1 USING STATA FOR DATA ANALYSIS

LEARNING OBJECTIVES

In this chapter you will learn how to:

- Use Stata's general syntax to execute commands
- Execute commands using Stata's graphical user interface
- Write and save a Do-file (a file that contains Stata syntax)
- Create a log file (a file that saves your commands and results)
- Print output and copy/paste output into Word
- Use Stata's help system

STATA COMMANDS AND FUNCTIONS USED

● `codebook`	Provides detailed coding and labeling information about a variable[1]
● `describe`	Provides descriptive information about the data currently in memory
● `graph box`	Creates a box plot showing the distribution of a variable's values (used as an example of command syntax)
● `help`	Accesses command syntax and options
● `log`	Creates a file that records commands and results
● `search`	Searches Stata's resources for desired commands and information
● `set more [on|off]`	Instructs Stata to display all results, even if they fill more than one screen (or turn this setting off)
● `tabulate`	Produces one-way frequency distributions (when applied to one variable)
● `#delimit`	Expression used in the Do-file to set an end-of-line character to a semicolon (or another special character); useful for clearly writing complex commands

Tutorials
Resources
Scan Code
1

[1] The underlined part of the command name shows you how it can be abbreviated in Stata. There aren't abbreviations for all commands. For example, you must type `codebook` for detailed coding and labeling information. But you can often use abbreviations to save time.

By this point, you've executed commands to describe the contents of a dataset and see how a variable is encoded. Commands, datasets, and variables are essential elements of analyzing political science data with Stata. We'll work through many specific examples in this chapter. Now that you've executed a couple of commands, this is a good time to provide a general overview of how Stata commands work so that you can apply them in many different situations.

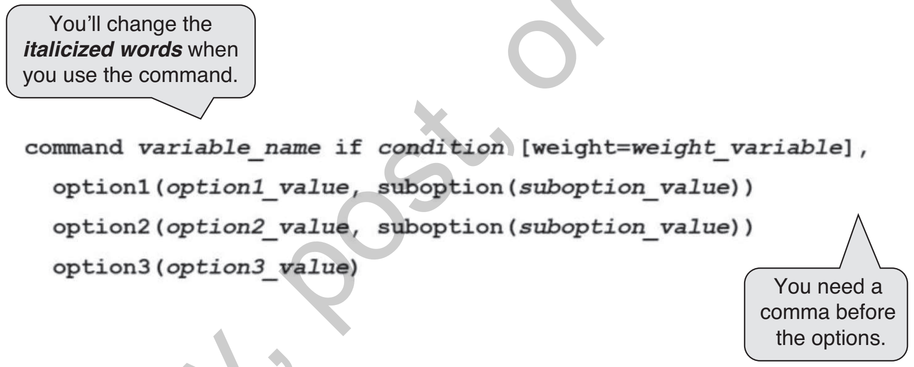
1.1 GENERAL SYNTAX OF STATA COMMANDS

All Stata **commands** have names—usually concise, descriptive names—and most commands operate on variables in datasets. Stata commands follow the same basic logic, regardless of what they do, and understanding the general pattern of commands makes it easier to read them and learn how to use them.

Consider the stylized usage statement in Figure 1.1. It's not a real command, but it can help you understand the logical structure of all Stata commands.

```
command variable_name if if_condition [weight=weight_variable],
  option1(option1_value, suboption(suboption_value))
  option2(option2_value, suboption(suboption_value))
  option3(option3_value)
```

FIGURE 1.1 ■ Standard Syntax of Stata Commands


You'll change the *italicized words* when you use the command.
command variable_name if condition [weight=weight_variable],
option1(option1_value, suboption(suboption_value))
option2(option2_value, suboption(suboption_value))
option3(option3_value)
You need a comma before the options.

In this example, `command` is the name of a Stata command. You know the names of two commands already: `describe` and `codebook`. Commands operate on one or more variables specified by name. You don't have to identify the dataset in your command because Stata can only have one dataset open at a time.

Command names can usually be abbreviated. Stata manual files underline what you need to type to execute a command; for example, `describe` can be abbreviated as `d`, but there is no abbreviation for `codebook`.

In this **generic usage statement**, *`variable _ name`* is a placeholder for the name of the variable. You should expect the *`variable _ name`* to change from one situation to the next. Sometimes, you'll run a command with a list of variables. There are just a few limitations on what you can name variables in Stata; for example, a variable's name can't start with a number or contain spaces. Variable names will often be a mash-up of multiple words. Because you can't have spaces in a variable name, researchers will use periods or underscores in place of spaces (like violent.crime.rate or violent_crime_rate) or write in "camel case" (like violentCrimeRate). Often, you'll work with whatever variable names the dataset author decided to use, but you will have occasions to generate new variables and name them.

For most tasks, you'll need to specify the command and the variable list. These are the basic, required statements. Sometimes, they are all you need. More often, to do things right, you'll need to add additional statements to a command. These additional statements fall into a few categories.

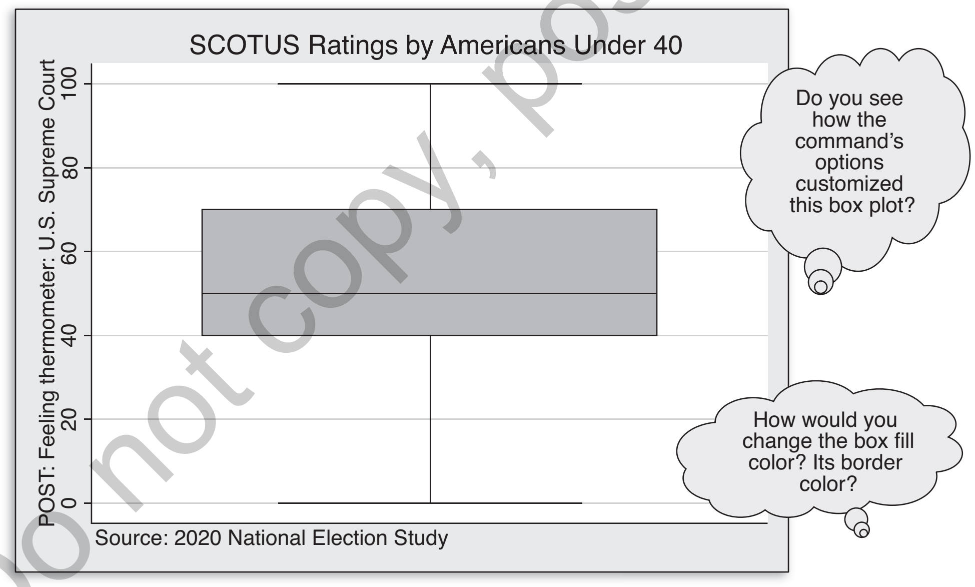
- **If conditions.** You may want the command to operate on the named variable(s) only if a certain **condition** (the *`if _ condition`*) is met. For example, you may want to apply the command to a subset of the dataset.

- **Weights.** You can weight some units of analysis more heavily than others. We'll weight observations in many functions so our analysis of survey data is more representative of the general population.
- **Options.** You can customize commands using their **options**. Some commands, for example, have options to generate more detailed results. Options are added after a comma at the end of the command line. Most options look something like `title("hello world")`, where `title` is one of the command's options and `"hello world"` is the text you want to use for the title. The option's values go inside parentheses after the option's keyword.
- **Suboptions.** Some command options have their own **suboptions**. For example, if a command has a `title` option, the `title` option may have suboptions that allow you to customize title properties like size, color, font, and so forth. Suboptions go inside the parentheses for the option they modify. To enter a suboption, you add a comma after the option's value and then enter the suboption.

To make this discussion of command syntax more concrete, examine the following Stata command and the graphic it generates (Figure 1.2). (You need to load and use the NES Dataset to run this command yourself.)

```
graph box ft_scotus if age < 40 [aweight=wt],                          ///
  box(1, color(blue))                                                  ///
  title("SCOTUS Ratings by Americans Under 40", size(large))           ///
  note("Source: 2020 National Election Study")
```

FIGURE 1.2 ■ Example of a Stata Command with an if Statement, Weights, and Options



This command applies Stata's `graph box` function to the NES Dataset's ft_scotus variable. The most basic expression of this command is graph box ft_scotus, which will produce a box plot with default settings. This `graph box` command is supplemented by an `if` statement, weights, and three options.

The `graph box` command applies only `if age < 40`. This if condition is met only when the value of the NES Dataset's age variable is less than 40; in other words, it will only show the Supreme Court ratings by respondents younger than age 40 years.

The statement `[aweight=wt]` causes Stata to **weight observations** based on the values of a variable named wt (which is included in the NES Dataset for this purpose). We'll discuss weighting observations more in Chapter 2.

Notice that there's a comma at the end of the first line after the command, variable list, if statement, and weight variable. All the command options follow the comma. We would omit the comma if we were calling `graph box` without any options, but we need the comma when we use command options. Also notice that we added `///` to the end of each line. The triple forward slash tells Stata that a command continues on the next line. The command, if statement, weights, and options go together and are run at the same time, but it's easier to write and read them over multiple lines.

We use three of the graph box command's options: `box`, `title`, and `note`. The `box` option allows us to style the box on the box plot. This `box` option applies to the first box (which happens to be the only one) and uses the `box` option's `color(blue)` suboption to make the box blue. We set the box plot's `title` to "SCOTUS Ratings by Americans Under 40" and use the `title` option's `size(large)` suboption to make the title larger than a default title. Notice that when we use a suboption, the option closes with two closing parentheses, `))`, one to close the suboption and another to close the option. We add a note about the data source with the `note` option.

If you look closely at Stata's GUI, you'll see that the tab for `if` statements also lets you to create `in` statements. An `in` statement is used when you want to apply a command to a limited number of observations, such as the first 5 or 10 observations only. They are not used as often as `if` statements, so we won't discuss `in` statements in detail, but we'll use them in Section 6.1.3.

When you're doing political analysis with Stata, we encourage you to take one step at a time and start by executing the most basic statement possible. In the earlier example, the most basic statement is `graph box ft _ scotus`. The basic statement will give you a general idea of what a function does. After you successfully execute the basic command, refine it by adding optional statements. Remember, you can recall earlier commands by clicking on them in the Review Window (or better yet by writing them in a Do-file, discussed in Section 1.3). Working with a program like Stata is usually an iterative process with a lot of trial and error. This is especially true when you're writing commands to display data graphically because the options literally aren't just black and white.

1.2 USING STATA'S GRAPHICAL USER INTERFACE EFFECTIVELY

Stata's pull-down menus can help you get started with a new, unfamiliar command and see what options are available (see Figure 1.3). When we introduce a new command in this book, we'll show where it can be found in Stata's graphical user interface (GUI). Stata's GUI can be an alternate way of executing commands. The GUI command dialogs are also a great way to explore a command's options and learn how to use it better. If, for example, you want to explore the options and suboptions for Stata's `graph box` command, select Graphics ► Box plot and explore the GUI fields. This can be a great way to learn how to use a command. When you submit a command using the GUI, Stata will print the command-line equivalent.

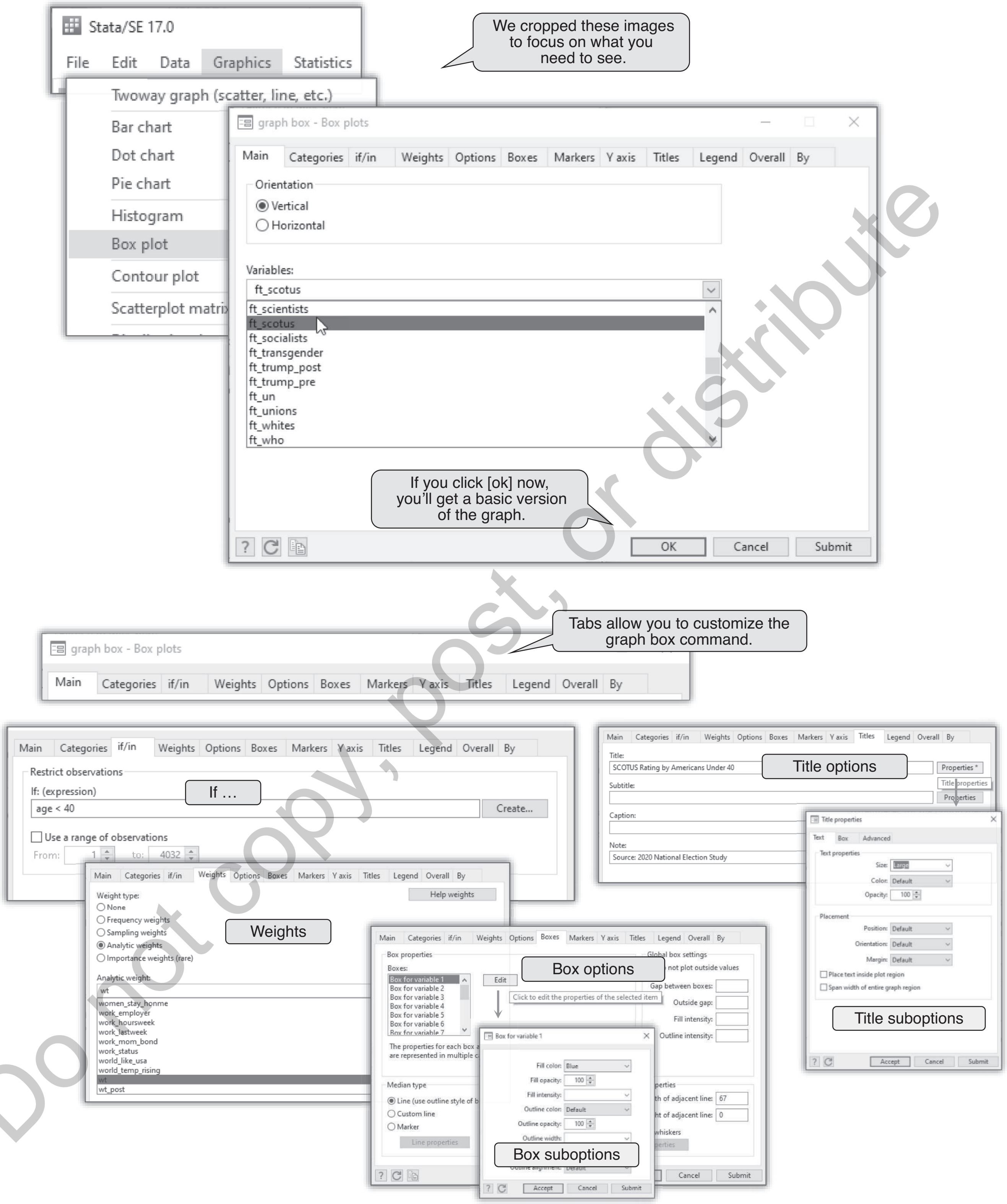
To reproduce the result more quickly in the future, copy and paste the command into a Do-file. What's a Do-file? Read on.

1.3 DO-FILES

When you are after quick information about a dataset or variables or when you are running preliminary analyses of interesting variables, the Command line will serve you well. However, when you become immersed in a line of analysis—or (especially) when you are building command-intensive, presentation-quality graphs—you will want to create Do-files.

A **Do-file** is a simple text file that lists command statements used to perform some task. You can open Stata's Do-file editor by clicking the icon on the menu bar (see Figure 1.4). The Stata Do-file editor is a primitive word processor, reminiscent of Microsoft Word's Notepad/WordPad applications.

FIGURE 1.3 ■ Executing a Command with the Graphical User Interface



Orientation
[✓ Vertical] Horizontal

Variables: ft_scotus

Restrict observations
If: (expression) age < 40
Use a range of observations From: 1 to: 4032

Title: SCOTUS Rating by Americans Under 40
Subtitle:
Caption:
Note: Source: 2020 National Election Study

Title properties — Text properties
Size: Large
Color: Default
Opacity: 100
Placement
Position: Default
Orientation: Default
Margin: Default
Place text inside plot region
Span width of entire graph region

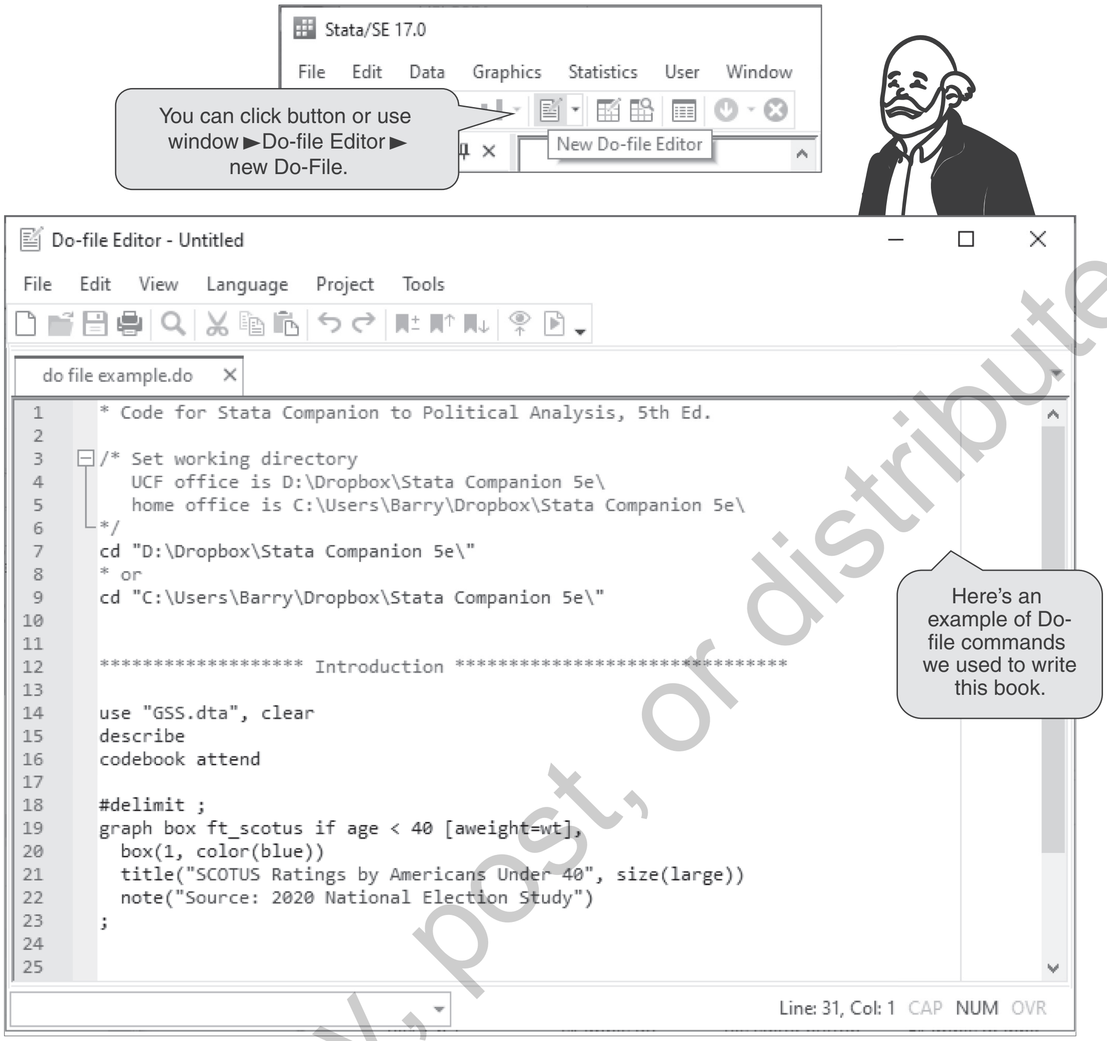
Weight type: None, Frequency weights, Sampling weights, [✓ Analytic weights], Importance weights (rare)
Analytic weight: wt

Box properties
The properties for each box are represented in multiple
Median type: [✓ Line (use outline style of b], Custom line, Marker
Global box settings
not plot outside values
Gap between boxes:
Outside gap:
Fill intensity:
Outline intensity:
th of adjacent line: 67
ht of adjacent line: 0

Box for variable 1
Fill color: Blue
Fill opacity: 100
Fill intensity:
Outline color: Default
Outline opacity: 100
Outline width:

The user types a command, marks the line(s), and clicks the Execute Selection button (see Figure 1.4). Running a command(s) from a Do-file is equivalent to entering the command(s) in the Command Window.

FIGURE 1.4 ■ The Do-file Editor


Stata/SE 17.0
File Edit Data Graphics Statistics User Window
New Do-file Editor
You can click button or use window ►Do-file Editor ► new Do-File.
Do-file Editor - Untitled
File Edit View Language Project Tools
do file example.do
Here's an example of Do-file commands we used to write this book.
Line: 31, Col: 1 CAP NUM OVR

```
* Code for Stata Companion to Political Analysis, 5th Ed.

/* Set working directory
   UCF office is D:\Dropbox\Stata Companion 5e\
   home office is C:\Users\Barry\Dropbox\Stata Companion 5e\
*/
cd "D:\Dropbox\Stata Companion 5e\"
* or
cd "C:\Users\Barry\Dropbox\Stata Companion 5e\"


******************* Introduction ******************************

use "GSS.dta", clear
describe
codebook attend

#delimit ;
graph box ft_scotus if age < 40 [aweight=wt],
  box(1, color(blue))
  title("SCOTUS Ratings by Americans Under 40", size(large))
  note("Source: 2020 National Election Study")
;
```

You can think of a Do-file as a step-by-step recipe that an analyst uses to produce results with Stata. If researchers want to replicate the analyst's results, they can run the Do-file. When should you write commands in a Do-file instead of the Command Window? If you're entering a one-off command, like accessing the help file for a command, use the Command Window. If you're trying to solve homework problems, complete problem sets, create graphs, work on independent research, or anything like this, work from a Do-file. For analysts, saving a script that makes it easy to repeat commands makes it easier to conduct similar analyses in the future.[2]

Do-files are especially helpful for writing and executing commands supplemented by multiple options. Complex commands, like complex sentences, require more words and their length creates an issue. By default, Stata expects a command to occupy only one line. The default line break is the carriage return (cr) **delimiter**. For simple commands, this default works fine. For longer commands, such as regression commands with many variables or graphing commands with many options, you will want to override the cr default with the semicolon, ;, delimiter. Once the semicolon delimiter is in place,

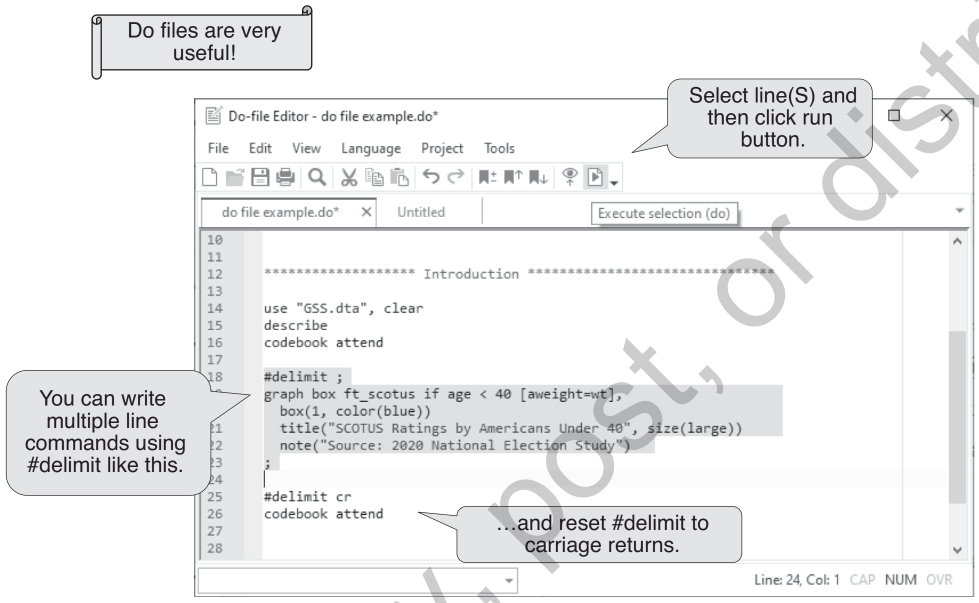
[2] If you play video games, you can think of it like saving a game while you're playing. This way, you can pick up where you left off, rather than start the game over and repeat all of the stages to get to where you were before. If you're not thrilled about writing Stata commands to solve a homework problem one time, ask yourself if you'd be happier writing those commands over and over again.

Stata will read multiple lines as a single command until it encounters a semicolon. To declare the semicolon delimiter, type the following command in the Do-file editor:

```
Do-file Editor
#delimit ;
```

Figure 1.5 illustrates how the delimiter works. If you execute one semicolon-delimited command or a series of them, Stata will expect the semicolon delimiter for all commands. When you return to the Do-file and run a carriage-return command, Stata will run that command, too, because it resets to the default between runs.[3] The `#delimit` command only works in a Do-file; it will throw an error if entered in the Command Window.

FIGURE 1.5 ■ The Semicolon Delimiter


Do files are very useful!
Select line(S) and then click run button.
Do-file Editor - do file example.do*
File Edit View Language Project Tools
do file example.do*
Untitled
Execute selection (do)
You can write multiple line commands using #delimit like this.
...and reset #delimit to carriage returns.
Line: 24, Col: 1 CAP NUM OVR

```


****************** Introduction ******************************

use "GSS.dta", clear
describe
codebook attend

#delimit ;
graph box ft_scotus if age < 40 [aweight=wt],
  box(1, color(blue))
  title("SCOTUS Ratings by Americans Under 40", size(large))
  note("Source: 2020 National Election Study")
;

#delimit cr
codebook attend
```

Another option for writing long commands in Do-files is to insert triple forward slashes, ///, at the end of the line to indicate to Stata that the command does not end but rather continues to the next line.

In addition to breaking over multiple lines, it is helpful to write one option per line and indent the command's options. See the formatting of the multiline commands in Section 1.1 as examples. Indenting suggests visually that the option lines belong to the command and aren't on the level of a command. Similarly, if an option has multiple suboptions, you can break them over multiple lines and use an additional level of indentation; this way, the suboptions will look like they belong to the option they modify and are not on the same level as the first level of options.

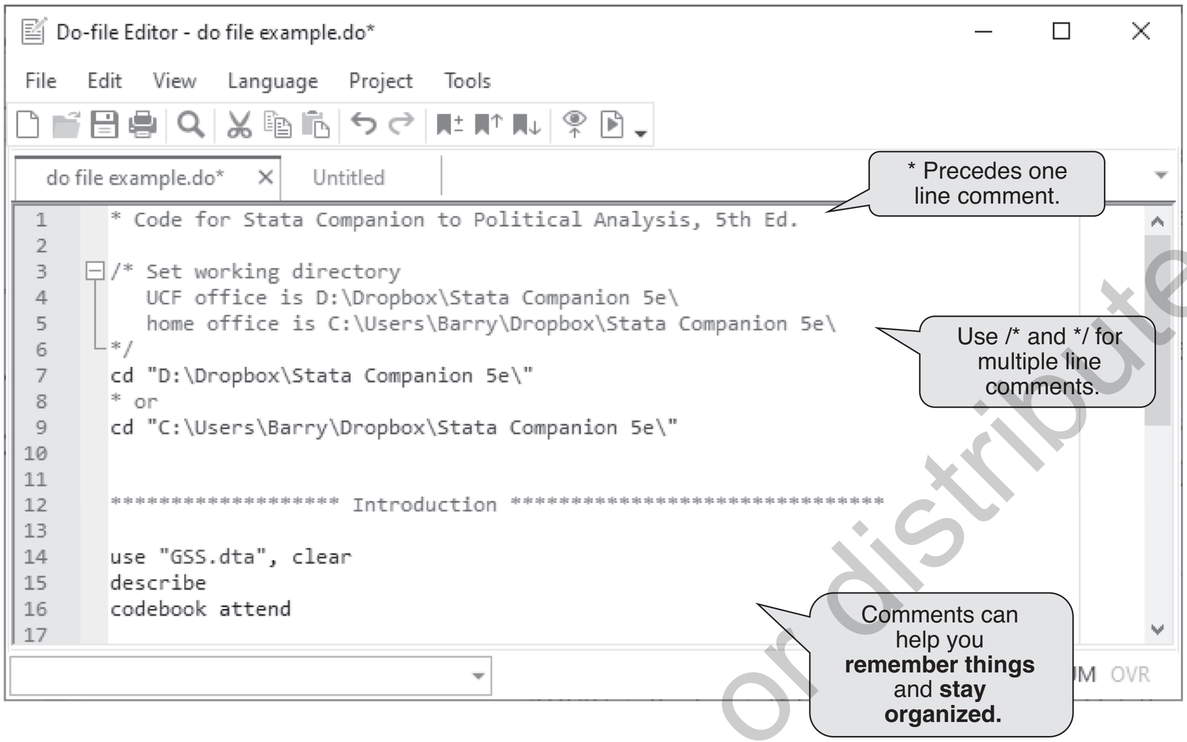
Insert short comments by starting the comment line with an asterisk, `*`. Insert longer, multiline comments by enclosing them between `/*` at the beginning and `*/` at the end, as shown in Figure 1.6.

In addition to inserting plain English comments into your Do-files to make them more user friendly, we can offer a few more suggestions to help you learn from our mistakes.

Save the Do-files you write with simple, descriptive names. Click on the Save File icon, select a location (such as your USB device), and save the Do-file, giving it a mundane but descriptive name (e.g., describe_dataset_variables.do). At the beginning of each chapter in this book, open the Do-file editor, write the syntax as described in the chapter, and save the file.

[3] If you run semicolon-delimited and carriage return–delimited commands in the same group of commands, then you must reset the carriage-return default within the group by using the command `#delimit cr`.

FIGURE 1.6 ■ Making Comments in a Do-file


Do-file Editor - do file example.do*
File Edit View Language Project Tools
do file example.do*
Untitled
* Precedes one line comment.
Use /* and */ for multiple line comments.
Comments can help you **remember things** and **stay organized.**
NUM OVR

```
* Code for Stata Companion to Political Analysis, 5th Ed.

/* Set working directory
   UCF office is D:\Dropbox\Stata Companion 5e\
   home office is C:\Users\Barry\Dropbox\Stata Companion 5e\
*/
cd "D:\Dropbox\Stata Companion 5e\"
* or
cd "C:\Users\Barry\Dropbox\Stata Companion 5e\"


******************* Introduction *******************************

use "GSS.dta", clear
describe
codebook attend
```

You'll frequently execute commands with quotation marks, parentheses, and brackets. When you insert quotation marks, parentheses, or brackets in a command, they need to come in matching pairs. For every opening `"`, `(`, or `[`, there must be a closing `"`, `)`, or `]`. It's easy to forget the closing mark, so get in the habit of typing matching pairs of quotation marks, parentheses, or brackets and then moving the cursor back to fill in whatever goes between them. The Do-file editor offers some visual assistance, but it's a good habit to start with paired sets.

Beware of curly quotation marks. If you copy a line of code from a word-processing document or web page, it might use curly quotes, like "hello world," rather than straight quotes, as in `"hello world"`. You can't type curly quotes on Stata's command line or in a Do-file but you can copy and paste them into Stata, which creates a nasty, hard-to-identify bug.

1.4 PRINTING RESULTS AND COPYING OUTPUT

Any tabular or text output can be selected and printed directly from the screen or from a log file. To illustrate, we will use the `tabulate` command to obtain a frequency distribution of government regime types around the world. (The `tabulate` command is detailed in Chapter 2; don't worry about exactly what it's doing yet.)

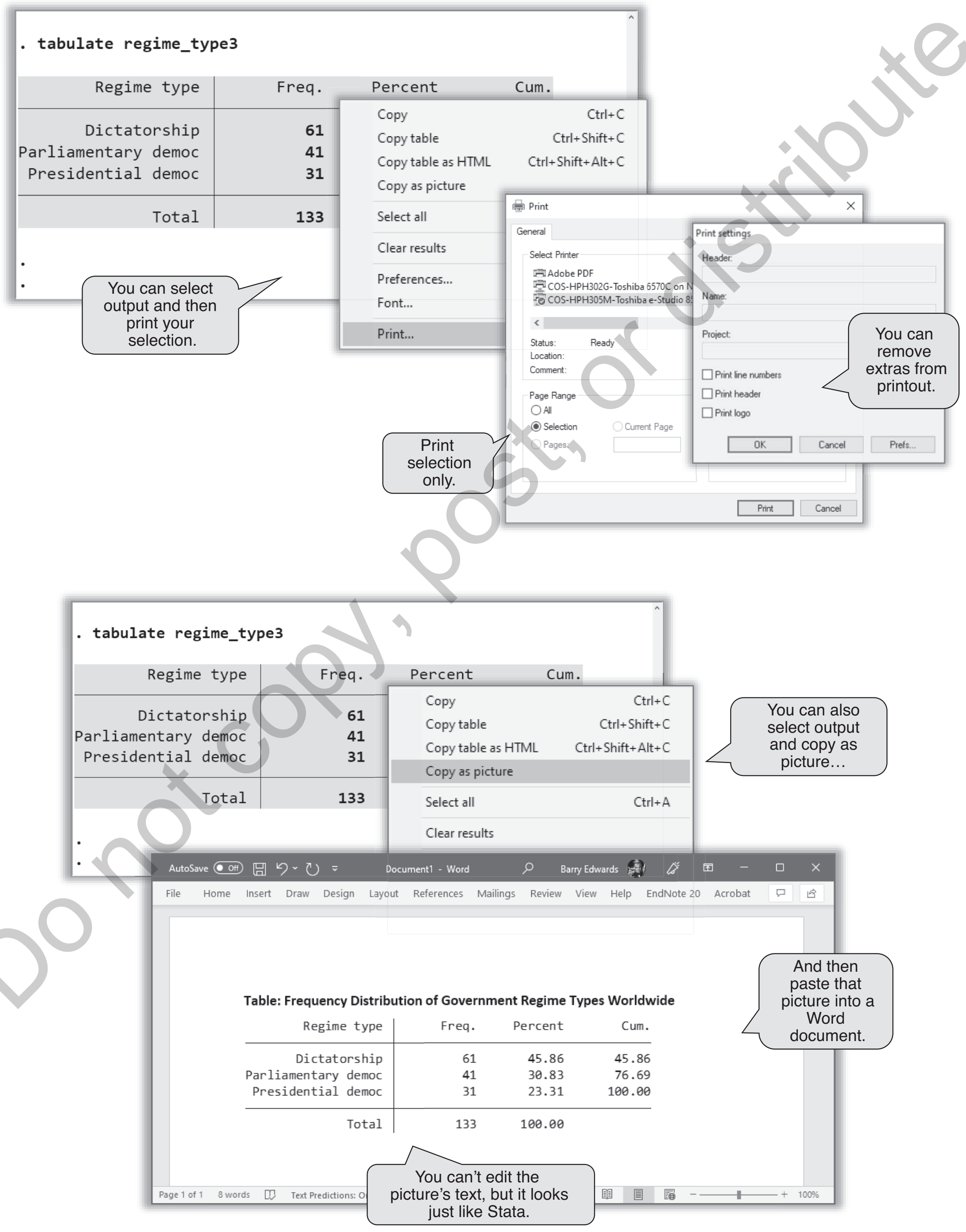
Open the World Dataset (World.dta) and run the following command. You can also execute a `tabulate` command using Stata's GUI by selecting Statistics ► Summaries, tables, and tests ► Frequency tables ► One-way table and then choosing regime_type3 as the categorical variable.

```
* use "world.dta"
tabulate regime_type3
```

```
         Regime type |      Freq.     Percent        Cum.
---------------------+-----------------------------------
         Dictatorship |         61       45.86       45.86
Parliamentary democ |         41       30.83       76.69
  Presidential democ |         31       23.31      100.00
---------------------+-----------------------------------
               Total |        133      100.00
```

Stata produces the desired frequency distribution table. To print the **table**, select it, right-click on the selection, and then click Print. In the Print Window, click the Selection radio button. (The All button is Stata's default assumption, but you rarely want to print all the output you have produced during a session.) When the Output Settings Window appears, uncheck all the boxes along the top of the window (see Figure 1.7).

FIGURE 1.7 ■ Printing Results



```
. tabulate regime_type3

      Regime type |      Freq.     Percent        Cum.
------------------+-----------------------------------
      Dictatorship |         61
Parliamentary democ |         41
 Presidential democ |         31
------------------+-----------------------------------
            Total |        133
```

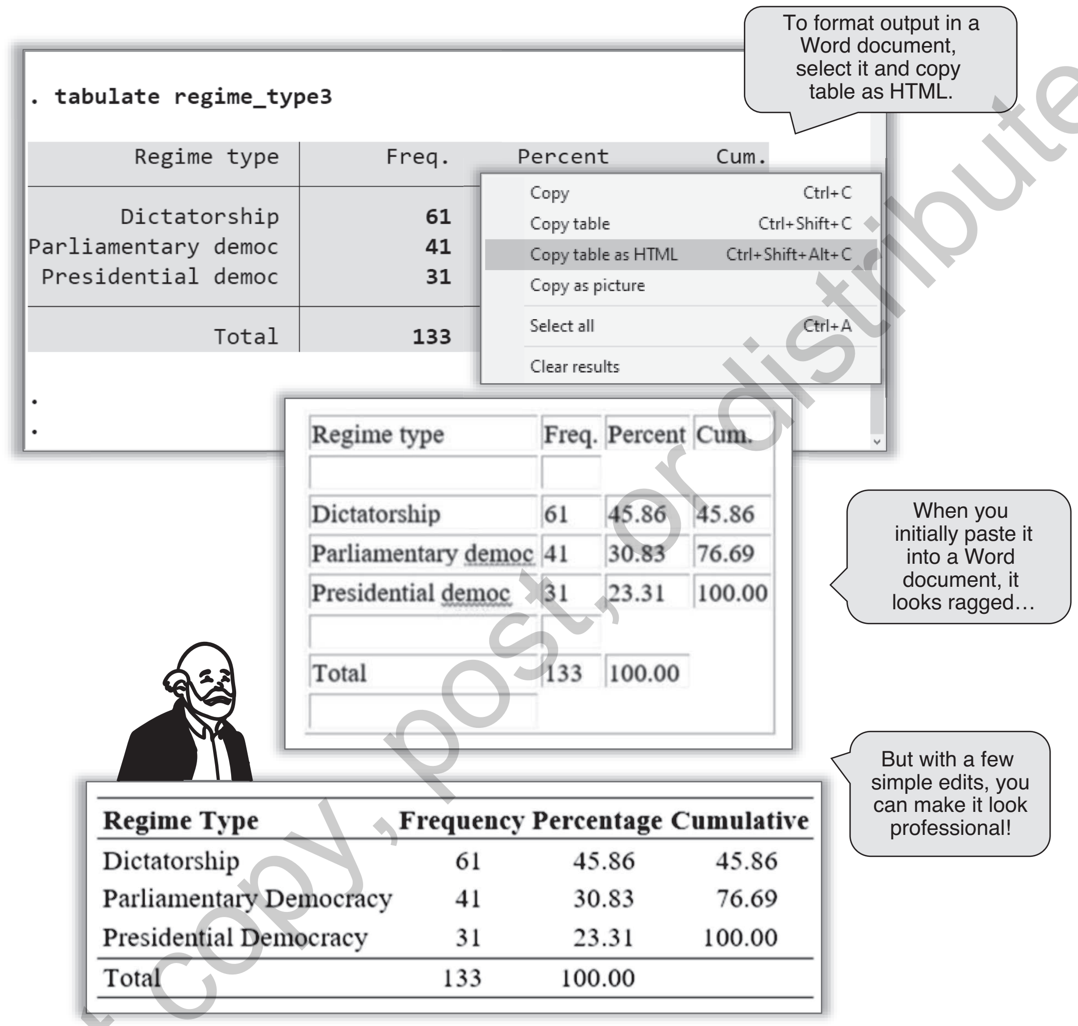
Table: Frequency Distribution of Government Regime Types Worldwide

```
        Regime type |      Freq.     Percent        Cum.
--------------------+-----------------------------------
       Dictatorship |         61       45.86       45.86
Parliamentary democ |         41       30.83       76.69
 Presidential democ |         31       23.31      100.00
--------------------+-----------------------------------
              Total |        133      100.00
```

Copy Ctrl+C
Copy table Ctrl+Shift+C
Copy table as HTML Ctrl+Shift+Alt+C
Copy as picture
Select all Ctrl+A
Clear results
Preferences...
Font...
Print...
Print
General
Select Printer
Adobe PDF
COS-HPH302G-Toshiba 6570C on N
COS-HPH305M-Toshiba e-Studio 85
Status: Ready
Location:
Comment:
Page Range
All
Selection
Current Page
Pages:
Print
Cancel
Print settings
Header:
Name:
Project:
Print line numbers
Print header
Print logo
OK
Cancel
Prefs...
You can select output and then print your selection.
Print selection only.
You can remove extras from printout.
You can also select output and copy as picture…
AutoSave Off
Document1 - Word
Barry Edwards
File Home Insert Draw Design Layout References Mailings Review View Help EndNote 20 Acrobat
Page 1 of 1 8 words Text Predictions: On 100%
And then paste that picture into a Word document.
You can't edit the picture's text, but it looks just like Stata.

You can also copy and paste output into a word processor, but (truth be told) this is one of Stata's less satisfactory features. To obtain the most workable result, select the tabular output, right-click, and choose Copy Table as HTML, as shown in Figure 1.8. When pasted into a Word document, the table is editable but not of presentation quality. A bit of editing, however, produces a presentable result.

FIGURE 1.8 ■ Copying and Pasting Results into Word


To format output in a Word document, select it and copy table as HTML.
Copy Ctrl+C
Copy table Ctrl+Shift+C
Copy table as HTML Ctrl+Shift+Alt+C
Copy as picture
Select all Ctrl+A
Clear results
When you initially paste it into a Word document, it looks ragged…
But with a few simple edits, you can make it look professional!

```
. tabulate regime_type3

          Regime type        Freq.     Percent        Cum.

         Dictatorship           61
 Parliamentary democ            41
  Presidential democ            31

                Total          133

.
.
```

Regime type	Freq.	Percent	Cum.
Dictatorship	61	45.86	45.86
Parliamentary democ	41	30.83	76.69
Presidential democ	31	23.31	100.00
Total	133	100.00	

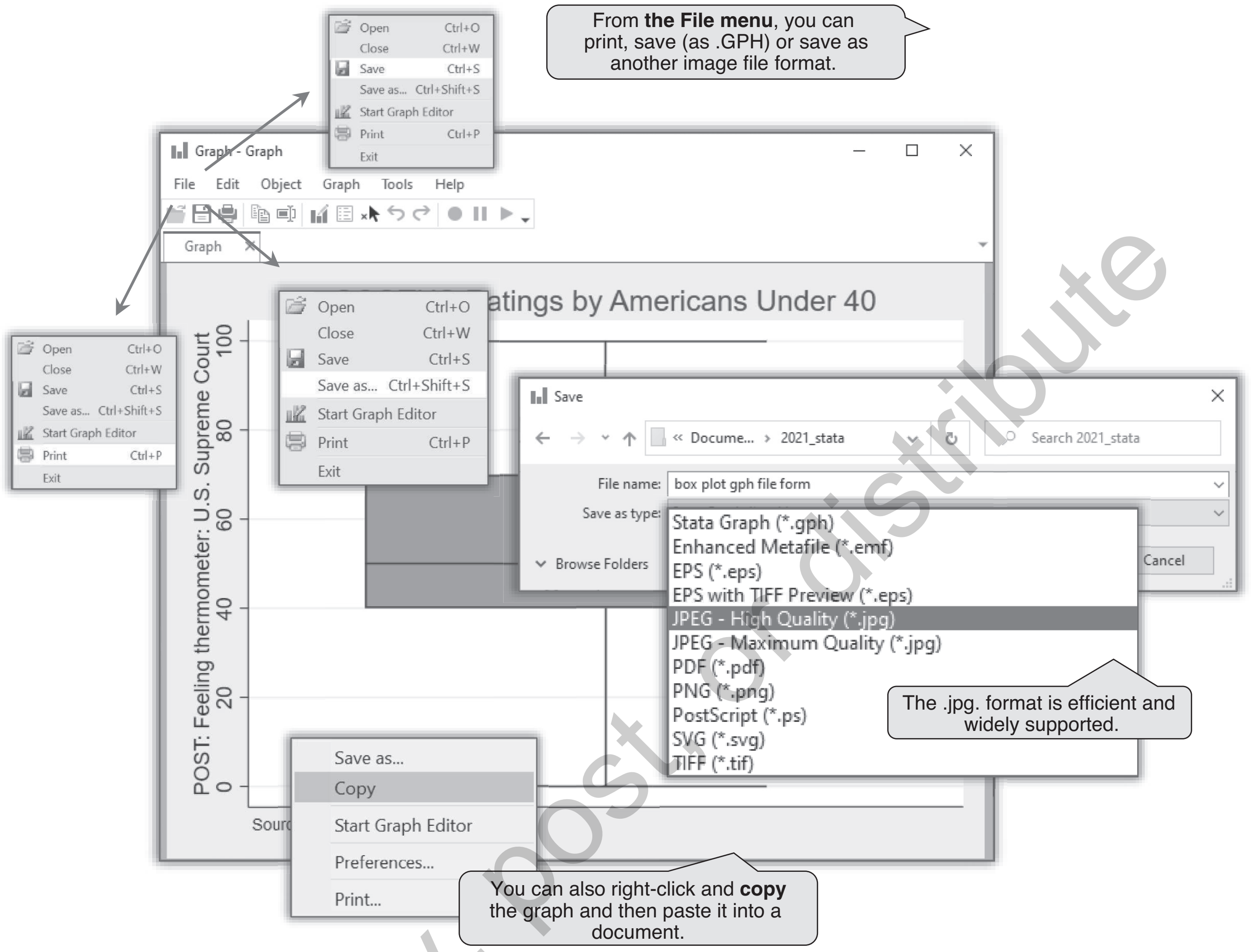
Regime Type	Frequency	Percentage	Cumulative
Dictatorship	61	45.86	45.86
Parliamentary Democracy	41	30.83	76.69
Presidential Democracy	31	23.31	100.00
Total	133	100.00	

Alternatively, if your results need to be clear but not necessarily presentation quality, you can highlight the relevant results, right-click, and choose Copy Table or Copy as Picture (see Figure 1.8; these options are available as an alternative to Print). If you copy tabular output and paste to a Word processor, be sure to change the Stata output to a monospace font (like Courier or Lucida Console) that gives each character the same amount of width. It is likely that your word processor defaults to a proportional font that will throw off Stata's column alignment. Copying the results as a picture will give a nice, crisp image but makes further editing difficult.

What about printing and downloading the **graphs** you create with Stata? Many of this book's end-of-chapter exercises ask you to create and submit graphs, so these are important skills for you to learn. There are a few ways to perform these tasks. Figure 1.9 shows the box plot we made earlier in Stata's Graph Window. You can print the graph from this window by selecting File ► Print, clicking the printer icon button, or right-clicking the graph and selecting the "Print…" option.

To save a graph as an image file on your computer, you can select File ► Save to save the graph in Stata's native image file format: *.gph (see Figure 1.9). The .gph format is fine if you want to save a graph in progress and reopen it later for further editing in Stata, but this file format

FIGURE 1.9 ■ Printing, Saving, and Copying Graphs


From **the File menu**, you can print, save (as .GPH) or save as another image file format.
Graph - Graph
File Edit Object Graph Tools Help
Open Ctrl+O
Close Ctrl+W
Save Ctrl+S
Save as... Ctrl+Shift+S
Start Graph Editor
Print Ctrl+P
Exit
Ratings by Americans Under 40
POST: Feeling thermometer: U.S. Supreme Court
Save
File name: box plot gph file form
Save as type:
Stata Graph (*.gph)
Enhanced Metafile (*.emf)
EPS (*.eps)
EPS with TIFF Preview (*.eps)
JPEG - High Quality (*.jpg)
JPEG - Maximum Quality (*.jpg)
PDF (*.pdf)
PNG (*.png)
PostScript (*.ps)
SVG (*.svg)
TIFF (*.tif)
Browse Folders
Cancel
The .jpg. format is efficient and widely supported.
Save as...
Copy
Start Graph Editor
Preferences...
Print...
You can also right-click and **copy** the graph and then paste it into a document.

is not supported by other programs (i.e., you can't insert a .gph image into a Word document or PowerPoint slide).

To save a graph as an image file you can use in other programs, you can select File ► Save as… and choose a widely supported image file format such as *.jpg, *.png, or *.tif. We recommend the .jpg format to save a graph you want to upload to a class website for an assignment. If you just want to copy the graph so you can paste it into a Word document or PowerPoint slide, simply right-click the graph and select "Copy" or select Edit ► Copy. The program you paste the graph into should automatically convert the image to a supported file type; if this doesn't work, save the graph in a widely supported image format and then insert the saved image into the other document.

1.5 CUSTOMIZING YOUR DISPLAY

By default, Stata will pause long output and display the "more" message. Perhaps you like this default. Perhaps you don't. To switch off the default—that is, to instruct Stata to scroll through all the output without pausing—execute the following command:

```
set more off
```

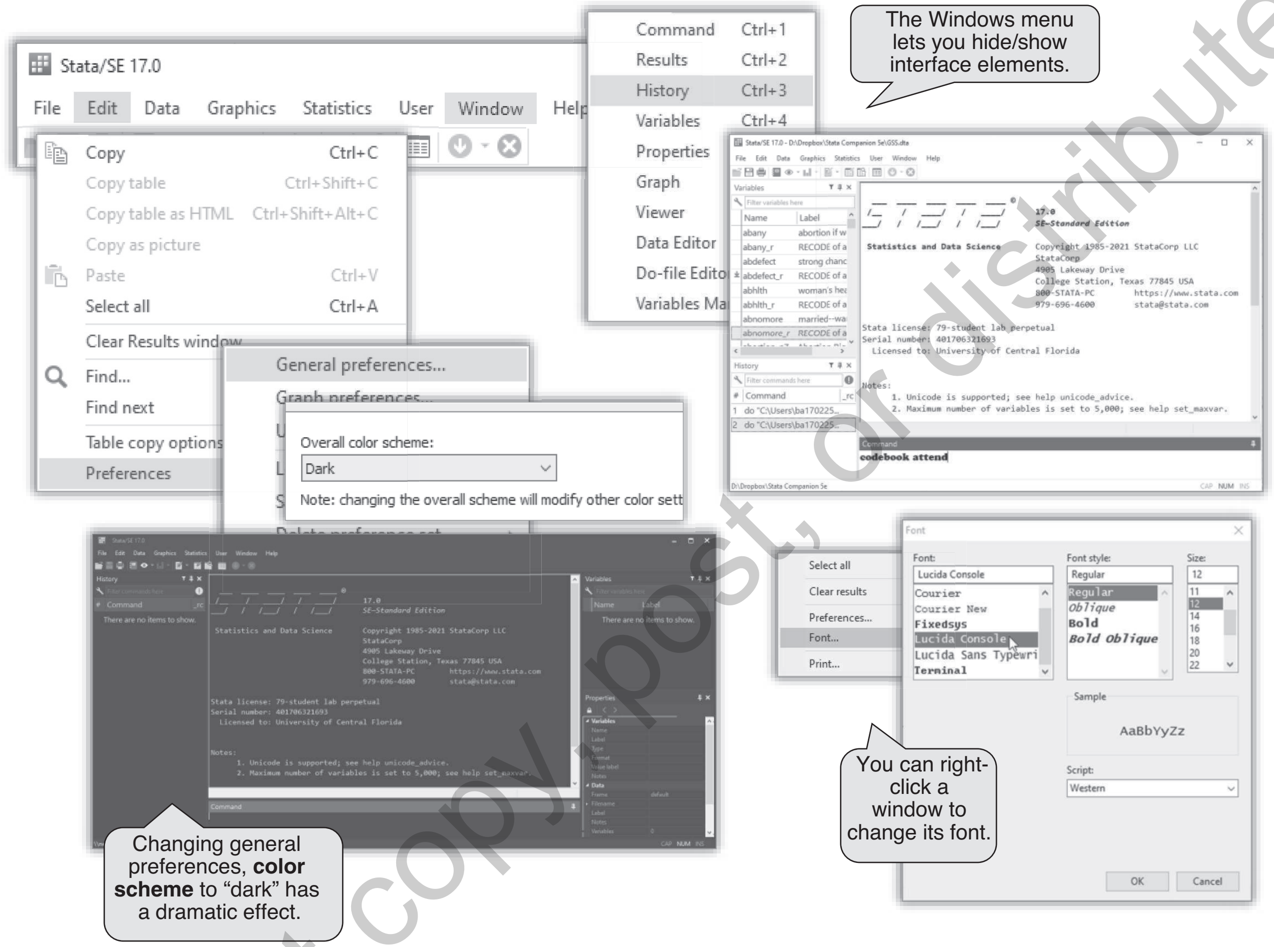
To reinstate the default, execute this command:

```
set more on
```

When you run the `set more off` and `set more on` commands, make sure to type them in the Command Window, not in a Do-file. If these commands appear in a Do-file, Stata ignores them.

To further customize Stata to your liking, you can toggle which windows are visible during your session by using the menu option under the heading "Window" or by simply pressing the X icon on windows you want to hide from view. Additionally, as shown in Figure 1.10, you can right-click Stata windows for a dialog box to change the default font size and style. If you right-click the Results Window, you can choose from a variety of tasteful color schemes and even create your own custom color scheme.

FIGURE 1.10 ■ Customizing Stata's Display



Overall color scheme: Dark

Note: changing the overall scheme will modify other color sett

```
17.0
SE-Standard Edition

Statistics and Data Science       Copyright 1985-2021 StataCorp LLC
                                  StataCorp
                                  4905 Lakeway Drive
                                  College Station, Texas 77845 USA
                                  800-STATA-PC        https://www.stata.com
                                  979-696-4600        stata@stata.com

Stata license: 79-student lab perpetual
Serial number: 401706321693
  Licensed to: University of Central Florida

Notes:
      1. Unicode is supported; see help unicode_advice.
      2. Maximum number of variables is set to 5,000; see help set_maxvar.
```

Command: codebook attend

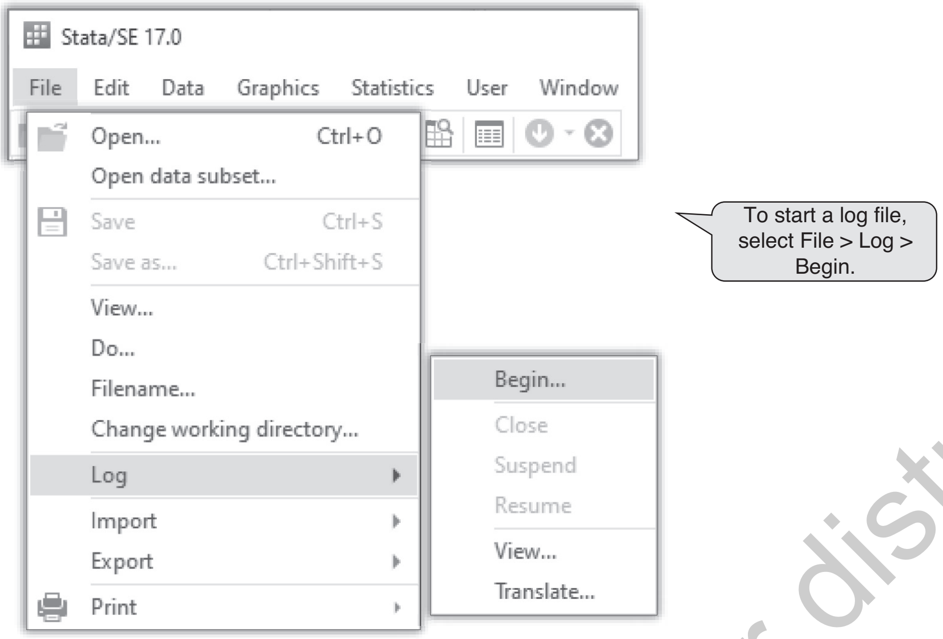
Font: Lucida Console (Courier, Courier New, Fixedsys, Lucida Console, Lucida Sans Typewriter, Terminal)
Font style: Regular (Regular, Oblique, Bold, Bold Oblique)
Size: 12 (11, 12, 14, 16, 18, 20, 22)
Sample: AaBbYyZz
Script: Western
OK Cancel

1.6 LOG FILES

Unlike most data analysis packages, Stata does not automatically save results to an output file. This is not a huge disadvantage, because Do-files permit you to replicate your work. You can also select the desired output and print it directly from the Results Window. Even so, you may want to keep a **log file**.

To begin a log file, click File ► Log ► Begin, as shown in Figure 1.11. Find a suitable location, name the file, and click Save. Stata will now record everything, including commands that are typed in the Command Window and Do-file syntax that sends output to the Results Window. From within Stata, you can open and view log files, including the current log, by clicking File ► Log ► View. Ordinarily, Stata correctly assumes that you want to view the active log file. If Stata is remiss, you can browse to the current log's location. Of course, by using the File ► Log menu, log files can be suspended, resumed, and closed.

FIGURE 1.11 ■ Beginning a Log File



1.7 GETTING HELP

To view the formal how-to manual for any Stata command, use the aptly named `help` command. The syntax of the help command is as follows:

help ***command_ name***

For example, if you want detailed instructions on the `codebook` command you used earlier in this chapter to see the **Stata Manual** entries about a variable, you would execute `help codebook`. The command `help codebook` retrieves specific syntax information; for example, the technical manual for the command will list all of the `codebook` command's options, summarize what the options do, and often provide examples of the command. Stata displays the requested information in a separate window, the Viewer (Figure 1.12). You can also access help files using Stata's GUI by selecting Help ► Stata Command... Once you get comfortable with the basic syntax of commands, help files can help you master them.

help codebook

In addition to a raft of information on options and useful links to command elements, Stata communicates—by way of underlined letters—the minimum acceptable command abbreviation. As it happens, there is no abbreviation for the `codebook` command, but many of its options can be abbreviated. If none of the letters in the command syntax are underlined, then the command cannot be abbreviated.

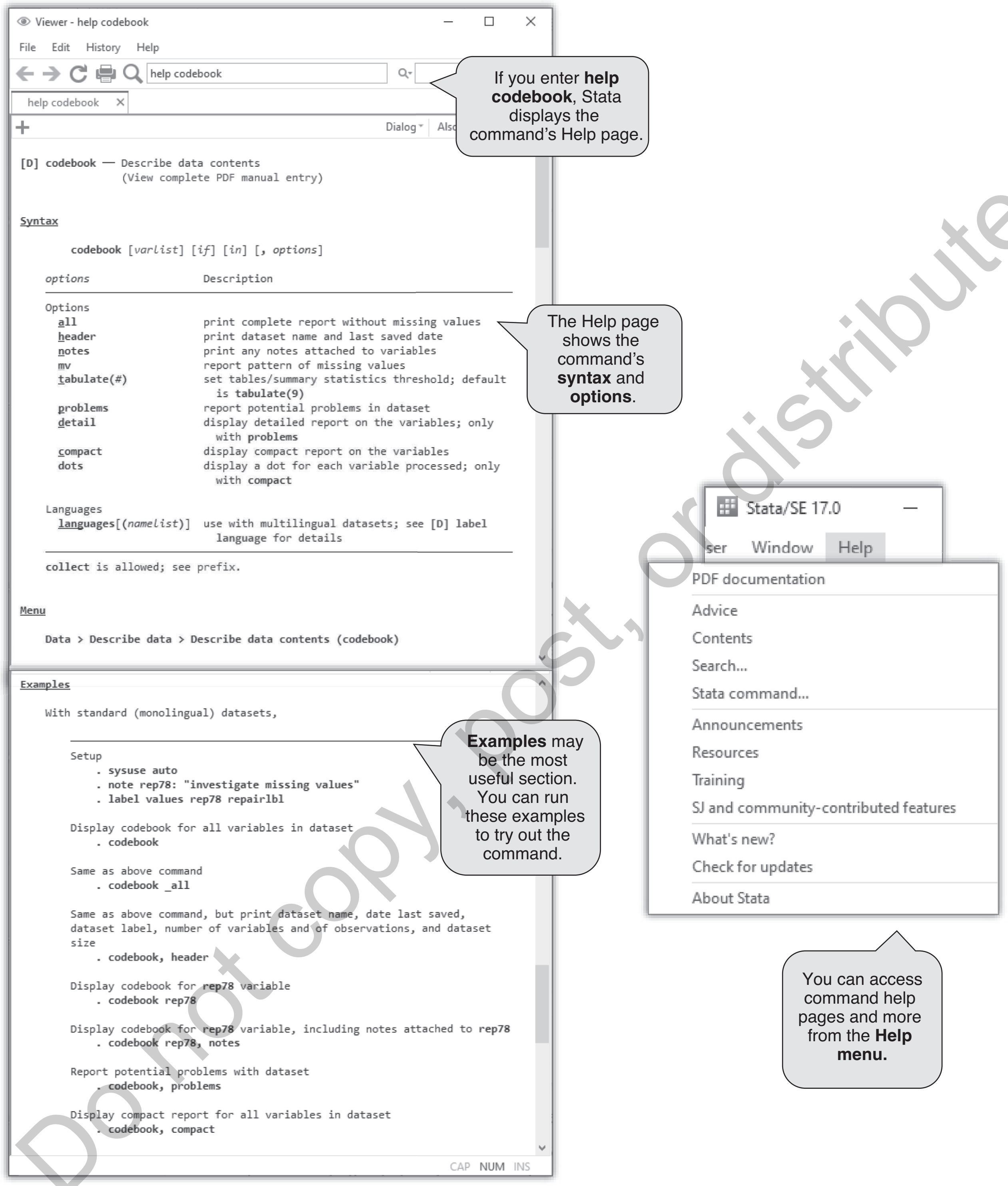
The more commonly used commands, such as `describe`, generally have shorter acceptable abbreviations. Commands that are destructive of data may not be abbreviated. The `recode` command, discussed in Chapter 3, is an example of a destructive command.

If you don't know or can't remember the name of a Stata command or if you want to delve through Stata's resources on a particular topic, use the `search` command. You can execute it from the command line or by selecting Help ► Search... The syntax of the `search` command is blissfully simple:

search ***keyword***

Suppose we are interested in doing probit analysis, a specialized modeling technique similar to logistic regression analysis (covered in Chapter 14), and we want access to Stata's information

FIGURE 1.12 ■ The help Command


Viewer - help codebook
File Edit History Help
help codebook
Dialog Also
If you enter **help codebook**, Stata displays the command's Help page.
The Help page shows the command's **syntax** and **options**.
Examples may be the most useful section. You can run these examples to try out the command.
Stata/SE 17.0
ser Window Help
PDF documentation
Advice
Contents
Search...
Stata command...
Announcements
Resources
Training
SJ and community-contributed features
What's new?
Check for updates
About Stata
You can access command help pages and more from the **Help menu.**
CAP NUM INS

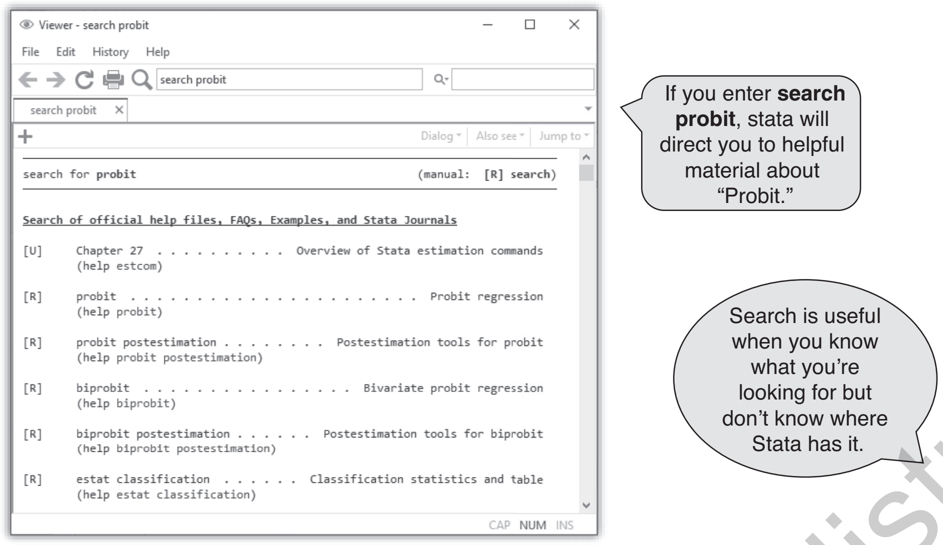
```
[D] codebook — Describe data contents
                (View complete PDF manual entry)


Syntax

        codebook [varlist] [if] [in] [, options]

    options                    Description
    ------------------------------------------------------------------------
    Options
      all                      print complete report without missing values
      header                   print dataset name and last saved date
      notes                    print any notes attached to variables
      mv                       report pattern of missing values
      tabulate(#)              set tables/summary statistics threshold; default
                                 is tabulate(9)
      problems                 report potential problems in dataset
      detail                   display detailed report on the variables; only
                                 with problems
      compact                  display compact report on the variables
      dots                     display a dot for each variable processed; only
                                 with compact

    Languages
      languages[(namelist)]    use with multilingual datasets; see [D] label
                                 language for details
    ------------------------------------------------------------------------
    collect is allowed; see prefix.


Menu

    Data > Describe data > Describe data contents (codebook)

Examples

    With standard (monolingual) datasets,

        ---------------------------------------------------------------
        Setup
            . sysuse auto
            . note rep78: "investigate missing values"
            . label values rep78 repairlbl

        Display codebook for all variables in dataset
            . codebook

        Same as above command
            . codebook _all

        Same as above command, but print dataset name, date last saved,
        dataset label, number of variables and of observations, and dataset
        size
            . codebook, header

        Display codebook for rep78 variable
            . codebook rep78

        Display codebook for rep78 variable, including notes attached to rep78
            . codebook rep78, notes

        Report potential problems with dataset
            . codebook, problems

        Display compact report for all variables in dataset
            . codebook, compact
```

on the topic. Entering the command `search probit` returns many pages of output (see Figure 1.13), including hyperlinks to descriptions of every probit-related Stata command, links to frequently asked questions (FAQs), examples of probit analysis, and references to technical reports and bulletins.[4]

```
search probit
```

[4]Another help-related command, `findit`, casts an even wider net than the `search` command. The `findit` command is equivalent to running a `search` command with the `all` option.

FIGURE 1.13 ■ Results of the search Command


Viewer - search probit
File Edit History Help
search probit
Dialog Also see Jump to
CAP NUM INS
If you enter **search probit**, stata will direct you to helpful material about “Probit.”
Search is useful when you know what you’re looking for but don’t know where Stata has it.

```
search for probit                                    (manual:  [R] search)

Search of official help files, FAQs, Examples, and Stata Journals

[U]     Chapter 27 . . . . . . . . . .  Overview of Stata estimation commands
        (help estcom)

[R]     probit . . . . . . . . . . . . . . . . . . . . .  Probit regression
        (help probit)

[R]     probit postestimation . . . . . . . .  Postestimation tools for probit
        (help probit postestimation)

[R]     biprobit . . . . . . . . . . . . . . . .  Bivariate probit regression
        (help biprobit)

[R]     biprobit postestimation . . . . . .  Postestimation tools for biprobit
        (help biprobit postestimation)

[R]     estat classification . . . . . .  Classification statistics and table
        (help estat classification)
```

KEY TERMS

Commands	Log file
Conditions	Options
Delimiter	Stata Manual
Do-files	Suboptions
Generic usage statement	Table
Graphs	Weight observations

CHAPTER 1 EXERCISES

1. From Stata’s user interface, select File ► Open, locate NES.dta on your computer, select it, and click Open. Run the `describe` command. According to the results, the NES Dataset has (fill in the blanks) ___________________observations and ____________ variables.

2. The NES Dataset contains a variable named terrorism_worry. The variable records NES respondents’ answers when researchers asked how worried they are about a terrorist attack in the near future. With the NES Dataset open, run `codebook terrorism _ worry`.
   A. Respondents who are extremely worried about a terrorist attack in the near future have what numeric code on terrorism_worry? (select one)
   1 2 3 4 5
   B. According to the `codebook` results, how many respondents have missing values on terrorism_worry because they didn’t have a postelection interview, refused to answer, or had some other reason for not responding? ________________________

3. Obtain tabular output for the NES variable leader_compromise by running `tabulate leader _ compromise [aw=wt]`. Using one of the methods discussed in Chapter 1, submit the table.

4. In Chapter 2, you will learn how to interpret output from the `summarize` command. Which of the following are acceptable abbreviations for the `summarize` command? (select all that apply)

   su sum sum summa summar summary summariz

5. Stata maintains a list of commands you have executed from the command line in the History Window. Sometimes it's useful to recall a command you previously executed and run it again (possibly with some minor changes). How do you get a prior command listed in the History Window to appear in the Command Window?

Additional exercises on fillable PDF forms are available to instructors online at edge.sagepub.com/pollock