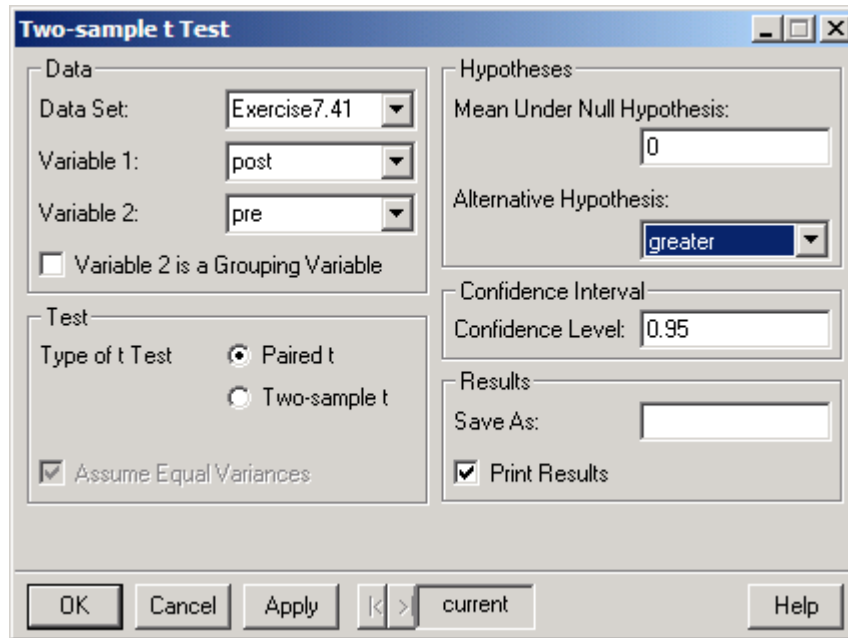


Figure 7.3

- Click **OK**.

```

Paired t-Test

data:  x: post in Exercise7.41 , and y: pre in Exercise7.41
t = 3.8649, df = 19, p-value = 0.0005
alternative hypothesis: mean of differences is greater than 0
95 percent confidence interval:
 1.381502      NA
sample estimates:
mean of x - y
           2.5

```

With $t = 3.86$ and $P < 0.01$, we conclude the improvement in scores was significant. S-PLUS returns a one-sided 95% confidence interval $(1.38, \infty)$.

Remark: In the data set provided (**Exercise7.41**), the difference in scores (**gain**) was also present in the data set. Hence you could then do this problem as a one-sample t test, using the menu (**Statistics > Compare Samples > One sample > t test...**) for the variable **gain**.

In other situations you can use the **Transform** menu to compute differences. Select the **Data > Transform...** menu; for **Target Column**: field, type a name for the new column to store differences (for this example **gain**). For **Expression**:, type the algebraic expression for the difference, for example, **post - pre**.

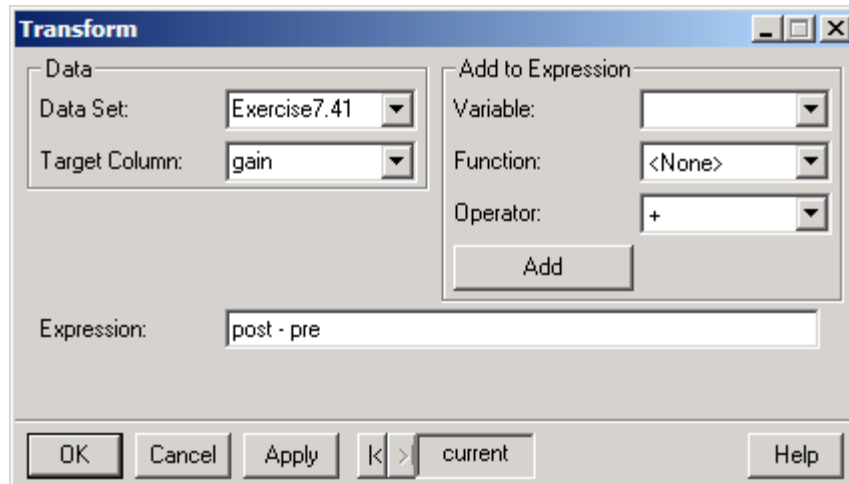


Figure 7.4

End of Example 7.3

Command Line Notes

The `t.test` command can be used for the matched-pairs t test.

```
> attach(Exercise7.41)
```

```
> t.test(post, pre, paired = T, alternative = "greater")
```

```
> detach()
```

7.1.4 Inference for nonnormal populations

See Chapters 14 and 15 for ways to handle nonnormal populations.

7.2 Comparing Two Means

We next consider *two-sample* problems in which we wish to compare the means of two independent, normal populations. Draw a sample of size n_1 from a normal population with true mean μ_1 , and a sample of size n_2 from another normal population with unknown μ_2 . Let \bar{x}_1 and \bar{x}_2 denote the sample means, and s_1 and s_2 denote the sample standard deviations. To test the hypothesis $H_0 : \mu_1 - \mu_2 = 0$, we form the two-sample t statistic

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\text{standard error}}$$

and use P -values or critical values for a t distribution.

A confidence interval for the true mean difference $\mu_1 - \mu_2$ is given by

$$(\bar{x}_1 - \bar{x}_2) \pm t^* \cdot \text{standard error}$$

If the population standard deviations are assumed to be equal, $\sigma_1 = \sigma_2$, then

$$\text{standard error} = s_p \sqrt{\frac{1}{n_1} + \frac{1}{n_2}}$$

where s_p^2 denotes the pooled variance

$$s_p^2 = \frac{((n_1 - 1)s_1^2 + (n_2 - 1)s_2^2)}{n_1 + n_2 - 2}$$

and the degrees of freedom for the t distribution is $n_1 + n_2 - 2$.

If the population standard deviations are not assumed to be equal, $\sigma_1 \neq \sigma_2$, then

$$\text{standard error} = \sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}$$

and the degrees of freedom is given by

$$\text{df} = \frac{\left(\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}\right)^2}{\frac{1}{n_1 - 1} \left(\frac{s_1^2}{n_1}\right)^2 + \frac{1}{n_2 - 1} \left(\frac{s_2^2}{n_2}\right)^2}.$$

Example 7.4 Degree of reading power (IPS Example 7.14, Table 7.4)

In an experiment to determine whether certain reading activities help school children improve reading ability, a group of 21 students receive a treatment—the reading activities—while a group of 23 students follows their usual curriculum.

Let μ_c and μ_t denote the true mean reading score for the control group and the treatment group, respectively. We wish to test the hypothesis $H_0 : \mu_t - \mu_c = 0$ against $H_a : \mu_t - \mu_c > 0$.

1. Open the data set [Table7.4](#) from the `IPSdata` library.

Note that columns one and two give the same information: whether a particular individual belonged to the treatment group or control group. Column two (a factor, see [Section 0.4.5](#)) uses character strings to identify the two levels while column three uses an integer coding.

2. At the menu, select [Statistics > Compare Samples > Two Samples > t Test ...](#)
3. For [Data Set](#): enter [Table7.4](#).
4. For [Variable 1](#): select `drp`, and for [Variable 2](#): select `group`. Check [Variable 2 is a Grouping Variable](#).
5. In the [Test](#) box, make sure the box [Equal Variances](#) is not checked.
6. In the [Hypotheses](#) box, for [Mean Under Null Hypothesis](#): keep the default 0 and select `less` for [Alternative Hypothesis](#).

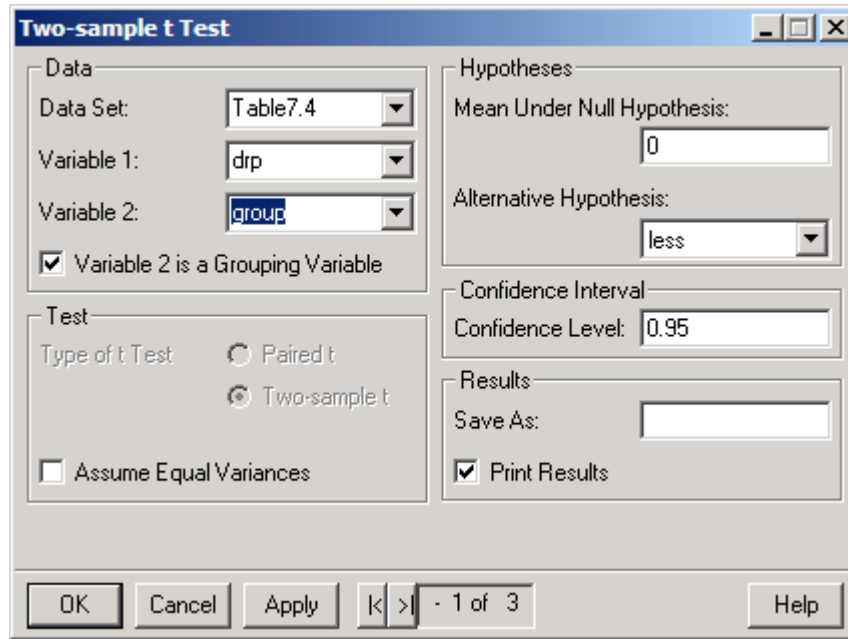


Figure 7.5

7. Click [OK](#).

```
Welch Modified Two-Sample t-Test

data: x: drp with group = Control, and y: drp with group = Treat
t = -2.3109, df = 37.8554006594391, p-value = 0.0132
alternative hypothesis: difference in means is less than 0
95 percent confidence interval:
      NA -2.691293
sample estimates:
mean of x mean of y
 41.52174  51.47619
```

The t test statistic is -2.3 based on 37.9 degrees of freedom. The P -value 0.0132 indicates that we can conclude that the reading activities improve scores at the $\alpha = 0.05$ significance level. Note that for the t statistic and confidence interval, S-PLUS computed the difference $\bar{x}_c - \bar{x}_t$ (rather than $\bar{x}_t - \bar{x}_c$ as done in IPS) so that the signs are reversed.

End of Example 7.4

Command Line Notes

```
> attach(Table7.4)
> t.test(drp[group == "Control"], drp[group == "Treat"],
+       alternative = "less", var.equal = F)
> # The following switches groups, and uses "greater"
> t.test(drp[group == "Treat"], drp[group == "Control"],
+       alternative = "greater", var.equal = F)
> detach(Table7.4)
```

7.3 Exercises

1. Exercise 6.71 gives readings for radon detectors in a test setting. In that exercise you are told to “Assume (unrealistically) that you know the standard deviation...” Exercise 7.37 follows up by dropping that unrealistic assumption. It asks: “Is there convincing evidence that the mean reading of all detectors of this type differs from the true value of 105? Carry out a test in detail and write a brief conclusion.”

Perform an appropriate hypothesis test and calculate a confidence interval.

2. Examples 7.19–7.21 in IPS describe an experiment to determine whether or not calcium intake affects blood pressure. The data are in [Table 7.5](#) in the `IPSdata` library. Perform a significance test to see whether or not calcium reduces blood pressure.

7.4 Solutions

1. Open [Exercise7.37](#) from the IPSdata library. From the menu, select [Statistics > Compare Samples > One Sample > t Test...](#)

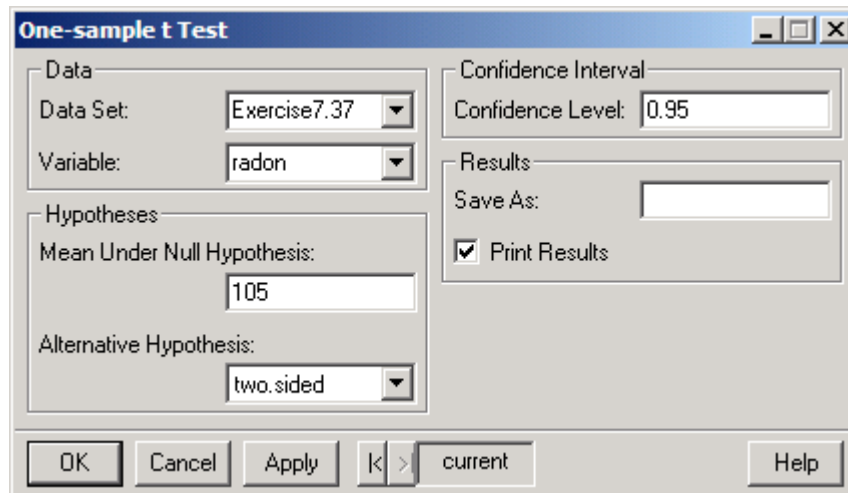


Figure 7.6

```
> t.test(Exercise7.37$radon, mu = 105)
```

2. Open [Table7.5](#) from the IPSdata library. From the menu, select [Statistics > Compare Samples > Two Samples > t Test...](#) For **Variable 1**: select **dec** (decrease), for **Variable 2**: select **group**, and check the **Variable 2 is a Grouping Variable** box.

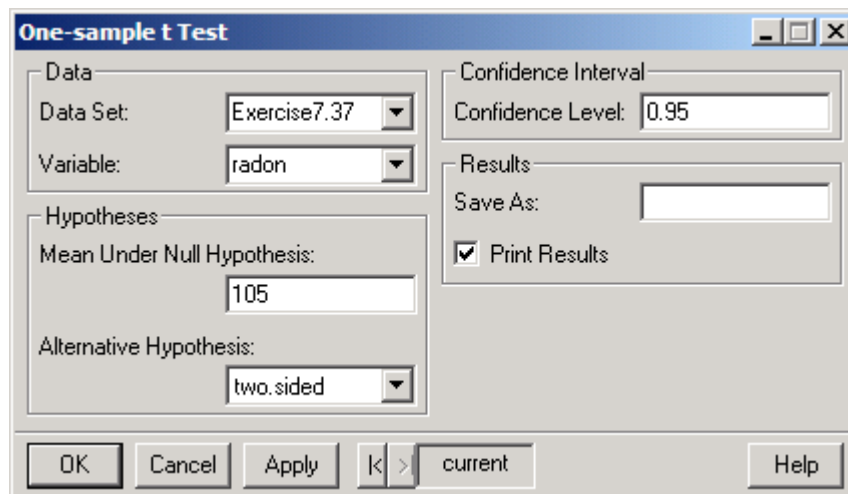


Figure 7.7

```
> attach(Table7.5)
> t.test(dec[group == "Calcium"], dec[group == "Placebo"],
+       alternative = "greater", var.equal = F)
> attach(Table7.5)
```

Chapter Order

We encourage teachers to consider using Chapter 14 (on bootstrap methods and permutation tests) next, to complement the material in Chapter 7 and later. The reasons are given in the preface.

We also encourage students to use Chapter 14 to supplement other chapters, even if that chapter is not included in your course. Bootstrap methods and permutation tests provide concrete visual representations, based on computer simulation, for some of the more abstract and difficult concepts treated in the later chapters. This complements the traditional mathematical approach, and may help you understand the concepts better.

Chapter 8

Inference for Proportions

In the previous chapters, we focused on inference for a population mean μ . In this chapter, we focus on counts and proportions.

We encourage you to use Chapter 14 together with this chapter, and in particular Section 14.3. The approach taken there, based on computer simulation and visualization, complements the more mathematical approach taken here, and may help you better understand the concepts in this chapter. It also provides a way for you to check your work.

8.1 Inference for a Single Proportion

If we let p denote the true proportion of the population that has a certain outcome (we will call this outcome a “success”), then we estimate p by taking a simple random sample of size n and computing the *sample proportion*,

$$\hat{p} = \frac{X}{n} = \frac{\text{count of successes in the sample}}{n}$$

For large n , we know that the sampling distribution of \hat{p} is approximately $N(p, \sqrt{\frac{p(1-p)}{n}})$. The calculations in IPS for confidence intervals and the P -values for significance tests are based on this assumption.

For hypothesis testing, to test $H_0 : p = p_0$, we use the z test statistic

$$z = \frac{\hat{p} - p_0}{\sqrt{\frac{p_0(1-p_0)}{n}}}$$

The [Proportions Test](#) in S-PLUS can be used for hypothesis testing and for finding confidence intervals.

Example 8.1 Buffon’s coin tosses

French naturalist Count Buffon (1707–1788) tossed a coin $n = 4040$ times and came up with 2048 heads. The sample proportion is $\hat{p} = \frac{2048}{4040} = 0.5069$. Is this strong evidence that Buffon’s coin was not a fair coin? We wish to test $H_0 : p = .50$ against $H_a : p \neq .50$. In this example $p_0 = 0.5$.

1. At the menu, select [Statistics > Compare Samples > Counts and Proportions > Proportions Parameters...](#). This brings up the [Proportions Test](#) dialog box.
2. In the [Data](#) box, for [Success Variable](#): type [2048](#), and for [Trials Variable](#): type [4040](#).
3. In the [Hypotheses](#) box, for [Proportions Variable](#): type [.5](#), and for [Alternative Hypothesis](#): select [two.sided](#).

4. For a 95% confidence interval leave the **Confidence Level:** field at the default of **.95**.

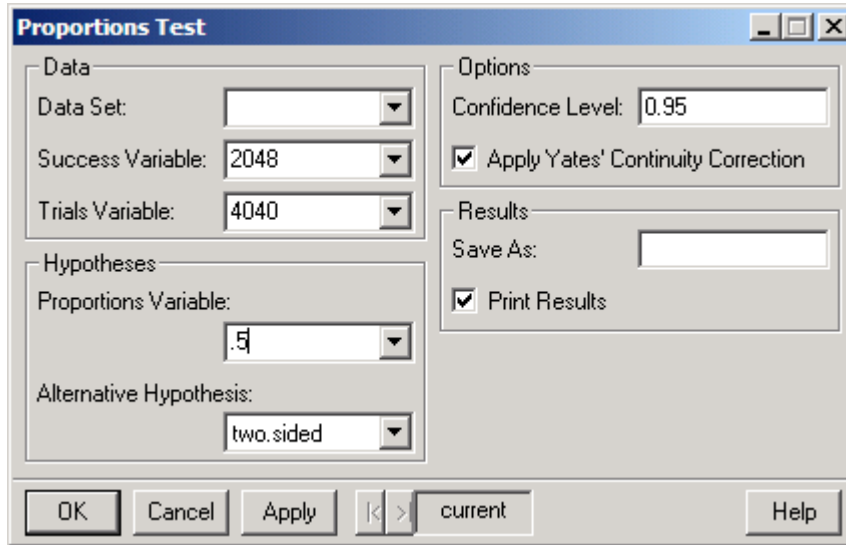


Figure 8.1

5. Click **OK**.

```
1-sample proportions test with continuity correction
data: 2048 and 4040
X-square = 0.7488, df = 1, p-value = 0.3869
alternative hypothesis: P(success) in Group 1 is not equal to 0.5
95 percent confidence interval:
 0.4913912 0.5224569
sample estimates:
prop'n in Group 1
 0.5069307
```

The P -value for the test is $P = 0.39$, so we do not reject the null hypothesis.

Note that the **Proportions Test** command also outputs a 95% confidence interval: we are 95% confident that the true proportion of heads from this coin is in the interval $(0.491, 0.522)$. This confidence interval is computed using the formula $\hat{p} \pm z^* \cdot \sqrt{\frac{\hat{p}(1-\hat{p})}{n}}$, where z^* denotes the relevant z critical value.

End of Example 8.1

Remarks:

- The proportions test in S-PLUS uses a *continuity correction* in performing the calculations. This improves the accuracy of the results but means that the S-PLUS output does not correspond exactly to the results obtained in the text. Clear the **Yates Continuity Correction** checkbox if you wish to see the results without this correction.

- To improve accuracy, another estimator of the population proportion is the **Wilson estimate** of the population proportion, namely

$$\tilde{p} = \frac{X + 2}{n + 4},$$

where X represents the number of “successes”. The corresponding standard error is

$$SE_{\tilde{p}} = \sqrt{\frac{\tilde{p}(1 - \tilde{p})}{n + 4}}.$$

To compute confidence intervals using the Wilson estimate, just increase the **Success Variable:** entry by 2 and the **Trials Variable:** by 4 in the **Proportions Test** dialog box. For the Buffon coin problem, this would mean typing 2050 in the **Success Variable:** field and 4044 in the **Trials Variable:** field.

Command Line Notes

The function `prop.test` performs the proportions test.

```
> prop.test(2048, 4040)
```

This produces output similar to that above. From the command line, you have the option of extracting just the P -value or confidence interval.

```
> prop.test(2048, 4040)$p.value
```

```
[1] 0.3868684
```

```
> prop.test(2048, 4040)$conf.int
```

```
[1] 0.4913912 0.5224569
```

```
attr(,"conf.level"):
```

```
[1] 0.95
```

If you wish to specify different hypotheses or a different confidence level, say for example, $H_0 : p = 0.6$ against $H_a : p < 0.6$ at $\alpha = 0.01$, then type

```
> prop.test(2048, 4040, p = .6, alternative = "greater",
+           conf.level = .99)
```

8.2 Two Sample Proportions

In this section, we compare the proportion of successes in two independent samples. If we let p_1 and p_2 denote the true proportion of successes for Population 1 and Population 2, respectively, we perform inference on the difference $p_1 - p_2$.

Example 8.2 Binge drinking study

This example considers a study of binge drinking behavior in college students. The random variable X is the number of students classified as binge drinkers.

Pop.	n	X	$\hat{p} = X/n$.
1 (men)	$n_1 = 7180$	1630	$\hat{p}_1 = 0.227$
2 (women)	$n_2 = 9916$	1684	$\hat{p}_2 = 0.170$
Total	17096	3314	0.194

We are testing $H_0 : p_1 = p_2$ against $H_a : p_1 \neq p_2$.

We first need to create a data set that holds this information.

1. Create a new data set (Section 0.4.6).

2. Enter the counts X into the first column, then the sample sizes n into the second column and label the columns X and n , respectively.
3. At the menu, select **Statistics > Compare Samples > Counts and Proportions > Proportions Parameters ...**. This brings up the **Proportions Test** dialog.
4. In the **Data** box, for **Data Set:** enter the name of the data set.
5. For **Success Variable:** select X and for **Trials Variable:** select n .

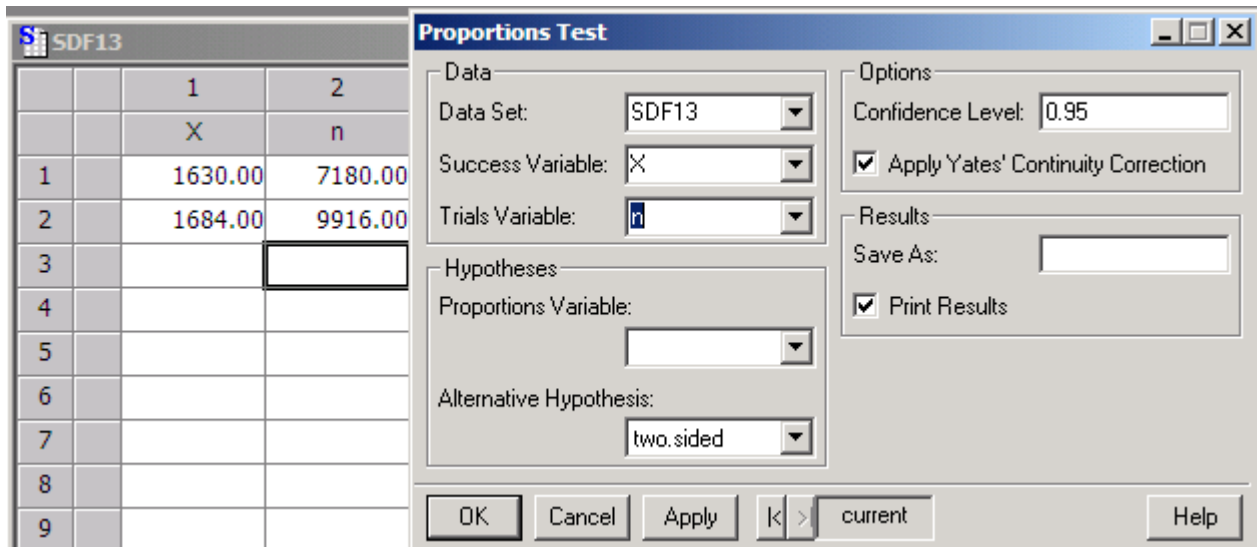


Figure 8.2

6. Click **OK**.

```

2-sample test for equality of proportions with continuity correction

data: X and n from data set SDF13
X-square = 86.8062, df = 1, p-value = 0
alternative hypothesis: two.sided
95 percent confidence interval:
 0.04488666 0.06949925
sample estimates:
 prop'n in Group 1 prop'n in Group 2
      0.2270195      0.1698265

```

With a P -value of 0, we conclude that there is strong evidence that the proportion of men who binge drink is not the same as the proportion of women who binge drink.

From the report, we see that a 95% confidence interval for the true difference in proportions is (0.045, 0.069). Note that this interval does not contain 0.

End of Example 8.2

Command Line Notes

For two independent samples, provide `prop.test` with two vectors: the first one contains the successes (X), the second contains the sample sizes (n) corresponding to the successes.

```
> prop.test(c(1630, 1684), c(7180, 9916))
```

8.3 Exercises

1. A poll taken during 2001 indicated that about 73% of all Americans were extremely worried about privacy over the Internet. You suspect that a higher proportion of students at your college have the same concern. You poll a random sample of 300 students at your school and find 240 of those questioned are extremely worried about privacy over the Internet. What can you conclude from your data?
2. Suppose your survey of privacy issues over the Internet included 138 women and 162 men, of which 120 of the women and 110 of the men were extremely worried about privacy issues. Is there evidence that men and women have different concerns about privacy issues over the Internet?

8.4 Solutions

1. From the GUI,

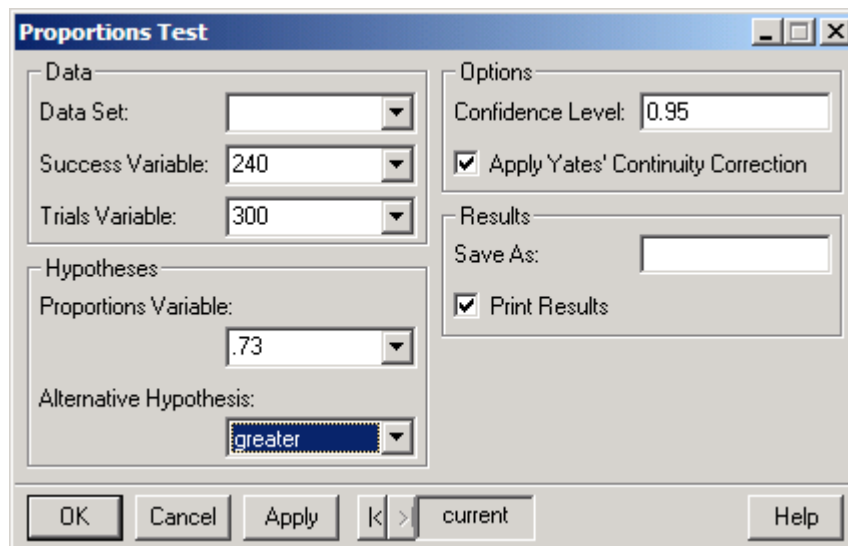


Figure 8.3

From the command line,

```
> prop.test(240, 300, p = .73, alternative = "greater")
```

```
1-sample proportions test with continuity correction
```

```
data: 240 out of 300, null probability 0.73
```

```
X-square = 7.1072, df = 1, p-value = 0.0038
```

```
alternative hypothesis: P(success) in Group 1 is greater than 0.73
```

```
95 percent confidence interval:
```

```
0.7576395 1.0000000
```

```
sample estimates:
```

```
prop'n in Group 1  
0.8
```

A P -value of 0.0038 gives strong evidence that the proportion of students who are concerned about privacy issues over the Internet is higher than the general population.

2. From the GUI,

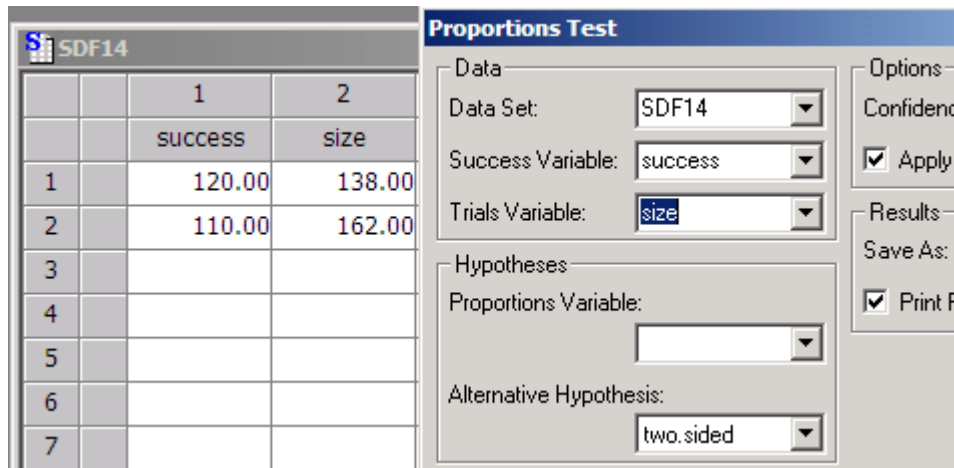


Figure 8.4

From the command line,

```
> prop.test(c(120, 110), c(138, 162))
```

2-sample test for equality of proportions with continuity correction

```
data: c(120, 110) out of c(138, 162)
X-square = 14.0794, df = 1, p-value = 0.0002
alternative hypothesis: two.sided
95 percent confidence interval:
 0.0925987 0.2885070
sample estimates:
 prop'n in Group 1 prop'n in Group 2
      0.8695652      0.6790123
```

A P -value of 0.0002 gives strong evidence that there is a difference in the proportions of men and women at your college who have concerns about privacy issues over the Internet.

Chapter 9

Inference for Two-Way Tables

In this chapter, we consider two related questions: a) Is the distribution of a categorical variable different for several independent populations? b) Is there an association between two categorical variables from the same population?

9.1 Two-Way Tables and the Chi-Square Test

Two-way tables (also called *contingency tables*) are a convenient way to summarize categorical variables. The categorical variables can arise in two different settings. A researcher may have a single categorical variable on samples taken from two or more independent populations. For instance, a sociologist surveys a random sample of 100 low-income, 100 medium-income, and 100 high-income individuals to compare their primary means of obtaining news. A *test of homogeneity* addresses the question, “Is the proportion of individuals who read the daily newspaper the same for all three populations?” Or, a researcher may have several categorical variables obtained from a single population. For example, an educator conducts a survey of 100 students at a high school and obtains information on their gender and their level of extracurricular activity. A *test of association* (also called a *test of independence*) addresses the question, “Is there an association between gender and level of participation in extracurricular activity?”

Example 9.1 Binge drinking and gender

A study compared binge drinking behavior and gender:

Frequent binge drinker	Men	Women	Total
Yes	1630	1684	3314
No	5550	8232	13782
Total	7180	9916	17096

We can enter the data into S-PLUS in one of three ways: 1) a contingency table similar to the one above, but leaving out the (marginal) totals; 2) three columns: a categorical variable for gender, a categorical variable for outcome, and a numeric variable giving counts; 3) a single row for each subject with two entries (one for the row response and one for the column response). The third option is usually only seen when there are many different variables collected. In the third case we could add a column of 1’s as the counts, and it would be a variation on the second method.

1. Create two new data sets by entering the data using the first and second methods above.

		1	2	3
		Men	Women	
1	Yes	1630.00	1684.00	
2	No	5550.00	8232.00	
3				

		1	2	3
		Sex	Binge	Count
1		Men	Yes	1630.00
2		Men	No	5550.00
3		Women	Yes	1684.00
4		Women	No	8232.00

Figure 9.1

Remark: You can convert between these two types of data entry using [Data > Restructure > Stack](#) and [Data > Restructure > Unstack](#).

		1	2	3	4
		Men	Women	Binge	
1	Yes	1630.00	1684.00	Yes	
2	No	5550.00	8232.00	No	

		1	2	3	4
		Count	Binge	Sex	
1		1630.00	Yes	Men	
2		5550.00	No	Men	
3		1684.00	Yes	Women	
4		8232.00	No	Women	

		1	2	3	4
		Men.Count	Men.Binge	Women.Count	Women.Binge
1		1630.00	Yes	1684.00	Yes
2		5550.00	No	8232.00	No
3					

Figure 9.2

Now, we conduct a test to see if the proportion of binge drinkers is the same for both sexes. First, use the data entered as a contingency table (SDF15 in Figure 9.1).

- At the menu, select [Statistics > Compare Samples > Counts and Proportions > Chi-square test...](#) This brings up the [Pearson's Chi-Square Test](#) dialog.

3. For **Data Set**: enter the name of the data (for example, **SDF15**).
4. Check the box labeled **Data Set is a Contingency Table**. The two **Variables** fields turn gray.
5. Click **OK**.

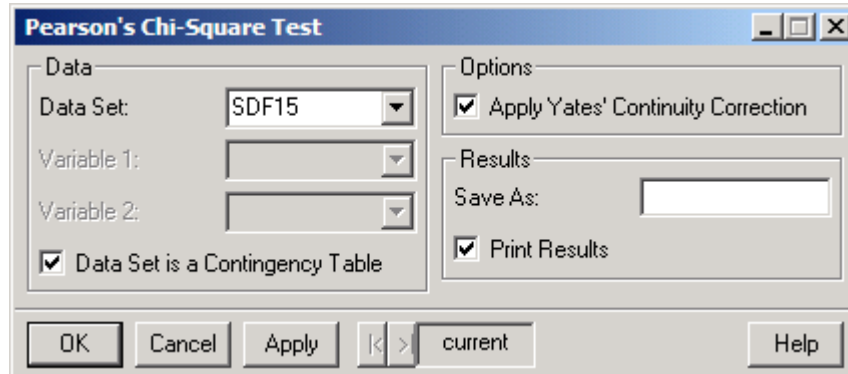


Figure 9.3

The results are sent to a Report window:

```
Pearson's chi-square test with Yates' continuity correction
data:  SDF15
X-square = 86.8062, df = 1, p-value = 0
```

From the P -value of 0, we reject the null hypothesis that the proportion of binge drinkers is the same for both men and women.

We repeat the analysis using the data entered into the data frame via the second method (SDF16 in Figure 9.1).

6. At the menu, select **Statistics > Data Summaries > Crosstabulations**.
7. In the **Crosstabulations** dialog box, select the name of the **Data Set**.
8. Click on **Binge**, then CTRL-click on **Sex** in the **Variables**: box. Select **Count** in the **Counts Variable**: box (this would be left empty if there were one row per subject).
9. Click **OK**.

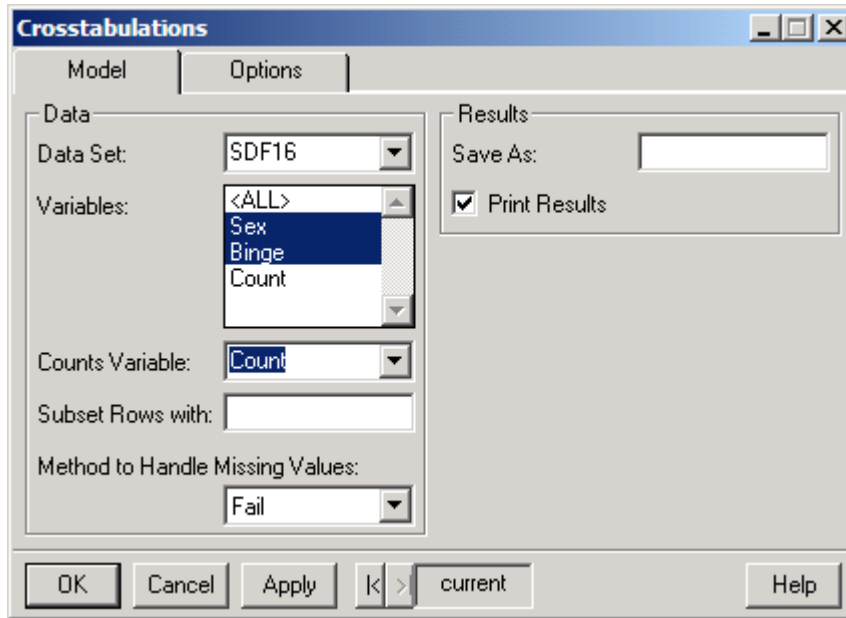


Figure 9.4

The results that are sent to the report window include the test, the table, and the marginal and conditional distributions.

```

*** Crosstabulations ***
Call:
crosstabs(formula = Count ~ Binge + Sex, data = SDF16,
          na.action = na.fail, drop.unused.levels = T)
17096 cases in table
+-----+
|N      |
|N/RowTotal|
|N/ColTotal|
|N/Total  |
+-----+
Binge |Sex
      |Men   |Women |RowTotal|
+-----+-----+-----+
Yes   |1630  |1684  |3314   |
      |0.49  |0.51  |0.19   |
      |0.23  |0.17  |        |
      |0.095 |0.099 |        |
+-----+-----+-----+
No    |5550  |8232  |13782  |
      |0.4    |0.6    |0.81   |
      |0.77  |0.83   |        |
      |0.32  |0.48   |        |
+-----+-----+-----+
ColTotl|7180  |9916  |17096  |
      |0.42  |0.58  |        |
+-----+-----+-----+
Test for independence of all factors
      Chi^2 = 87.17176 d.f.= 1 (p=0)
      Yates' correction not used

```

End of Example 9.1

```

Command Line Notes
To do the chi-square test from the command line, you first need to create a
matrix. Then, to do Cross tabulations, the data should be in a data frame.
> men = c(1630, 5550)
> women = c(1684, 8232)
> binge = cbind(men, women) # This is a matrix (not data frame)
> chisq.test(binge)
...
> binge2 = data.frame(Sex = rep(c("Men", "Women"), each = 2),
+                     Binge = rep(c("Yes", "No"), 2),
+                     Count = c(men, women))
> crosstabs(Count ~ Binge + Sex, data = binge2)

```

Example 9.2 Chi-square test for raw data

In many settings data come in a raw form, not summarized in a two-way table. An example is the S-PLUS built-in data set `market.survey`, containing the results of a marketing survey of 1000 households conducted for AT&T.

Open the data set `market.survey` (Section 0.4.1) and note that it consists of 10 variables and 1000 observations.

	1	2	3	4	5	6	7
	pick	income	moves	age	education	employment	usage
1	OCC	<7.5	0	35-44	HS	F	9.00
2	ATT	45-75	2	25-34	HS	F	2.00
3	OCC	NA	0	NA	NA	NA	6.00
4	OCC	NA	2	65+	<HS	R	7.00
5	OCC	NA	0	65+	HS	H	0.00

Figure 9.5

The variable `pick` has two levels: `ATT`, if the household surveyed chose AT&T for their telephone service, and `OCC`, if the household chose a different company. The variable `education` has five levels based on the highest level of education achieved by the respondent.

Suppose you want to know if there is a relationship between the choice respondents made in telephone service and their education level. That is,

H_0 : There is no association between choice of telephone service and education level, against

H_a : There is an association between choice of telephone service and education level.

To perform a chi-square test on these two variables,

1. At the menu, select `Statistics > Compare Samples > Counts and Proportions > Chi-square test...`. This brings up the `Pearson's Chi-Square Test` dialog.
2. For `Data Set`: select `market.survey`.
3. For `Variable 1`: select `pick`, and for `Variable 2`: select `education`.
4. Click `OK`.

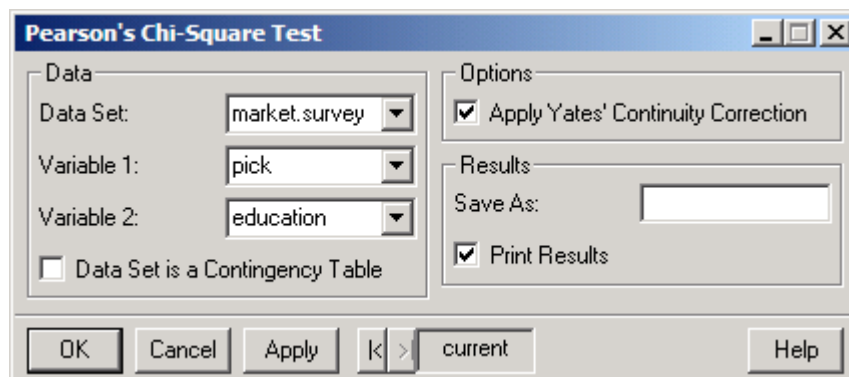


Figure 9.6

Pearson's chi-square test without Yates' continuity correction

```
data: pick and education from data set market.survey  
X-square = 28.623, df = 5, p-value = 0
```

Note that Yates' continuity correction is not used here; it is only used for a two-by-two table.

We reject the null hypothesis and conclude that there is an association between choice of service and education level.

In addition to the Report window, there is a warning message indicating that one or both of the variables contained missing observations (NA's). S-PLUS automatically excludes them from the analysis.

If you want to see more detail on the analysis, you can create a table of cross-classified data.

5. At the menu, select **Statistics > Data Summaries > Crosstabulations....** This brings up the **Crosstabulations** dialog.
6. For **Data Set:**, type `market.survey`.
7. For **Variables:** select `pick` and `education` (hold down the CTRL key to make these multiple selections).
8. For **Method to Handle Missing Values:** select `omit`.

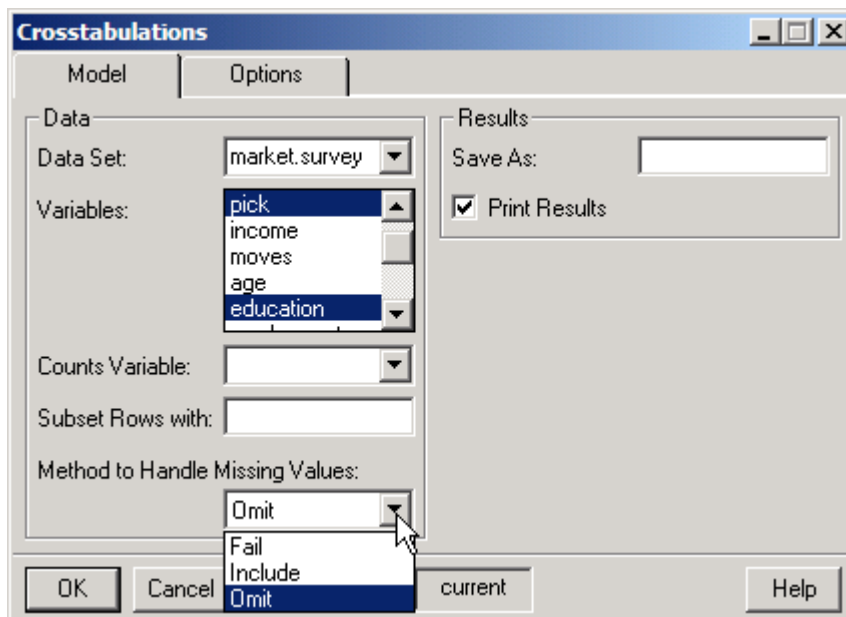


Figure 9.7

9. Click **OK**.

```

Call:
crosstabs(formula = ~ pick + education, data = market.survey,
           na.action = na.exclude, drop.unused.levels = T)
965 cases in table
+-----+
|N      |
|N/RowTotal|
|N/ColTotal|
|N/Total |
+-----+
pick    |education
        |<HS   |HS    |Voc   |Coll  |BA    |>BA   |RowTotal|
+-----+-----+-----+-----+-----+-----+-----+
OCC     | 96   |180   | 35   | 82   | 60   | 21   |474   |
        |0.2   |0.38  |0.074 |0.17  |0.13  |0.044 |0.49  |
        |0.63  |0.5   |0.65  |0.44  |0.4   |0.35  |      |
        |0.099 |0.19  |0.036 |0.085 |0.062 |0.022 |      |
+-----+-----+-----+-----+-----+-----+-----+
ATT     | 57   |181   | 19   |105   | 90   | 39   |491   |
        |0.12  |0.37  |0.039 |0.21  |0.18  |0.079 |0.51  |
        |0.37  |0.5   |0.35  |0.56  |0.6   |0.65  |      |
        |0.059 |0.19  |0.02  |0.11  |0.093 |0.04  |      |
+-----+-----+-----+-----+-----+-----+-----+
ColTotal|153   |361   |54    |187   |150   |60    |965   |
        |0.16  |0.37  |0.056 |0.19  |0.16  |0.062 |      |
+-----+-----+-----+-----+-----+-----+-----+
Test for independence of all factors
Chi^2 = 28.62297 d.f.= 5 (p=0.00002748984)
Yates' correction not used

```

We see that there were 965 households that answered both questions of interest ([pick](#) and [education](#)). Find the cell corresponding to row ATT and column BA. We note that 491 of the 965 respondents chose AT&T, and of these, 90 or 18% (from $90/491 = .1832$, rounded to 0.18 in the print-out), had a B.A. degree. We also note that of those surveyed, 150 had a B.A. degree, and of these 90, or 60% (from $90/150 = 0.60$), chose AT&T.

S-PLUS does not report the expected counts for each cell.

End of Example 9.2

Command Line Notes

The function [chisq.test](#) performs the chi-square test, and [crosstabs](#) creates cross-tabulations.

```

> chisq.test(market.survey$pick, market.survey$education)
> crosstabs(~ pick + education, data = market.survey,
+           na.action = na.omit)

```

9.2 Exercises

1. A college dean wonders if student participation in extracurricular sports is related to their area of study. A questionnaire sent to a simple random sample of 140 students at his college provides the following information:

	No participate	Yes participate
Math/Science	32	15
Social Science	43	20
Humanities	22	10
Fine Arts	20	8

Based on this data, what can this dean conclude?

- Using the [market.survey](#) data, determine whether or not there is an association between choice of telephone service and income.

9.3 Solutions

-

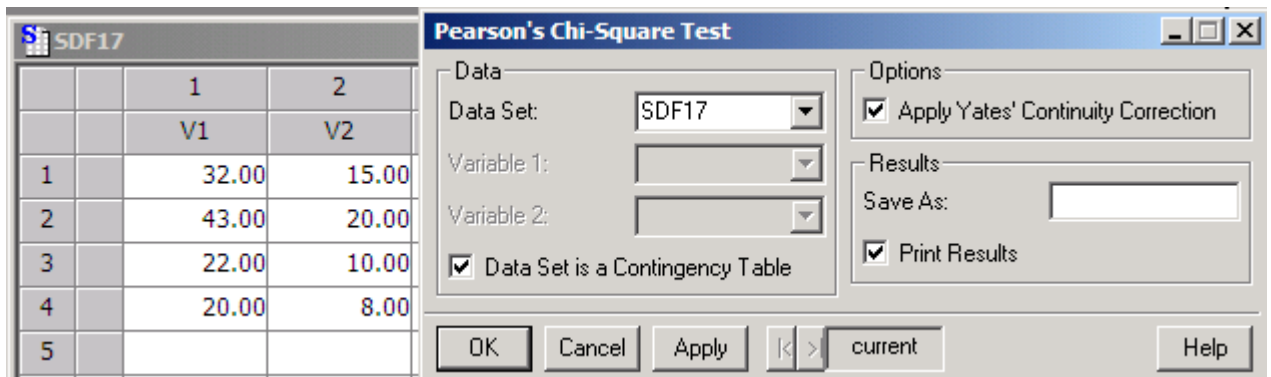


Figure 9.8

At the command line,

```
> yes = c(32, 43, 22, 20)
> no = c(15, 20, 10, 8)
> participate = cbind(yes, no)
> participate
  yes no
[1,] 32 15
[2,] 43 20
[3,] 22 10
[4,] 20  8
> chisq.test(participate)
```

```

Pearson's chi-square test without Yates'
continuity correction
```

```
data: participate
X-square = 0.1101, df = 3, p-value = 0.9906
```

From the P -value of 0.9906, the dean cannot conclude that there is an association between area of study and participation in extracurricular sports.

2.

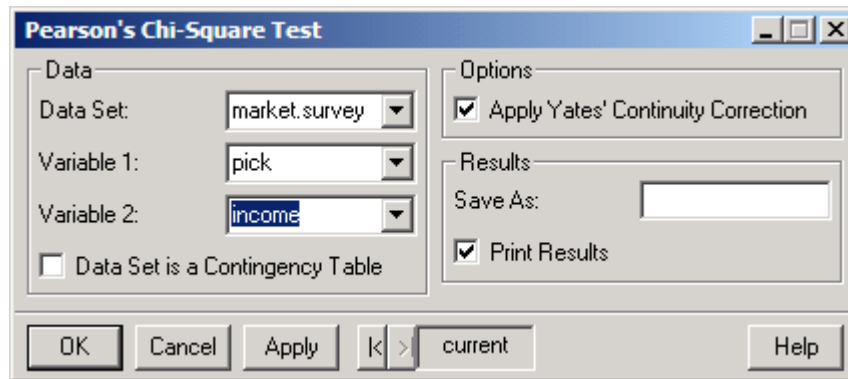


Figure 9.9

At the command line,

```
> chisq.test(market.survey$pick, market.survey$income)
```

```
    Pearson's chi-square test without Yates'
    continuity correction
```

```
data: market.survey$pick and market.survey$income
```

```
X-square = 11.1541, df = 6, p-value = 0.0837
```

```
Warning messages:
```

```
 215 observations (x[i],y[i]) with NA in 'x' or 'y' removed.
```

There does not appear to be an association between choice of telephone service and income.

Chapter 10

Inference for Regression

In Chapter 2 we saw how to fit a least-squares regression line using [Statistics > Regression > Linear....](#) In this chapter we explore in more detail some of the other options available with this command.

We encourage you to use Chapter 14 together with this chapter, and in particular Section 14.5. The approach taken there, based on computer simulation and visualization, complements the more mathematical approach taken here and may help you better understand the concepts in this chapter. It also provides a way for you to check your work.




10.1 Graphing the Data

We should always begin by graphing the data.

Example 10.1 Fuel efficiency (IPS Example 10.1)

Computers in some vehicles calculate various quantities related to the performance. Example 10.1 in IPS discusses a sample of 60 observations from one vehicle, in which miles per gallon (MPG) and speed in miles per hour (MPH) were recorded each time the gas tank was filled.

We begin by plotting the data. A review of plotting commands in Chapter 2 of this manual may be helpful.

1. Open the [Example10.1](#) data set from the `IPSdata` library. There are 60 observations.
2. Begin by plotting the data. Select `MPH`, and control-click on `MPG` (to select these as x and y (explanatory and response) variables, respectively). Bring up the [Plots 2D](#) palette and click on the [Scatter](#) button .
3. It would be helpful to show a regression line and/or smooth, as in Figure 10.3 in IPS. It is difficult to do both simultaneously without using the command line, but we can do one at a time. Click on the [Linear Fit](#)  button for a line, and the [Spline](#)  button for a spline smooth.

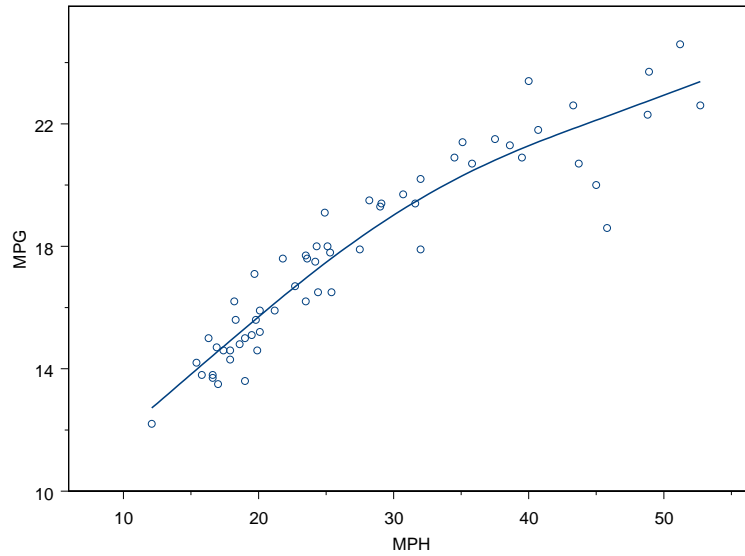



Figure 10.1

The relationship is clearly nonlinear. Following IPS, we next consider `LOGMPH` as the explanatory variable. If that variable were not already present in the data set, we would create it using the `Data > Transform...` menu.

4. Bring up the data. Select `LOGMPH`, and control-click on `MPG`. Bring up the `Plots 2D` palette and click on the `Spline` button . This relationship appears close to linear. We will proceed with linear regression using `LOGMPH` as the explanatory variable.

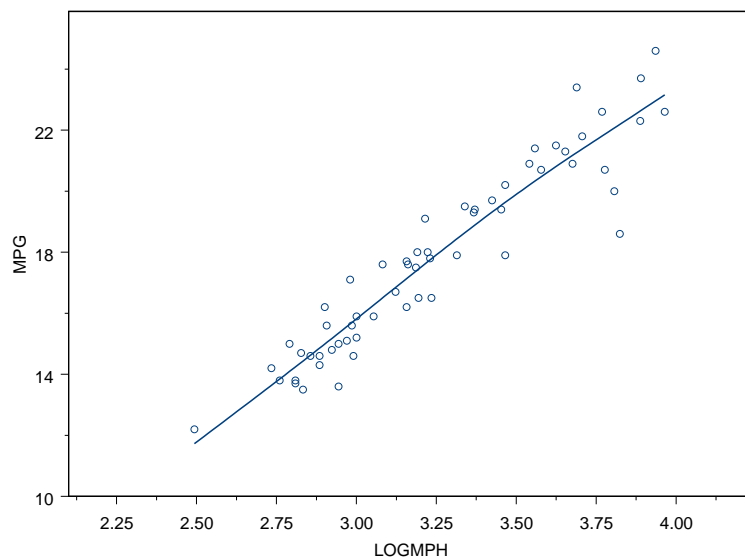


Figure 10.2

Command Line Notes

We use the functions `plot`, `lm`, `abline`, `lines` and `smooth.spline`.

```
> attach(Example10.1)
> plot(MPH, MPG)
> lines(smooth.spline(MPH, MPG, df = 3))
> abline(lm(MPG ~ MPH, data = Example10.1), col=2)
>
> plot(LOGMPH, MPG)
> lines(smooth.spline(LOGMPH, MPG, df = 3))
> abline(lm(MPG ~ LOGMPH, data = Example10.1), col=5)
> detach()
```

End of Example 10.1

10.2 Inference About the Model

Now that we understand confidence intervals and tests of hypothesis, we can apply these ideas to regression analysis to get a sense of how accurate our predictions are and to test to see if the regression equation differs significantly from a horizontal line.

Example 10.2 Fuel efficiency

1. At the menu, select **Statistics > Regression > Linear...** This brings up the **Linear Regression** dialog.
2. For **Data Set:** enter **Example10.1**.
3. Under **Variables**, for **Response:** select **MPG**, and for **Explanatory:** select **LOGMPH**. (In some versions of S-PLUS the fields are called **Dependent:** and **Independent:**, respectively.)

The **Formula:** field should read **MPG ~ LOGMPH**.

4. In the **Save Model Object** box, for **Save As:** type **fitMPG**.

This saves the regression calculations as a *model object* for later use.

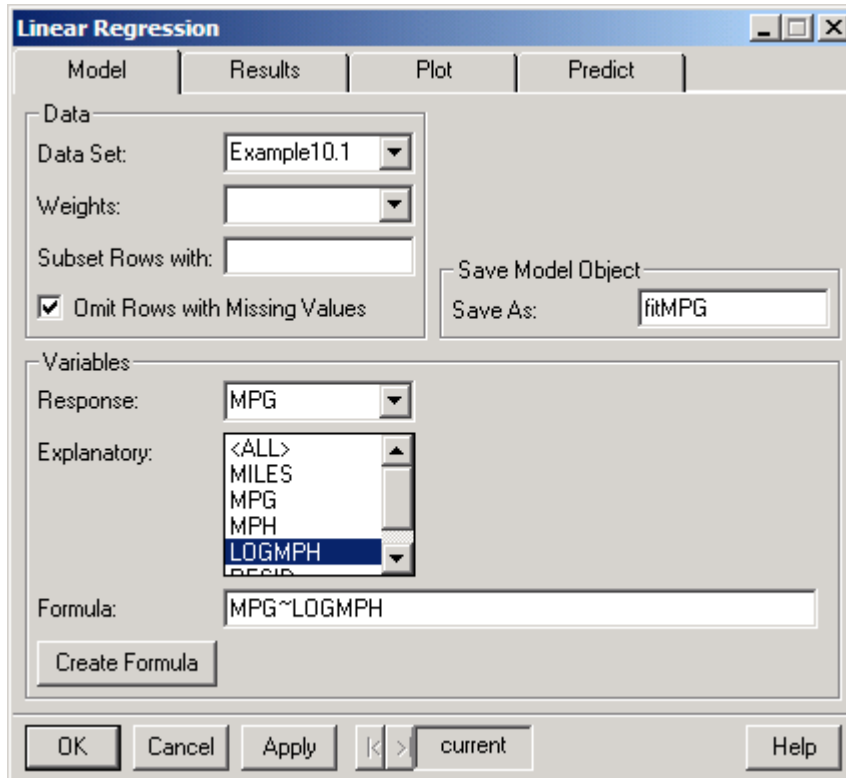


Figure 10.3

5. Click [OK](#).

```

*** Linear Model ***

Call: lm(formula = MPG ~ LOGMPH, data = Example10.1,
          na.action = na.exclude)
Residuals:
    Min       1Q   Median       3Q      Max
-3.717 -0.5187  0.1121  0.6593  2.149

Coefficients:
              Value Std. Error  t value Pr(>|t|)
(Intercept) -7.7963   1.1549   -6.7506  0.0000
      LOGMPH  7.8742   0.3541   22.2373  0.0000

Residual standard error: 0.9995 on 58 degrees of freedom
Multiple R-Squared:  0.895
F-statistic: 494.5 on 1 and 58 degrees of freedom,
the p-value is 0

```

We see in the Report window that

- The estimated slope (b) is $b = 7.8742$, and the estimated intercept is $a = -7.7963$. Thus, the fitted line is $\widehat{MPG} = -7.80 + 7.87\text{LOGMPH}$.

- The standard error of the estimated slope is $SE_b = 0.03541$. This measures the variability in the estimate of the slope and is used in both the confidence interval and the hypothesis test.
- The t statistic for the slope is 22.2373; this is the ratio $t = b/SE_b = 7.8742/0.3541$ (these numbers are rounded). We use t to test the hypothesis H_0 : True slope is zero against H_a : True slope is not zero. Under the null hypothesis and if various other assumptions are satisfied, then t follows a t distribution with $n - 2 = 58$ degrees of freedom. S-PLUS prints both the t statistic and the corresponding P -value, $P = 0.0$.
S-PLUS also gives the corresponding information for the intercept.
- The residual standard error is $s = 0.9995$. This measures the variability of the residuals around the regression line and is computed from $s^2 = \frac{1}{n-2} \sum \text{residuals}^2$.

This report does not give a confidence interval for the true slope (or intercept), although it is easy to compute with the information given above: $b \pm t^*SE_b$. You just need to find t^* , the upper $(1 - C)/2$ critical value (Table C in IPS, or see Chapter 4 of this manual).

End of Example 10.2

Command Line Notes

```
We use lm to fit the regression and summary to describe the results.
> fitMPG = lm(MPG ~ LOGMPH, data = Example10.1)
> summary(fitMPG)
:
> 7.8742 + c(-1,1) * qt(.975,58) * 0.9995
[1] 5.873483 9.874917
```

Remark: S-PLUS uses a special syntax for expressing formulas in models. The response variable is written first, followed by the tilde, \sim . Any explanatory variables are written to the right of the tilde, separated by $+$. The intercept term is included by default. For instance, $Y \sim X$ represents $Y = a + bX$. To omit the intercept term, write $Y \sim X - 1$.

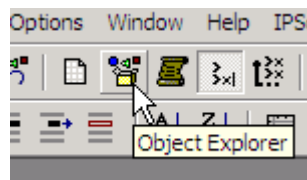
10.3 Inference About Prediction

A level C confidence interval for the true mean response μ_y for $x = x^*$ is given by $\hat{y} \pm t^*SE_{\hat{\mu}}$, where the standard error $SE_{\hat{\mu}}$ is

$$SE_{\hat{\mu}} = s \sqrt{\frac{1}{n} + \frac{(x^* - \bar{x})^2}{\sum (x - \bar{x})^2}}$$

and t^* represents the upper $(1 - C)/2$ critical value of the t distribution on $n - 2$ degrees of freedom; s is the residual standard error.

Example 10.3 Confidence intervals for the mean response



1. Bring up the Object Explorer (Section 0.5)

- Click on the **Data** icon in the left panel to select (highlight) it and find `fitMPG` (the saved regression object) in the right panel.
- Right-click on `fitMPG` and select `Predict...` from the shortcut menu.

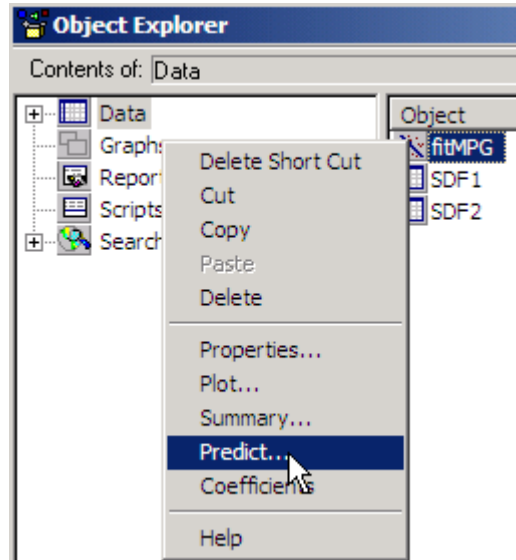


Figure 10.4

- In the `Linear Regression Prediction` dialog box, enter `Example10.1` in both the `New Data:` and `Save In:` fields.
- Check the boxes `Predictions` and `Confidence Intervals`.

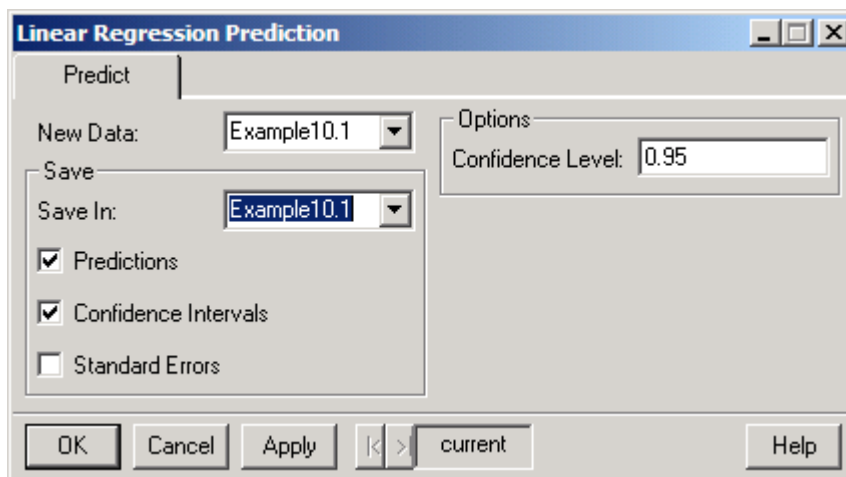


Figure 10.5

- Click on `OK`.

The data set now has three new columns: a column (`fit`) of fitted values (\hat{y}), and columns `LCL95` and `UCL95` with the lower and upper limits, respectively, of the confidence intervals.


Example10.1								
	1	2	3	4	5	6	7	8
	MILES	MPG	MPH	LOGMPH	RESID	fit	LCL95	UCL95
1	12457.00	14.80	18.60	2.92	-0.42	15.22	14.88	15.56
2	12658.00	15.10	19.50	2.97	-0.49	15.59	15.27	15.92
3	13439.00	17.50	24.20	3.19	0.21	17.29	17.03	17.56

Figure 10.6

For example, the 95% confidence interval for the true mean response at MPH= 18.60 is (14.88, 15.56).

End of Example 10.3

Remarks:

- To find fitted values and confidence intervals for x -values different from the ones used in the regression, create a new data set with a column holding the new x -values. This column of new x values must have the same name as the original column (and be spelled exactly the same). Then repeat the above steps (1–6) but enter the name of this data set in the **Data Set:** and **Save In:** fields.
- To see a plot of confidence bounds for mean responses, create a linear fit plot of the data. For example, using [Example10.1](#):
 1. Select **LOGMPH**, then CTRL-click on **MPG**.
 2. Click on the **Linear Fit** button  on the **Plots 2D** palette.
 3. Right-click on the line and choose **By Conf Bound...** from the shortcut menu. This brings up the **Curve Fitting Plot** dialog, with the **By Conf Bound** tab active.
 4. Type the desired **Level:** of confidence.
 5. Under **Line Attributes**, select a line **Style:** (solid, dashed, and so on).
 6. Click **OK**.

These confidence bounds are a little wider than the pointwise confidence intervals to adjust for multiple comparisons (that is, to have the confidence of including the true line at all x -values simultaneously).

- Through the GUI, it is possible to compute only confidence intervals for the mean response from a regression equation. See the **Command Line Notes** section below for how to compute the prediction intervals (for a single observation).
- The prediction interval for a single observation uses the standard error for prediction $SE_{\hat{y}}$

$$SE_{\hat{y}} = s \sqrt{1 + \frac{1}{n} + \frac{(x^* - \bar{x})^2}{\sum(x - \bar{x})^2}}.$$

Note this is larger than $SE_{\hat{\mu}}$ because of the extra term (the 1) under the radical sign.

Caution—while most common regression inferences are approximately correct for large sample sizes even when residuals are not normally distributed, this is not true for prediction intervals. That number 1 means that the effective sample size is always 1 for prediction intervals for a new observation; the number of previous observations is nearly irrelevant.

Command Line Notes

The `predict` command computes confidence and prediction intervals (options `ci.fit=T` and `pi.fit=T`, respectively). We first demonstrate how to create the graph that includes the confidence and prediction intervals of the mean responses.

```
> predMPG = predict(fitMPG, ci.fit = T, pi.fit = T)
> predMPG$ci.fit
$ci.fit:
      lower      upper
1 14.87890 15.56444
2 15.27157 15.91510
3 17.03299 17.55536
...
> attach(Example10.1)
> plot(LOGMPH, MPG, ylim = range(predMPG$pi.fit))
> lines(sort(LOGMPH), predMPG$fit[order(LOGMPH)])
> lines(sort(LOGMPH), predMPG$ci.fit[order(LOGMPH),1], lty=3)
> lines(sort(LOGMPH), predMPG$ci.fit[order(LOGMPH),2], lty=3)
> lines(sort(LOGMPH), predMPG$pi.fit[order(LOGMPH),1], lty=4)
> lines(sort(LOGMPH), predMPG$pi.fit[order(LOGMPH),2], lty=4)
> detach()
```

The `predict` command can also be used to obtain predicted values for new values of the explanatory variable:

```
> predict(fitMPG, newdata=data.frame(LOGMPH=c(1.6, 1.7)))
      1      2
1.062009 1.055697
```

Thus, $\hat{MPG} = 1.062$ is predicted for $LOGMPH = 1.6$, and $\hat{MPG} = 1.056$ is predicted for $LOGMPH = 1.7$.

10.4 Checking the Regression Assumptions

S-PLUS generates several diagnostic plots to help you detect any extreme points that need investigating and to determine if the regression assumptions are satisfied.

Example 10.4 Diagnostic plots

1. Bring up the Object Explorer.
2. Click on the **Data** icon in the left panel to select (highlight) it and find `fitMPG` (the saved regression object) in the right panel.
3. Right-click on `fitMPG` and select **Plot...** from the shortcut menu.
4. In the **Linear Model Plots** dialog box, under **Plots**, check the top six boxes in the left panel—all except **Partial Residuals** (which is useful when there are multiple predictor variables).
5. Click **OK**.

The following six plots are produced:

- **Residuals vs. fitted values.** This plot provides the same information as the residuals vs. independent (x) plot. (The fitted values are used because this plot generalizes nicely to the case of more than one predictor variable.) Look for nonlinear patterns or increasing variance in this plot.

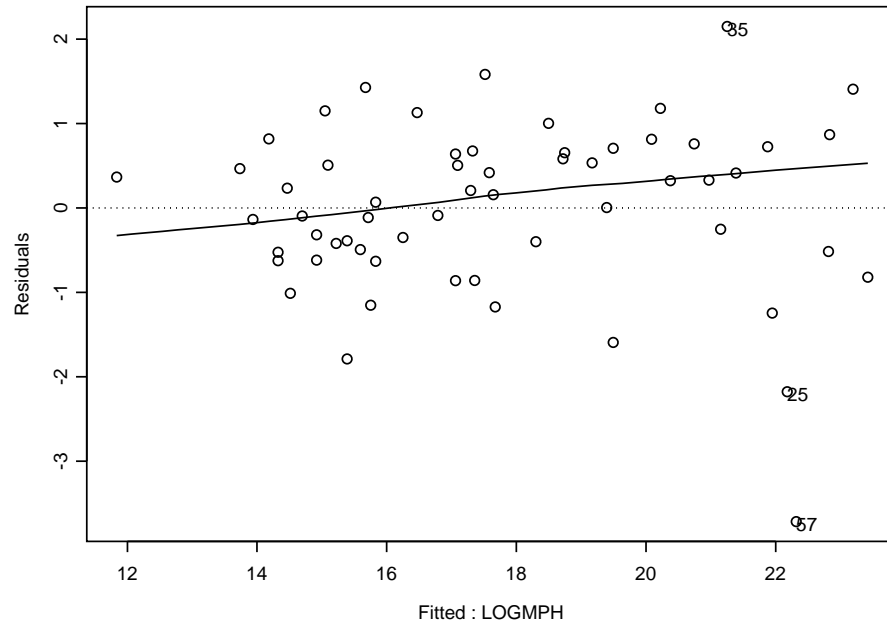


Figure 10.7

We look for a nonrandom pattern in this plot. The solid line in the plot is a curve produced by a smoother, and we note how much it deviates from the dashed horizontal reference line at $y = 0$. In this case, the curve appears nearly straight, but with a positive slope. This curve is produced by loess, a procedure which is resistant to outliers. In this case it indicates observations 57, 25, and 35 are possible outliers. It gives a positive slope because it is less influenced by observations 57 and 25 than is the least-squares line (which has zero intercept and zero slope).

- **Square root of absolute value of residuals vs. fitted values.** This plot is similar to the first one, except that the square root of the absolute value of the residuals is plotted on the y axis. This puts all potential outliers at the top. Unequal variances are also easier to spot in this plot.

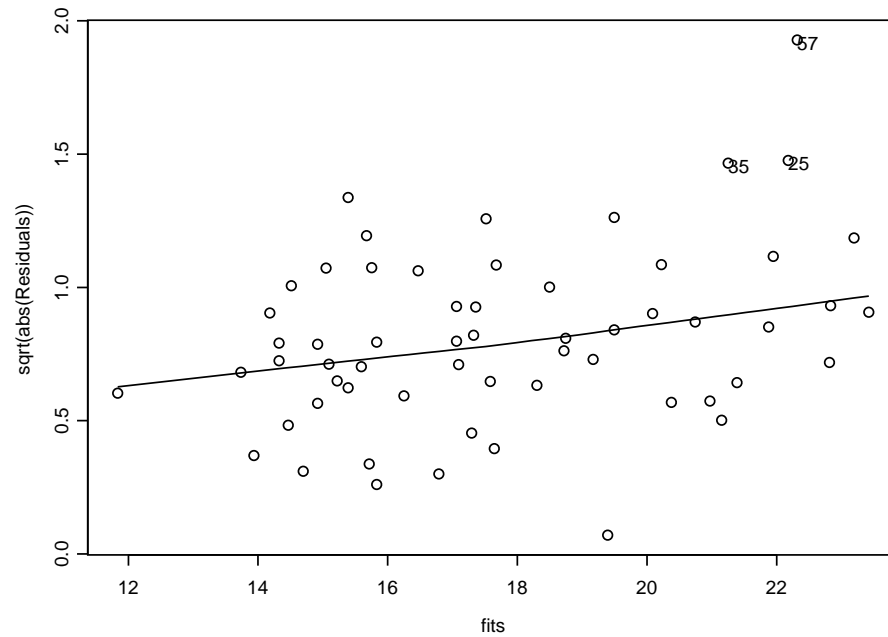


Figure 10.8

The smoother indicates that there is a positive relationship between the fitted values and the size of the residuals. This suggests that there are unequal variances. A common remedy in cases like this, where the variance *increases* with the fitted values, is to transform the response variable, for example using $\sqrt{\text{MPG}}$ or $\log(\text{MPG})$ as the response variable (this may also require transforming explanatory variables for a good fit).

- **Response vs. Fit.** The third plot graphs the observed y -values against the predicted (fitted) y -values. In a perfect fit, all of the points would fall along the dotted line. We can see how much of the y -variable is predicted by the x -variable. We can also check for nonlinearity or unequal variances.

When there is only one explanatory variable this plot is not very useful. It gives exactly the same points as a simple plot of y vs. x , except that the x axis is rescaled (so the x -values have the same mean as y). And the first diagnostic plot is more useful for checking for unequal variances, and either of the first two plots is more useful for checking for unequal variances.

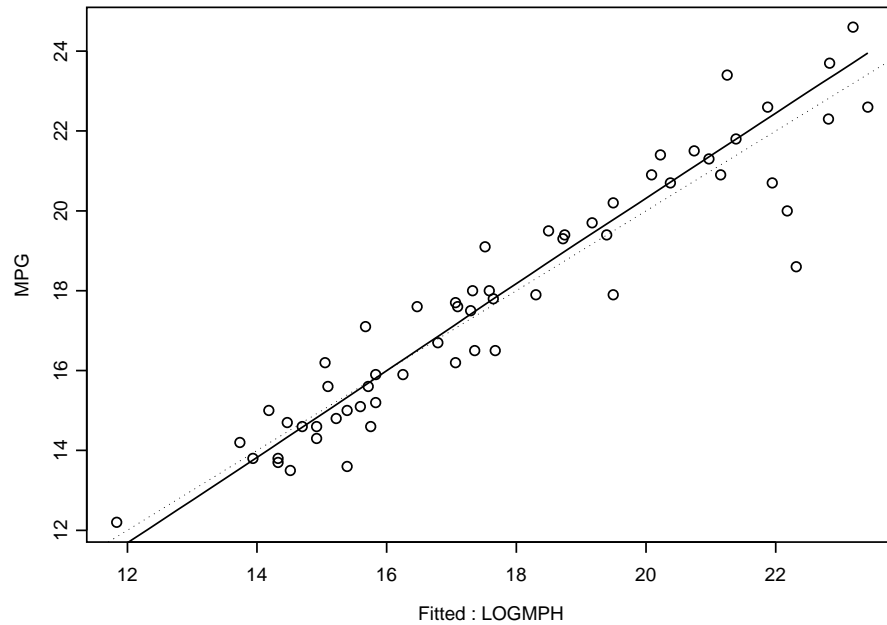


Figure 10.9

- **Residuals Normal QQ.** This is the normal quantile plot of the residuals. It is used to see if the residuals follow a normal distribution. If the assumption is met, then the points should lie roughly on a line.

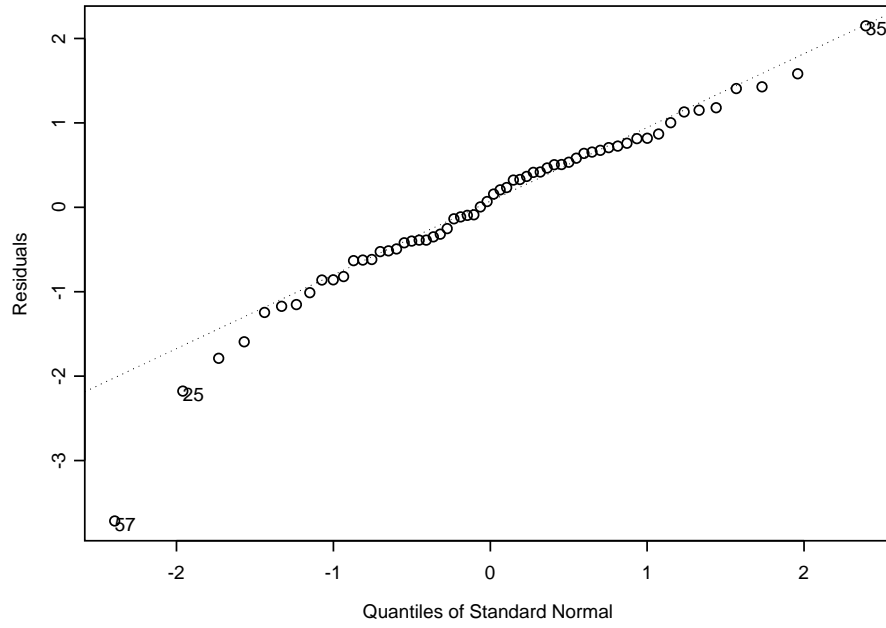


Figure 10.10

In this example observations 57 and perhaps 25 do appear to be outliers.

- **Residual-Fit Spread.** This plot is a visual analog of the r^2 term. It shows a plot of the (centered) sorted fitted values next to the sorted residuals. The better the regression fits, the higher the r^2 , and the larger the spread of the fitted values relative to the residuals.

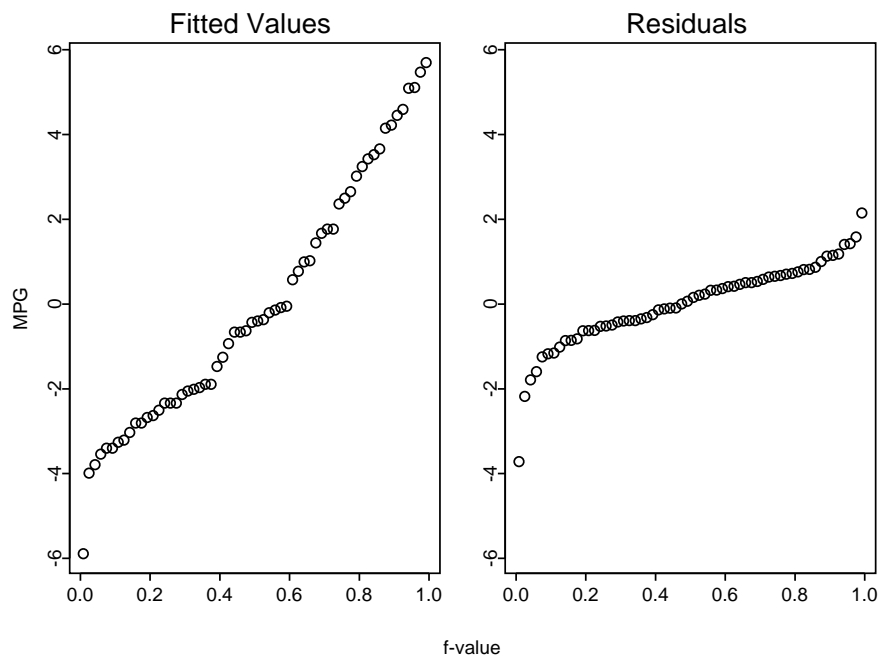


Figure 10.11

In this example, we note that there is substantially more spread in the fitted values than the residual values; this corresponds to a high r^2 (recall that $r^2 = 0.895$).

- **Cook's Distance.** Cook's distance is a measure of how *influential* an observation is. An observation is influential if its omission greatly alters the regression equation (see IPS, Chapter 2). This is indicated by the length of the vertical line in the Cook's distance plot: the longer the line, the more influential the observation.

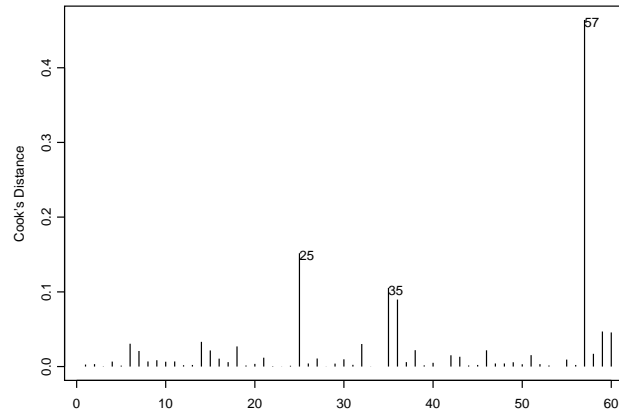


Figure 10.12

In this example, observation 57 is quite influential, followed by observations 25, 35, and 36.

End of Example 10.4

Command Line Notes

The `plot` function automatically gives the same six plots when called with a model object. The command `par(mfrow=c(2,3))` sets up the graphsheet to display all six graphs on one page.

```
> plot(fitMPG, smooth = T, ask = T)
> par(mfrow = c(2,3)) # set layout parameter to 2x3 arrangement
> plot(fitMPG, smooth = T)
> par(mfrow = c(1,1)) # reset layout parameter to default
```

10.5 More Detail About Simple Linear Regression

The ANOVA table for regression provides information to perform an F-test for the slope coefficient.

Example 10.5 Fuel efficiency, continued

We will use the model object `fitMPG` that we created earlier.

1. In the Object Explorer, right-click on the icon for `fitMPG`.
2. In the resulting menu, select `Summary...`

- In the [Linear Regression Results](#) dialog box, check the box [ANOVA Table](#) (and clear the box next to [Long Form](#)).
- Click [OK](#).

```

*** Linear Model ***

Analysis of Variance Table

Response: MPG

Terms added sequentially (first to last)
      Df Sum of Sq  Mean Sq  F Value Pr(F)
LOGMPH  1  493.9918  493.9918  494.4971    0
Residuals 58   57.9407    0.9990

```

End of Example 10.5

Remark: In this chapter, we demonstrated how to create a model object ([fitMPG](#)) from a linear regression (see the first example) and then with this object, access the dialog boxes to do additional analysis, such as prediction, diagnostic plots, and so forth. It is possible to combine all of these steps at once. In the initial dialog box ([Statistics > Regression > Linear...](#)), enter the information (see [Figure 10.3](#)), then click on the tabs [Results](#), [Plot](#), and [Predict](#) for additional options.

10.6 Exercise

- In Germany, fuel efficiency for cars is measured in “liters per hundred kilometers.” The analog using gallons and miles measurements is “gallons per hundred miles” (GPHM). This system more directly measures the cost of driving.

For example, if you drive 100 miles per week, how much cheaper is a Jeep Liberty at 17 MPG than a Ford Explorer at 14 MPG? Or a Honda Insight at 57 MPG than a VW Golf at 38 MPG? It is easier given the GPHM ratings, of 5.9, 7.1, 1.8, and 2.6, respectively. The Liberty saves 1.2 gallons over the Explorer, while the Insight saves 0.8 gallons over the Golf. Note that a difference of 3 MPG between two SUVs matters more than a difference of 19 MPG between two smaller cars.

Convert the [MPG](#) into GPHM, using the [Data > Transform](#) menu, using [100 / MPG](#) for [Expression](#).

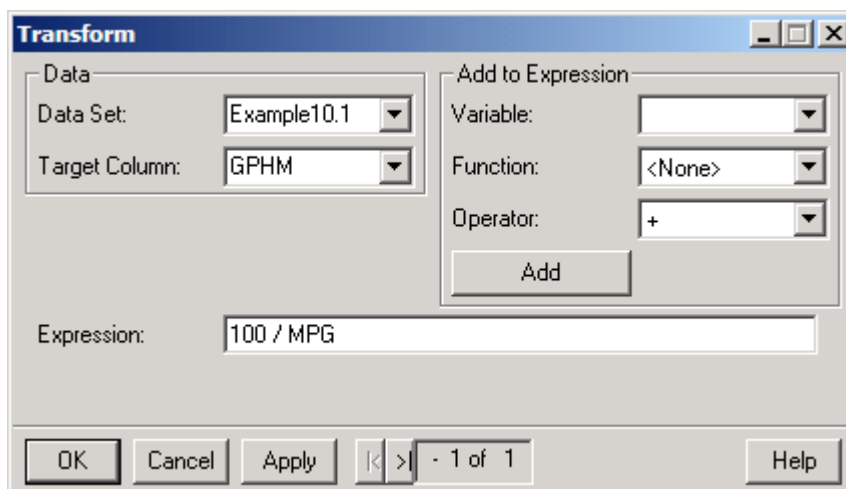


Figure 10.13

Then regress **GPHM** against **MPH** (or a transformation of **MPH**). Do the steps outlined in this chapter, in particular: plotting the data, finding the equation of the regression line, obtaining the standard error and a confidence interval for the regression slope, obtaining confidence intervals for the predictions, and checking the regression assumptions.

10.7 Solution

1. From the GUI,

Open the data set, select **MPH**, and control-click to select **GPHM**. Use a spline curve to check for nonlinearity; there is a clear nonlinear relationship.

Try again with **LOGMPH** in place of **MPH**. The relationship is closer to linear, but there is still nonlinearity apparent.

Use the **Data > Transform** menu, to create a new variable **inverseMPH** using the expression $1 / MPH$ and do the plot; now the relationship appears linear.

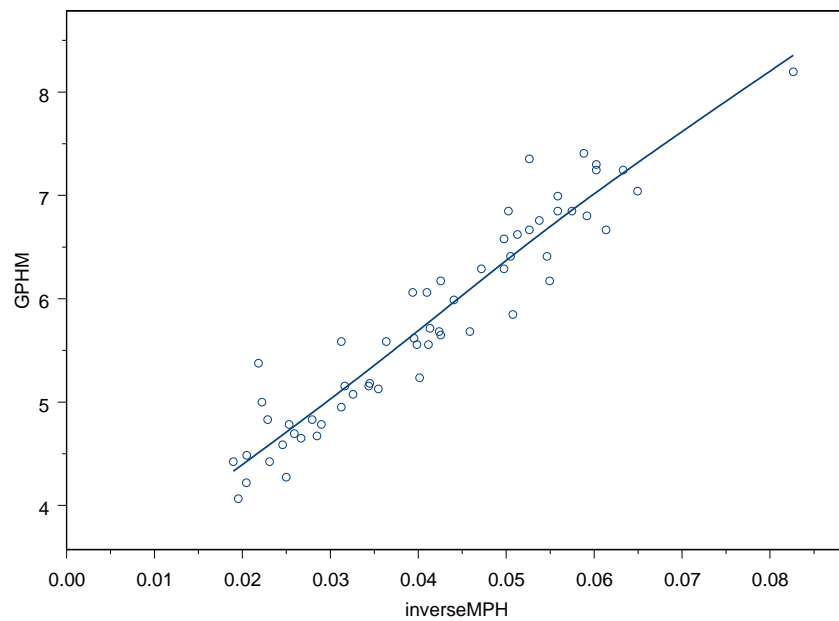


Figure 10.14

Then run the regression and create plots and predictions:

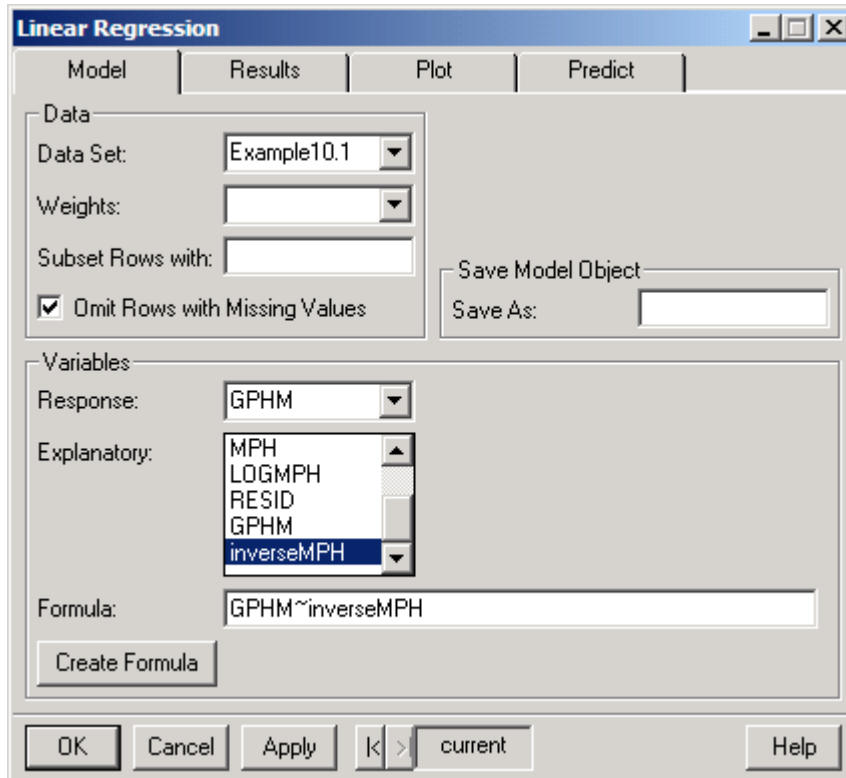


Figure 10.15

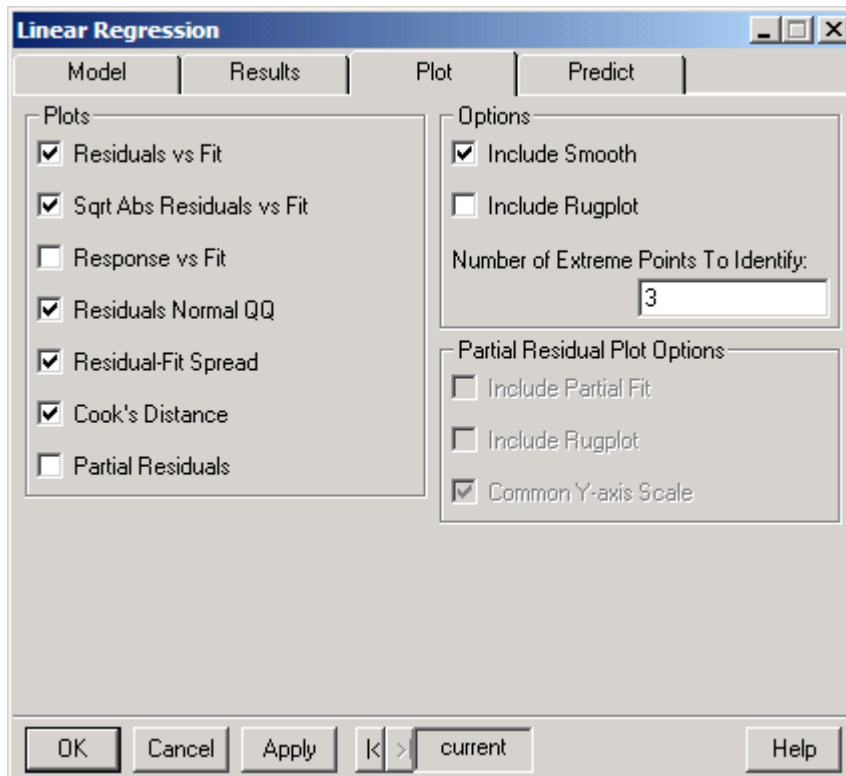


Figure 10.16

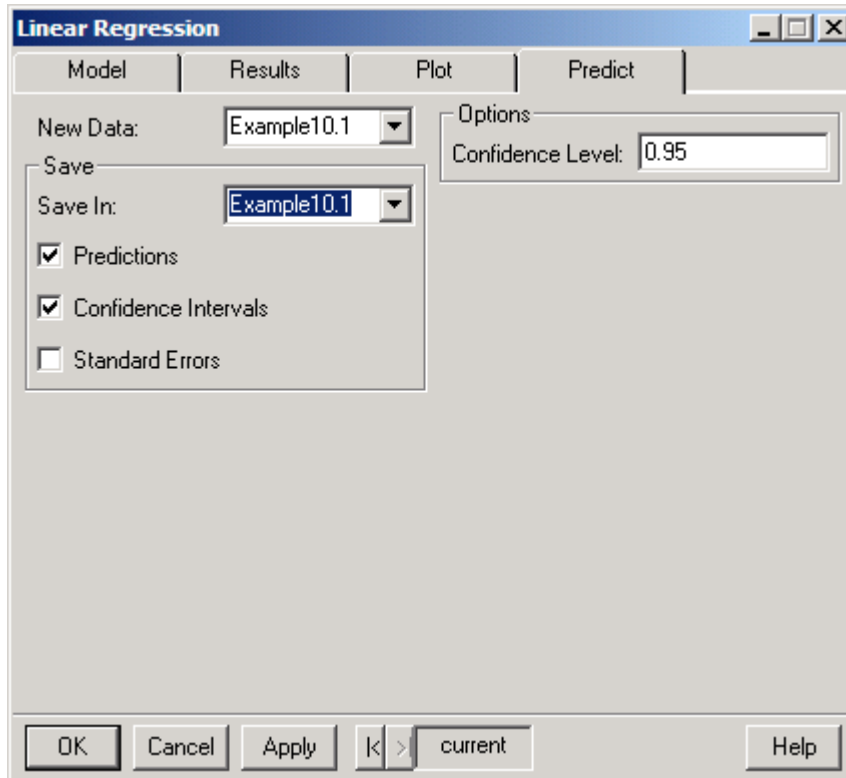


Figure 10.17

```

*** Linear Model ***

Call: lm(formula = GPHM ~ inverseMPH, data = Example10.1,
          na.action = na.exclude)
Residuals:
    Min       1Q   Median       3Q      Max
-0.5538 -0.1941 -0.0475  0.1657  0.8671

Coefficients:
            Value Std. Error t value Pr(>|t|)
(Intercept)  3.0808   0.1214   25.3688  0.0000
inverseMPH  65.4213   2.7577   23.7229  0.0000

Residual standard error: 0.3043 on 58 degrees of freedom
Multiple R-Squared:  0.9066
F-statistic: 562.8 on 1 and 58 degrees of freedom,
the p-value is 0

```

The regression equation is $GPHM = 3.08 + 65.4 \text{ inverseMPH}$. The standard error for the slope is 2.76; a 95% confidence interval for the slope is $65.4 \pm t^*2.76$. The r^2 is .907, slightly higher than before. The regression plots indicate there may be some nonlinearity remaining, that residual variances are not quite constant, that residuals are reasonably close to normal though observations 57 and 59 may be outliers, and that observation 57 is influential.

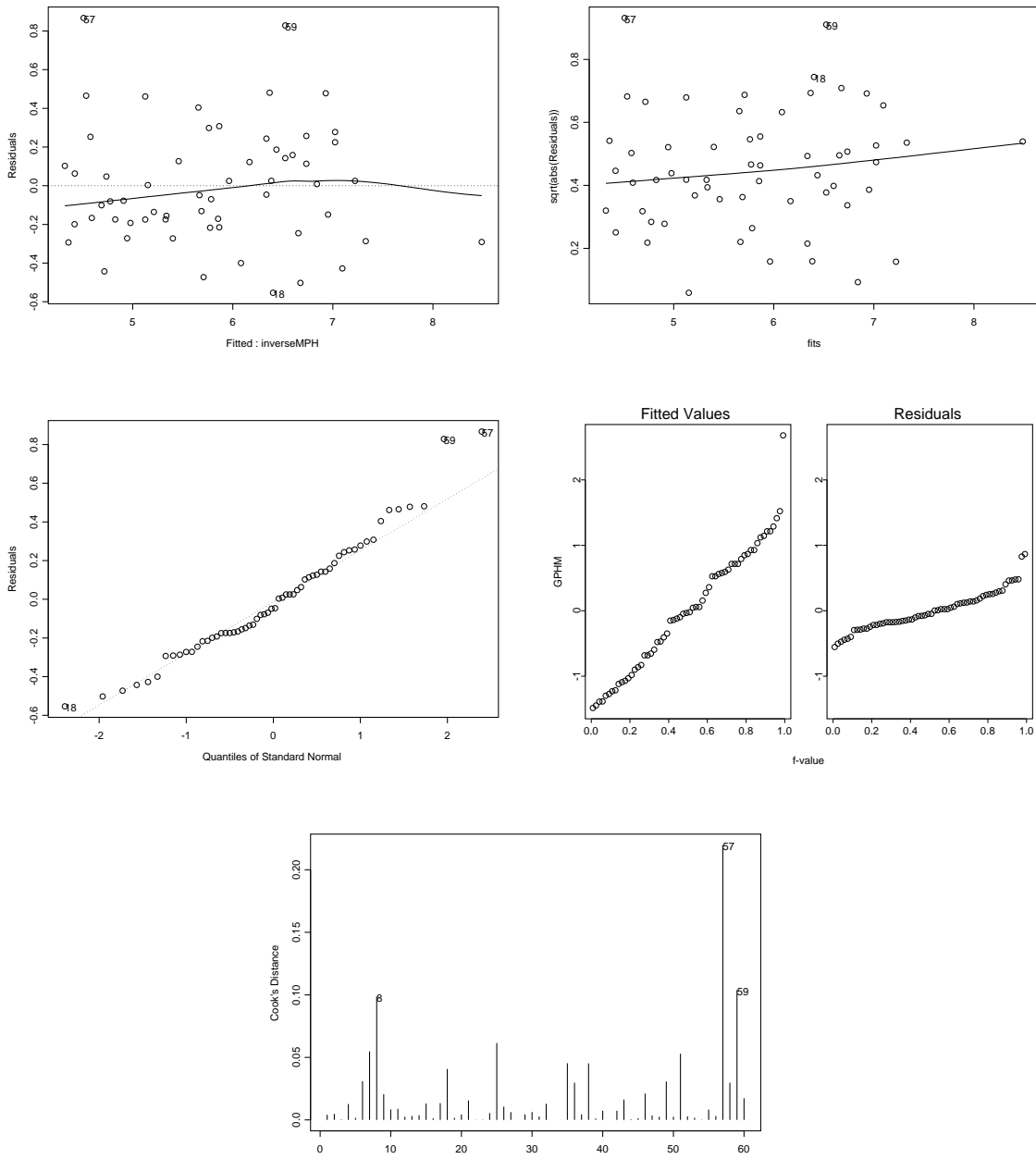


Figure 10.18

Chapter 11

Multiple Regression

Multiple regression models a linear relationship between a numeric response variable y and several numeric explanatory variables x_1, x_2, \dots, x_p . Let μ_y denote the mean response:

$$\mu_y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p$$

11.1 Inference for Multiple Regression

Example 11.1 Computer science majors

Information from a study on 244 computer science majors at a large university is described in the Data Appendix of IPS. The researchers wanted to predict success in the early university years, as measured by GPA. This data set is included in the `IPSdata` library as `csdata`.

The first step is to perform some exploratory data analysis. We will compute summary statistics and correlations.

1. Open the data set (using `IPSdata > Select Data...`).
2. At the menu, select `Statistics > Data Summaries > Summary Statistics...`. This brings up the `Summary Statistics` dialog.
3. In the `Data` box, for `Data Set:` select `csdata`.
4. For `Variables:`, select `<ALL>`.
5. Click `OK`.

This gives numeric summaries of the data, and frequencies for the factor variable `sex` (not printed here).

6. At the menu, select `Statistics > Data Summaries > Correlations...`. This brings up the `Correlations and Covariances` dialog.
7. For `Data Set:`, select `csdata` and for `Variables:` select (use CTRL-click) all numerical variables. Or leave `Variables` set to `<ALL>`; and ignore the warning `Dropping nonnumeric column(s) sex`.
8. Click `OK`.

```


*** Correlations for data in:  csdata ***

      gpa      hsm      hss      hse      satm      satv
gpa  1.000000  0.4364988  0.3294253  0.2890013  0.2517143  0.1144905
hsm  0.4364988  1.0000000  0.5756865  0.4468865  0.4535139  0.2211203
hss  0.3294253  0.5756865  1.0000000  0.5793746  0.2404793  0.2616975
hse  0.2890013  0.4468865  0.5793746  1.0000000  0.1082849  0.2437146
satm 0.2517143  0.4535139  0.2404793  0.1082849  1.0000000  0.4639419
satv 0.1144905  0.2211203  0.2616975  0.2437146  0.4639419  1.0000000

```

From this matrix, we can note, for example, that the correlation between **hsm** (high school mathematics grade) and **satm** (SAT mathematics score) is 0.4535.

Scatterplot matrices can also be a helpful tool in exploratory data analysis of several numeric variables (see Section 2.1).

9. Select all numerical columns, and click on the **Scatter Matrix** button  on the **Plots 2D** palette.
10. Right-click on a point, and select **Smooth...** from the shortcut menu.
11. For **Smoothing Type**: select **Smoothing Spline**. Leave **Deg. of Freedom** at the default value of 2 (numbers close to 1 give straight lines; larger numbers allow more curvature).
12. Click **OK**.

Note in particular the top row of the figure, which has **GPA** on the *y* axis and the potential explanatory variables on the *x* axes.

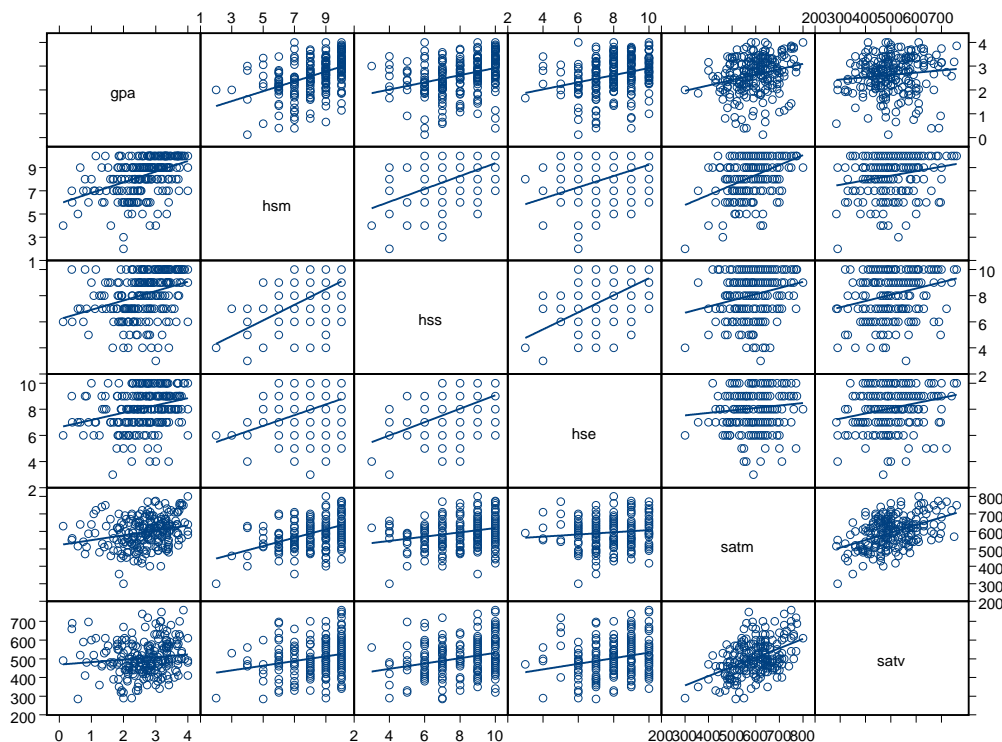


Figure 11.1

We now consider a multiple regression on high-school grades. We first work with the model using `gpa` for the response variable and using only the three high school grades for the explanatory variables:

$$\mu_{\text{gpa}} = \beta_0 + \beta_1 \text{hsm} + \beta_2 \text{hss} + \beta_3 \text{hse}$$

13. At the menu, select `Statistics > Regression > Linear....`
14. Select `csdata` for the `Data Set:`.
15. For `Response:` variable, select `gpa`.
16. For `Explanatory:`, select `hsm`, `hss`, `hse` (use control-click). The `Formula:` field should read `gpa ~ hsm + hss + hse`.

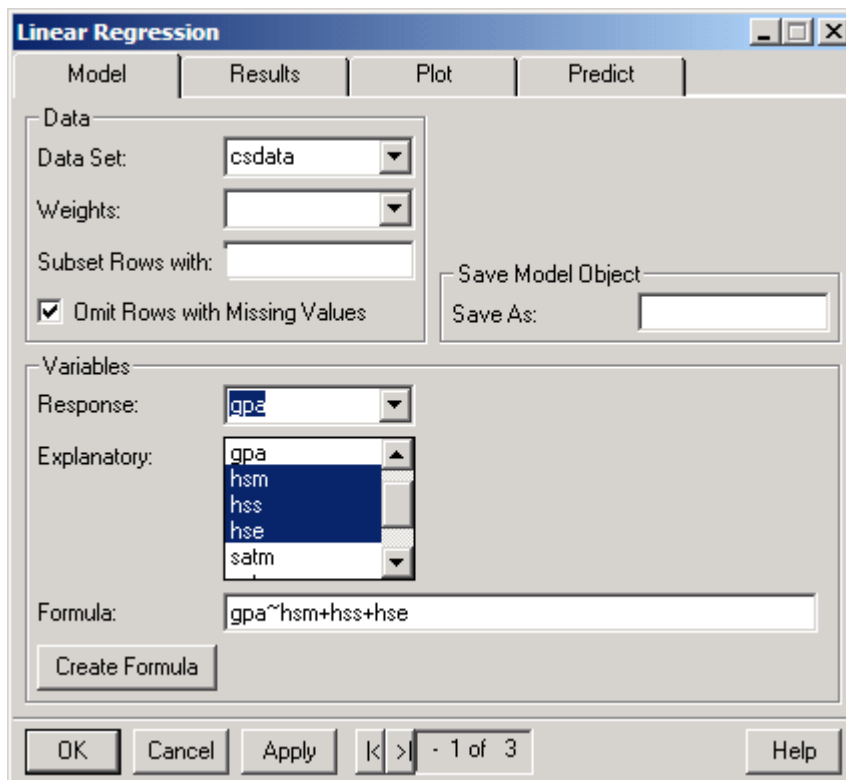


Figure 11.2

This formula syntax in S-PLUS indicates that `gpa` (the variable to the left of the tilde, `~`) is the response variable, and that the model includes (+) the terms `hsm`, `hss`, and `hse`. By default, the intercept term is included.

17. Click on the `Plot` tab and check the boxes next to all plots.
18. Click `OK`.

```

*** Linear Model ***

Call: lm(formula = gpa ~ hsm + hss + hse, data = csdata,
          na.action = na.exclude)
Residuals:
    Min       1Q   Median       3Q      Max
-2.129 -0.3407  0.07568  0.4744  1.754

Coefficients:
            Value Std. Error t value Pr(>|t|)
(Intercept)  0.5899  0.2942     2.0047  0.0438
          hsm  0.1686  0.0355     4.7494  0.0000
          hss  0.0343  0.0376     0.9136  0.3599
          hse  0.0451  0.0387     1.1655  0.2425

Residual standard error: 0.6998 on 220 degrees of freedom
Multiple R-Squared:  0.2046
F-statistic: 18.86 on 3 and 220 degrees of freedom, the
p-value is 6.359e-011

```

From the report, we obtain the regression equation $\text{gpa} = 0.5899 + 0.1686\text{hsm} + 0.0343\text{hss} + 0.0451\text{hse}$. We can also obtain the F statistics 18.86 which follows an $F(3, 220)$ distribution. The P -value is essentially 0 so we can conclude that at least one of the slope coefficients is nonzero. We also see that about 20.5% of the variation in gpa is explained by this regression.

The plot of the residuals against the fitted values shows random noise about the $y = 0$ line.

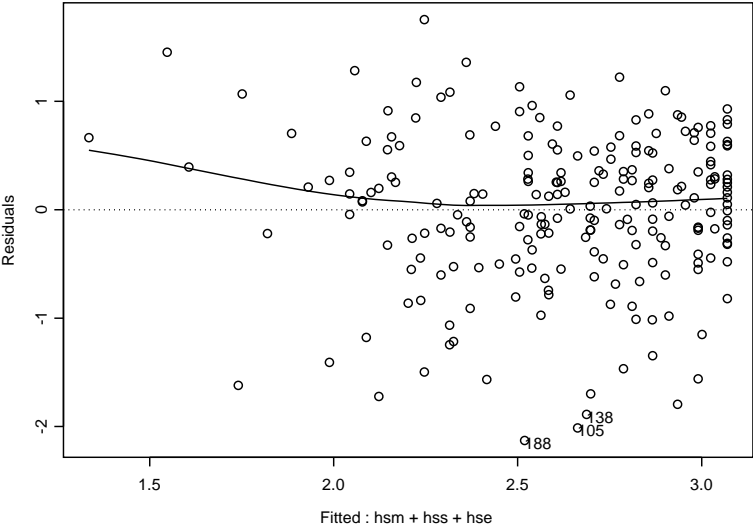


Figure 11.3

The plot of $\sqrt{\text{abs}(\text{Residuals})}$ against fitted values (not shown here) suggests that variances are slightly smaller for large fitted values, though not enough to worry about.

The plot of gpa against fitted values suggests some possible nonlinearity in the relationship. However, much of the nonlinearity is due to the left side of the figure, where there are few points, so it may just be an artifact.

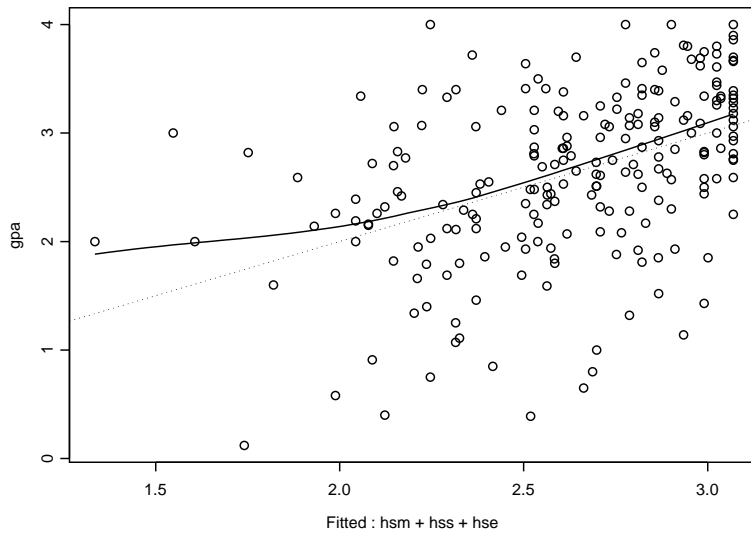


Figure 11.4

The residuals fall off the straight line on the lower left side of the normal quantile plot. This indicates some deviation from normality, though not enough to worry about, except for prediction intervals.

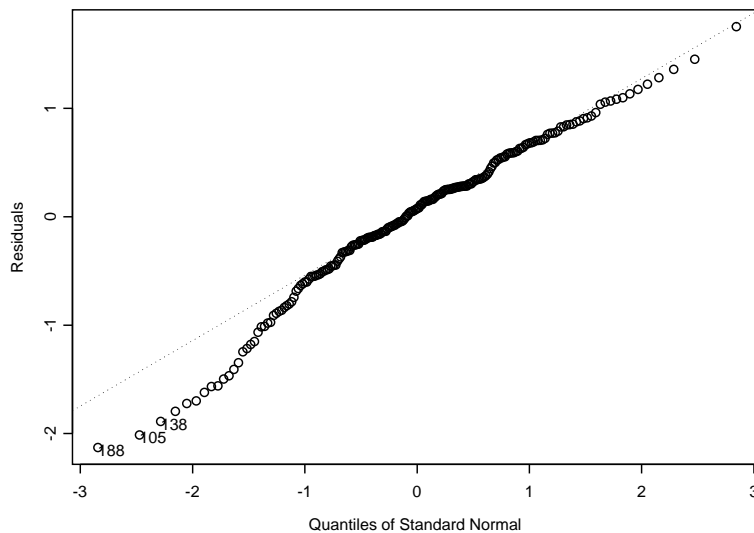


Figure 11.5

The plots of sorted **Fitted Values** and sorted **Residuals** show that there is substantially more variation in the residuals; this is because there is a low r^2 .

The plot of Cook's distance (not shown) labels three points as influential, but they are not substantially more influential than other points.

The three remaining plots, the partial residual plots (not shown), show no patterns that would be a cause for further investigation.

End of Example 11.1

Command Line Notes

```
> summary(csdata)
> cor(csdata[,-7]) # omit "sex"
```

For the regression,

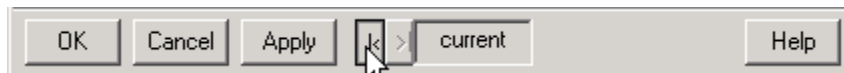
```
> fit1 = lm(gpa ~ hsm + hss + hse, data = csdata)
> summary(fit1)
> plot(fit1) # or: plot(fit1, ask = T)
```

Example 11.2 Comparing regression models

Sometimes we wish to compare two regression models to see if the simpler model fits the data as well as the more complicated model, or if the additional terms improve the fit.

1. Repeat the previous example, but in addition, type `fit1` in the **Save As:** field on the **Model** page of the **Linear Regression** dialog box.
2. Click **Apply**. This saves the results of the regression in the model object `fit1`.

A shortcut when repeating part of a previous analysis is to bring up the regression dialog, then use the back arrow at the bottom of the dialog to bring up the previous analysis.



We now consider the model $\mu_{gpa} = \beta_0 + \beta_1\text{hsm} + \beta_2\text{hss} + \beta_3\text{hse} + \beta_4\text{satm} + \beta_5\text{satv}$ (that is, we add the SAT variables to the original model).

3. On the **Model** page of the **Linear Regression** dialog box, select `gpa` for **Dependent:** and choose all of `hsm`, `hss`, `hse`, `satm`, and `satv` in the **Independent:** box.
4. Type `fit2` in **Save As:**.
5. Click **OK**.

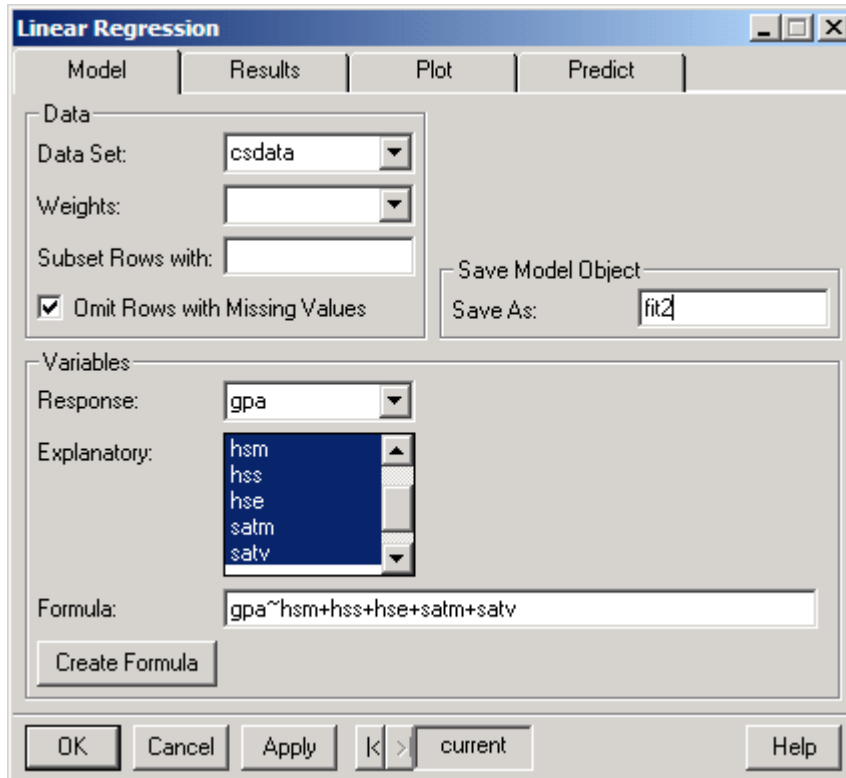


Figure 11.6

6. From the main menu choose [Statistics > Compare Models](#).
7. Select `fit1` and `fit2` in the [Model Objects](#): box.
8. Click [OK](#).

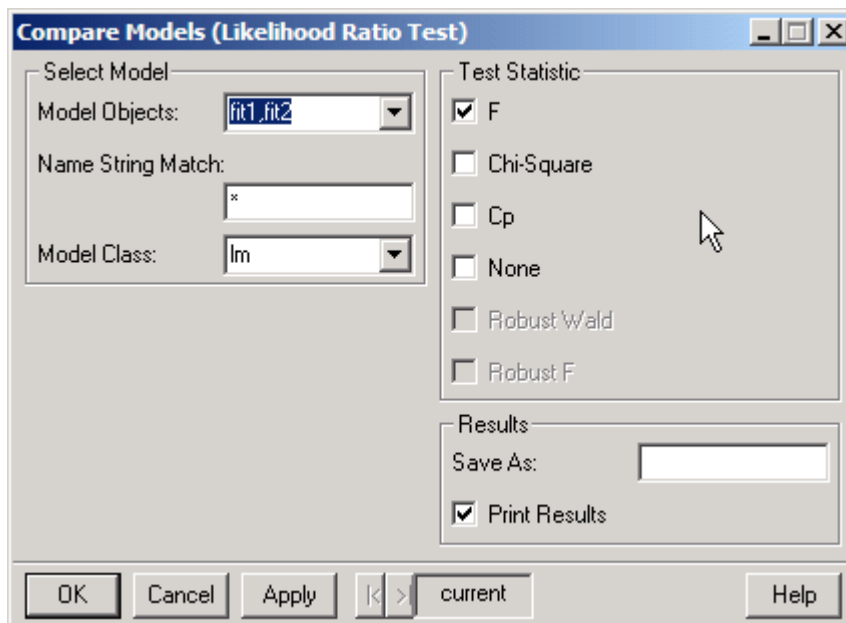


Figure 11.7

Analysis of Variance Table					
Response: gpa					
	Terms	Resid. Df	RSS		
1	hsm + hss + hse	220	107.7505		
2	hsm + hss + hse + satm + satv	218	106.8191		
	Test	Df	Sum of Sq	F Value	Pr(F)
1					
2	+satm+satv	2	0.9313136	0.9503276	0.38821

This tests to see if the two nested models fit equally well (the null hypothesis) or if adding either or both of the **SAT** variables improves the fit. The *P*-value of 0.38821 indicates that we cannot reject the null hypothesis that the simple model fits just as well as the more complicated model.

End of Example 11.2

```

Command Line Notes

The anova function compares different models.
> fit1 = lm(gpa ~ hsm + hss + hse, data = csdata)
> fit2 = lm(gpa ~ hsm + hss + hse + satm + satv,
+          data = csdata)
> # Or, use the following to create fit2 from fit1,
> # where "." is short for "like before":
> fit2 = update(fit1, . ~ . + satm + satv)
> anova(fit1, fit2)

```

11.2 Exercise

- Using the `csdata`, perform the linear regression of `gpa` against `satm` and `satv` and interpret the output.

11.3 Solution

1. From the GUI,

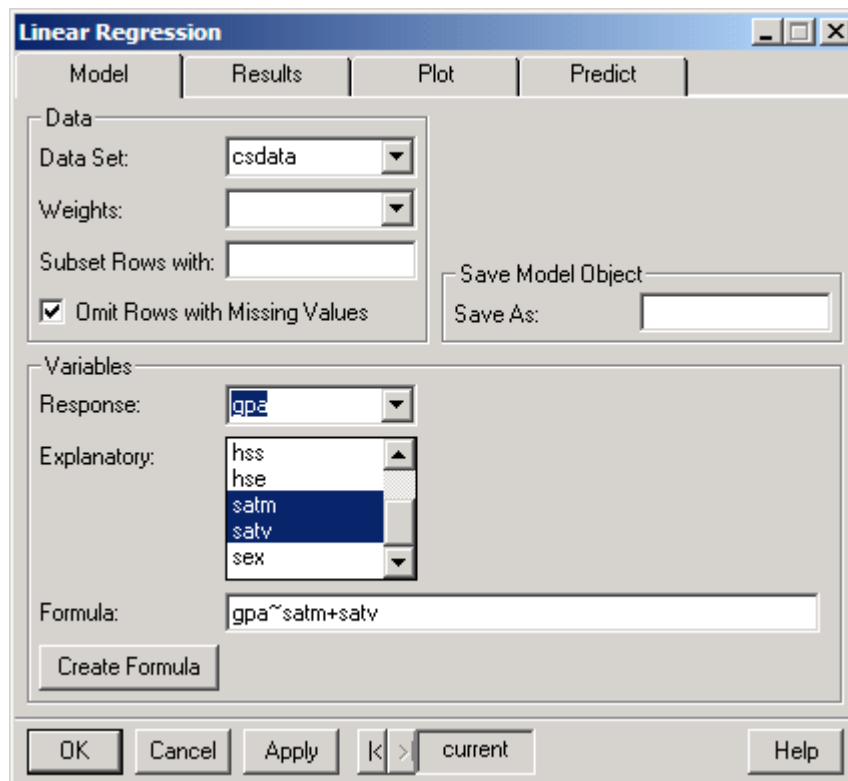


Figure 11.8

Click on the **Plot** tab and check the boxes to all desired plots. Then click **OK**.

From the command line,

```
> cs2.fit = lm(gpa ~ satm + satv, data = csdata)
> summary(cs2.fit)
> plot(cs2.fit, ask = T)
```

(partial output)

```
Coefficients:
              Value Std. Error t value Pr(>|t|)
(Intercept)  1.2887   0.3760     3.4270  0.0005
          satm  0.0023   0.0007     3.4436  0.0005
          satv  0.0000   0.0006    -0.0397  0.9683

Residual standard error: 0.7577 on 221 degrees of freedom
Multiple R-Squared:  0.06337
F-statistic: 7.476 on 2 and 221 degrees of freedom,
the p-value is 0.0007218
```

The estimated coefficients are 1.2887 for intercept, 0.0023 for `satm`, and 0.000 for `satv`. We caution that before we conclude that both slopes are 0, we should notice that the variable `gpa` ranges from 0.4 to 4.0 while the SAT scores range from 285 to 760 for the verbal, 300 to 800 for the math.

The P -value of 0.00072 for the regression indicates that at least one slope is nonzero. The P -value of 0.9684 for the t test on the `satv` slope indicates that `satv` is not significant, given that `satm` is included in the model.

The residuals against fitted values plot shows random noise about the line $y = 0$, but the normal quantile plot of the residuals hints at left-skewness in the residuals.

Chapter 12

One-Way Analysis of Variance

In Chapter 7, we used the two-sample t procedures to compare the means of two independent populations. In many settings, we wish to compare the means of three or more independent populations. The one-way analysis of variance (ANOVA) is the generalization of the two-sample t test.

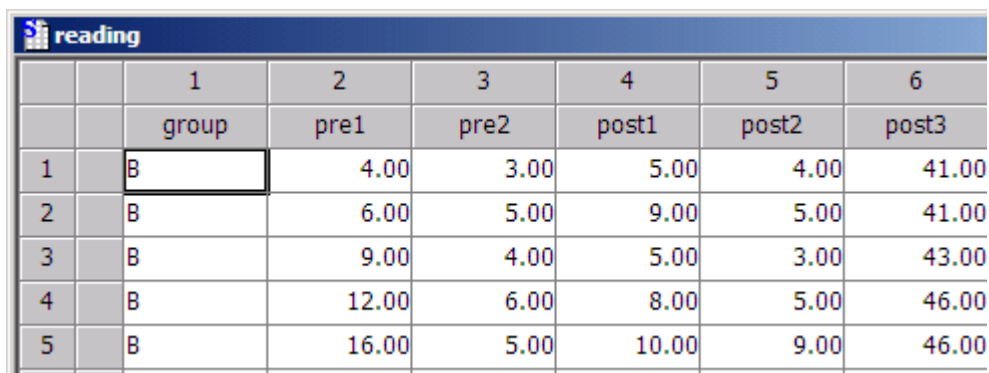
12.1 The Analysis of Variance F Test

The hypothesis we wish to test in a one-way ANOVA is $H_0 : \mu_1 = \mu_2 = \dots = \mu_k$ against $H_a : \mu_i \neq \mu_j$ for at least one pair of indices $i \neq j$, where μ_i represents the mean of a numeric variable grouped by level i . The ANOVA test is based on a distribution known as the F distribution. Analogous to the two-sample t test, a test statistic is computed, and this value is compared to an F distribution with degrees of freedom that depend on the data.

Example 12.1 Reading in children

The `reading` data set in the `IPSdata` library is described in the Data Appendix in IPS. The data is from a study of reading in children.

1. Open the `reading` data set from the `IPSdata` library.



		1	2	3	4	5	6
		group	pre1	pre2	post1	post2	post3
1		B	4.00	3.00	5.00	4.00	41.00
2		B	6.00	5.00	9.00	5.00	41.00
3		B	9.00	4.00	5.00	3.00	43.00
4		B	12.00	6.00	8.00	5.00	46.00
5		B	16.00	5.00	10.00	9.00	46.00

Figure 12.1

This data set uses the abbreviations **B** for **Basal**, **D** for **DRTA**, and **S** for **Strat**.

We begin by obtaining summary statistics for each group. In addition to being instructive in its own right, this provides one way to check the ANOVA assumptions.

2. At the menu, select [Statistics > Data Summaries > Summary Statistics ...](#). This brings up the [Summary Statistics](#) dialog.
3. In the [Data](#) box, for [Data Set:](#) enter [reading](#), and for [Variables:](#) select [pre1](#).
4. In the [Summaries by Group](#) box, for [Group Variables:](#) select [group](#).
5. Click on the [Statistics](#) tab to go the next page. Check the boxes for [Mean](#) and [Std. Deviation](#) and clear all other boxes.
6. Click [OK](#).

```
*** Summary Statistics for data in:  reading ***

group:B
      pre1
Mean: 10.50000
Std Dev.: 2.972092
-----
group:D
      pre1
Mean: 9.727273
Std Dev.: 2.693587
-----
group:S
      pre1
Mean: 9.136364
Std Dev.: 3.342304
```

The standard deviations are similar, suggesting that the ANOVA assumption of equal variances is not violated, so we continue with our ANOVA analysis.

7. At the menu, select [Statistics > ANOVA > Fixed Effects](#).
8. For [Data Set:](#) select [reading](#), for [Response:](#) select [pre1](#), and for [Explanatory:](#) select [group](#).

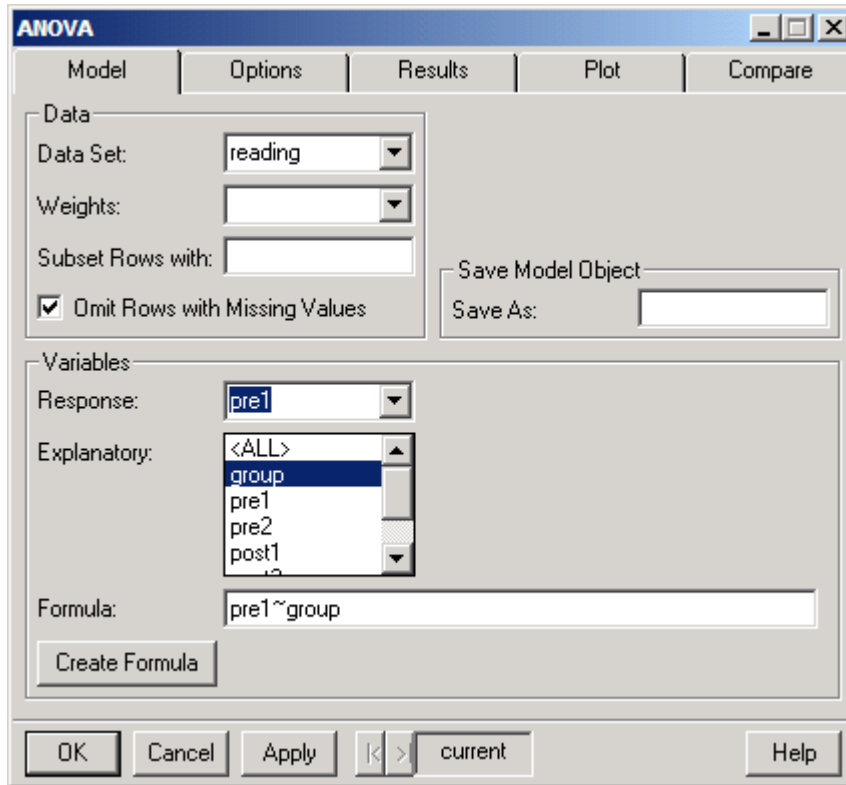


Figure 12.2

9. Click [OK](#).

```
*** Analysis of Variance Model ***

Short Output:
Call:
  aov(formula = pre1 ~ group, data = reading,
       na.action = na.exclude)

Terms:
              group Residuals
Sum of Squares 20.5758  572.4545
Deg. of Freedom      2      63

Residual standard error: 3.014395
Estimated effects are balanced

      Df Sum of Sq Mean Sq F Value Pr(F)
group  2  20.5758 10.28788 1.132206 0.3287909
Residuals 63  572.4545  9.08658
```

From the output, we see that the P -value is 0.3287, so we have no evidence to reject the null hypothesis that the reading scores are different for the three groups.

We would not expect a difference here, because the response variable was a pretest. Keep in mind, however, that 5% of similar studies will show a significant difference in pretest scores between the groups, at the 5% level.

S-PLUS also returns the Mean Square values of (using the notation in IPS) $SSG/DFG = 10.287$ and $SSE/DFE = 9.08658$.

End of Example 12.1

Command Line Notes

The `tapply` command is used to apply a function to a vector grouped by levels of a factor or several factors. The first argument is the vector whose values you are interested in. The second argument can be a factor or a list of factors. The third argument is the function.

```
> attach(reading)
> boxplot(split(pre1, group))
> tapply(pre1, group, stdev)
> detach()
> read.fit = aov(pre1 ~ group, data = reading)
> summary(read.fit)
```

12.2 Comparing the Means

The ANOVA test tells us if there is a difference or not. It does not tell us which means are different (remember only two need to differ, so all but one could all be equal). S-PLUS provides the multiple comparisons needed to answer the more specific questions on which means are different.

Example 12.2 All pairwise comparisons

We analyze the data in `fuel.frame` to see which types of cars have significantly different gas mileages.

1. Choose **Statistics > ANOVA > Fixed Effects...** For **Data Set:** enter `fuel.frame`, for **Response:** select **Fuel** (gallons to go 100 miles), and for **Explanatory:** select **Type**.
2. Click on the **Compare** tab.
3. For **Levels Of:** select **Type**, and make sure that both **Print Results** and **Plot Intervals** boxes are checked, then click on **OK**.

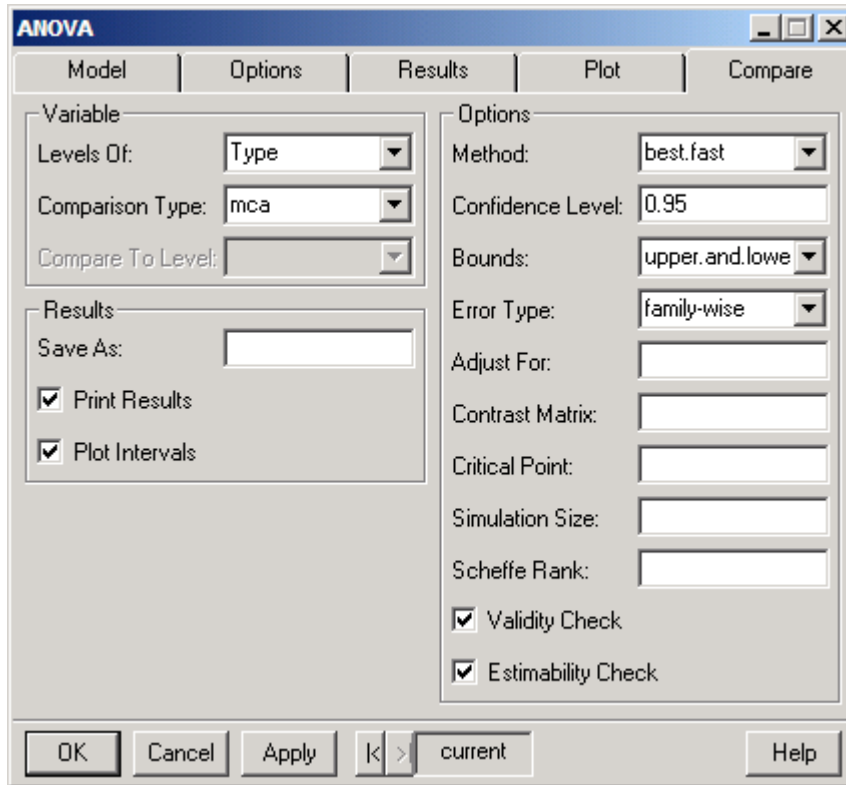


Figure 12.3

Partial output:

```

95 % simultaneous confidence intervals for specified
linear combinations, by the Tukey method

critical point: 2.9545
response variable: Fuel

intervals excluding 0 are flagged by '****'

      Estimate Std.Error Lower Bound Upper Bound
Compact-Large   -0.800    0.267   -1.590   -0.0116 ****
Compact-Medium  -0.434    0.160   -0.906    0.0387
Compact-Small    0.894    0.160    0.422    1.3700 ****
...

```

From this output we can estimate that the difference in fuel consumption between **Compact** and **Small** cars is between 0.422 and 1.37 gallons. Since this interval does not include 0, we can conclude that small cars take significantly less fuel to go 100 miles. The confidence interval comparing **Compact** and **Medium** cars does contain 0, which means that we do not have enough evidence to say that the fuel consumption is different for the two types of cars.

We could continue to look at other pairs of means and know that we are 95% confident (since we did not change the default confidence level) that all 15 comparisons contain the true difference. (S-PLUS has adjusted for the fact that we are doing 15 different comparisons and still maintains the correct confidence level; this is more accurate than doing 15 different *t* tests.)

We can also look at the confidence intervals in the graph produced. The vertical line is at 0, and any confidence interval not including 0 indicates a significant difference. The advantage of computing confidence intervals over doing only significance tests is that we can also judge if the differences are important in practical terms. (Some people may consider that 1.37 gallons is not enough to worry about even though the difference is statistically significant.)

End of Example 12.2

Remarks:

- We used the default value of `mca` for the **Comparison Type:** field, which gives all pairwise comparisons. There are two other options: `mcc` gives all comparisons with one single level (specified in **Compare To Level:**); for example, comparing each other type of car to the **Small** cars. The `none` option does no comparisons, but instead computes a confidence interval for each type of car (six confidence intervals).
- Other fields allow us to specify other options. **Method:** allows us to specify the method to use (the default chooses the best method among all those appropriate for the given data) and **Confidence Level:** allows us to choose how confident we want to be in the results.

Example 12.3 Contrasts

Sometimes it is overkill to look at all pairwise comparisons, or we want to look at a different linear combination of the means than just a simple difference. S-PLUS allows us to compute confidence intervals (and test) contrasts on our variables.

We proceed to use the data in `fuel.frame` to answer the questions: Are **Small** cars different from **Compact** cars? Are **Vans** different from **Large** cars? Is the fuel consumption of **Medium** cars equal to the average fuel consumption of **Small** and **Large** cars?

1. We first need to create a data frame with the contrasts. Each column represents a contrast and the rows represent the values to compare/combine. Click on the new data set button and enter the contrasts to match the figure below (the row and column names are not necessary, but do help in keeping the contrasts and terms straight). **Note:** the first row represents the intercept or overall mean; it is important to include this row even though we will not use it. The order of the rows is based on the order of the levels in the data (alphabetical by default).

SDF1		1	2	3
		Small.vs.Compact	Large.vs.Van	Med.vs.L.S.
1	Intercept	0.00	0.00	0.00
2	Compact	1.00	0.00	0.00
3	Large	0.00	-1.00	-0.50
4	Medium	0.00	0.00	1.00
5	Small	-1.00	0.00	-0.50
6	Sporty	0.00	0.00	0.00
7	Van	0.00	1.00	0.00

Figure 12.4

2. Choose **Statistics > ANOVA > Fixed Effects....** For **Data Set:** enter `fuel.frame`, for **Response:** select **Fuel**, and for **Explanatory:** select **Type**.

3. Click on the **Compare** tab.
4. For **Levels Of:** select **Type**, and make sure that both **Print Results** and **Plot Intervals** boxes are checked.
5. Change **Comparison Type:** to **none**. For **Contrast Matrix:** enter the name of the new data set created above. Then click on **OK**.

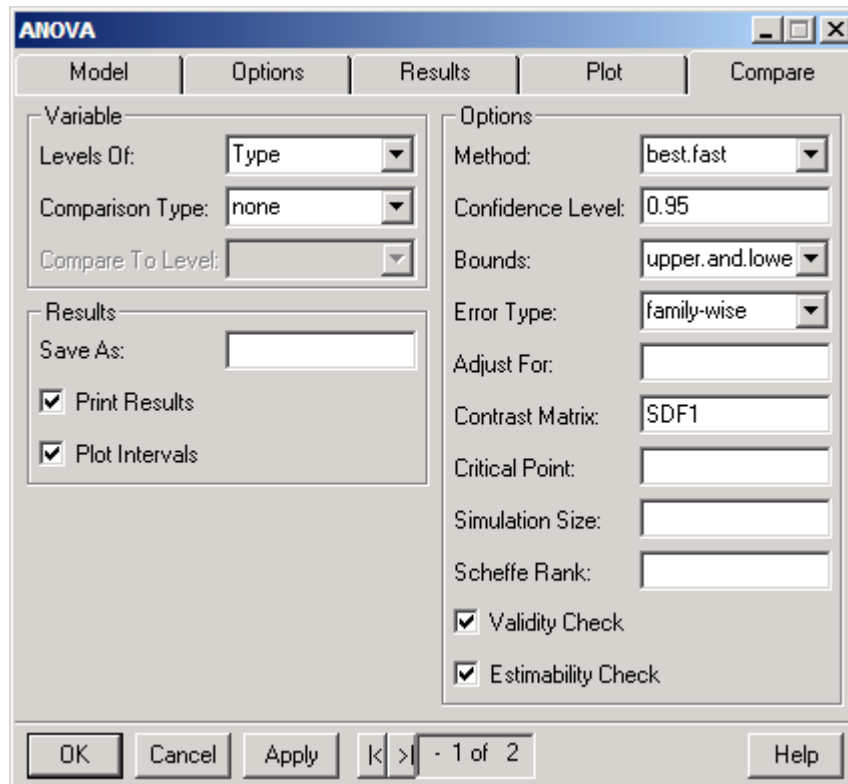


Figure 12.5

Partial results:

```

95 % simultaneous confidence intervals for specified
linear combinations, by the Tukey method

critical point: 2.41
response variable: Fuel

intervals excluding 0 are flagged by '****'

      Estimate Std.Error Lower Bound Upper Bound
Small.vs.Compact  0.894   0.160    0.5090    1.280 ****
  Large.vs.Van    0.345   0.291   -0.3560    1.050
    Med.vs.L.S    0.481   0.179    0.0499    0.912 ****

```

This shows that **Compact** cars use more fuel than **Small** cars, that we do not have enough evidence to show that **Large** cars and **Vans** differ, and that **Medium** cars use more fuel than the average of **Small** and **Large** cars.

Compare the line for **Small** vs. **Large** cars in this output to the output for all pairwise comparisons. The estimate and standard error are the same, but the interval is narrower, because we only needed to adjust the critical point for 3 comparisons rather than 15.

End of Example 12.3

Command Line Notes

```
The function multicomp does multiple comparisons and contrasts.  
> fuel.fit = aov(Fuel ~ Type, data = fuel.frame)  
> multicomp(fuel.fit, plot = T)  
> temp.mat = cbind(SvC = c(0,1,0,0,-1,0,0),  
+                 LvV = c(0,0,-1,0,0,0,1),  
+                 MvLS = c(0,0,-1/2,1,-1/2,0,0))  
> multicomp(fuel.fit, comparisons = "none", lmat = temp.mat)
```

12.3 Exercises

1. For the reading comprehension data (the `reading` data set used earlier in this chapter), compare the results of a reading comprehension test (`post3`) for the three reading groups.
2. Test the two hypotheses $H_{01} : \frac{1}{2}(\mu_D + \mu_S) - \mu_B = 0$ and $H_{02} : \mu_D - \mu_S = 0$ for the reading comprehension test.

12.4 Solutions

1. Check the sample standard deviations. From the [Statistics > Data Summary > Summary Statistics...](#) command, we obtain the mean and standard deviations for each color. Because

$$\frac{\text{largest sd}}{\text{smallest sd}} = \frac{7.388}{5.636} = 1.31 < 2, \text{ we can safely use ANOVA.}$$

Select [Statistics > Anova > Fixed Effects...](#) from the menu and fill out the dialog box:

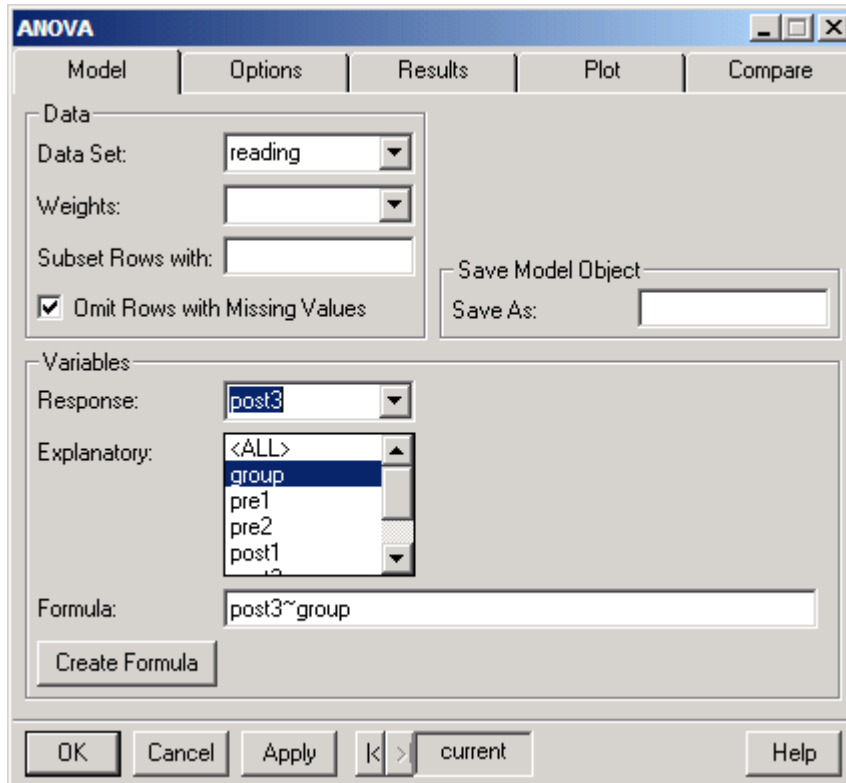


Figure 12.6

The large F value of 4.48 and the small P -value of 0.01515 leads us to conclude that the means are different.

Here are some of the commands that replicate the above results.

```
> attach(reading)
> tapply(post3, group, stdev)
> read2.fit = aov(post3 ~ group, data = reading)
> summary(read2.fit)
```

2. Add the following changes to the above solution:

		1	2
		B.vs.D.S.	D.vs.S
1	Int	0.00	0.00
2	B	-1.00	0.00
3	D	0.50	1.00
4	S	0.50	-1.00
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			

ANOVA

Model | Options | Results | Plot | Compare

Variable:

Levels Of:

Comparison Type:

Compare To Level:

Options:

Method:

Confidence Level:

Bounds:

Error Type:

Adjust For:

Contrast Matrix:

Critical Point:

Simulation Size:

Scheffe Rank:

Validity Check

Estimability Check

Results:

Save As:

Print Results

Plot Intervals

OK Cancel Apply < > - 1 of 5 Help

Figure 12.7

```
> multcomp(read2.fit, comparisons = "none",
+          lmat = cbind(BvDS=c(0,-1,.5,.5), DvS=c(0,0,1,-1)) )
```

Chapter 13

Two-Way Analysis of Variance

In the previous chapter, we compared the means of several populations. In this chapter, we compare the means of populations that are classified in two ways. We assume that the data are approximately normal and that the standard deviations for the different groups are equal.

13.1 Preliminary Data Analysis

Before doing any ANOVA, we need to take a preliminary look at the data. Do the samples come from populations with the same standard deviation? Is there any indication of interactions between the two explanatory variables? We will perform our exploratory analysis on the data set presented in Example 13.11 of IPS.

Example 13.1 Cardiovascular risk factors

Two hundred subjects consisting of runners and those who described themselves as sedentary (the control group) had their heart rate measured after six minutes of exercise on a treadmill.

The factor (categorical) variable `group` has two levels, Control and Runners; the factor variable `sex` has two levels, Male and Female. All combinations appear, so we have four populations to consider: Control-Female, Control-Male, Runners-Female, Runners-Male.

1. Open the data set `Example13.11.txt` from the `IPSdata` library.
2. At the menu, select `Statistics > Data Summaries > Summary Statistics...`
3. For `Data Set` enter `Example13.11`. For `Variables:` select `beats`.
4. For `Group Variables:`, select `group` and `sex`.
5. On the `Statistics` page, check `Std. Deviation` and clear all other options.
6. Click `OK`.

From the report, we see that the largest standard deviation, 17.10 for Control-Male, is less than twice the smallest standard deviation, 12.49 for the Runners-Male. Thus, we may proceed with our analysis.

Next, we take a graphical look at the data.

7. At the menu, select `Statistics > Design > Factor Plot...`
8. For `Data Set:` enter `Example13.11`.
9. In the `Variables` box, for `Response:` select `beats` and for `Explanatory:` select `group` and `sex`.
10. In the `Layout` box, change the number of `Columns:` to 2.

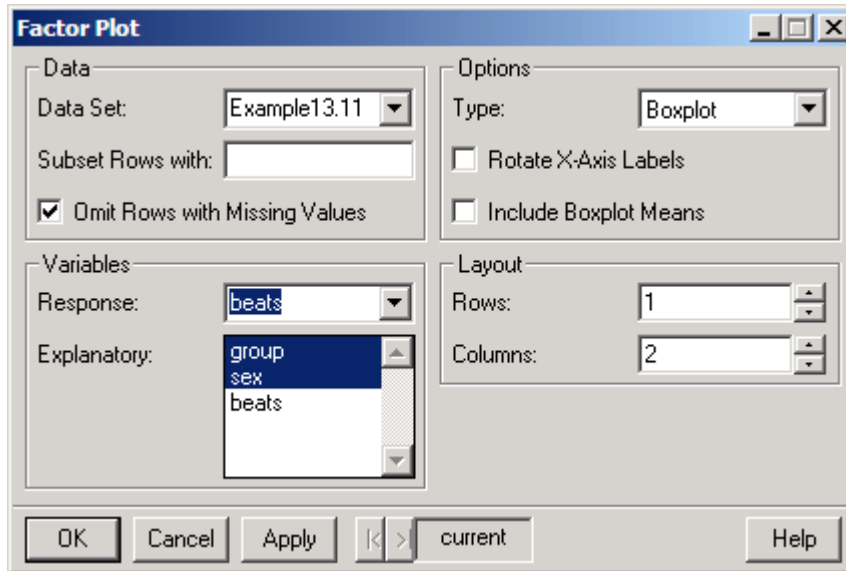


Figure 13.1

11. Click [OK](#).

This gives two boxplots, of heart rate grouped by gender and by experimental group. We can compare medians, spread, and skewness between the different groups. We note that in the control group, there are three outliers, and in the runners group, one outlier.

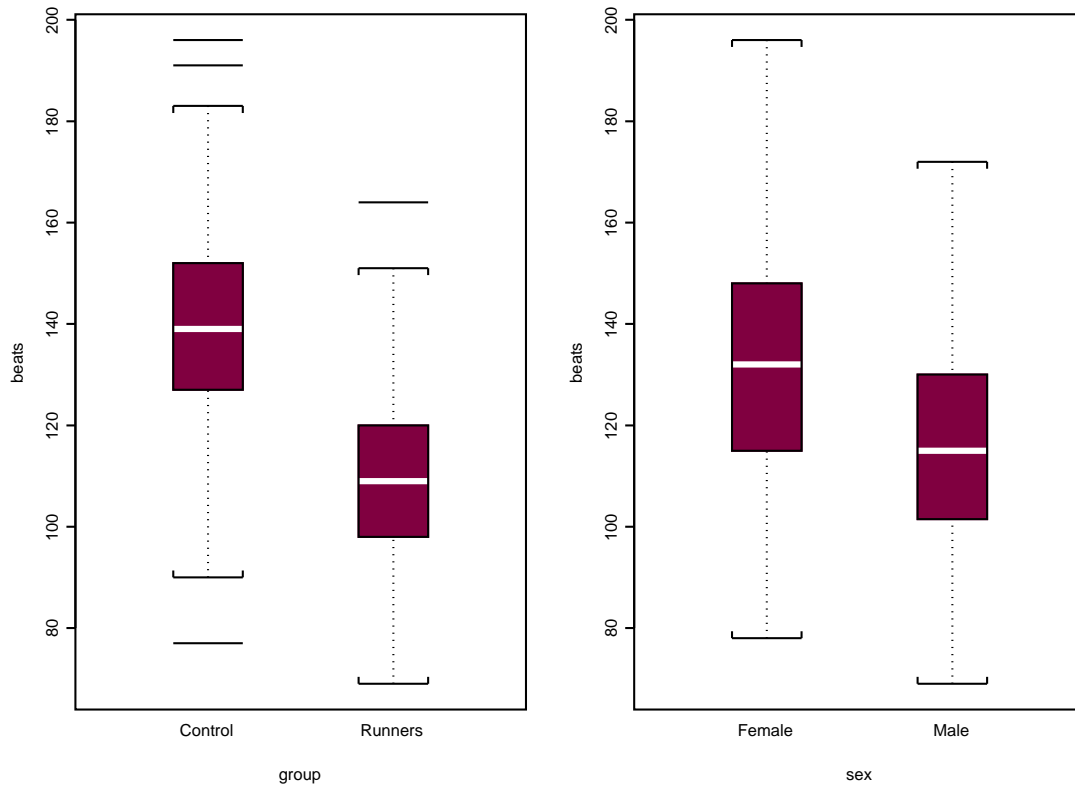


Figure 13.2

Remark: We can also obtain the side-by-side boxplots by using the [Plots 2D Palette](#) and clicking on the [Box](#) button (see [Section 1.2](#)).

12. At the menu select [Statistics > Design > Design Plot...](#)
13. For [Data Set:](#) enter [Example13.11](#).
14. In the [Variables](#) box, for [Response:](#) select [beats](#) and for [Explanatory:](#) select [group](#) and [sex](#).

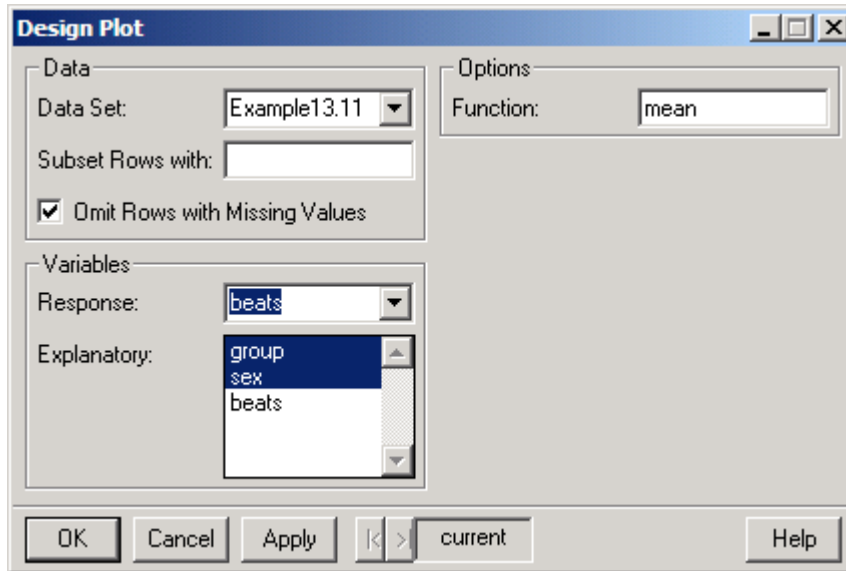


Figure 13.3

15. Click [OK](#).

This plot allows us to compare the mean heart rates for each group. The horizontal line at approximately $y = 125$ is the overall mean for [beats](#).

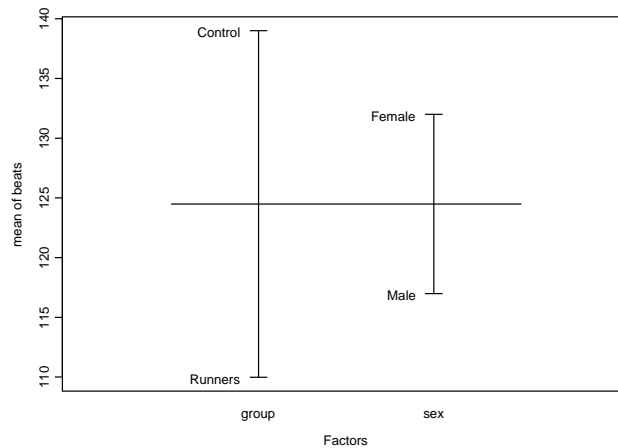


Figure 13.4

We next assess interactions between [group](#) and [sex](#).

16. At the menu, select [Statistics > Design > Interaction Plot...](#)

17. For [Data Set](#): enter [Example13.11](#).

18. In the [Variables](#) box, for [Response](#): select [beats](#) and for [Explanatory](#): select [group](#) and [sex](#).

19. Under [Options](#), check [Both Orderings for Each Pair](#).

20. Under [Layout](#), change the number of [Columns](#): to [2](#).

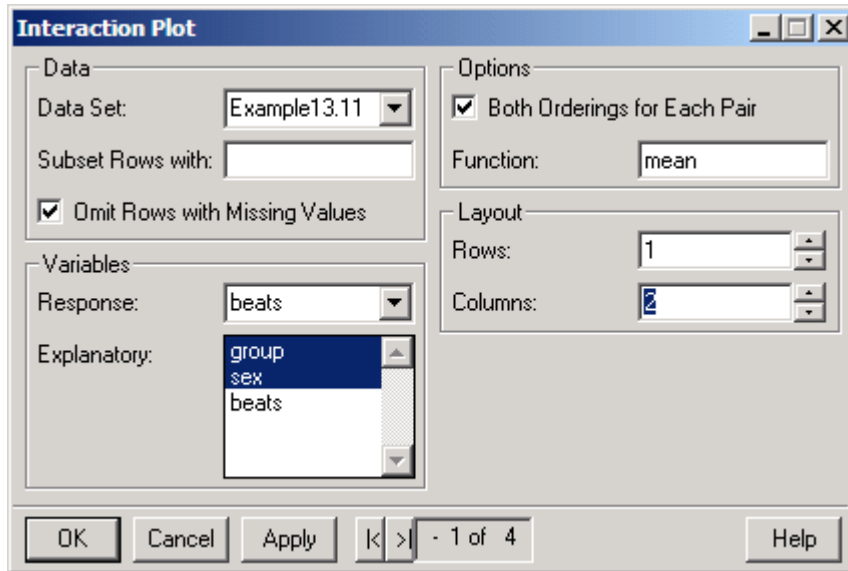


Figure 13.5

21. Click [OK](#).

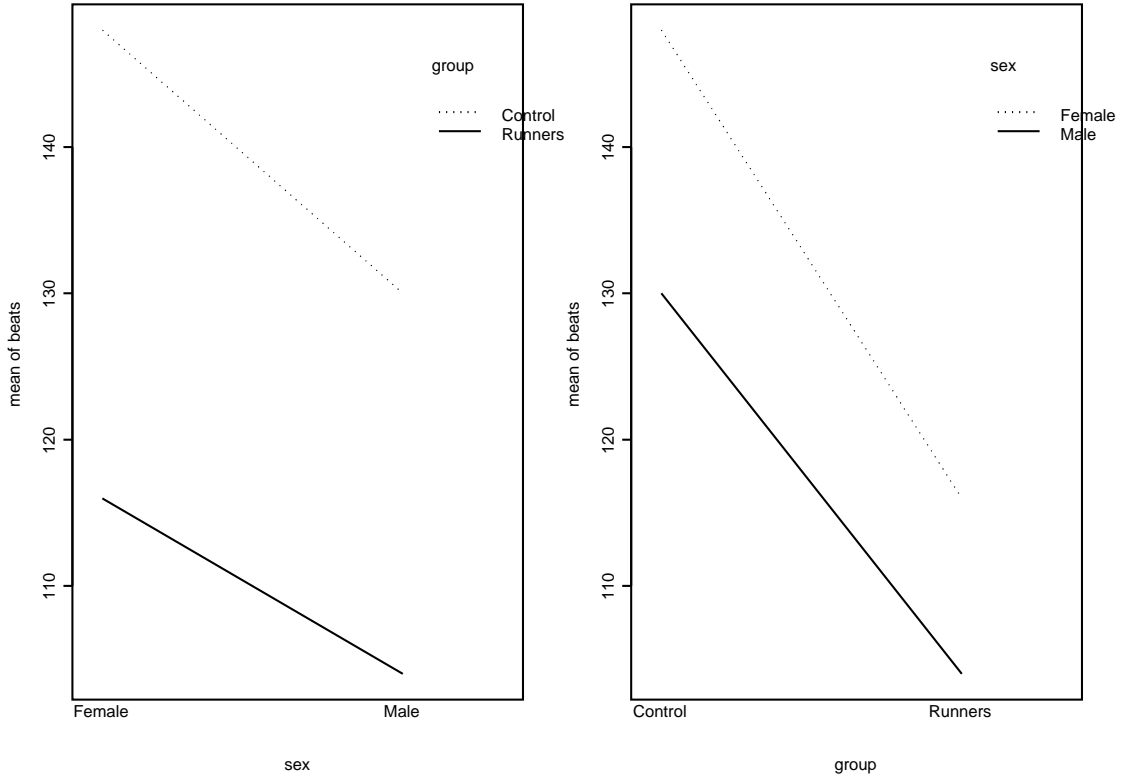


Figure 13.6

The endpoints of each line in this plot are at the mean of `beats` for the relevant group: for instance, in the left plot above, the upper endpoint for the dotted line indicates that for females in the control group, the mean heart rate is approximately 148.

The two lines in each graph are not quite parallel, indicating some interaction between the explanatory variables `group` and `sex`.

End of Example 13.1

13.2 Two-Way ANOVA

We will consider a model with interaction in the two-way ANOVA.

Example 13.2 Cardiovascular risk, continued

1. At the menu, select `Statistics > ANOVA > Fixed Effects...`
2. For `Data Set:` enter `Example13.11`.
3. In the `Variables` box, for `Response:` select `beats` and for `Explanatory:` select `group` and `sex`.
4. To include the interaction term, add the term `group:sex` in the `Formula:` field. The complete formula should read:

```
beats ~ group + sex + group:sex
```

The model formula indicates that `beats` is the response variable, that `group` and `sex` are included as main effects, and that an interaction between `group` and `sex` is included (`group:sex`).

5. Click `OK`.

```
*** Analysis of Variance Model ***

Short Output:
Call:
  aov(formula = beats ~ group + sex + group : sex,
      data = Example13.11, na.action = na.exclude)

Terms:
              group          sex group:sex Residuals
Sum of Squares 168432.1  45030.0   1794.0 192729.8
Deg. of Freedom      1          1         1      796

Residual standard error: 15.5603
Estimated effects are balanced

              Df Sum of Sq Mean Sq F Value    Pr(F)
group         1 168432.1 168432.1 695.6470 0.000000000
sex           1  45030.0  45030.0 185.9799 0.000000000
group:sex     1   1794.0   1794.0   7.4095 0.006629953
Residuals   796 192729.8    242.1
```

From the output, we see that the F value for the `group` main effect is 695.647, which is highly significant. The F -values and the P -values for `sex` and the interaction term `group:sex` are also significant.

End of Example 13.2

Remark: The `ANOVA` dialog box has options for producing diagnostic plots and making comparisons between the levels of the various categorical variables. These plots and comparisons are typically covered in a more advanced course on ANOVA.

Command Line Notes

For the standard deviations grouped by `group` and `sex`, use the `tapply` command.

```
> attach(Example13.11)
> tapply(beats, list(group, sex), stdev)
```

For the preliminary plots, use the `plot.factor`, `plot.design`, and `interaction.plot` commands.

```
> par(mfrow = c(1,2))
> plot.factor(beats ~ group + sex)
> graphsheet() # new graph sheet
> plot.design(beats ~ group + sex)
> graphsheet()
> par(mfrow = c(1,2))
> interaction.plot(group, sex, beats)
> interaction.plot(sex, group, beats)
> detach()
```

The `par` command sets layout parameters. In the above, we specified a layout of one row and two columns.

The command `aov` can be used to perform two-way ANOVA.

```
> heart.fit = aov(beats ~ group + sex + group:sex,
+               data = Example13.11)
> summary(heart.fit)
```

13.3 Exercise

1. Exercise 13.19 in IPS describes a study to compare certain biological materials for use as new tissue. The data are in `Exercise13.19` in the `IPSdata` library. The `group` variable indicates the type of material used, the `material` variable is a numeric version of the `group` variable, and the `weeks` variable is when the response was measured. The `gpi` variable is the response, the percent of glucose phosphated isomerase (Gpi) cells in the region of a wound.

Run the two-way ANOVA and include any preliminary analyses.

Note: The explanatory variables must be factors (see Section 0.4.5). You will need to convert `weeks` to a factor.

13.4 Solution

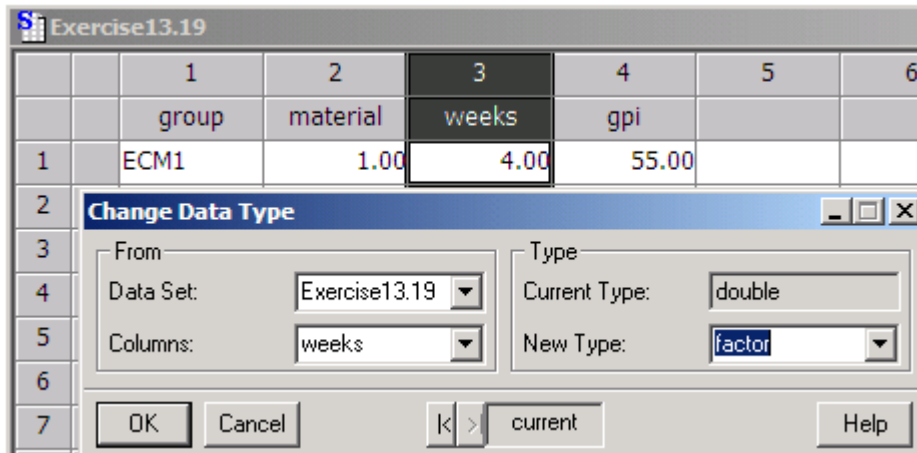


Figure 13.7

Because there are only three data values for each level of the variable `Material1`, a boxplot of `Gpi` grouped by `Material1` would not be informative. We look at the means using the `Design Plot...` option.

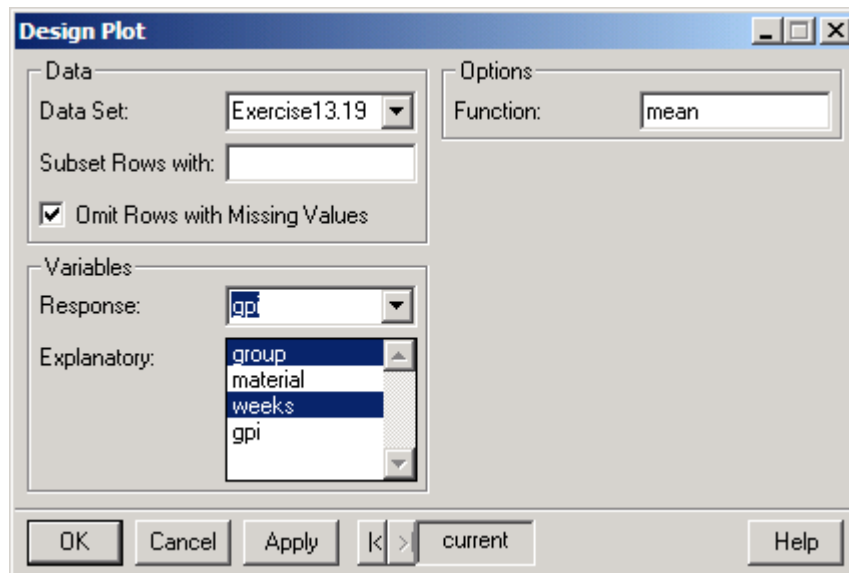


Figure 13.8

That plot indicates that `week` has little effect, but the type of material has a strong effect. An `Interaction Plot...` option shows the same information and shows there are possible interactions, albeit weak.

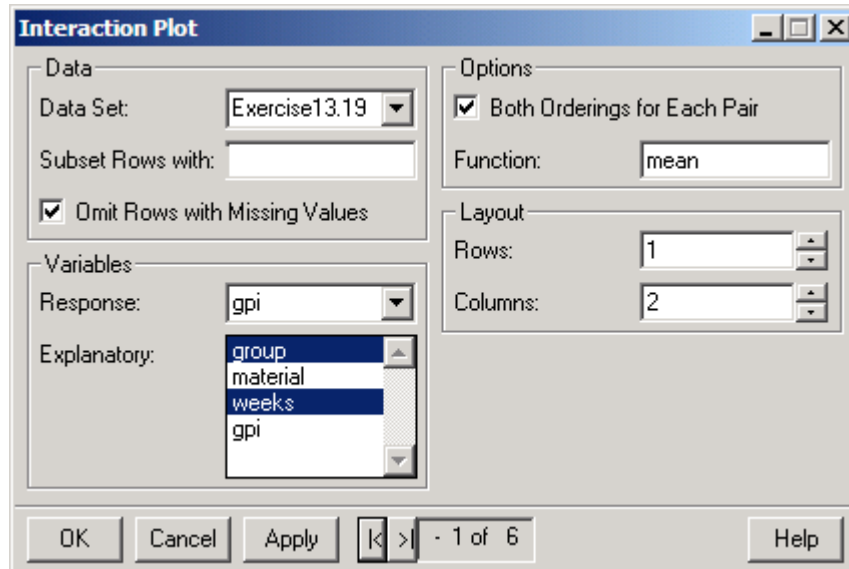


Figure 13.9

The numerical calculations are done from the [ANOVA](#) menu. They show that both [weeks](#) and the interaction are not significant.

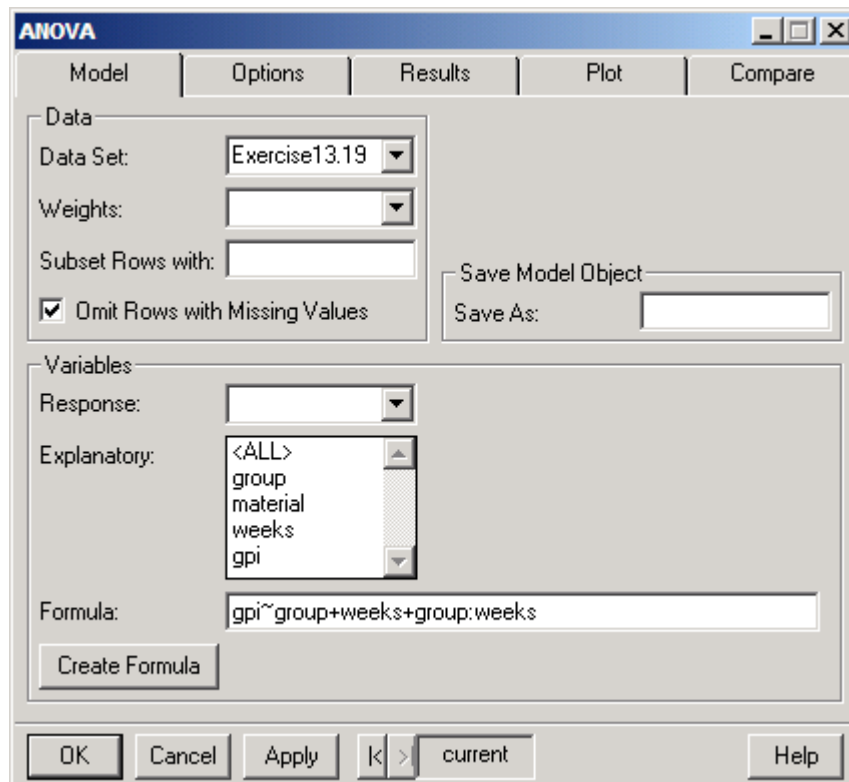


Figure 13.10

From the command line,

```
> Exercise13.19$weeks = as.factor(Exercise13.19$weeks)
> attach(Exercise13.19)
```



```
> par(mfrow = c(1,2))
> plot.factor(gpi ~ group + weeks)
> graphsheet()
> plot.design(gpi ~ group + weeks)
> interaction.plot(group, weeks, gpi)
> detach()
> gpi.fit = aov(gpi ~ group + weeks + group:weeks,
+             data=Exercise13.19)
> summary(gpi.fit)
> detach()
```

Chapter 14

Bootstrap Methods and Permutation Tests

This chapter describes how to use resampling methods, the bootstrap and permutation tests, to calculate standard errors, to determine whether distributions are close enough to normal to use t tests and confidence intervals, to calculate confidence intervals, and for significance testing.

The techniques in this chapter can be applied to problems found in other chapters, especially Chapters 7, 8, 10, and 11. This chapter can supplement those chapters, providing an alternate approach with certain advantages, including a consistent approach that applies to a wide variety of problems (obviating the need for a cookbook of formulas to apply in different cases), and concrete graphical representations of some of the concepts that students find most difficult, including sampling distributions, standard errors, confidence intervals, and P -values.

Most of this chapter is laid out according to the type of problem to be handled, starting with the one- and two-sample means problems of Chapter 7. You may choose to do the beginning of this chapter after Chapter 7, then read additional parts as supplements to later chapters when you come to them.

Additional sources of information: There are some additional sources about resampling in connection with IPS besides this chapter. When the [IPSData](#) library is loaded (see Section 0.4.2) it adds a menu to the right of the main menu bar with four options, of which the first three are informational:

IPSData Library User's Manual is a guide to use the GUI menus for doing bootstrap and permutation test calculations.

Example Code for Exercises contains example command-line code for all exercises in IPS Chapter 14.

IPSData help is primarily a listing of the data sets in the library and the variables they contain.

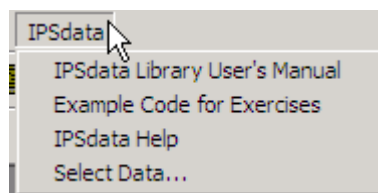


Figure 14.1

Loading the [IPSData](#) automatically loads the [resample](#) library, which adds an additional manual at the main menu under [Help > Online Manuals > Resample Library User's Manual](#) and online help files for resampling functions and the resampling GUI at [Help > Available Help > resample](#).

You can also load the [resample](#) library separately (for analyzing data not in [IPSData](#)), see Section 0.4.3 or 0.7.1.

14.1 One and Two Means (Chapter 7)

In Chapter 7 we looked at confidence intervals and significance tests for a single mean, for the mean difference of matched pairs, and for the difference of two independent means, using t tests and confidence intervals. In this section we look at using resampling for five of those six situations—everything except significance tests for a single mean. We also do three things that are not possible with t tests and intervals:

- use robust alternatives like trimmed means in place of simple means,
- accurate handling nonnormal populations with small samples (and knowing when this is the case), and
- confidence intervals and tests for the ratio of means.

14.1.1 One mean

Example 14.1 Corn soy blend (CSB)—IPS Example 7.1

Researchers wished to evaluate appropriate vitamin C levels (mg/100gm) in a random sample of eight batches of CSB from a production run.

Our goal is to find a 95% confidence interval for the mean vitamin C content of CSB produced during this run.

In Chapter 7 we did this using t confidence intervals, based on the standard error s/\sqrt{n} . Here we use a bootstrap approach.

1. Open the `IPSdata` library (Section 0.4.2). This automatically opens the `resample` library.
2. Open the `Example7.1` data set.
3. At the menu, select `Statistics > Compare Samples > One Sample > t Test/Resample...`
4. For `Data Set` enter `Example7.1`, and for `Variable:` select `vitc`.
5. In the `Hypotheses` box, for `Mean Under the Null Hypothesis:` enter `40` and for `Alternative Hypothesis:` select `two.sided`.

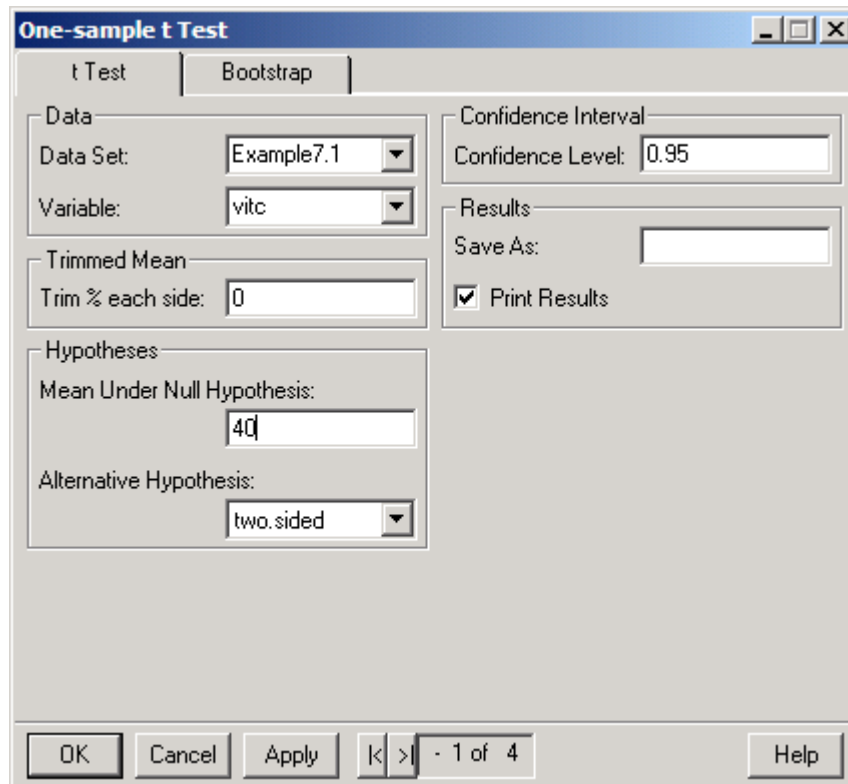


Figure 14.2

6. Select the [Bootstrap](#) tab, and check the box to [Perform Bootstrap](#).
7. In the [Summary Results](#) box, check the box for [t Intervals using Bootstrap SE](#).

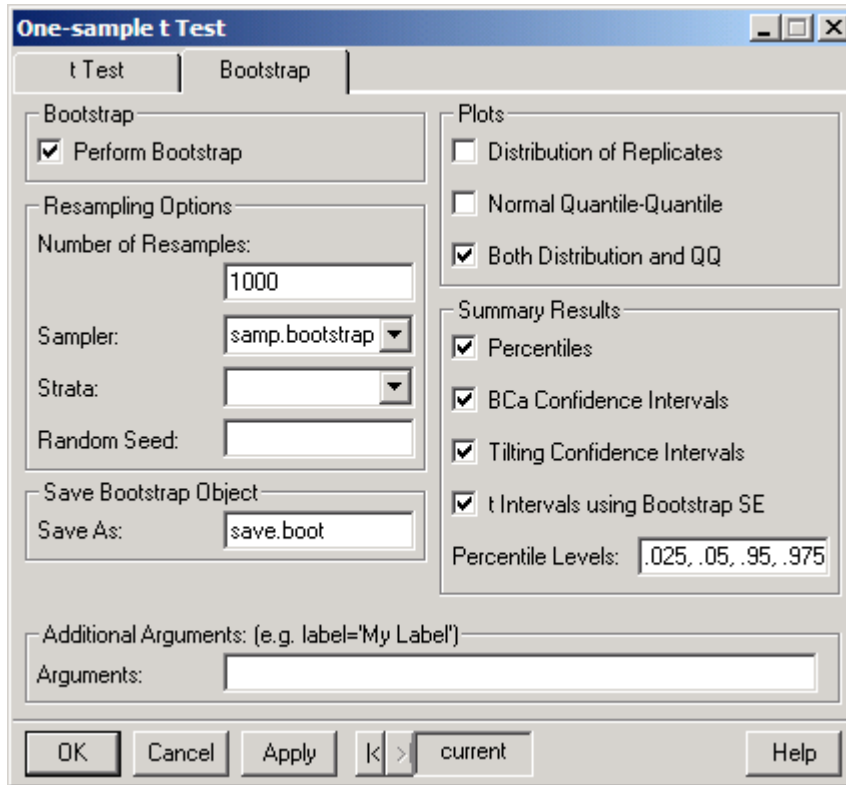


Figure 14.3

8. Click [OK](#).

bootstrap : Example7.1\$vitc : mean

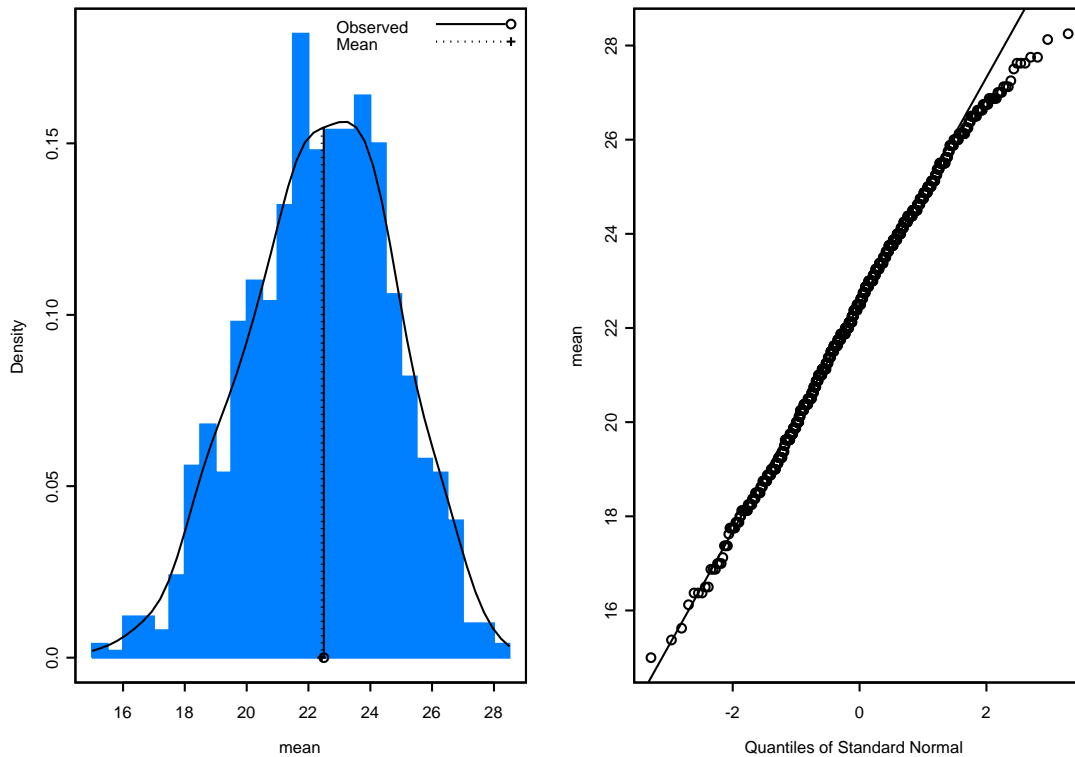


Figure 14.4

The plots are a histogram and normal quantile plot of the *bootstrap distribution*. This is the distribution of means of many random samples. The key things to look for are spread and normality.

First, the spread of the bootstrap distribution gives an idea of the uncertainty in the statistic—how much does the statistic (the mean) vary when the data it uses come from random sampling?

The middle is about 22.5 (the statistic, the mean of the original data), but there is a fairly substantial spread. We get a qualitative sense of the uncertainty just looking at the bootstrap distribution. We can also quantify the uncertainty in two ways: (1) the middle 95% of the bootstrap distribution gives a quick-and-dirty 95% confidence interval, and (2) computing the standard deviation.

That standard deviation measures the uncertainty in the statistic; in other words, it is a *standard error* (SE) for the statistic. This may give you a better idea what a standard error means—it is a standard deviation for the uncertainty of a statistic. With the bootstrap we compute the standard error directly, from the bootstrap distribution. With a formula like s/\sqrt{n} we do so indirectly, based on statistical theory.

The second thing to look for is that the bootstrap distribution is quite normal—we see this in both the histogram and the normal quantile plot. Hence, even with the small sample size, Chapter 7 methods based on normality should be accurate.

Now turn to the printed results, which include:

Summary Statistics:				
	Observed	Mean	Bias	SE
mean	22.5	22.48	-0.0225	2.45
Percentiles:				
	2.5%	5%	95%	97.5%
mean	17.5	18.25	26.25	27.125
T Confidence Intervals using Bootstrap Standard Errors:				
	2.5%	5%	95%	97.5%
mean	16.69778	17.85287	27.14713	28.30222

The **observed** value is the mean of the original data; the **SE** is the standard error, 2.45. For comparison, the formula SE is $s/\sqrt{n} = 2.54$. The bootstrap provides a way for you to check if you calculated the standard error correctly using the formula.

The range of the middle 95% of the bootstrap distribution, between the 2.5% and 97.5% percentiles, (17.5, 27.1) is a quick-and-dirty confidence interval, and provides a way to check formula confidence intervals. For comparison, the 95% t confidence interval is (16.5, 28.5) (this quick-and-dirty bootstrap interval tends to be too narrow for very small samples).

A second quick-and-dirty way to use bootstrapping for confidence intervals is to do a t interval, but using the bootstrap SE in place of a formula SE. This is labeled here as **T Confidence Intervals using Bootstrap Standard Errors** (called *bootstrap t* intervals in IPS), and is (16.7, 28.3), which is fairly close to the formula t interval.

We can use the same procedures for one-sided confidence intervals. Here a one-sided 95% percentile interval is the bottom 95% of the bootstrap distribution, $(-\infty, 26.2)$. For comparison, the formula t interval is $(-\infty, 27.3)$, and the t interval using the bootstrap standard error is $(-\infty, 27.1)$.

In this problem, all the intervals gave roughly the same results, so we would not have to use the bootstrap. The main things the bootstrap gives us here are assurance that it is OK to use t intervals (because the bootstrap distribution is normal) and a concrete picture of the variability of the statistic.

```

Command Line Notes
> save.boot = bootstrap(Exercise7.1$vitc, mean)
> save.boot # print the results
> plot(save.boot)
> qqnorm(save.boot)
> limits.percentile(save.boot)
> limits.t(save.boot)

```




End of Example 14.1

Remark: Resampling is random, so the plots and numbers you create will differ from ours. But they should not be much different, unless you use less than 1000 resamples.

Example 14.2 CLEC data

For comparison, here is a situation in which t intervals and confidence intervals should not be used because the underlying distribution is not normal, and the sample size is not large enough to allow use of t methods in spite of the non-normality.

The **CLEC** data is a subset of the **Verizon** data set described in the chapter. This data set consists of 23 observations from a very skewed distribution. To bootstrap the distribution of the mean, using the GUI:

1. Open the [CLEC](#) data set (Section [0.4.1](#)).
2. Begin by plotting the data, using a histogram  and normal quantile plot  from the [2D Plots](#) palette . Note that the data are very skewed. You should make the histogram bars narrower, for example width 5 (see Section [1.1.2](#)).
3. At the menu, select either [Statistics > Compare Samples > One Sample > t Test/Resample...](#) or (equivalently) [Statistics > Resample > One sample t.](#)
4. For [Data Set](#): enter [CLEC](#). For [Variable](#): select [Time](#).
5. Select the [Bootstrap](#) tab, and check the box to [Perform Bootstrap](#).
6. In the [Summary Results](#) box, check the box by [t Intervals using Bootstrap SE](#).

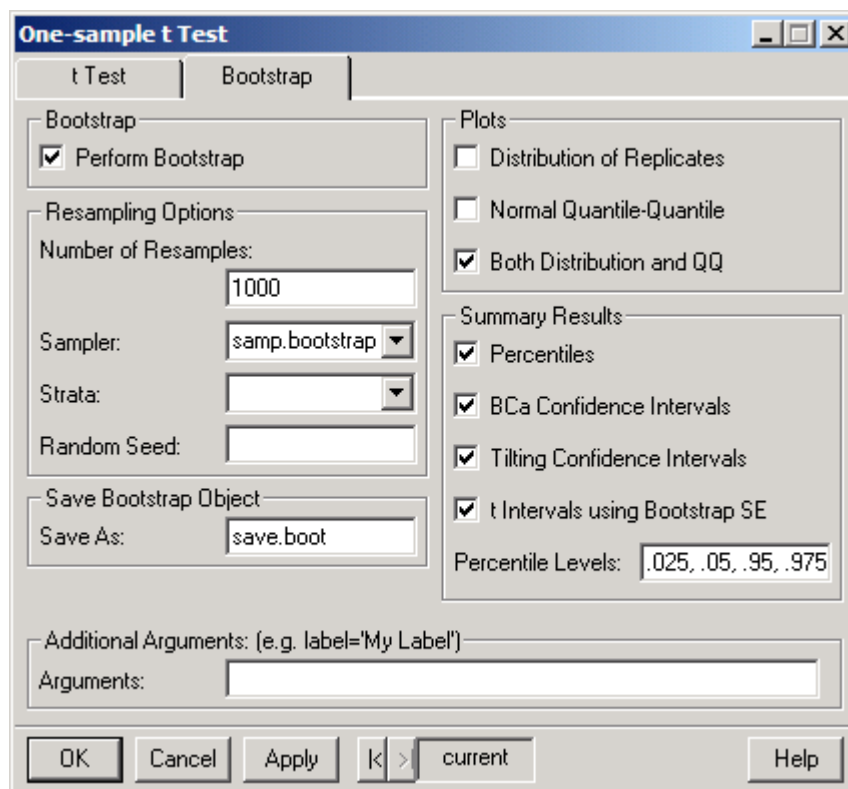


Figure 14.5

7. Click [OK](#).

bootstrap : CLEC\$Time : mean

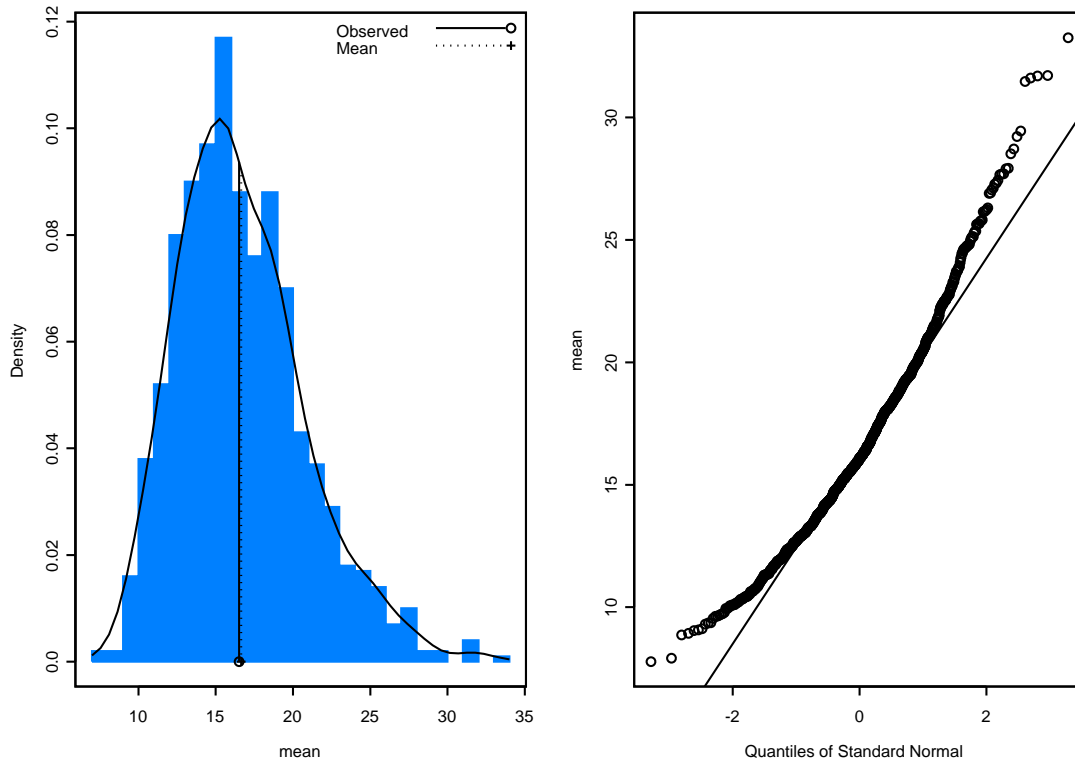


Figure 14.6

The bootstrap distribution shows substantial skewness, indicating that t intervals or significance tests would be inaccurate. Note that while this amount of skewness in data might be acceptable, it is not acceptable in a sampling distribution, where the central limit theorem has already had its chance to work. Any differences from normality here translate directly into errors if you use t confidence intervals or significance tests.

The printed results include t confidence intervals and significance tests (not shown here), saddlepoint inferences (which you may ignore), and the following bootstrap results:

```

*** Bootstrap Results ***
Call:
bootstrap(data = CLEC$Time, statistic = mean,
          trace = F, save.indices = T)

Number of Replications: 1000

Summary Statistics:
      Observed Mean   Bias   SE
mean    16.51 16.65 0.1394 4.092

Percentiles:
      2.5%      5%      95%   97.5%
mean 10.10785 10.74915 24.62728 26.15751
:

T Confidence Intervals using Bootstrap Standard Errors:
      2.5%      5%      95%   97.5%
mean 8.011573 9.475426 23.54284 25.00669

```

The printout includes the following items:

Call is the command you could give to create this bootstrap object from the command line. (Note that you would not get exactly the same results, because you would be using different random numbers.)

Observed is the mean of the original data, 16.51.

Mean is the mean of the bootstrap distribution (that is, the average of the 1000 bootstrap means).

Bias is the estimated bias, **Mean** – **Observed**.

SE is the bootstrap standard error.

Percentiles are percentiles of the bootstrap distribution. The interval (10.1, 26.2) is the 95% *bootstrap percentile confidence interval*.

(Other confidence intervals) The BCa and Tilting confidence intervals are omitted here.

T Confidence Intervals using Bootstrap SE are called *bootstrap t confidence intervals* in IPS. These are $16.51 \pm t * 4.092$.

Note that the bootstrap percentiles and the *t* confidence intervals are quite different; this is another indication that it would not be appropriate to use *t* confidence intervals. Instead, we would prefer the percentile intervals, or better yet either of the other two intervals (not shown), the BCa or Tilting intervals.

Command Line Notes The basic function for one-sample bootstrapping is `bootstrap`, for which the first two arguments are the data to bootstrap, and the statistic to apply to each bootstrap sample.

```
> save.boot = bootstrap(data = CLEC$Time, statistic = mean)
> save.boot
> par(mfrow = c(1,2)) # 1 row, 2 columns
> plot(save.boot)
> qqnorm(save.boot)
> par(mfrow = c(1,1)) # restore default layout
```

There are other arguments you can pass to `bootstrap`, including `label` (used for printing and plotting) `seed` (to make results reproducible), and `save.indices` (to speed up computing confidence intervals). For a more complete list see

```
> help(bootstrap)      # most common arguments
> help(bootstrap.args) # more arguments, and more detail
```

To compute confidence intervals, do for example:

```
> limits.percentile(save.boot)
> limits.percentile(save.boot, probs = c(.025, .975))
> limits.t(save.boot)
> limits.bca(save.boot)
> limits.tilt(save.boot)
```

Trimmed Mean

One nice thing about the bootstrap is that it lets us handle a wide variety of statistics easily, without knowing a formula for the standard error of every statistic.

We may prefer to use a trimmed mean rather than an ordinary mean. For example, scoring in Olympic gymnastics and diving uses trimmed means, in which the high and low scores are discarded and the mean of the remaining observations is computed. A trimmed mean is particularly useful for data that may have outliers, or otherwise have long tails. An extreme example of a trimmed mean is a median, where all observations but the middle one or two are discarded.

The procedure for bootstrapping a trimmed mean is identical, except that after step 4 above, there is an additional step:

- For **Trim % each side**: enter the percentage of observations to trim; for a 25% trimmed mean, enter **25** (not 0.25).

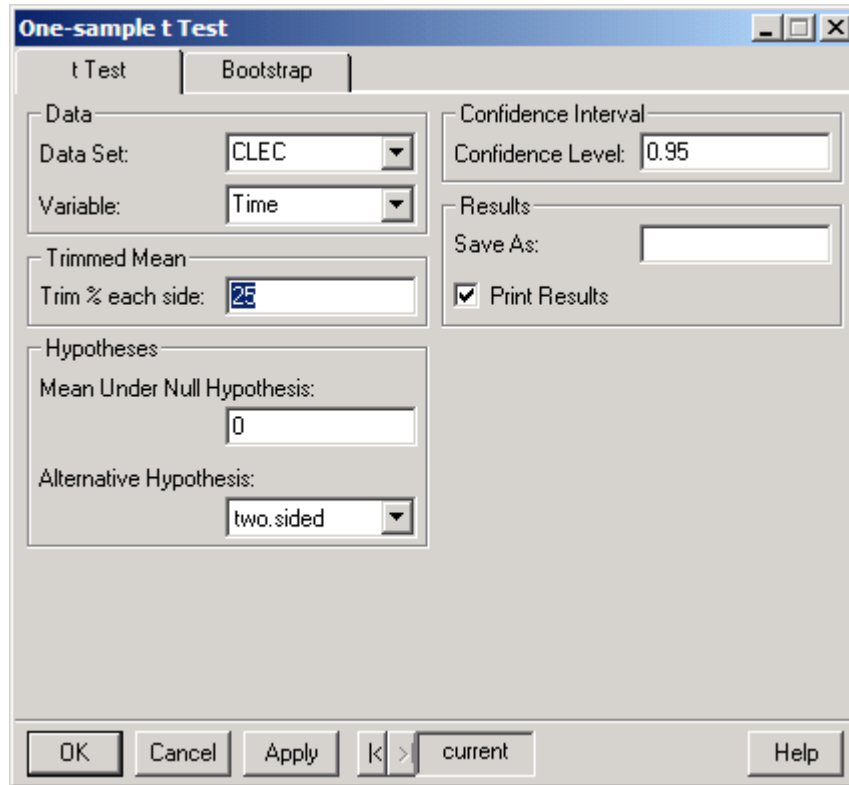


Figure 14.7

The resulting bootstrap distribution is nearly normal, is centered at the trimmed mean of the original data (13.8), and the bootstrap distribution appears narrower than for the untrimmed mean. This is born out by standard error of 2.7, compared to 4.1 for the untrimmed mean. This is reasonable, as we expect the trimmed mean to be less affected by the outlier in this data.

```
*** Bootstrap Results ***
Label: bootstrap 25% trimmed mean: CLEC$Time

Call:
bootstrap(data = CLEC$Time, statistic = mean,
  args.stat = list(trim = 0.25),
  label = "bootstrap 25% trimmed mean: CLEC$Time",
  trace = F, save.indices = T)

Number of Replications: 1000

Summary Statistics:
      Observed Mean      Bias      SE
mean    13.84 13.78 -0.06184 2.732
```

Command Line Notes

To bootstrap a trimmed mean we need to pass an additional argument to `mean`, the trimming fraction. One way is to use the `args.stat` argument, for example,

```
> save.boot = bootstrap(data = CLEC$Time,
+                       statistic = mean)
> save.boot2 = bootstrap(data = CLEC$Time,
+                       statistic = mean,
+                       args.stat = list(trim = 0.25))
> save.boot2
:
```

An alternative is that instead of passing a *function* (`mean`) as the second argument, you can pass an *expression* that indicates how you would call the function if you were working from the command line. As a further simplification, when the data is a data frame (almost all data sets in the `IPSdata` library are data frames), you can pretend that you have called `attach(that.data.frame)`. For example:

```
> save.boot = bootstrap(data = CLEC,
+                       statistic = mean(Time))
> save.boot2 = bootstrap(data = CLEC,
+                       statistic = mean(Time, trim = 0.25))
```

End of Example 14.2

14.1.2 Matched pairs

We turn now to comparing matched pairs. In a matched pairs design, subjects are matched in pairs and each treatment is given to one subject in each pair. In many cases, the same person is used for each pair and the responses are before-and-after measurements.

We look at the same data set as in Chapter 7. Twenty French teachers took a language test before and after an intense four-week summer French course. Did the course improve the test scores of these teachers?

1. Open the `Exercise7.41` data set from the `IPSdata` library.
2. At the menu, select `Statistics > Compare Samples > Two Samples > t Test/Resample...`
3. For `Data Set:` enter `Exercise7.41`.
4. For `Variable 1:` select `post`, and for `Variable 2:` select `pre`.
5. For `Type of t Test` select `Paired t`.
6. Optionally, select a percentage of observations to trim from each side. (We will stick with 0 in this example.)
7. In the `Hypotheses` box, for `Mean Under Null Hypothesis:` type `0` and for `Alternative Hypothesis:` select `greater`.

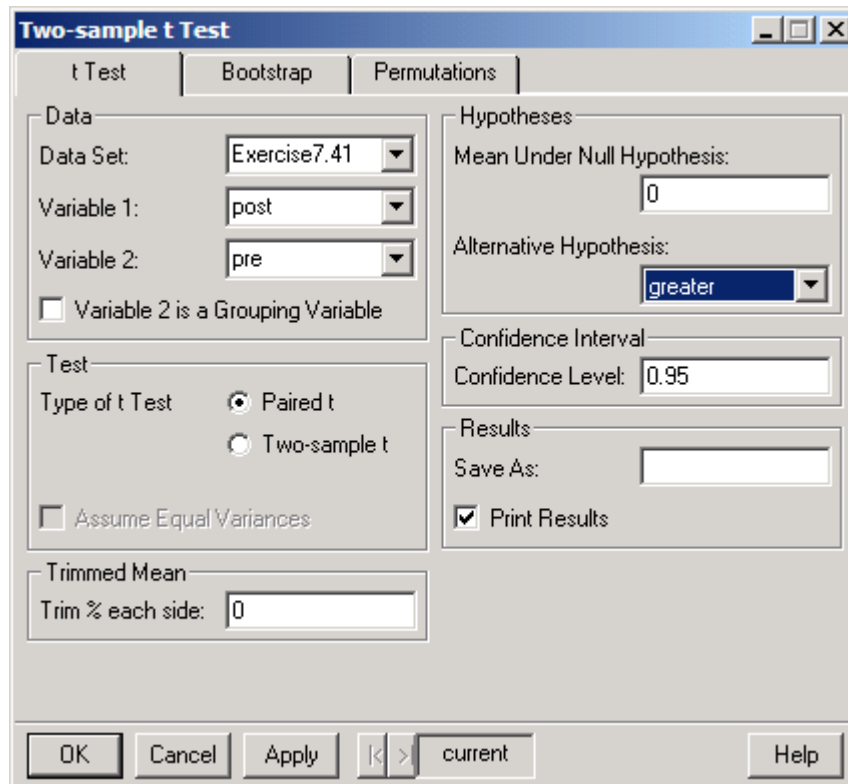


Figure 14.8

8. On the [Bootstrap](#) tab, check the box to [Perform Bootstrap](#).
9. In the [Summary Results](#) box, check the box for [t Intervals using Bootstrap SE](#).
10. On the [Permutations](#) tab, check the box to [Perform Permutation Test](#).
11. Click [OK](#).

Bootstrap Results The bootstrap distribution is roughly normal, so t tests and intervals should be reasonably accurate (though not quite normal, so some of the other bootstrap confidence intervals would be more accurate).

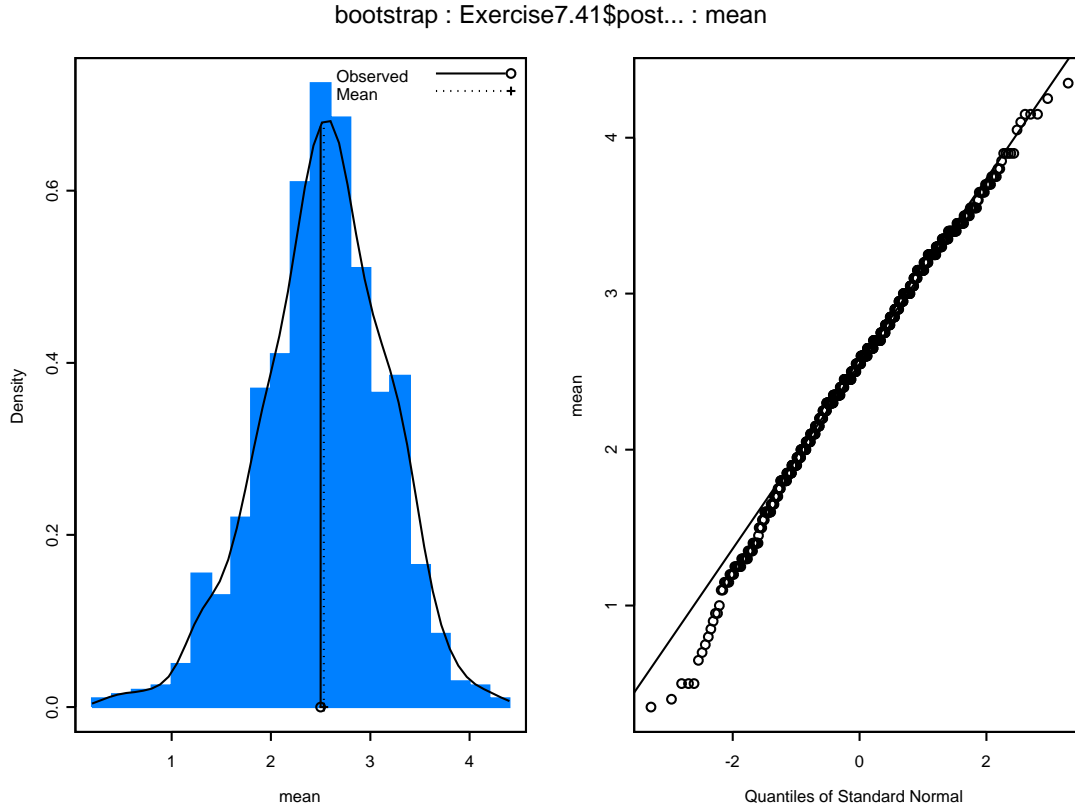


Figure 14.9

Summary Statistics:				
	Observed	Mean	Bias	SE
mean	2.5	2.532	0.0324	0.6252
Percentiles:				
	2.5%	5%	95%	97.5%
mean	1.25	1.4	3.4975	3.65

The **observed** value of 2.5 is the mean improvement. The **SE** of 0.6252 is the standard error; for comparison, the formula standard error is 0.647. The upper 95% of the bootstrap distribution, $(1.4, \infty)$, gives a quick-and-dirty one-sided 95% confidence interval for the improvement; this compares to the formula interval of $(1.38, \infty)$.

Permutation Test Results The bootstrap gives standard errors and confidence intervals, but does not provide significance tests. For significance testing we use permutation tests. The permutation distribution shows the variability of the statistic (for example, the difference in sample means) *assuming H_0 is true*.

Here is the permutation distribution for the French language data; note that it is centered about zero, as we expect if H_0 is true.

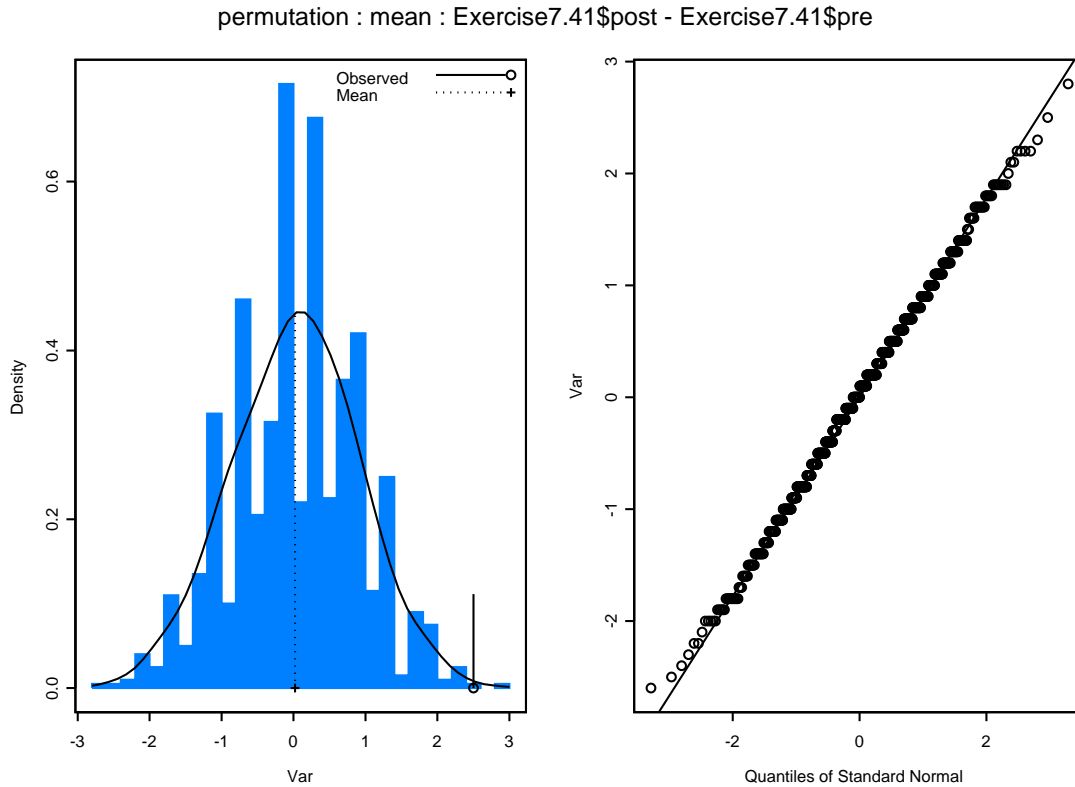


Figure 14.10

The key thing to look for in the permutation test plot is where the observed statistic lies. Here it is on the far right; only rarely does random chance give a statistic as large as the observed value. This permutation test tells us that if H_0 is true then the probability that the test statistic would take a value as extreme or more extreme than that actually observed is small.

We quantify that using the printed results from the permutation test:

Summary Statistics:					
	Observed	Mean	SE	alternative	p.value
Var	2.5	0.02202	0.875	greater	0.003

The P -value is 0.003.

Note that the histogram looks odd, with an up-and-down pattern. That is a problem with histograms—it is possible to get funny pictures depending on where the bar edges are. The density curve, overlaid on the histogram, is more reliable, as is the normal quantile plot.

14.1.3 Two independent means

The third situation we consider is comparing two independent samples. We consider the example from Chapter 7.

Example 14.3 Degree of reading power (IPS Example 7.14, Table 7.4)

In an experiment to determine whether certain reading activities help school children improve reading ability, a group of 21 students receive a treatment of these reading activities while a group of 23 students follows their usual curriculum.

Let μ_c and μ_t denote the true mean reading score for the control group and the treatment group, respectively. We wish to test the hypothesis

$H_0 : \mu_t - \mu_c = 0$ against $H_a : \mu_t - \mu_c > 0$.

1. Open the data set [Table7.4](#) from the `IPSdata` library.
2. At the menu, select [Statistics > Compare Samples > Two Samples > t Test/Resample...](#) or equivalently [Statistics > Resample > Two-Sample t](#).
3. For [Data Set](#): enter [Table7.4](#).
4. For [Variable 1](#): select `drp`, and for [Variable 2](#): select `group`. Check the box [Variable 2 is a Grouping Variable](#).
5. In the [Test](#) box, make sure the box [Equal Variances](#) is not checked.
6. Optionally, select a percentage of observations to trim from each side. (We will stick with 0 in this example.)
7. For [Alternative Hypothesis](#): select `greater`.

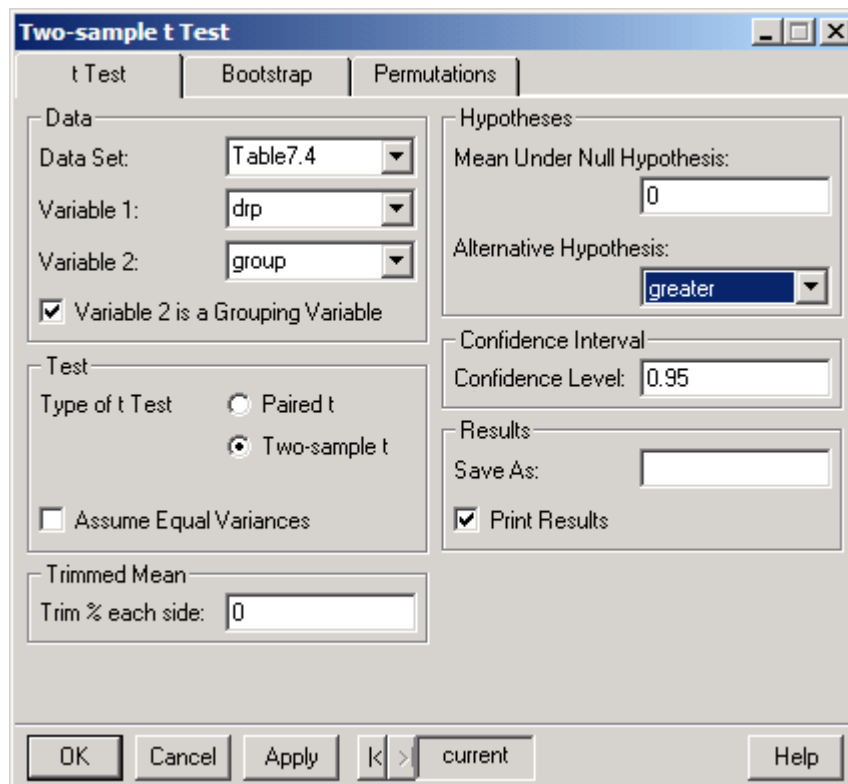


Figure 14.11

8. On the [Bootstrap](#) tab, check the box to [Perform Bootstrap](#).
9. On the [Permutations](#) tab, check the box to [Perform Permutation Test](#).
10. Click [OK](#).

Bootstrap Results The bootstrap distribution appears very normal, and gives an idea of uncertainty.

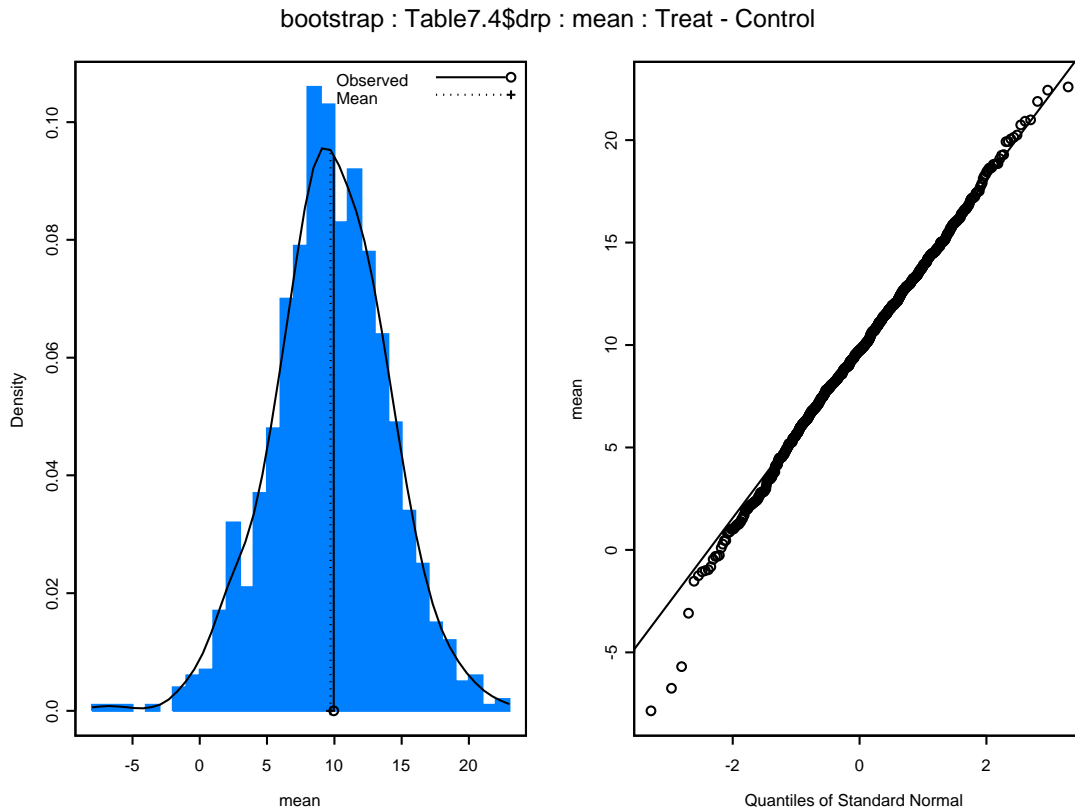


Figure 14.12

From the printed bootstrap results, we see that the standard error is 4.27 (this does not assume that the variances are equal); for comparison the formula SE is 4.31. A quick-and-dirty upper 95% confidence interval, the range of the upper 95% of the bootstrap distribution, is $(2.36, \infty)$; the formula interval is $(2.69, \infty)$.

Summary Statistics:				
	Observed	Mean	Bias	SE
mean	9.954	9.713	-0.2419	4.266
Percentiles:				
	2.5%	5%	95%	97.5%
mean	1.104917	2.357557	16.55839	18.25166

Permutation Test Results The observed value is near the right side of the permutation distribution, indicating that a statistic this large seldom occurs by random chance; the P -value should be small.

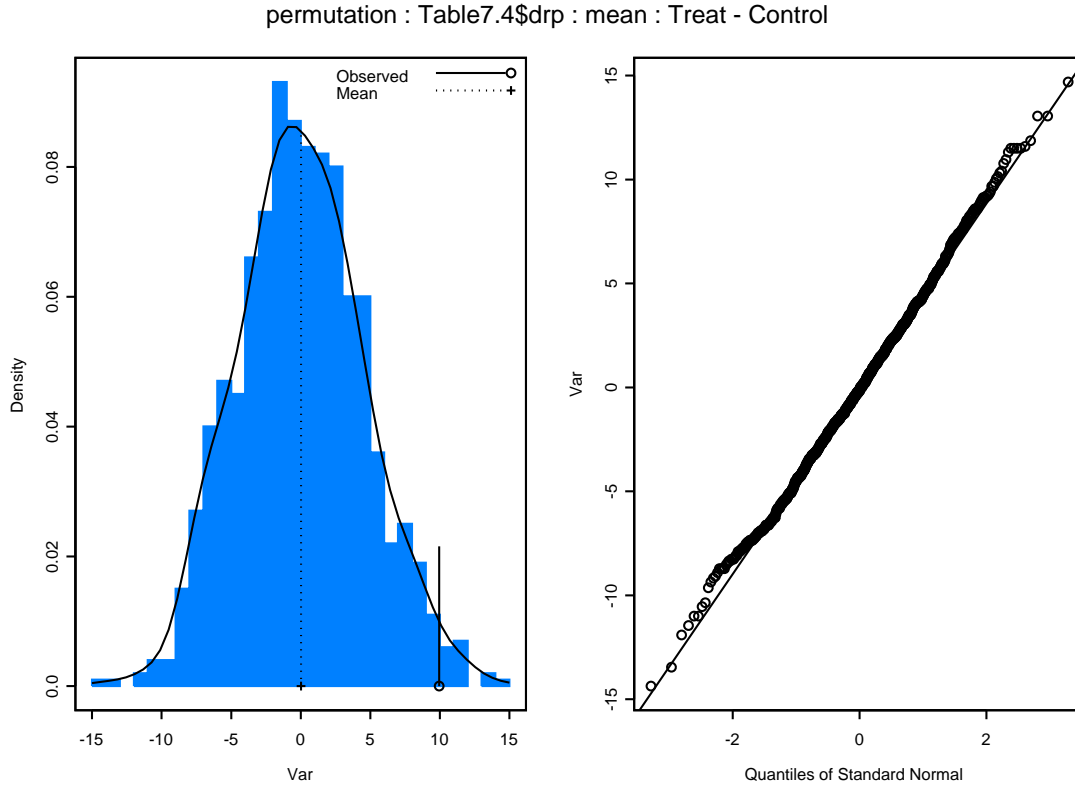


Figure 14.13

From the printed results, we see that the one-sided P -value is 0.017.

Summary Statistics:					
	Observed	Mean	SE	alternative	p.value
Var	9.954	0.02074	4.481	greater	0.017

For comparison, the formula P -value is 0.0132. We can conclude that the reading activities improve scores.

Here again is a case where the bootstrap is not needed because t procedures are acceptably accurate, but the bootstrap provides pictures that may help us understand P -values, sampling variability, and standard errors.

Command Line Notes

```
> attach(Table7.4)
> t.test(drp[group == "Control"], drp[group == "Treat"],
+       alternative = "less", var.equal = F)
> # The following switches groups, and uses "greater"
> t.test(drp[group == "Treat"], drp[group == "Control"],
+       alternative = "greater", var.equal = F)
> save.boot = bootstrap2(drp, mean, treatment = group)
> save.boot
> plot(save.boot)
> qqnorm(save.boot)
> save.perm = permutationTestMeans(drp, treatment = group,
+                                 alternative = "greater")
> save.perm
> plot(save.perm)
> qqnorm(save.perm)
> detach(Table7.4)
```

End of Example 14.3

14.2 Ratios of Means, Standard Deviations, or Other Statistics

Resampling lets us calculate standard errors, confidence intervals, and significance tests for the *ratio* of two means, of trimmed means, or of other statistics.

14.2.1 Ratio of means

Example 14.4 Vitamin A in blood

In the study described in IPS Exercise 14.56, researchers are interested in comparing the ratio of mean blood levels of serum retinol (vitamin A), between two groups of children, those who had or had not had recent infections.

The menus we used above work specifically for the difference, not ratio, of means or trimmed means. To work with ratios we turn to different menus.

1. Open the [Table14.6](#) data set from the IPSdata library.
2. At the menu, select [Statistics > Resample > Bootstrap...](#)
3. For [Data Set:](#) enter [Table14.6](#).
4. For [Expression:](#) enter `mean(retinol)`.
5. In the [Difference between two samples](#) box, for [Treatment:](#) select [group](#).
6. For [Compare Using:](#) select [Ratio](#).
7. On the [Results](#) tab, pick your choice of confidence intervals.
8. Click [OK](#) (or [Apply](#), to keep the menu open, in case you want to rerun this with different confidence intervals).

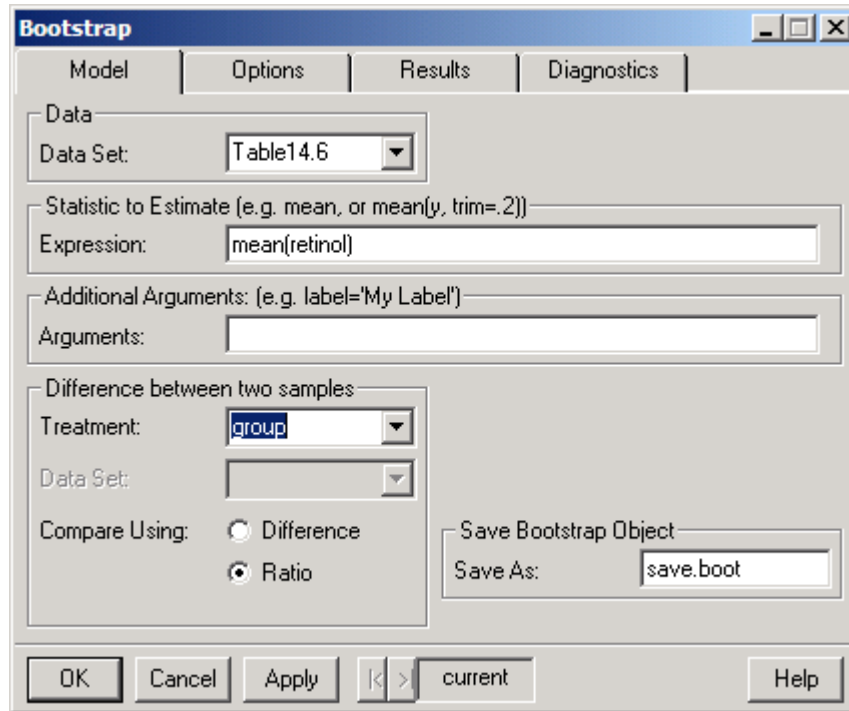


Figure 14.14

The plots of the bootstrap results show that the bootstrap distribution is nonnormal, so t methods are not appropriate. Instead we would use one of the bootstrap hypothesis tests

bootstrap : Table14.6 : mean(retinol) : uninfected / infected

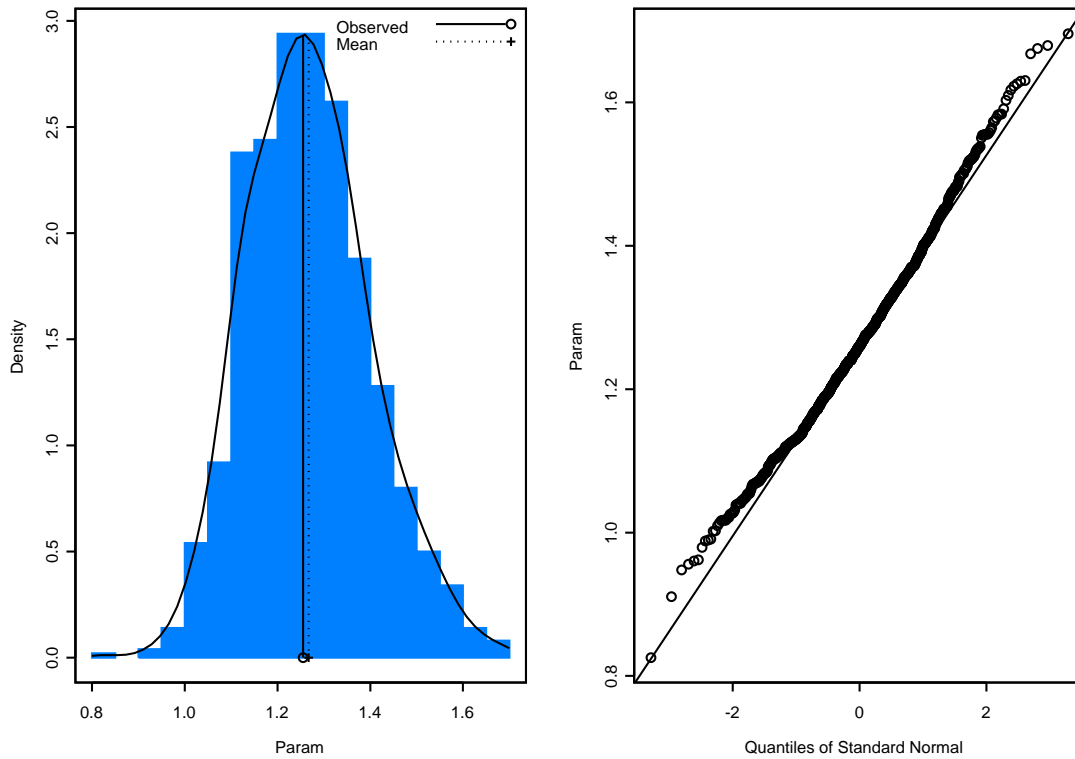


Figure 14.15

For significance testing we use permutation testing.

9. At the menu, select [Statistics > Resample > Permutation Test...](#)
10. For [Data Set](#): enter [Table14.6](#).
11. In the [Difference between two samples](#) box, for [Treatment](#): select [group](#).
12. For [Compare Using](#): select [Ratio](#).
13. For [Expression](#): enter [mean\(retinol\)](#).
14. For [Alternative](#): select [greater](#).
15. Click [OK](#).

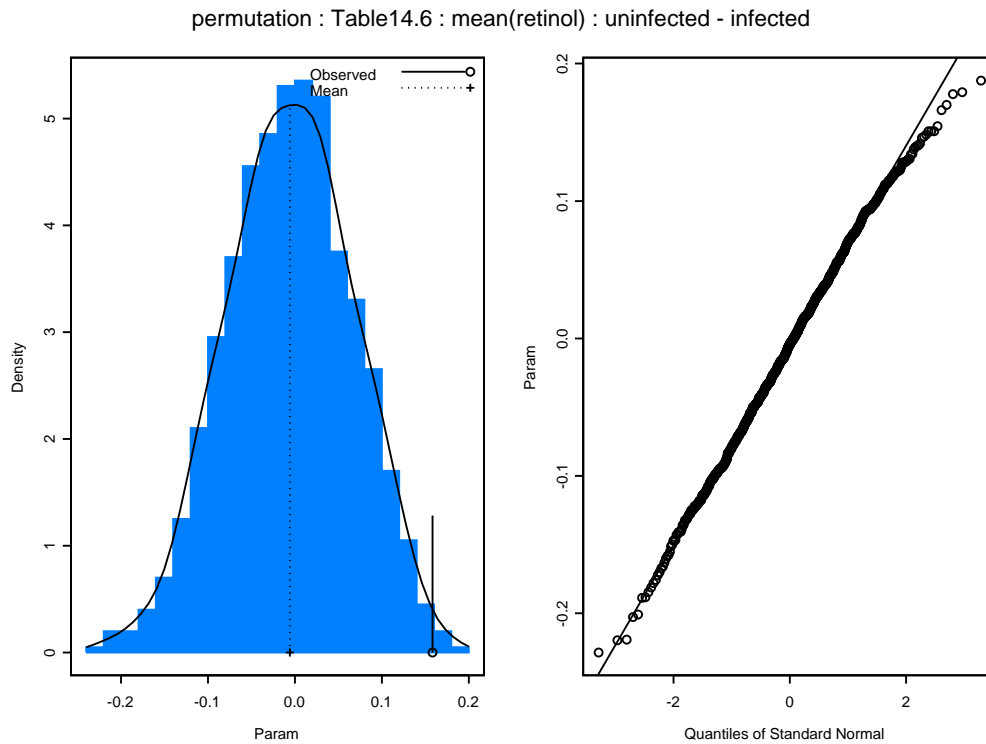


Figure 14.16

The observed value is in the right tail of the distribution; the P -value will be small. The printed results contain the P -value (this is part of IPS Exercise 14.56).

Command Line Notes Bootstrapping for two samples is done by `bootstrap2`:

```
> save.boot = bootstrap2(data = Table14.6,
+                         statistic = mean(retinol),
+                         treatment = group, ratio = T)
> save.boot
> # continue with plot, qqnorm, limits.percentile, etc.
```

The permutation test can be done either by a general-purpose function, or a version specifically for means that runs faster:

```
> save.perm = permutationTest2(data = Table14.6,
+                              statistic = mean(retinol),
+                              treatment = group, ratio = T,
+                              alternative = "greater")
> save.perm2 = permutationTestMeans(Table14.6$retinol,
+                                   treatment = Table14.6$group, ratio=T,
+                                   alternative = "greater")
> # continue by printing and plotting
```

End of Example 14.4

14.2.2 Ratio of standard deviations, or other statistics

The procedure for bootstrapping the ratio of standard deviations, or other statistics, is very similar to the procedure for the ratio of means. We use the same menus, either for the bootstrap or permutation tests. The difference is that we specify a different statistic. In the [Expression:](#) field, instead of specifying `mean(x)`, we specify a different expression, for example:

- `stdev(x)` (for standard deviations)
- `var(x)` (for variances)
- `mean(x, trim = 0.25)` (for a 25% trimmed mean)

14.3 Proportions (Chapter 8)

Resampling proportions using the GUI is straightforward. We can bootstrap a single proportion or difference in two proportions, or do a permutation test for the difference of two proportions, using augmented versions of the menus we used in Chapter 8. The basic procedure is

1. At the menu, select [Statistics > Compare Samples > Counts and Proportions > Proportions Parameters/Resample...](#) or equivalently [Statistics > Resample > Proportions](#).
2. Enter data and select other options as before.
3. Select the [Bootstrap](#) and/or [Permutations](#) tabs, and check the boxes for [Perform Bootstrap](#) and/or [Perform Permutation Test](#).

Command Line Notes

Resampling proportions from the command line is more complicated. You must expand summary data into a vector or data frame with one row per subject, with a variable containing 0 or 1 (or [FALSE](#) or [TRUE](#)) for failures and successes, respectively, then resample the mean of this variable.

There are examples of bootstrapping proportions for Exercises 14.60 and 14.77 in the file of examples that comes with the [IPSDdata](#) library. You can open the file using the menu: [IPSDdata > Example Code for Exercises](#).

14.4 Two-Way Tables and the Chi-Square Test (Chapter 9)

We do not recommend using resampling for the problems considered in Chapter 9. These problems are purely significance testing problems, so bootstrapping would not apply. Permutation tests could be used, but there is a better alternative—instead of using the Chi-Square test, to use Fisher’s Exact Test. From the main menu, select [Statistics > Compare Samples > Counts and Proportions > Fisher’s Exact Test...](#) In fact, it gives the answer we would get using permutation testing, if we were to use an infinite number of permutations.

Fisher’s Exact Test is an alternative to the Chi-Square test that is accurate for small samples. The exact calculations done by Fisher’s Exact test are slow for large samples; in that case the Chi-Square test is preferred.

14.5 Regression (Chapter 10)

We can bootstrap in linear regression problems using an augmented version of the same menus we used in Chapter 10, and do permutation testing with the same menu we used for the ratio of means or standard deviations.

14.5.1 Bootstrap for Regression

We will work with the first example from Chapter 10, relating fuel efficiency to the log of speed.

Example 14.5 Fuel efficiency

1. At the menu, select **Statistics > Regression > Linear/Resample...**, or equivalently, **Statistics > Resample > Linear Regression**.
2. For **Data Set:**, enter **Example10.1**.
3. Under **Variables**, for **Response:** select **MPG**, and for **Explanatory:** select **LOGMPH**.
The **Formula:** field should read **MPG ~ LOGMPH**.
4. In the **Save Model Object** box, for **Save As:** type **fitMPG**.
5. Select the **Bootstrap** tab, and check the box to **Perform Bootstrap**.
Note that the **Statistic:** field is **coef**. This bootstraps regression coefficients. We will see other options later.
6. Click **OK**. The calculations may take a while.

In addition to the same printed results as before, this produces bootstrap results for both regression coefficients—the intercept and slope. We are primarily interested in the slope coefficient. The bootstrap histogram and normal quantile plot show that bootstrap distribution is quite normal, suggesting that t confidence intervals and tests should be accurate.

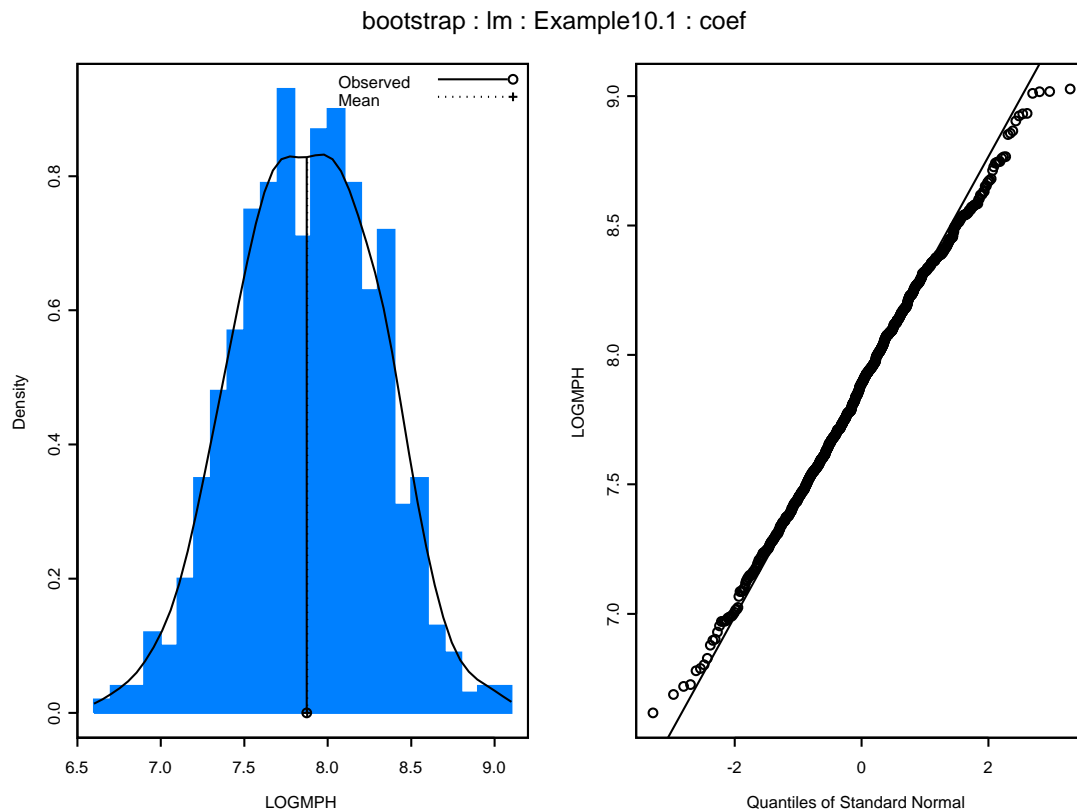


Figure 14.17

The printed report includes results for both coefficients:

Summary Statistics:				
	Observed	Mean	Bias	SE
(Intercept)	-7.796	-7.801	-0.004659	1.307
LOGMPH	7.874	7.877	0.003186	0.421

Percentiles:				
	2.5%	5%	95%	97.5%
(Intercept)	-10.180389	-9.891102	-5.610312	-5.253948
LOGMPH	7.017872	7.184364	8.544028	8.653138

The bootstrap standard error for the slope coefficient is 0.421, substantially larger than the formula standard error of 0.3541 (from the printout in Chapter 10). This suggests that one or more of the assumptions required for the formula standard error may be violated. In particular, recall from Section 10.4 that the residuals tended to be larger for larger values of the fit, a violation of the assumption that the residual standard deviation is constant. This violation makes the formula standard error too low.

End of Example 14.5

Graphical Bootstrap

We investigate this discrepancy further using graphical bootstrapping. We begin by plotting `MPG` against `LOGMPH`. For technical reasons, we do this using the command line (Section 0.7) rather than the GUI.

Example 14.6 Graphical Bootstrap

1. At the menu, select `Window > Commands Window` to bring the `Commands` window to the front.
2. At the end of the `Commands` window, paste these three lines:

```
attach(Example10.1)
plot(LOGMPH, MPG)
detach()
```

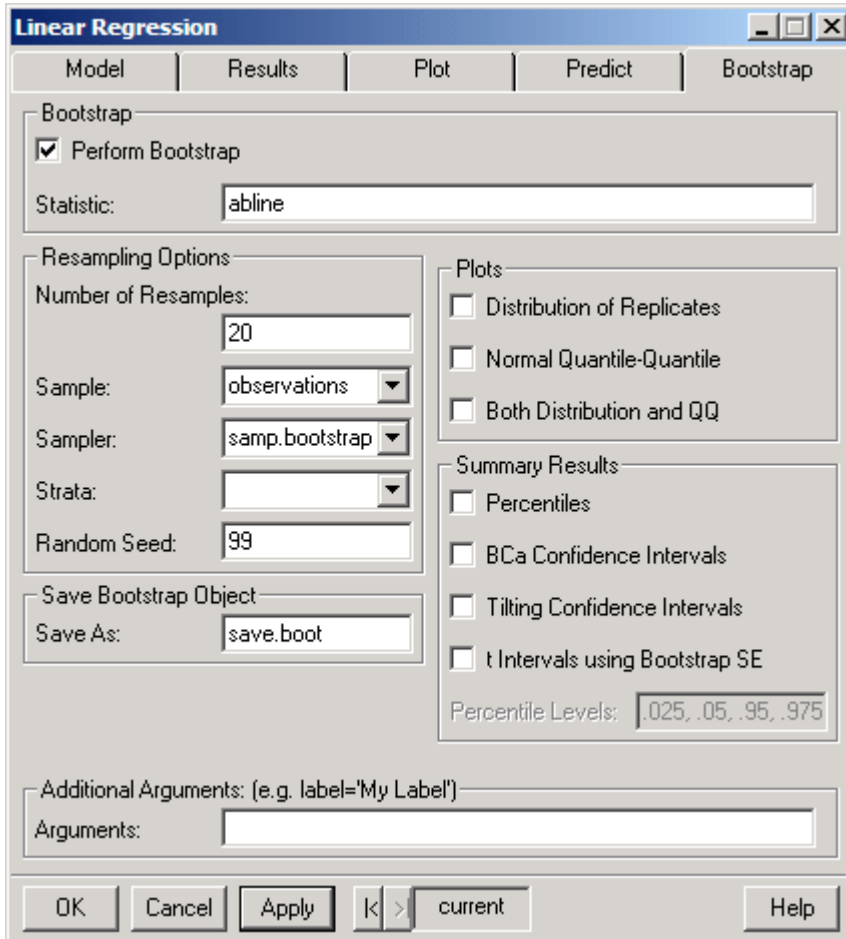
Now we are ready to add bootstrap regression lines to this plot.

3. At the menu, select `Statistics > Resample > Linear Regression`,
4. Enter everything on the first page just like before, *or* use the back arrow at the bottom of the dialog to scroll back.



5. On the `Bootstrap` tab, change `Statistic:` to `abline`. (`abline` adds a slope-intercept line to a plot.)
6. Change `Number of Resamples:` to 20.
7. Change `Random Seed` to 99. (This is optional; it gives you the same random numbers we are using, so you can reproduce the figure below.)

8. In the **Plots** and **Summary Results** boxes, turn everything off.
9. Click **OK**.



The resulting plot gives an idea of the variability of regression lines due to random sampling. One thing we notice is that the point of smallest variability appears to be somewhat to the left of center. If the assumptions underlying the formula standard errors were correct, the minimum variability would be at the mean of the explanatory variable.

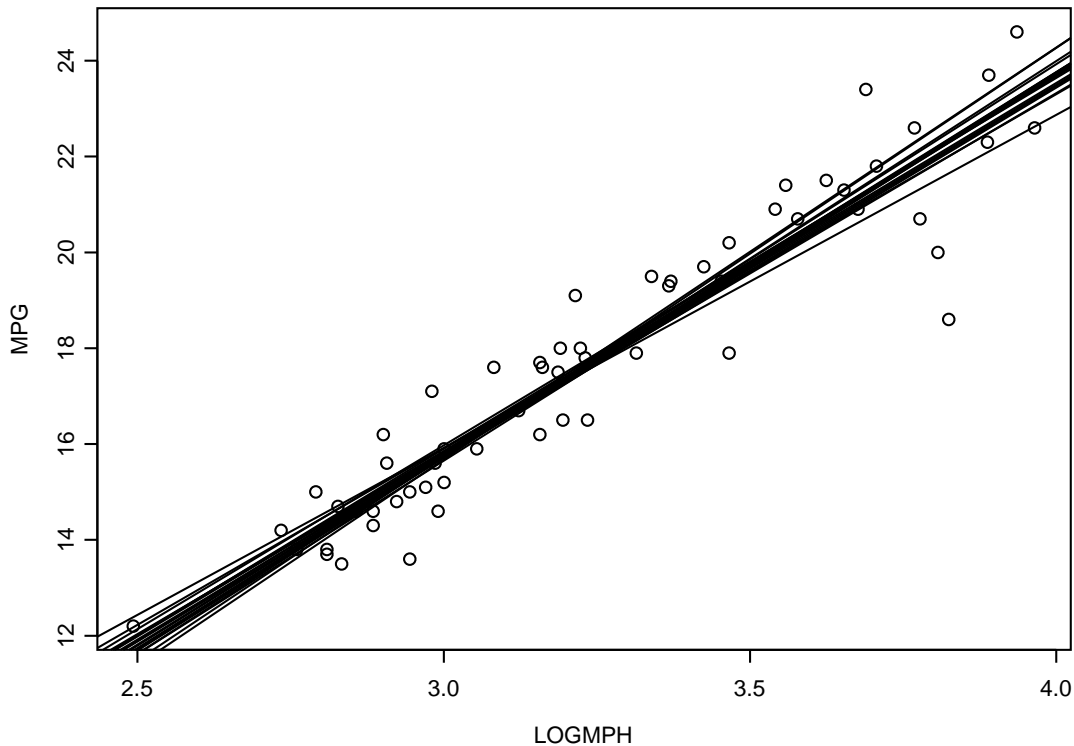


Figure 14.18

We used only 20 random lines here because too many lines cause a mess of black ink, but the small number does mean that the result is rather sensitive to randomness. You are encouraged to repeat this experiment with more lines, or without setting the random number seed, to see how the results differ.

Command Line Notes

We will make use of another handy feature here. If we want to bootstrap linear regression results, we can give `bootstrap` the model object rather than the original data set. The bootstrap code will find the original data automatically.

```
> fitMPG = lm(MPG ~ LOGMPH, data = Example10.1)
> attach(Example10.1)
> plot(LOGMPH, MPG)
> detach()
> bootstrap(fitMPG, abline, B = 20, seed = 99)
```

End of Example 14.6

Bootstrap Confidence Intervals for Predictions

Our next step is to quantify what we noticed in the graphical bootstrap numerically, by creating confidence intervals for the bootstrap regression lines. We will do this by picking 16 values of x in advance; then for every bootstrap sample we will compute and save the value of the regression line at those 16 x -values. In

effect we are bootstrapping 16 statistics simultaneously, where each statistic is the y -value of the regression line above one of the x 's.

Example 14.7 Fuel efficiency

1. Create a new data set with two variables, `LOGMPH` containing values 2.5, 2.6, ..., 4.0 and `MPG` containing anything (the values are not used, but the variable must be present). You may use the `Data > Fill...` dialog (see page 36 for an example).
2. At the menu, select `Statistics > Resample > Linear Regression`.
3. Enter everything on the first page as before.
4. On the `Bootstrap` tab, change `Statistic:` to `predict(fitMPG, newdata = SDF1)`. (Replace `SDF1` with the name of the data set you created).
5. Set `Number of Resamples:` to 1000.
6. Change `Random Seed` to blank.
7. In the `Plots` box, select `Both Distribution and QQ`.
8. In the `Summary Results` box, select `Percentiles` and `t Intervals using Bootstrap SE`.
9. Click `OK`.

	1	2
	MPH	LOGMPH
1	NA	2.50
2	NA	2.60
3	NA	2.70
4	NA	2.80
5	NA	2.90
6	NA	3.00
7	NA	3.10
8	NA	3.20
9	NA	3.30
10	NA	3.40
11	NA	3.50
12	NA	3.60
13	NA	3.70
14	NA	3.80
15	NA	3.90
16	NA	4.00
17		
18		

For every bootstrap sample, this fits a regression line and gives the value of the line at the 16 values of `LOGMPH` between 2.5 and 4.0.

Because we are bootstrapping 16 statistics, there are 16 plots produced (not shown here; they all indicate that predictions are approximately normally distributed), and the printed output includes 16 standard errors:

Summary Statistics:				
	Observed	Mean	Bias	SE
1	11.89	11.88	-0.00790069	0.2792
2	12.68	12.67	-0.00703319	0.2386
3	13.46	13.46	-0.00616569	0.1997
4	14.25	14.25	-0.00529819	0.1638
5	15.04	15.03	-0.00443068	0.1332
6	15.83	15.82	-0.00356318	0.1123
7	16.61	16.61	-0.00269568	0.1070
8	17.40	17.40	-0.00182818	0.1194
9	18.19	18.19	-0.00096068	0.1450
10	18.98	18.98	-0.00009317	0.1783
11	19.76	19.76	0.00077433	0.2157
12	20.55	20.55	0.00164183	0.2553
13	21.34	21.34	0.00250933	0.2964
14	22.13	22.13	0.00337684	0.3383
15	22.91	22.92	0.00424434	0.3809
16	23.70	23.71	0.00511184	0.4238

The smallest standard error (and narrowest confidence intervals) occurs at the seventh value, where `LOGMPH` = 3.2, slightly less than $\bar{x} = 3.24$. They increase in either direction. For comparison, the formula standard error

$$SE_{\hat{\mu}} = s \sqrt{\frac{1}{n} + \frac{(x^* - \bar{x})^2}{\sum(x - \bar{x})^2}}$$

is smallest exactly at \bar{x} .

To add the confidence intervals to a plot is simplest using the command line:

```

Command Line Notes These commands run the bootstrap, and add
confidence intervals to the plot. You may skip the first steps if you have
created save.boot using the GUI.
> fitMPG = lm(MPG ~ LOGMPH, data = Example10.1)
> SDF1 = data.frame(MPG = NA,
+                   LOGMPH = seq(2.5, to = 4.0, by = 0.1))
> save.boot = bootstrap(fitMPG, predict(fitMPG, newdata = SDF1))
> save.boot # print the standard errors
> plot(LOGMPH, MPG)
> abline(fitMPG)
> save.intervals = limits.t(save.boot, c(.025, .975))
> lines(SDF1$LOGMPH, save.intervals[,1])
> lines(SDF1$LOGMPH, save.intervals[,2])

```

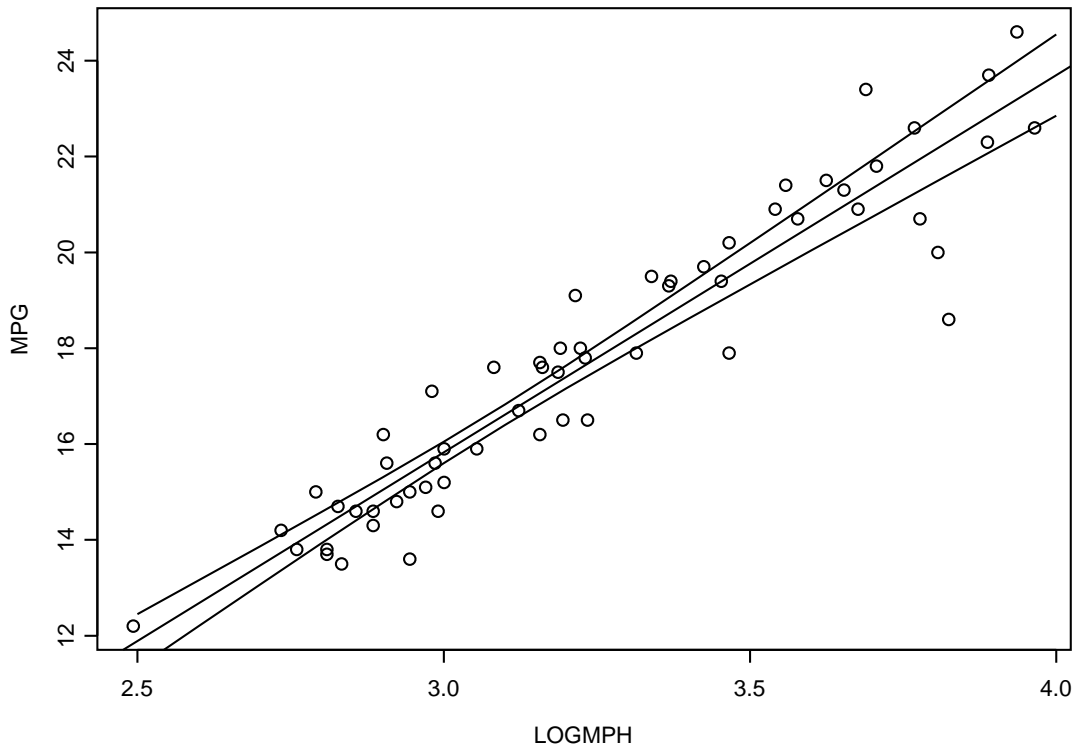


Figure 14.19

End of Example 14.7

Bootstrap Prediction Intervals

We do not recommend that you try using the bootstrap for prediction intervals (intervals for individual new values). The usual formula prediction intervals require two assumptions that are often not true in practice—that the residual distribution is exactly normal (large samples do not help!) with exactly the same standard deviation everywhere. The bootstrap does not like to make assumptions like that. While there are ways to use the bootstrap to calculate prediction intervals, they become complicated unless you make at least one of those two problematic assumptions.

14.5.2 Permutation tests for regression

To do permutation testing in regression, we choose a test statistic, and obtain the distribution of that test statistic when H_0 is true by permutation sampling. In contrast to formula-based methods, the test statistic does not have to be a t statistic. It can be the statistic we are directly interested in, for example b_1 .

The Linear Regression dialog does not support permutation testing, but we can do it easily from the general-purpose [Permutation Test](#) menu. The key is that we must permute just one of the two variables, either the response or the explanatory variable, not both.

1. At the menu, select [Statistics > Resample > Permutation Test...](#)
2. For [Data Set:](#), enter [Example10.1](#).

- In the **Statistic to Estimate** box, for **Expression:** type `coef(lm(MPG ~ LOGMPH, data = Example10.1))`
As an alternative, if you have saved the `fitMPG` model object, you can type `coef(update(fitMPG))` which says to update `fitMPG` using resampled data.
- In the **Permute these columns** box, select `MPG`.
- Click **OK**.

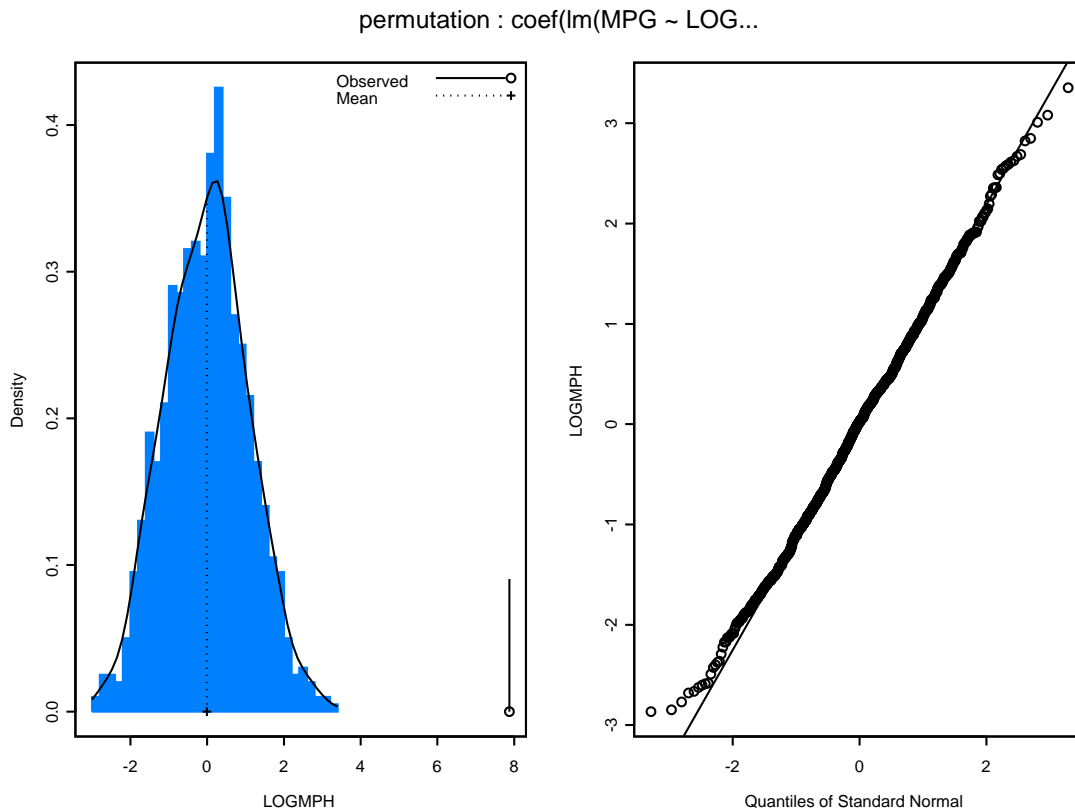


Figure 14.20

The observed statistic is well outside the range of the permutation distribution, indicating that the P -value should be almost zero, and the slope coefficient is statistically significant.

The printed report shows that the P -value for the intercept term is 0.002, the smallest possible value for a two-sided permutation test with 999 replications:

Summary Statistics:					
	Observed	Mean	SE	alternative	p-value
(Intercept)	-7.796	17.754580	3.494	two.sided	0.002
LOGMPH	7.874	-0.009126	1.078	two.sided	0.002

14.6 Multiple Regression (Chapter 11)

Resampling for multiple regression is very similar to resampling for simple linear regression. We obtain standard errors and confidence intervals by bootstrapping, and do significance testing using permutation tests. We use the same menus as for simple linear regression, either [Statistics > Regression > Linear/Resample...](#) or [Statistics > Resample > Linear Regression](#) for bootstrapping, and [Statistics > Resample > Permutation Test...](#) for permutation tests.

There is one major difference. For permutation testing, we may not permute one of the explanatory variables, with the intent of testing whether the contribution of that variable is significant. To do that would lose all correlations between that explanatory variable and other explanatory variables, and the P -value would be incorrect, sometimes dramatically so. For example, if the correlations between explanatory variables are strong, there could be greater than a 49% chance of rejecting H_0 at the 5% level, even when H_0 is true.

What we can do is to permute the response variable, to test whether the response is related to the explanatory variables as a whole set.

14.7 Analysis of Variance (Chapters 12 and 13)

We do not recommend using resampling for the problems considered in Chapters 12 and 13. These problems are purely significance testing problems, so bootstrapping would not apply. Permutation testing could be used for one-way ANOVA problems (Chapter 12) and some two-way ANOVA problems (Chapter 13), but this is more complicated than we want to get into here.

14.8 Correlation (Chapter 2)

Correlation is an example of a statistic which is useful for describing data, but for which simple formula confidence intervals and significance tests are not available or accurate. We can do the confidence intervals and tests using resampling.

Exercise 2.9 in IPS discusses the relationship between social distress and brain activity. In a study of 13 subjects, there is correlation 0.88 between score on a questionnaire and brain activity. We will use resampling to obtain confidence intervals for the true correlation, and to test whether the correlation is significantly different from zero.

1. Open the [Exercise2.9](#) data set.
2. At the menu, select [Statistics > Data Summaries > Correlations/Resample](#) or equivalently [Statistics > Resample > Correlations](#).
3. For **Data Set:** enter [Exercise2.9](#). For **Variables:** select **<ALL>**.

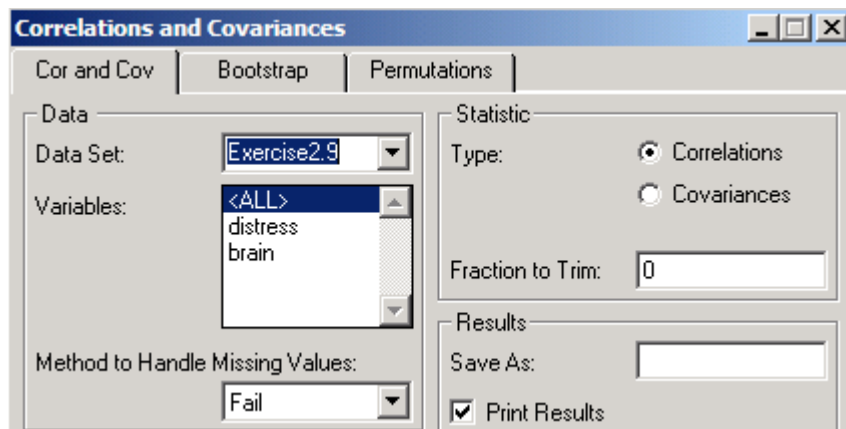


Figure 14.21

4. Select the [Bootstrap](#) tab, and check the box to [Perform Bootstrap](#).
5. Select the [Permutations](#) tab, and check the box to [Perform Permutation Test](#).
6. In the [Permute these columns](#) box, ensure that one (but not both) of the columns are selected.
7. Click [OK](#).

The bootstrap distribution is highly skewed. This is not surprising—correlation is bounded by ± 1 , so it could be substantially lower than 0.88 but not much higher. In any case, t intervals and tests should not be used.

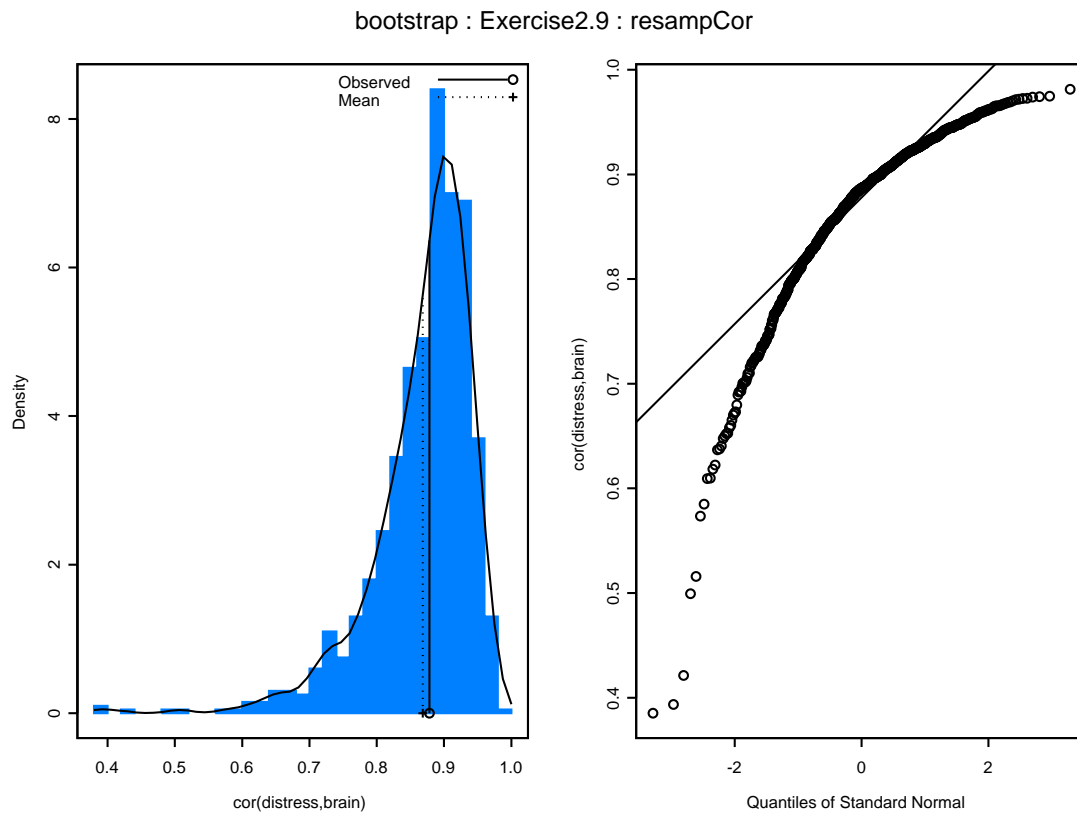


Figure 14.22

What is perhaps surprising is how much variability there is. The estimated correlation of 0.88 is just not very precise. The quick-and-dirty bootstrap percentile confidence interval gives a 95% confidence interval ranging from 0.68 to 0.96.

Percentiles:	2.5%	5%	95%	97.5%
cor(distress,brain)	0.6800324	0.7255228	0.951649	0.9608618

The permutation distribution is centered about 0, corresponding to H_0 that the true correlation is zero. The observed correlation is in the far right tail of the permutation distribution, so the P -value is small and the correlation is probably statistically significantly different from zero.

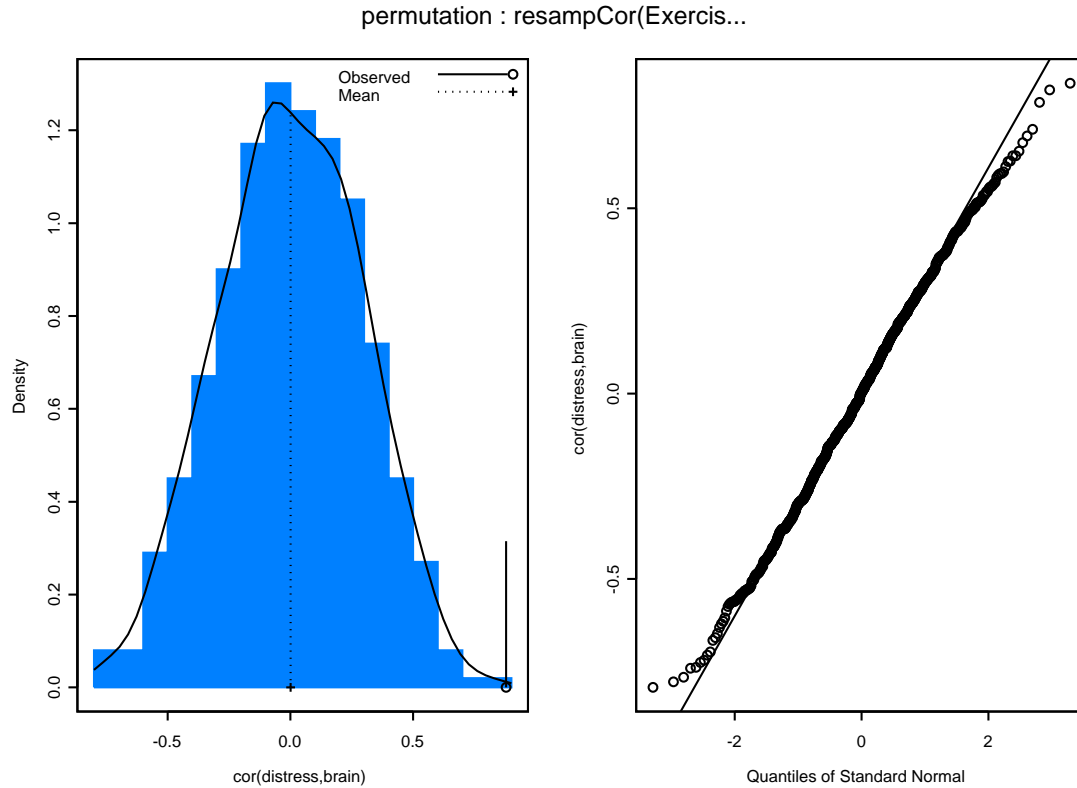


Figure 14.23

This is born out by the permutation test printed report, which indicates that the P -value is 0.002, the smallest possible for a two-sided alternative based on 999 replications:

Summary Statistics:					
	Observed	Mean	SE	alternative	p-value
cor(distress,brain)	0.8782	0.0008145	0.2897	two.sided	0.002

We cannot use permutation tests for a null hypothesis that the correlation is some value other than zero, because permuting either variable corresponds to sampling in a way that is consistent with a correlation of zero, not some other value.

Chapter 15

Nonparametric Tests

Throughout this text we have made the assumption that our samples come from populations that are normally distributed. However, in many settings, this assumption may not be met: the sample may be strongly skewed, or there may be extreme outliers that cannot be ignored. Nonparametric tests do not make any distributional assumptions on the population.

15.1 Wilcoxon Rank Sum Test

The Wilcoxon rank sum test is a nonparametric analog to the parametric t test and is used to test the hypothesis H_0 : Two populations follow the same distribution, against H_a : Two populations do not follow the same distribution.

Draw simple random samples of size n_1 and n_2 from each of two independent populations, rank all $N = n_1 + n_2$ observations, and let W denote the sum of the ranks for the first sample; this is the *Wilcoxon rank sum statistic*. If the two populations have the same continuous distribution, then W has mean and standard deviation

$$\mu_W = \frac{n_1(N+1)}{2} \quad \sigma_W = \sqrt{\frac{n_1 n_2 (N+1)}{2}}.$$

We reject the null hypothesis if the rank sum W is far enough from its mean. If n_1 and n_2 are large then W is approximately normally distributed (even if the population is nonnormal), while for small samples the distribution can be calculated exactly.

Example 15.1 Weeds among the corn (IPS Example 15.1)

Researchers performed a study to see if weeds (lamb's-quarter plants) reduced corn yield. Four small plots were weeded completely (no weeds) and four small plots were weeded so that three weed plants per meter of row were present. Here are the yields of corn, in bushels per acre:

Weeds per meter	Yields			
0	166.7	172.2	165.0	176.9
3	158.6	176.4	153.1	156.0

We wish to test the hypothesis H_0 : Corn yields are the same regardless of weeds, against H_a : Corn yields are higher in weed-free lots.

1. Open the [Example15.1](#) data set from the `IPSdata` library. The variables are `weeds` and `yield`.
2. At the menu, select `Statistics > Compare Samples > Two Samples > Wilcoxon Rank Test...`
3. For `Data Set`: enter `Example15.1`.
4. For `Variable 1`: select `yield`.

5. For **Variable 2**: select **weeds**.
6. Check the box **Variable 2 is a Grouping Variable**.
7. Under **Hypotheses**, for **Mean Under Null Hypothesis**: enter 0, and for **Alternative Hypothesis**: select **greater**.
8. For more accuracy, check the boxes for **Continuity Correction** and **Use Exact Distributions**.
If there are ties among the ranks, then there is no exact distribution, and S-PLUS generates a warning message.

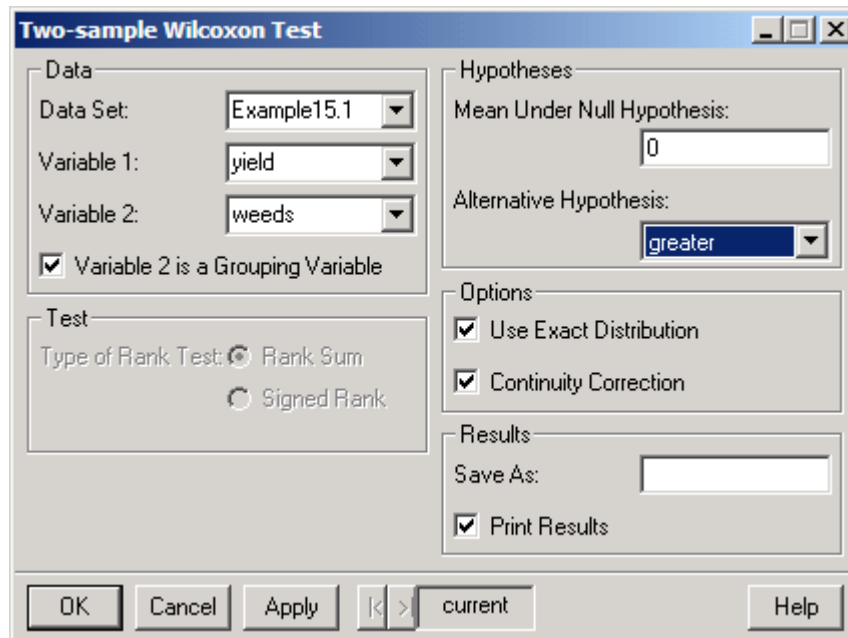


Figure 15.1

9. Click **OK**.

```

Exact Wilcoxon rank-sum test

data:  x: yield with weeds = 0 , and y: yield with weeds = 3
rank-sum statistic W = 23, n = 4,  m = 4, p-value = 0.1
alternative hypothesis:  mu is greater than 0

```

Using a significance level of $\alpha = 0.05$, we do not have strong enough evidence (P -value = 0.1) to reject the null hypothesis.

Remark: This test can be also executed in the S-PLUS GUI if the data are entered by creating two columns of yield information: one containing yields from lots with no weeds, and the second containing yields from lots with three weeds. To do this enter the two columns as the two variables, and do *not* check the box **Variable 2 is a Grouping Variable**.

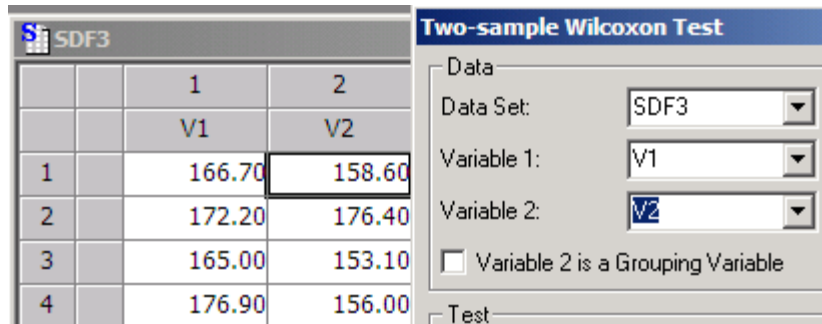


Figure 15.2

End of Example 15.1

Command Line Notes

The function `wilcox.test` performs the Wilcoxon rank sum test.

```

> attach(Example15.1)
> wilcox.test(yield[weeds == 0], yield[weeds == 3],
+             alternative="greater")
> detach()

```

15.2 The Wilcoxon Signed Rank Test

The Wilcoxon signed rank test is the nonparametric analog to the matched pairs t test. To test the hypothesis H_0 : There are no systematic differences within pairs against H_a : There are systematic differences within pairs, compute the absolute values of the differences in responses within pairs of a matched pairs study and rank these values. Let W^+ denote the sum of the ranks for the positive differences. If the responses have a continuous distribution that is not affected by the different treatment means within pairs, then W^+ has mean and standard deviation

$$\mu_{W^+} = \frac{n(n+1)}{4} \quad \sigma_{W^+} = \sqrt{\frac{n(n+1)(2n+1)}{24}}.$$

We reject the null hypothesis when W^+ is far from its mean.

Example 15.2 Tell me a story (IPS Example 15.8)

How well do children recall stories read to them? The five children in this study recount two fairy tales read to them: one fairy tale is just read to them while the second fairy tale is read *and* illustrated with pictures. An expert assigns scores for certain uses of language.

Child	1	2	3	4	5
Story 2	0.77	0.49	0.66	0.28	0.38
Story 1	0.40	0.72	0.00	0.36	0.55
Difference	0.37	-0.23	0.66	-0.08	-0.17

We wish to test the hypothesis H_0 : Scores have the same distribution for both stories against H_a : Scores are systematically higher for Story 2.

1. Open the [Example15.8](#) data set.
2. At the menu, select [Statistics > Compare Samples > Two Samples > Wilcoxon Rank Test...](#)
3. For [Data Set](#): select [Example15.8](#).
4. For [Variable 1](#): select [story2](#), and for [Variable 2](#): select [story1](#). (Note the reversal.)
5. Under [Test](#), check the radio button next to [Signed Rank](#).
6. Under [Hypotheses](#), for [Alternative Hypothesis](#): select [greater](#).

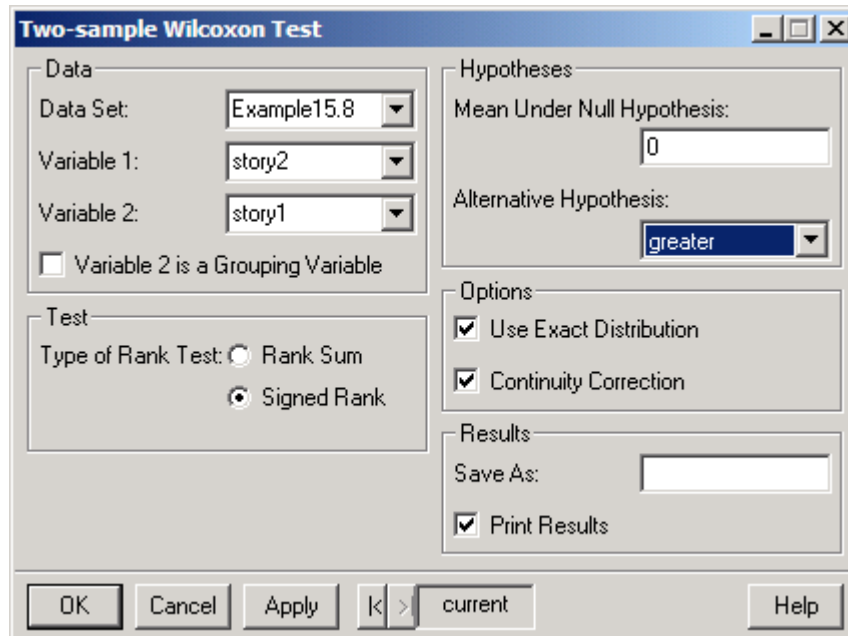


Figure 15.3

7. Click [OK](#).

```
Exact Wilcoxon signed-rank test

data:  x: story2 in Example15.8 , and y: story1 in Example15.8
signed-rank statistic V = 9, n = 5, p-value = 0.4062
alternative hypothesis: mu is greater than 0
```

The large P -value of 0.4062 gives no evidence that seeing illustrations improves the storytelling recall of children.

Note: If you have the differences [story2-story1](#), then you could also do this test using the menu [Statistics > Compare Samples > One Sample > Wilcoxon Signed Rank Test...](#)

End of Example 15.2

Command Line Notes

```
The function wilcox.test also performs the Wilcoxon signed rank test.  
> attach(Example15.8)  
> wilcox.test(story2, story1, alternative = "greater", paired=T)  
> detach()
```

15.3 The Kruskal-Wallis Test

The Kruskal-Wallis test is the nonparametric analog of the one-way ANOVA. Simple random samples of sizes n_1, n_2, \dots, n_I are drawn from I independent populations and these $N = n_1 + n_2 + \dots + n_I$ observations are ranked. If R_i denotes the sum of the ranks for the i^{th} sample, then H , the Kruskal-Wallis statistic, is computed by

$$H = \frac{12}{N(N+1)} \sum \frac{R_i^2}{n_i} - 3(N+1).$$

For populations with the same underlying continuous distribution and for large n_i , H is approximately a chi-square distribution with $I - 1$ degrees of freedom. We use H to test the hypothesis H_0 : All treatments (populations) have the same distribution, against H_a : At least two of the treatments do not follow the same distribution. We reject H_0 for large values of H .

Example 15.3 Weeds among the corn (IPS example 15.13)

Sixteen small plots of ground were planted with corn, and each plot weeded to allow a fixed number of lamb's-quarter plants to grow in each meter of corn row. The corn yields (bushels per acre) are shown below.

Weeds per meter	Corn yield			
0	166.7	172.2	165.0	176.9
1	166.2	157.3	166.7	161.1
3	158.6	176.4	153.1	156.0
9	162.8	142.4	162.7	162.4

We wish to test the hypothesis H_0 : Yields have the same distribution in all groups against H_a : Yields are not the same across groups.

1. Open the data set [Example15.13](#) from the `IPSdata` library.
2. At the menu, select `Statistics > Compare Samples > k Samples > Kruskal-Wallis Rank Sum Test...`
3. For `Data Set:` enter [Example15.13](#).
4. For `Variable:` enter `yield`, and for `Grouping Variable:` enter `weeds`.

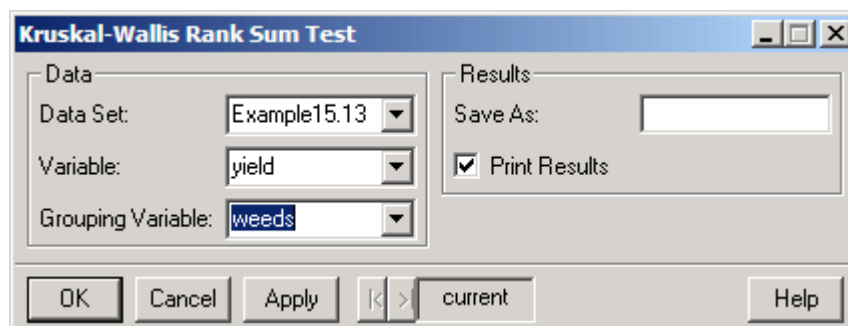


Figure 15.4

5. Click [OK](#).

```
Kruskal-Wallis rank sum test

data: yield and weeds from data set Example15.13
Kruskal-Wallis chi-square = 5.5725, df = 3, p-value = 0.1344
alternative hypothesis: two.sided
```

We do not have strong evidence that weeds decrease corn yield.

End of Example 15.3

```
Command Line Notes

The function kruskal.test performs the Kruskal-Wallis test.

attach(Example15.13)
kruskal.test(yield, weeds)
detach()
```

15.4 Exercises

Use a nonparametric test for all of these problems.

1. A psychologist wishes to test the hypothesis that biology majors are better than statistics majors at a certain memory test. He randomly selects eight statistics majors and eight biology majors at his university and administers this test. Suppose he obtains the following scores (out of 100):

Statistics	70	94	86	60	74	65	82	72
Biology	93	95	85	90	75	81	80	82

Is there sufficient evidence to conclude that biology majors score higher on this test?

2. Suppose a junior-high track coach wants to see if motivational tapes can help her runners run a faster 400 meters. She chooses six boys randomly from her team, times them in a 400-meter run, then has them spend 20 minutes a day listening to motivational tapes for a week, before timing them again in a 400-meter run. She obtained the following times (in seconds):

Before	58.9	62.0	55.5	63.2	61.5	60.0
After	57.5	61.0	57.0	61.4	62.0	58.1

Is there any evidence that motivational tapes influence runner's speed?

3. An economist wants to know if stock prices of companies in his state differ by education of the CEO (Chief Executive Officer). He groups the companies according to the CEO's highest degree—a Bachelor's, Master's, or Doctorate—and randomly selects seven companies from each group. He obtains the following data on stock prices:

B.A.	65	64	87	53	70	85	69
M.A.	85	82	71	64	91	90	83
Ph.D.	55	60	63	69	89	82	58

Is there any evidence that stock prices differ by highest degree of the CEO?

15.5 Solutions

1. Create a data set with one column holding the biology test scores and the second column holding the statistics test scores.

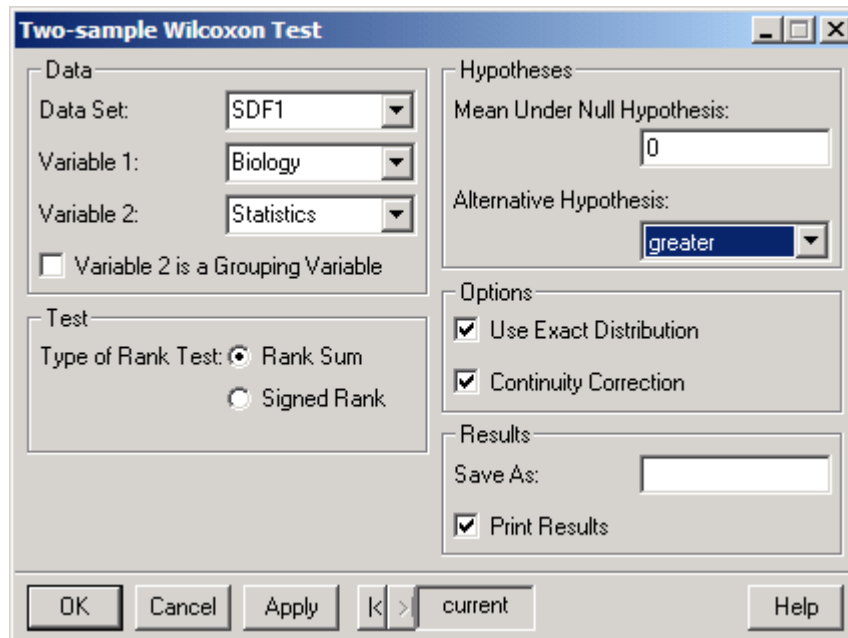


Figure 15.5

```
> Statistics = c(70, 94, 86, 60, 74, 65, 82, 72)
> Biology = c(93, 95, 85, 90, 75, 81, 80, 82)
> wilcox.test(Biology, Statistics, alternative = "greater")
```

Note that by choosing the first variable to be `Biology` above, the alternative hypothesis is that biology majors score higher than statistics majors on this test (`alternative="greater"`).

The P -value of 0.0463 is marginally significant at $\alpha = 0.05$.

2. Create a data set with one column holding the “before” times and the second column holding the “after” times.

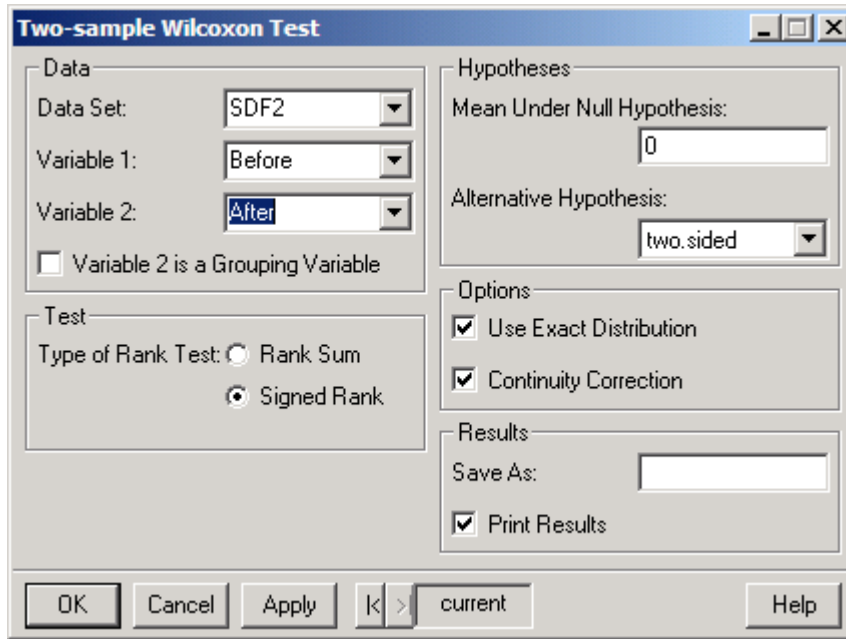


Figure 15.6

From the command line,

```
> before = c(58.9, 62, 55.5, 63.2, 61.5, 60)
> after = c(57.5, 61, 57, 61.4, 62, 58.1)
> wilcox.test(before, after, paired = T)
```

The P -value is 0.3125, so we cannot conclude that motivational tapes help the runners on the 400-meter run.

3. Create a data set with the first column containing all 21 prices and the second column containing the degree information for each observation.

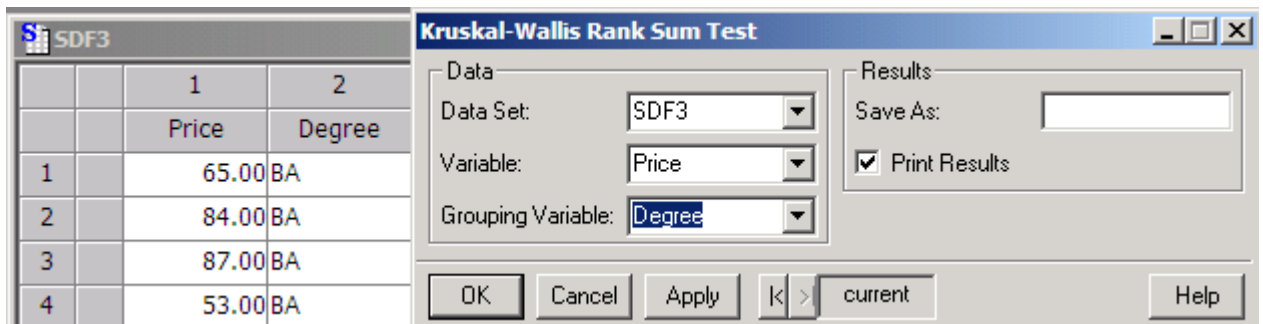


Figure 15.7

From the command line,

```
> price = scan()
1: 65 64 87 53 70 85 69
```

```
8: 85 82 71 64 91 90 83
15: 55 60 63 69 89 82 58
22:
> degree = rep(c("BA", "MA", "PhD"), each = 7)
> kruskal.test(price, degree)
```

With a P -value of 0.0997, we do not have strong evidence that stock prices differ by degree of the CEO.

Chapter 16

Logistic Regression

16.1 The Logistic Regression Model

If p represents the proportion of successes in a population, we consider the model

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x,$$

where x is a numeric explanatory variable.

Example 16.1 The `cheese` data

The data set `cheese` is described in the Data Appendix of IPS. The response variable is `taste`, a score on a taste test. The explanatory variables are `acetic`, `H2S`, and `lactic`. We will convert the taste variable into a binary (logical) response variable and use `acetic` as an explanatory variable.

1. Open the `cheese` data set from the IPSdata library.
2. At the menu, select `Data > Transform ...`.
3. In the `Transform` dialog box, select `cheese` for the `Data Set:` and type `tasteOK` for the `Target Column:`.
4. In the `Expression:` field, type `taste >= 37`. This returns a logical vector, with `TRUE` when `taste` is at least 37, and `FALSE` otherwise.

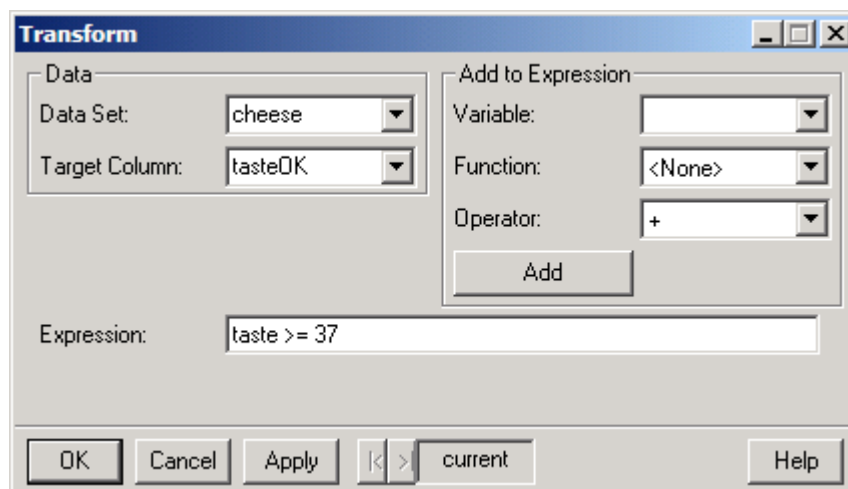


Figure 16.1

5. Click **OK**.

That creates a new column called **tasteOK**. The first five entries down the column are **F, F, T, T, F**.

6. At the menu, select **Statistics > Regression > Logistic...** This brings up the **Logistic Regression** dialog.

7. For **Data Set:** select **cheese**, for **Response:** select **tasteOK**, and for **Explanatory:** select **acetic**.

The **Link:** field should say **logit**.

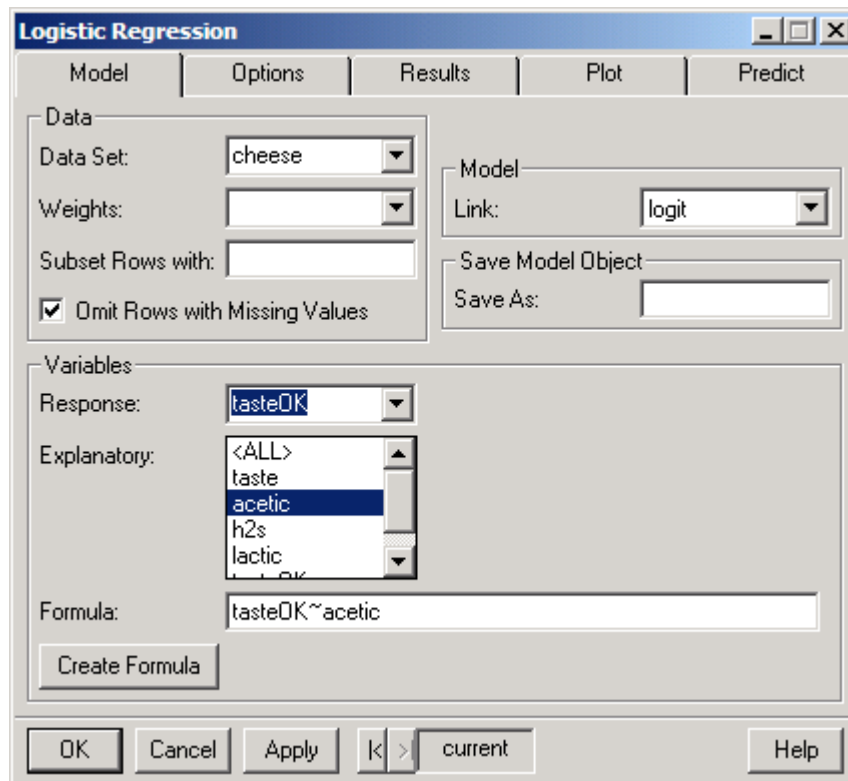


Figure 16.2

8. Click **OK**.

Here is part of the output:

```
...
Coefficients:
              Value Std. Error  t value
(Intercept) -13.704859   5.910583  -2.318698
          acetic   2.248974   1.023642   2.197030

(Dispersion Parameter for Binomial family taken to be 1 )

      Null Deviance: 34.79491 on 29 degrees of freedom
Residual Deviance: 28.22684 on 28 degrees of freedom
```

S-PLUS outputs estimated regression coefficients, their standard errors, and the t value (estimate/standard error). This t value is compared to a standard normal distribution.

For this example, the equation is $\log\left(\frac{\hat{p}}{1-\hat{p}}\right) = -13.704 + 2.248 \cdot \text{acetic}$. The standard error for the slope estimate is 1.023, so a 95% confidence interval for the slope can be found by computing

$$2.248 \pm (1.96)1.023 = (0.249, 4.253)$$

S-PLUS does not compute a χ -square statistic, but you can square the t value for the slope variable (2.197^2) to compute it.

End of Example 16.1

Command Line Notes

We first create a new logical column called `tasteOK` with `TRUE` if `taste` ≥ 37 and `FALSE` otherwise.

```
> cheese = cheese # Make copy of cheese, so can change the copy
> cheese$tasteOK = (cheese$taste >= 37)
> cheese$tasteOK
[1] F F T T F ...
```

For the regression, use the `glm` command.

```
> cheese.fit = glm(tasteOK ~ acetic,
+                 family = binomial(link = logit),
+                 data = cheese)
> summary(cheese.fit)
```

16.2 Multiple Logistic Regression

The procedure for finding coefficients for a multiple logistic regression is nearly identical to that for the multiple linear regression (see Chapter 11).

Example 16.2 Cheese, continued

We now consider the model $\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 \cdot \text{acetic} + \beta_2 \cdot \text{h2s} + \beta_3 \cdot \text{lactic}$. Return to the previous example starting at step 6 and fill in the [Logistic Regression](#) dialog box as follows:

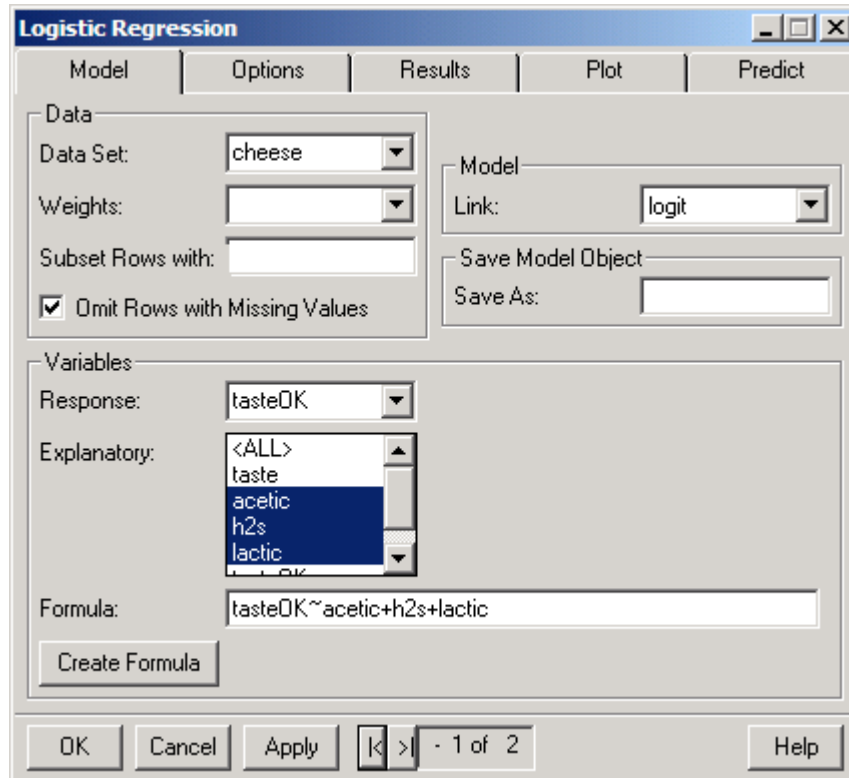


Figure 16.3

Here is part of the output:

```

*** Generalized Linear Model ***
...
Coefficients:
              Value Std. Error  t value
(Intercept) -14.2598841  8.2533771 -1.7277636
      acetic   0.5844239  1.5401678  0.3794547
        h2s   0.6848284  0.4027545  1.7003616
       lactic   3.4683719  2.6437532  1.3119121

(Dispersion Parameter for Binomial family taken to be 1 )

Null Deviance: 34.79491 on 29 degrees of freedom

Residual Deviance: 18.46055 on 26 degrees of freedom

```

From the output, we find

$$\log\left(\frac{\hat{p}}{1-\hat{p}}\right) = -14.259 + 0.584 \cdot \text{acetic} + 0.685 \cdot \text{h2s} + 3.469 \cdot \text{lactic}.$$

End of Example 16.2

Command Line Notes

```
> cheese.fit2 = glm(tasteOK ~ acetic + h2s + lactic,  
+                 data = cheese,  
+                 family = binomial(link = logit))  
> summary(cheese.fit2)
```

16.3 Exercise

1. An S-PLUS provided data set `kyphosis` gives data on 81 children who had corrective spinal surgery. The variable `Kyphosis` is binary indicating whether kyphosis (a postoperative deformity) is “present” or “absent”. The remaining variables `Age`, `Number`, `Start` are numeric. Perform a multiple logistic regression to relate the probability of developing Kyphosis to the three number variables.

16.4 Solution

1. Open the `kyphosis` data set using `Data > Select Data` (see Section 0.4.1). Then run the logistic regression:

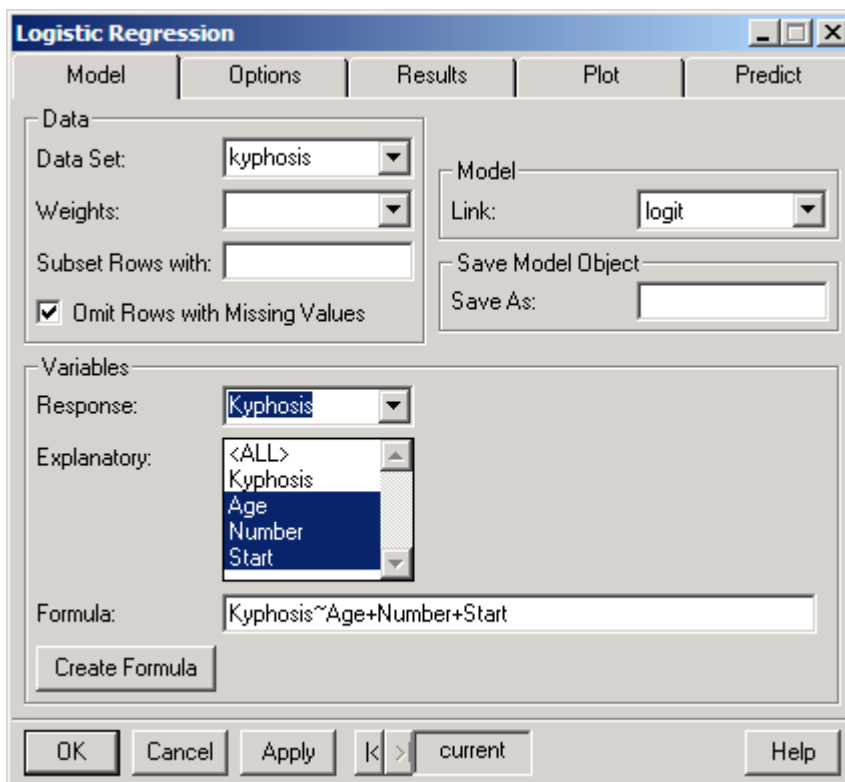


Figure 16.4

From the command line,

```
> kyph.fit = glm(Kyphosis ~ Age + Number + Start,  
+               family = binomial(link = logit),  
+               data = kyphosis)  
> summary(kyph.fit)
```

The solution is $\log\left(\frac{\hat{p}}{1-\hat{p}}\right) = -2.04 + 0.01 \cdot \text{Age} + 0.41 \cdot \text{Number} - 0.21 \cdot \text{Start}$.

Index

- #, 13
- \$, 17
- +, 12
- :, 15
- <-, 14
- =, 14
- >, 12
- [,], 17
- [], 16

- abbreviation of arguments, 19
- `abline`, 60
- annotations, 41
- ANOVA, 145–152, 203
 - contrasts, 150–152
 - multiple comparisons, 148–152
 - two-way, 155–161
- `aov`, 161
- `Apply`, 4
- `args`, 19
- `as.factor`, 163
- assign, 14
- assignment operator, 14
- `attach`, 18

- bar graphs, 20–26
- `barplot`, 26
- binomial distribution, 76–78
- bootstrap, 165–198
 - bias estimate, 173
 - confidence interval, 173
 - distribution, 169
 - graphical, 189
 - percentile confidence interval, 173
 - standard error, 169, 173
 - t* confidence interval, 173
- `boxplot`, 32, 34
- brush and spin, 50

- `c`, 15
- cancel, 16
- `cbind`, 110
- central limit theorem, 172
- charts, *see* graphs
- `chisq.test`, 110, 113
- column header, 9

- `Commands window`, 12–19
- comparison operator, 17
- confidence interval, 82–84, 88–89, 93, 99–103
 - regression
 - response, 120–123
 - slope, 120
- Consumer Reports*, 20, 31
- contingency table, *see* two-way table
- continuation prompt, 12
- continuity correction, 100
- contrasts, 150–152
- Cook’s distance, 129
- `cor`, 54
- correlation, 53–55
- critical values, 82
- `crosstabs`, 26, 113
- `crosstabulations`, 24, 108, 111–113
- `cu.summary`, *see* data sets
- `cumsum`, 72, 81

- data frame, 7, 17–19
- data sets
 - `car.test.frame`, 60
 - cheese, 208
 - CLEC, 170, 175
 - Corn soy blend, 166
 - `csdata`, 135
 - `cu.summary`, 20, 63
 - Degree of Reading Power, 94, 180
 - diabetes, 55
 - education, 22
 - French language test, 91, 176
 - `fuel.frame`, 3, 17, 31, 45, 63, 66, 148
 - kyphosis, 212
 - `market.survey`, 111
 - Newcomb, 96
 - reading, 145
 - Verizon, 170
- data tip, 8, 46
- `data.frame`, 19
- `dbinom`, 78
- Delete, 11
- density curve, 28, 179
- Design Plot, 157
- `detach`, 18

`dev.cur`, 42
`dev.list`, 42
Distribution Functions, 34–38, 76, 80
`dnorm`, 40
`dotchart`, 23, 26

editing graphs, *see* graphs, editing
Exit, 11

F distribution, 75, 145
factor, 7, 8, 20
Fill Numeric Columns, 36, 72
filter, 67
fitted, 60
fitted values, 56
formula, 120
fuel.frame, *see* data sets
function, 19

`glm`, 210
graphs
 bar graphs, 20–26
 boxplot, 32–34
 brush and spin, 50
 dotchart, 23
 editing, 40–42
 histogram, 26–29
 Interaction Plot, 158
 line plots, 29–31
 normal quantile plot, 39
 Pareto chart, 24
 pie charts, 20–26
 scatterplot matrix, 48
 scatterplot smoother, 51
 scatterplots, 45–53
 time plots, 29–31
graphsheet, 42, 161

header, 9, 10
`hist`, 29
histogram, 26, 29, 179
hypothesis test
 ANOVA, 145–148
 chi-square, 106–113
 Kruskal-Wallis, 203–204
 permutation test, 165–198
 proportions, 99–103
 regression
 slope, 120
 t test, 89–95
 Wilcoxon rank sum, 199–201
 Wilcoxon signed rank, 201–203
 z test, 82–85

import data, 6
 Import From File, 6
influential point, 129
Interaction Plot, 158
`interaction.plot`, 161
interrupt, 16
IPSdata, 4–5
IPSdata library, 1, 4, 165

Kernel, 51
Kruskal-Wallis test, 203–204
`kruskal.test`, 204
ksmooth, 53

Law of Large Numbers, 70
least-squares regression, 55–60
legend, 47
length, 19
level, 7, 20
line plots, 29–31
`lm`, 60
Loess, 51
logical operator, 17
logistic regression, 208–212

market.survey, *see* data sets
matched pairs, 91
matrix, 110
mean, 31, 34
mean, trimmed, 174
median, 31
missing value, *see* NA
mode, 17
model formula, 60, 62, 120, 137, 160
model object, 118, 129
`multicomp`, 152
multiple comparisons, 148–152

NA, 8, 13, 112
names, 10, 14
normal distribution, 34–40
normal quantile plot, 39
`normal.sample.size`, 85

Object Explorer, 11
objects, 11, 14
outlier, 125

P-value, 179
pairs, 53
`panel.smooth`, 53
`par`, 129, 161
Pareto chart, 24
`pbinom`, 78
Pearson’s chi-square test, 107, 111
permutation test, 165–198

- P*-value, 179
- pf, 81
- pie, 26
- pie charts, 20–26
- plot, 31, 53, 129
- plot.design, 161
- plot.factor, 161
- plots, *see* graphs
- pnorm, 40
- power, 84–85
- precision, 9
- predict, 123
- prediction interval, 122
- prompt, 12
- prop.test, 101–103
- Proportions Test, 99–102

- q, 19
- qnorm, 40
- QQ Normal, *see* normal quantile plot
- qqline, 40
- qqnorm, 40
- quantile, 148
- quartiles, 31, 32

- Random Numbers, 79
- random sample, 66–69
- rbinom, 81
- regression
 - diagnostics, 123–129
 - least-squares, 55–60
 - multiple, 135–142
- rep, 15
- Resample library, 1, 6, 165
- resampling, 165
- resid, 60, 144
- residuals, 56, 120
- rf, 81
- rm, 15
- rmvnorm, 54
- rnorm, 81
- row header, 10
- rowMeans, 81
- rt, 81

- sample, 69
- save, 14
- scatter.smooth, 53
- scatterplots, 45–53
 - brush and spin, 50
 - matrix, 48
 - smoother, 51
 - trellis, 62
- Script window, 12–14
- SE, 169, 173

- search path, 18
- seed, 67, 70
- Select Data, 3
- seq, 15
- Set Conditioning Mode, 61
- set.seed, 69
- significant digits, *see* precision
- smooth, 53
- smooth.spline, 53
- smoother, 124
- spin, *see* brush and spin
- Spline, 51
- split, 34, 148
- standard deviation, 31
- standard error, 169
- stdev, 34
- stem, 29
- subscripting, 16–18
- summary, 34, 60
- summary statistics, 31–34
 - conditional, 32
- Supersmooth, 51
- supsmu, 53

- t* distribution, 80, 88–97
- t* interval, 88–97
- t* test, 88–97
- t.test, 91, 93
- table, 26
- tapply, 148, 161
- test of association, 106
- test of homogeneity, 106
- test of independence, 106
- Text as Symbol, 46
- time plots, 29–31
- Transform, 72, 92
- trellis plot, 62
- trimmed mean, 174
- two-way table, 106–113

- vector, 15–17

- wilcox.test, 201, 203
- Wilcoxon rank sum test, 199–201
- Wilcoxon signed rank test, 201–203
- Wilson estimate, 100
- working database, 11, 12

- xyplot, 62

- Yates’ continuity correction, 100, 112

- z* interval, 82–85
- z* test, 82–85