Data Analysis Using Stata

Third Edition

First edition 2005
Second edition 2009
Third edition 2012

Published by Stata Press, 4905 Lakeway Drive, College Station, Texas 77845
Typeset in LaTeX 2$_\varepsilon$
Printed in the United States of America
10 9 8 7 6 5 4 3 2 1

ISBN-10: 1-59718-110-2
ISBN-13: 978-1-59718-110-5

Library of Congress Control Number: 2012934051

Contents

(Pages omitted)

Preface

As you may have guessed, this book discusses data analysis, especially data analysis using Stata. We intend for this book to be an introduction to Stata; at the same time, the book also explains, for beginners, the techniques used to analyze data.

Data Analysis Using Stata does not merely discuss Stata commands but demonstrates all the steps of data analysis using practical examples. The examples are related to public issues, such as income differences between men and women, and elections, or to personal issues, such as rent and living conditions. This approach allows us to avoid using social science theory in presenting the examples and to rely on common sense. We want to emphasize that these familiar examples are merely standing in for actual scientific theory, without which data analysis is not possible at all. We have found that this procedure makes it easier to teach the subject and use it across disciplines. Thus this book is equally suitable for biometricians, econometricians, psychometricians, and other "metricians"—in short, for all who are interested in analyzing data.

Our discussion of commands, options, and statistical techniques is in no way exhaustive but is intended to provide a fundamental understanding of Stata. Having read this book and solved the problems in it, the reader should be able to solve all further problems to which Stata is applicable.

We strongly recommend to both beginners and advanced readers that they read the preface and the first chapter (entitled *The first time*) attentively. Both serve as a guide throughout the book. Beginners should read the chapters in order while sitting in front of their computers and trying to reproduce our examples. More-advanced users of Stata may benefit from the extensive index and may discover a useful trick or two when they look up a certain command. They may even throw themselves into programming their own commands. Those who do not (yet) have access to Stata are invited to read the chapters that focus on data analysis, to enjoy them, and maybe to translate one or another hint (for example, about diagnostics) into the language of the statistical package to which they do have access.

Structure

The first time (chapter 1) shows what a typical session of analyzing data could look like. To beginners, this chapter conveys a sense of Stata and explains some basic concepts such as variables, observations, and missing values. To advanced users who already have experience in other statistical packages, this chapter offers a quick entry into Stata.

Advanced users will find within this chapter many cross-references, which can therefore be viewed as an extended table of contents. The rest of the book is divided into three parts, described below.

Chapters 2–6 serve as an introduction to the basic tools of Stata. Throughout the subsequent chapters, these tools are used extensively. It is not possible to portray the basic Stata tools, however, without using some of the statistical techniques explained in the second part of the book. The techniques described in chapter 6 may not seem useful until you begin working with your own results, so you may want to skim chapter 6 now and read it more carefully when you need it.

Throughout chapters 7–10, we show examples of data analysis. In chapter 7, we present techniques for describing and comparing distributions. Chapter 8 covers statistical inference and explains whether and how one can transfer judgments made from a statistic obtained in a dataset to something that is more than just the dataset. Chapter 9 introduces linear regression using Stata. It explains in general terms the technique itself and shows how to run a regression analysis using an example file. Afterward, we discuss how to test the statistical assumptions of the model. We conclude the chapter with a discussion of sophisticated regression models and a quick overview of further techniques. Chapter 10, in which we describe regression models for categorical dependent variables, is structured in the same way as the previous chapter to emphasize the similarity between these techniques.

Chapters 11–13 deal with more-advanced Stata topics that beginners may not need. In chapter 11, we explain how to read and write files that are not in the Stata format. At the beginning of chapter 12, we introduce some special tools to aid in writing do-files. You can use these tools to create your own Stata commands and then store them as ado-files, which are explained in the second part of the chapter. It is easy to write Stata commands, so many users have created a wide range of additional Stata commands that can be downloaded from the Internet. In chapter 13, we discuss these user-written commands and other resources.

Using this book: Materials and hints

The only way to learn how to analyze data is to do it. To help you learn by doing, we have provided data files (available on the Internet) that you can use with the commands we discuss in this book. You can access these files from within Stata or by downloading a zip archive.

Please do not hesitate to contact us if you have any trouble obtaining these data files and do-files.[1]

[1] The data we provide and all commands we introduce assume that you use Stata 12 or higher. Please contact us if you have an older version of Stata.

- If the machine you are using to run Stata is connected to the Internet, you can download the files from within Stata. To do this, type the following commands in the Stata Command window (see the beginning of chapter 1 for information about using Stata commands).

  ```
  . mkdir c:\data\kk3
  . cd c:\data\kk3
  . net from http://www.stata-press.com/data/kk3/
  . net get data
  ```

  These commands will install the files needed for all chapters except section 11.4. Readers of this section will need an additional data package. You can download these files now or later on by typing

  ```
  . mkdir c:\data\kk3\kksoep
  . cd c:\data\kk3\kksoep
  . net from http://www.stata-press.com/data/kk3/
  . net get kksoep
  . cd ..
  ```

  If you are using a Mac or Unix system, substitute a suitable directory name in the first two commands, respectively.

- The files are also stored as a zip archive, which you can download by pointing your browser to http://www.stata-press.com/data/kk3/kk3.zip.

  To extract the file `kk3.zip`, create a new folder: `c:\data\kk3`. Copy `kk3.zip` into this folder. Unzip the file `kk3.zip` using any program that can unzip zip archives. Most computers have such a program already installed; if not, you can get one for free over the Internet.[2] Make sure to preserve the `kksoep` subdirectory contained in the zip file.

Throughout the book, we assume that your current working directory (folder) is the directory where you have stored our files. This is important if you want to reproduce our examples. At the beginning of chapter 1, we will explain how you can find your current working directory. Make sure that you do not replace any file of ours with a modified version of the same file; that is, avoid using the command `save, replace` while working with our files.

We cannot say it too often: the only way to learn how to analyze data is to analyze data yourself. We strongly recommend that you reproduce our examples in Stata as you read this book. A line that is written `in this font` and begins with a period (which itself should not be typed by the user) represents a Stata command, and we encourage you to enter that command in Stata. Typing the commands and seeing the results or graphs will help you better understand the text, because we sometimes omit output to save space.

As you follow along with our examples, you must type all commands that are shown, because they build on each other within a chapter. Some commands will only work if

[2] For example, “pkzip” is free for private use, developed by the company PKWARE. You can find it at http://pkzip.en.softonic.com/.

you have entered the previous commands. If you do not have time to work through a whole chapter at once, you can type the command

```
. save mydata, replace
```

before you exit Stata. When you get back to your work later, type

```
. use mydata
```

and you will be able to continue where you left off.

The exercises at the end of each chapter use either data from our data package or data used in the Stata manuals. StataCorp provides these datasets online.[3] They can be used within Stata by typing the command `webuse` *filename*. However, this command assumes that your computer is connected to the Internet; if it is not, you have to download the respective files manually from a different computer.

This book contains many graphs, which are almost always generated with Stata. In most cases, the Stata command that generates the graph is printed above the graph, but the more complicated graphs were produced by a Stata do-file. We have included all of these do-files in our file package so that you can study these files if you want to produce a similar graph (the name of the do-file needed for each graph is given in a footnote under the graph).

If you do not understand our explanation of a particular Stata command or just want to learn more about it, use the Stata `help` command, which we explain in chapter 1. Or you can look in the Stata manuals, which are available in printed form and as PDF files. When we refer to the manuals, [R] **summarize**, for example, refers to the entry describing the `summarize` command in the *Stata Base Reference Manual*. [U] **18 Programming Stata** refers to chapter 18 of the *Stata User's Guide*. When you see a reference like these, you can use Stata's online help (see section 1.3.16) to get information on that keyword.

Teaching with this manual

We have found this book to be useful for introductory courses in data analysis, as well as for courses on regression and on the analysis of categorical data. We have used it in courses at universities in Germany and the United States. When developing your own course, you might find it helpful to use the following outline of a course of lectures of 90 minutes each, held in a computer lab.

To teach an introductory course in data analysis using Stata, we recommend that you begin with chapter 1, which is designed to be an introductory lecture of roughly 1.5 hours. You can give this first lecture interactively, asking the students substantive questions about the income difference between men and women. You can then answer them by entering Stata commands, explaining the commands as you go. Usually, the students

[3] They are available at http://www.stata-press.com/data/r12/.

name the independent variables used to examine the stability of the income difference between men and women. Thus you can do a stepwise analysis as a question-and-answer game. At the end of the first lecture, the students should save their commands in a log file. As a homework assignment, they should produce a commented do-file (it might be helpful to provide them with a template of a do-file).

The next two lectures should work with chapters 3–5 and can be taught a bit more conventionally than the introduction. It will be clear that your students will need to learn the *language* of a program first. These two lectures need not be taught interactively but can be delivered section by section without interruption. At the end of each section, give the students time to retype the commands and ask questions. If time is limited, you can skip over sections 3.3 and 5.7. You should, however, make time for a detailed discussion of sections 5.1.4 and 5.1.5 and the examples in them; both sections contain concepts that will be unfamiliar to the student but are very powerful tools for users of Stata.

One additional lecture should suffice for an overview of the commands and some interactive practice in the graphs chapter (chapter 6).

Two lectures can be scheduled for chapter 7. One example for a set of exercises to go along with this chapter is given by Donald Bentley and is described on the webpage http://www.amstat.org/publications/jse/v3n3/datasets.dawson.html. The necessary files are included in our file package.

A reasonable discussion of statistical inference will take two lectures. The material provided in chapter 8 shows necessary elements for simulations, which allows for a hands-on discussion of sampling distributions. The section on multiple imputation can be skipped in introductory courses.

Three lectures should be scheduled for chapter 9. According to our experience, even with an introductory class, you can cover sections 9.1, 9.2, and 9.3 in one lecture each. We recommend that you let the students calculate the regressions of the Anscombe data (see page 279) as a homework assignment or an in-class activity before you start the lecture on regression diagnostics.

We recommend that toward the end of the course, you spend two lectures on chapter 11 introducing data entry, management, and the like, before you end the class with chapter 13, which will point the students to further Stata resources.

Many of the instructional ideas we developed for our book have found their way into the small computing lab sessions run at the UCLA Department of Statistics. The resources provided there are useful complements to our book when used for introductory statistics classes. More information can be found at http://www.stat.ucla.edu/labs/, including labs for older versions of Stata.

In addition to using this book for a general introduction to data analysis, you can use it to develop a course on regression analysis (chapter 9) or categorical data analysis (chapter 10). As with the introductory courses, it is helpful to begin with chapter 1, which gives a good overview of working with Stata and solving problems using Stata's online help. Chapter 13 makes a good summary for the last session of either course.

(Pages omitted)

6.4 Multiple graphs

In Stata, you can create multiple graphs in several different ways. By “multiple graphs”, we mean graphs that consist of different graph parts, in particular,

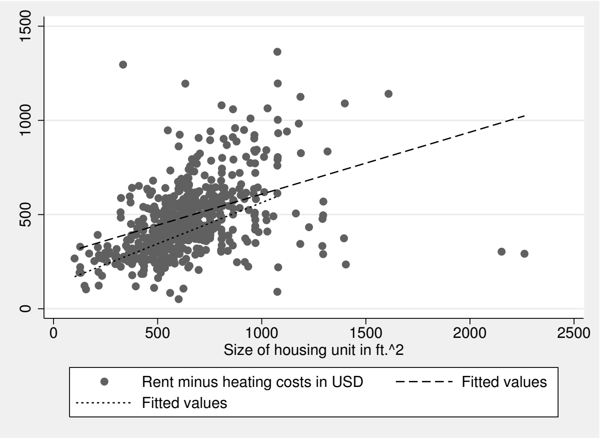
- `twoway` graphs that are plotted on top of each other,
- graphs that are broken out with the `by()` option and are then displayed together, and
- varying graphs that are combined using the `graph combine` command.

We will now quickly introduce these three types of graphs.

6.4.1 Overlaying many twoway graphs

You can overlay as many types of `twoway` graphs as you want in the same coordinate system. In the following example, three graphs are placed on top of each other: a scatterplot; a *linear fit* (or *regression line*; see chapter 9) for the same data, but restricted to the old federal states in West Germany (`rent_w`); and finally, a linear fit that is restricted to the new federal states in East Germany (`rent_e`).

```
. twoway || scatter rent size || lfit rent_w size || lfit rent_e size
```



To overlay twoway graphs, you consolidate the graphs in a single `twoway` command, separated by parentheses or two vertical lines. In this book and our own work, we use two vertical lines because there are already so many parentheses in the graph syntax. The two vertical lines are particularly readable in do-files containing the individual graphs one after another with line breaks commented out (see section 2.2.2).

If you are combining several twoway graphs, you can specify options that correspond to the respective graph types, as well as twoway options that apply to all the graphs to be combined. Generally, the syntax for overlaid twoway graphs is as follows:

`twoway`

`||` `scatter` *varlist*`,` *scatter_options*

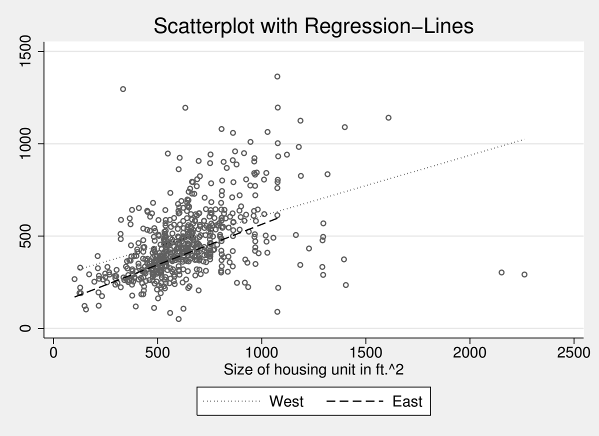
`||` `lfit` *varlist*`,` *lfit_options*

`||` *plottype varlist*`,` *plottype_options*`,`

`||` *twoway_options*

The first and the last two vertical bars are superfluous. However, we tend to use them for long graph commands to enhance readability. This syntax structure can be illustrated with an example (but note that commands of that length are often better typed into the Do-file Editor than into the command line):

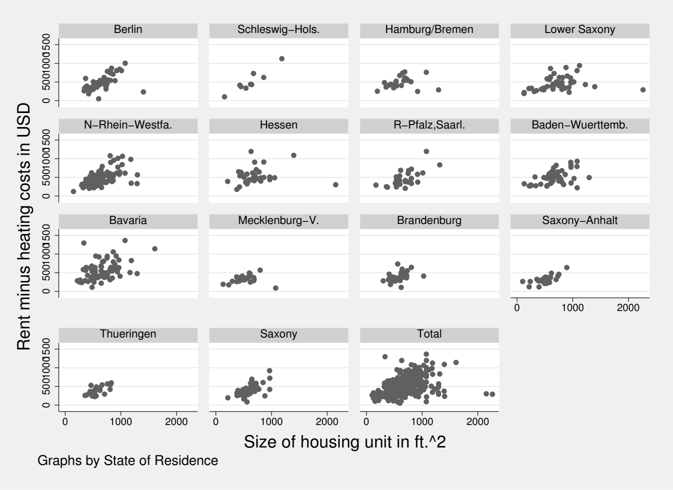
```
. twoway || scatter rent size, msymbol(oh)
> || lfit rent_w size, clpattern(dot)
> || lfit rent_e size, clpattern(dash)
> || , title("Scatterplot with Regression-Lines") legend(order(2 "West" 3 "East"))
```



6.4.2 Option by()

The `by()` option displays separate graphs for each group defined by the variable in the parentheses. If more than one variable is entered in the parentheses, graphs are provided for every combination of the chosen variables. If you also specify the `total` suboption, another graph is displayed without separating it by group. Other suboptions control the positioning (for example, `rows()` and `cols()`), the display or omission of individual axes (for example, [`no`]`ixaxes`), or the appearance of the margins between the individual graphs. For the list of suboptions, see `help by_option` or [G-3] ***by_option***. One example should be enough at this point:

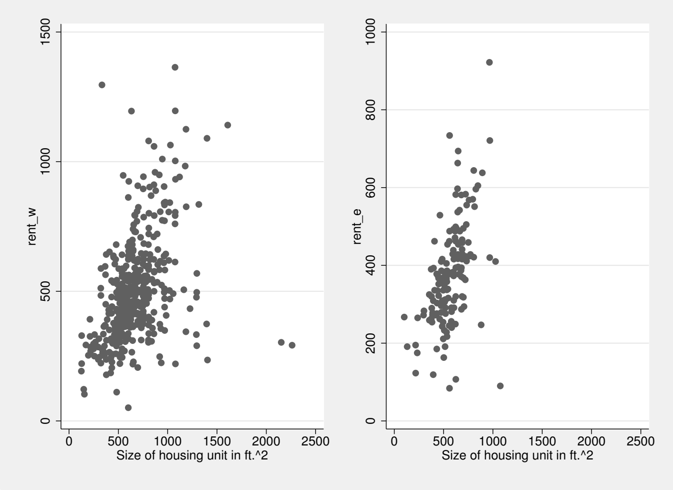
```
. scatter rent size, by(state, total)
```



6.4.3 Combining graphs

Stata allows you to combine as many graphs as you want into a joint graph. To do this, you first save the individual graphs and then combine them using `graph combine`. We will demonstrate this using a display of `rent` by `size`, separated by respondents from East and West Germany:

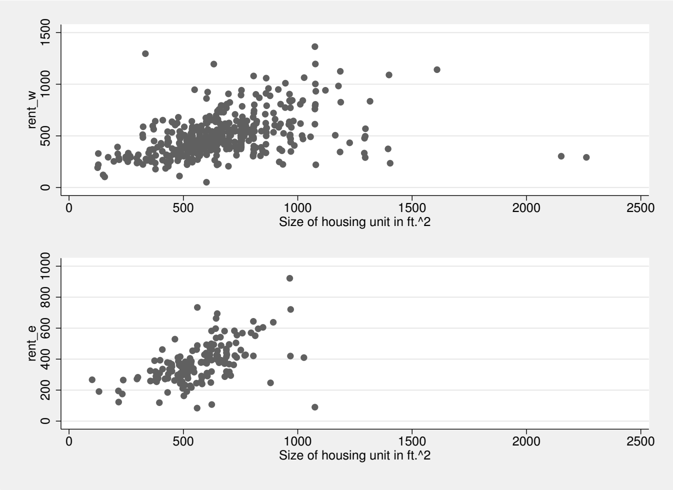
```
. scatter rent_w size, name(west, replace)
. scatter rent_e size, name(east, replace)
. graph combine west east
```



To save both graphs, we use the `name()` option, which specifies the name under which the graph will be saved in the computer's memory. The `replace` suboption tells Stata to delete any graphs already saved under this name. We then combine the two graphs using `graph combine`.

The `graph combine` command has a series of options for controlling how the combined graph is to be displayed. To begin with, it is important to set the number of rows and columns in the combined graph. The individual graphs are placed in the combined graph in rows and columns in a matrix-like fashion. The positioning of the individual graphs depends on how many rows and columns the matrix has. In the matrix above, one row and two columns were used. Here you will see what happens if we instead use two rows and one column:

```
. graph combine west east, rows(2)
```

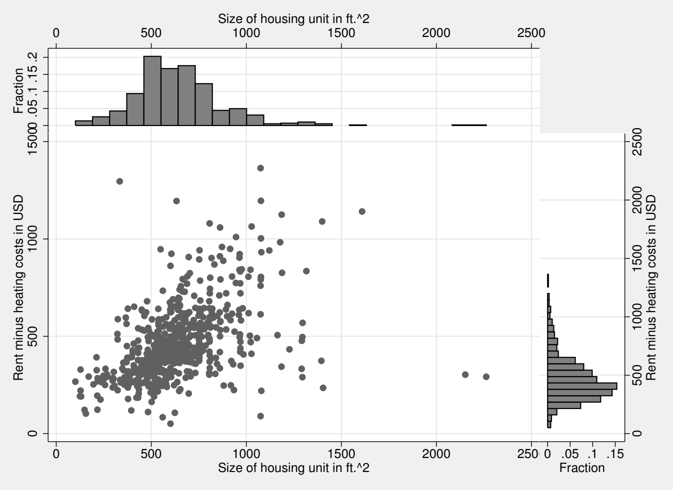


The number of individual graphs you can put in a multiple graph is limited only by printer and screen resolutions. If the main issue is the readability of the labels, you can increase their size with the `iscale()` option. The default font size decreases with every additional graph. With `iscale(1)`, you can restore the text to its original size; `iscale(*.8)` restores the text to 80% of its original size.

If you want the individual graph parts of the combined graph to have different sizes, you will have to save the graphs with different sizes before combining them. However, you cannot use the `xsize()` and `ysize()` options discussed in section 6.3.2, because these sizes are not taken into account by `graph combine`. Instead, you will have to use the forced-size options `fysize()` and `fxsize()`, which tell Stata to use only a certain percentage of the available space. For example, the `fxsize(25)` option creates a graph that uses only 25% of the width of the available space; correspondingly, a graph created using the `fysize(25)` option uses only 25% of the available height.

Here is a slightly more advanced example of **graph combine** using the forced-size options.

```
. twoway scatter rent size, name(xy, replace)
> xlabel(, grid) ylabel(, grid gmax)
. twoway histogram size,  name(hx, replace) fraction
> xscale(alt) xlabel(, grid gmax) fysize(25)
. twoway histogram rent, fraction name(hy, replace) horizontal
> yscale(alt) ylabel(0(500)2500, grid gmax) fxsize(25)
. graph combine hx xy hy, imargin(0 0 0 0) hole(2)
```



For more details on creating such graphs, see [G-2] **graph combine**. The Graph Editor has further capabilities regarding the positioning of graph elements. The repositioning is best done with the Grid Edit Tool. A description of its functionality can be found in [G-1] **graph editor**.

6.5 Saving and printing graphs

To print a Stata graph, you type

```
. graph print
```

The graph displayed in Stata's Graph window is then printed directly. If you create many graphs in a do-file, you can also print them with that do-file by typing the `graph print` command after every graph command. The `graph print` command also has many options that can be used to change the printout of a graph (see `help pr_options`).

You can also print graphs that are not (or are no longer) in the Graph window, but you must first save the graph to memory or to a file on the hard drive. You already learned how to save graphs to memory in the previous section. There you saved numerous graphs with the `name()` option, which we then combined with the `graph combine` command. To print out these graphs, you first display them with `graph display` and then print them using `graph print`. Let us try this with the graph saved as `east` above (see page 150):

```
. graph display east
. graph print
```

Saving graphs to memory is often useful, but they are lost when you close Stata. To print graphs that you created a few days ago, you must have saved them as a file by using the `saving()` option. It works the same way as the `name()` option except that it saves the file to the hard drive. You type the `filename` under which the file is to be saved in the parentheses. By default, Stata uses the `.gph` file extension. To overwrite an existing file of the same name, include the `replace` option in the parentheses. The option `replace` is commonly used when creating graphs with do-files.

Be careful when you use the Graph Editor. When you exit the Graph Editor, you will be asked if you want to save the changes you made to the graph. In most cases, you would want to answer yes. If you do so, you should make sure that you use a *new* filename. If you save the graph with its old name, you might be in trouble next time you run the do-file that created the graph originally. Running the do-file will re-create the original graph and therefore overwrite the changes you made with the Graph Editor. You would need to start over.

All saved files can be, at a later point in time, printed or edited with the Graph Editor. To do so, you simply call the saved graph on the screen with the command `graph use` and start printing or editing. The command

```
. graph combine hx xy hy, hole(2) imargin(0 0 0 0) saving(combined, replace)
```

saves the graph under the name `combined.gph` in the current working directory. If a file with the same name already exists in the working directory, it is overwritten because you specified `replace`. You can now close Stata, shut down the computer, or display another graph . . .

```
. graph display east
```

. . . and regardless of what you do, you can print the graph saved in a file by typing

```
. graph use combined
. graph print
```

Finally, you will usually be exporting a Stata graph to a word processing program or presentation program rather than printing it. For this, you can use the much-loved *copy-and-paste* procedure, where you first copy the graph displayed in a Graph window and then paste it into the respective document. If you are dealing with several graphs, it is better to save the graph in a suitable file format on the hard drive and then import to the desired program when needed.[5]

[5] When doing this and working with do-files, it is a good idea to document your work (see chapter 2).

To save a graph in a different file format, use the **graph export** command. You type the filename with the appropriate file extension after the command. Table 6.1 lists the formats that are available to you.

Table 6.1. Available file formats for graphs

Extension	File format	Restriction
`.ps`	PostScript	
`.eps`	Encapsulated PostScript	
`.wmf`	Windows Metafile	Windows
`.emf`	Windows Enhanced Metafile	Windows
`.pict`	Mac Picture Format	Mac
`.pdf`	Portable Document Format	Windows/Mac
`.png`	Portable Network Graphics	
`.tif`	Tagged-Image File Format	

Microsoft applications normally handle files saved in WMF or EMF formats well. The same applies to most other software that runs under Windows operating systems. PostScript and Encapsulated PostScript work well on Unix systems, and you should use them if you write reports using LaTeX. Mac users will usually prefer PDF or PICT files. In any case, if you want to save the graph in the Graph window as a WMF file called **mygraph1.wmf**, use

```
. graph export mygraph1.wmf
```

Use the other file formats as you wish.

6.6 Exercises

1. Get data from the National Health and Nutrition Examination Study (NHANES) by using the following command:

   ```
   . webuse nhanes2.dta, clear
   ```

2. Using the NHANES data, produce a scatterplot of weight in kg by height in cm. Use hollow circles as the marker symbol.
3. Change the title of the vertical axis to “Weight (in kg)”, and add a note for the data source of your graph.
4. Add reference lines to indicate the arithmetic means of weight and heights.
5. Add an axis label to explain the meaning of the reference lines.
6. Use blue marker symbols for male observations and pink marker symbols for female observations, and construct a self-explanatory legend. Remove the reference lines.

7. Plot the data for men and women separately, and produce a common figure of both plots placed on top of each other. Take care that the note on the data source does not appear twice in the figure.
8. Construct a graph similar to the previous one but this time with reference lines for the gender-specific averages of weight and height.
9. Create a variable holding the body mass index [BMI; see (5.1) on page 112], and classify the observations according to the table on page 112. Change the previous graph so that the colors of the marker symbols represent the categorized BMI.
10. Add the unique person identifier (`sampl`) to the symbols for the male and the female observations with the highest BMI.
11. Export your graphs so that they can be imported into your favorite word processing program.