Using Stata with *The Fundamentals of Political Science Research*

Paul M. Kellstedt and Guy D. Whitten
Department of Political Science
Texas A&M University

© Paul M. Kellstedt and Guy D. Whitten 2009

Contents

1 Overview

In this document we provide a guide for how to produce the statistics and graphs for our book using the Stata program. This guide is not intended as a comprehensive introduction to Stata. There are numerous books intended to serve this purpose.[1]

The ordering of topics in this appendix is intended to follow closely the ordering of topics in the book. After a couple of brief discussions of housekeeping issues (keeping track of your work and getting data into the program), we present the statistical and graphical tools in roughly the order in which they occur in the text. Throughout this guide we refer to specific sections of our book *The Fundamentals of Political Science Research (FPSR).*

[1] For an up-to-date listing of such books we recommend going to Stata's website at www.stata.com

1.1 Getting started—launching Stata

Once you are sitting in front to a computer on which Stata has been properly installed, you can launch the program by double-clicking on the Stata icon or by finding the Stata program on your start menu. We suggest launching the Stata program and working through this appendix in front of a computer with a copy of *FPSR*.

Once you have successfully launched the Stata program, you will normally see one large window with a row of textual headings (starting with "File," then "Edit" etc) at across the top followed by a row of icons (a picture of an envelop openning, a picture of a floppy disc, etc). Within the main Stata window, you will see three other windows labeled "Review," "Results," and "Command." If you are seeing all of this, you are ready to go.

1.2 Which version?

This document has been written using version 9.2 of the Stata program on a Windows-based computer. Most of these commands will work with different versions of the Stata program and on different platforms. The main exceptions to this involve graphics commands. Stataś graphic commands changed substantially between versions 7 and 8. Thus, most of the graphics commands that we cover in this appendix will not work in versions earlier than 8.0. All of the commands covered in this appendix should work in later versions of StataIf you are working with a later version that 9.2 and are having problems, type "version 9.2" in the command line and then hit the "Enter" key. This will revert the program back to version 9.2.

1.3 Reading commands in this appendix

Throughout this appendix, when we present Stata commands in their text versions, we write out commands in typewriter type and present generic statements within those commands in italics. So, for instance, the command to "use" a data set called "test.dta" which is located in the directory "c:\mydata" will be presented as:

```
use c:\mydata\test.dta
```

The command to use a data set without a reference to a specific data set will be presented as:

`use` *filename*

In the command above, the fact that "filename" appears in italics indicates that this is a generic (as opposed to specific) reference and therefor when you write this command, you must choose the appropriate text to replace what is in italics.

Another important part of Stata syntax is the various options that come with different commands (different commands have different options available with them). In the text versions of commands, options appear to the right of a comma after the main command syntax. So, for instance, if we see the following command:

`use` *filename*, `clear`

we know that Stata will use whatever filename we have specified and that the "clear" option has been chosen. In this case, the selected option tells Stata to clear its data area before opening the specified data set.

2 Keeping track of your work

In almost any mainstream statistical program today, there are multiple ways to accomplish the same tasks. In Stata , almost any command can be executed using pull-down menus, typed commands in the command window, or typed commands in a do-file window. The choice of which of these options to use is a matter of personal comfort. But regardless of which way you find that you like to issue commands, it is critical that you keep track of your work. Any time that you work with data, you should have the following questions in mind:

1. If I put these data and my work with them aside for a month, would I be able to go back and *exactly* recreate my work?
2. If I put these data and my work with them aside, would *someone else* be able to go back and *exactly* recreate my work?

If the answer to both questions is "yes," then you have done a good job keeping track of your work.

2.1 Replication is high flattery

All of this emphasis on keeping track of your work may seem tedious, especially when you consider the second question. But keep in mind the crucial role that our statistical hypothesis testing plays on the road to scientific knowledge (see Chapter 1). When a researcher comes

up with a hypothesis test that supports a new theory or challenges an established theory, the first thing that other researchers will want to do is to **replicate** their work. Initial replication of another researcher's work may involve either recreating the same results with the original data or the collection of new data and attempting to recreate the same results. Either way, it is crucial that the researcher who is replicating someone else's work be able to repeat what they did.

Replication may seem like a distrustful exercise but, keep in mind that one of our rules of the road (again, from Chapter 1) is to "consider only empirical evidence." Replication is a very serious consideration of another researcher's empirical evidence. Boring results seldom get replicated. Thus, rather than being distrustful or insulting, replication is actually high flattery.

2.2 Using a .log file

An excellent way to keep track of your work in Stata is to open a log file. This is usually done at the very beginning of your work session. You can choose one from the following three different ways to open a log file:

- *Using the pull-down menu commands*: To start a log from the pull-down menus, left-click on "File," then "Log," then "Begin... ." This will prompt you to name a file and the location in which this file should be saved. When you are finished with your Stata session, you
- *From the Command window or the Do-file editor*: Type "`log using` *filename*"

2.3 Working from a .do file

If you wish to work by writing your commands and running them from a command file, you can launch a fourth window, called a "Do-file" by left-clicking on "Window," then left-clicking on "Do-file Editor," and then left-clicking on "New Do-file." Once inside a do-file, you can write commands and execute them from your do-file. Your do-file can be saved, closed, and re-accessed as a separate file. A do-file has the advantage that you can keep notes in it about what your are doing so that you have the notes and the commands that you executed all in the same place.

2.3.1 Executing commands from a do-file

In a do-file, each line is assumed to be a separate command. To execute a line, you simply select it by left-clicking on it and dragging across the text.[2] You then execute the command line by clicking on the icon third icon from the left at the top of the do-file window. This icon, which looks like a piece of paper with writing on it next to a downward pointing blue arrow, tells Stata to "execute the current commands." Be sure to click on this icon and not the icon to the right which is the equivalent of telling Stata to "execute the entire do-file."

2.3.2 Notes in a do-file

You can keep notes in a do-file. In order to make a distinction between lines that contain your notes and lines that contain your commands, you should place a star "*" in front of your lines of notes. If you accidentally run such a line, there will not be a problem because, when Stata reads a * in front of a line, it automatically ignores the rest of that line.

2.4 Keeping notes

One of the best ways to keep track of what you have done is to write about it in a separate file or in a notebook. An obvious worry about writing about what you are doing while you are doing it is that you might work more slowly. However, what you lose in working speed is likely to be more than made up for by avoiding the mistakes that writing allows you to avoid. We agree wholeheartedly with the words of Diedre McCloskey that "writing is thinking." And, in this case, slowing down and thinking about what you are doing will often save you time in the long run.

3 Getting data into the program

Not surprisingly, Stata works best with Stata data sets. Stata data sets are identifiable by the extension ".dta." If you are working with a data set that is not a Stata data set, the best way to get it into Stata is to use a data translator program such as Stat/Transfer which can move data from whatever format you are using into the Stata format. Once you have done this, you can open the data set one of three ways:

- *Using the pull-down menu commands*: To start a log from the pull-down menus, left-

[2]You will find that you actually do not need to select an entire line. Once you have selected any portion of a line, Stata assumes that you wish to run that entire line.

click on "File," and then "Open." This will prompt you to select a file or to switch folders until you find the file that you want to open.

- *From the Command window*: Type "`use` *filename*" and hit return.
- *From the Do-File Editor*: Write "`use` *filename*" and run this command line.

Stata can also read in ASCII files using the "insheet" command.

3.1 Making your own data set

If you make your own data set, we recommend that you enter your data into a spreadsheet and then convert your spreadsheet into Stata format using a data translation program. As we mentioned earlier, a popular data translation program is Stat/Transfer.

3.1.1 Naming your variables

Variable names should convey as much information about a variable as possible. They should be designed for use in such a way that they prompt your memory of what you did when you created the variable. When creating variable names in Stata you should keep in mind the following list of concerns:

- Variable names must begin with a letter. So "vote1984" is allowed but "1984vote" is not.
- Variable names may not contain spaces. For instance, "presidential_vote" is allowed, but "presidential vote" is not a valid name for a single variable.
- Stata is sensitive to whether a letter is capitalized or not. This means that "vote1984" and "Vote1984" do not refer to the same variable.
- In practice, you will find that variable names longer than 10 characters are pretty unwieldy.

3.2 Reading in someone else's electronic file

When you acquire someone else's data, the first thing that you need to do to work with it in Stata format is to convert it into one of the data formats that Stata can read. The easiest way to to do this is with a data translation program like Stat/Transfer.

3.2.1 Getting to know someone else's data

When working someone else's data, it is important to get to know each variable with which you intend to work. To get to know each variable, we recommend reading any documentation that comes with their data set. Many data sets have accompanying text files, often called "codebooks" that explain how the cases were selected and the various variables measured. A careful read through of this material will help you to know a data set and to avoid making mistakes. We also recommend that you produce descriptive statistics and/or descriptive graphs for each variable before you run any hypothesis tests with more than one variable.

It is particularly important to know what values represent missing values when you are working with someone else's data. In Stata missing values are represented as a period (.), but in some data sets a missing value will be assigned a numeric value. If you do not know about this, you may mistake a missing value for an actual value and come to faulty conclusions as a result.

4 Making changes to a data set

When you are working with a data set, you will often want to make changes before you conduct your analyses. We have already stressed the important of keeping track of your work. This is especially crucial when you are making changes to a data set. Another important part of changing a data set is to always be able to go back to the data set with which you started working. We highly recommend that you save the original data under a different name after you make changes to it and that you preserve the original data set.

4.1 Recoding variables

It is often the case that we want to recode our variables before we use them to test hypotheses. As an example, let's say that we want to test a theory about what makes someone more or less likely to vote for an incumbent presidential candidate. If we are using survey data from the 2004 National Election Study, the variable V045026, with a variable label "Voter: R's vote for President" provides a useful measure of our dependent variable. We can see from the NES codebook that the possible values for this variable are 1. John Kerry, 3. George W. Bush, 5. Ralph Nader, 7. Other, and 9. Refused. For our purposes, the we want to distinguish between those voters who voted for the incumbent (the 3s), and those who voted for someone else (the 1s, 5s, and 7s). Survey respondents who refused to say who they voted for present a bit of a dilemma but, since we do not know whether they voted for or against the incumbent, the safe thing to do is to treat them as missing for our variable. One of the best ways to conduct a recode is to create a new variable equal to the old variable and then

recode the new variable. This is particularly helpful in this example because the old variable name "V045026" did not convey very much information to us. Here are the commands that we can use in a do-file (or submit one at a time in the command window) to create the new variable that we want:

```
generate bushvote=V045026
recode bushvote 3=1 1=0 5=0 7=0 9=.
```

If we read through the lines of code, in the first line the command "generate" tells Stata that we want it to create a new variable and then we name the new variable, "bushvote," and set it to equal "V045026." On the second line of code we tell Stata that we wish to recode the variable "bushvote" and then provide a list of first the old values and then the new values to which they will be changed such that 3s become 1s, 1s become 0s, 5s become 0s, 7s become 0s and 9s become missing values. Our newly recoded variable, "bushvote" is now equal to 1 for voters who voted for Bush, 0 for voters who voted for someone other than Bush, and missing for voters who refused to tell us for whom they voted. Following from the guidelines that we gave in the section on naming variables, our new variable name should help us to remember what the values mean. If we want to, we can also create a variable label and/or value labels.

4.1.1 Checking your recodes

Any time that you recode a variable, it is important to produce a table to check that you have done what you set out to do with your recode command. We provide an example of this below in the section on tables.

4.2 Selecting cases

When we are testing our hypotheses, it will sometimes be appropriate to analyze data from only a particular subset of the total number of cases in our data set. We have seen such an example in the section above on recoding variables. Because some of the respondents to the 2004 NES refused to report who they voted for, they are not appropriate for testing a theory about voting behavior. By setting values for such cases as missing (equal to .) for the variable "bushvote," we have ensured that these cases will not be a part of any analysis involving this one variable. Another way to select cases for an analysis is to delete all cases that we do not want from our data set or to issue an order as a part of our analysis command that only certain cases be included in the analysis being conducted. We recommend the second practice and will include examples of it throughout our discussion of specific commands. If, however, you decide that you wish to drop certain cases, you can issue the following command:

```
drop if condition
```

Where the "condition" can be a reference to the values of one or more variables. So if, for example, we wanted to drop all of the voters who refused to say who they voted for from the NES data set, we could issue the following command:

```
drop if V045026==9
```

In this example, the two consecutive equal signs translates into "is exactly equal to," so this expression tells Stata to drop all cases where the variable V045026 is exactly equal to 9. In general we recommend against dropping cases like this but, if you feel strongly compelled to do so, it is very important that you follow the recommendation in the section "Changing someone else's data" of saving the changed data set under a different name so that those values cases are not permanently lost.

5 Producing descriptive statistics and graphs

Once you have your data in Stata it is important to get to know your data. In Chapter 6 we discussed a variety of tools that can be used to get to know your data one variable at a time. In this section we discuss how to produce and present such information. An important first step to getting to know your data is to figure out the what is the measurement metric for each variable. For categorical and ordinal variables, we suggest producing frequency tables. For continuous variables, there are a wide range of descriptive statistics.

5.1 Describing categorical and ordinal variables

As we discussed in Chapter 6, a frequency table is the best way to numerically examine and present the distribution of values for a categorical or ordinal variable. In Stata the "tabulate" command is used to produce frequency tables. In a do-file or from the command line, the syntax for this command is:

```
tabulate variable
```

This command produces a four-column table in which the first column contains the variable values (or value labels if there are value labels for this variable), the second column is the number of cases in the data set that take on each value, the third column is the percentage of cases that take on each value and the fourth column is the cumulative percentage of cases from top to bottom. The data presented in Table 6.1 were obtained by using the tabulate command on the religious identification variable.

To get a frequency table from using drop-down menus, select "Statistics," then "Tables," and "One-way tables." Doing this pops up a new menu with a drop-down menu from which

you should select the variable for which you want a frequency table. Once you have selected your variable, click the "OK" button at the bottom on of the pop-up menu.

Pie graphs, such as Figure 6.1 in *FPSR*, are a graphical way to get to know categorical and ordinal variables. The commands for producing this type of figure in Stata are below.

The syntax to produce a pie graph in a do-file or from the command line is:

```
graph pie, over(variable)
```

To produce a pie graph using drop-down menus, select "Graphics," then "Pie chart." Doing this pops up a new menu containing a drop-down menu labeled "Category Variable." Once you have selected your variable, click the "OK" button at the bottom on of the pop-up menu.

5.2 Describing continuous variables

Figure 6.3 displays the output from Stataś "summarize" command with the "detail" option. This command produces a full battery of descriptive statistics for a continuous variable. From a do-file or from the command line, the syntax for this command is:

```
summarize variable, detail
```

To get this same output from using drop-down menus, select "Statistics," then "Summaries, tables, & tests," "Summary Statistics," and "Summary Statistics" (again). Doing this pops up a new menu containing a drop-down menu from which you should select the variable for which you descriptive statistics. Once you have selected your variable, click the circle under "Options" labeled "Display additional statistics," and then click the "OK" button at the bottom on of the pop-up menu.

From the discussion in Chapter 6, we can see that Stataś "summarize" command present both rank statistics and moment statistics to describe the values taken on by continuous variables. To get a visual depiction of rank statistics, we recommend producing a box-whisker plot like that displayed in Figure 6.4. From a do-file or from the command line, the syntax for a box-whisker plot of the values for a single continuous variable is:

```
graph box variable
```

To produce a box-whisker plot using drop-down menus, select "Graphics," then "Box plot." Doing this pops up a new menu containing a drop-down menu labeled "Variables." Once you have selected your variable, click the "OK" button at the bottom of the pop-up menu.

To get a visual depiction of moment statistics, we recommend producing either a histogram (Figure 6.5 and 6.6) or a kernel density plot (Figure 6.7). From a do-file or from the command line, the syntax for a histogram of the values for a single continuous variable is:

`hist` *`variable`*

To produce a histogram using drop-down menus, select "Graphics," then "Histogram." Doing this pops up a new menu containing a drop-down menu labeled "Variable." Once you have selected your variable, click the "OK" button at the bottom of the pop-up menu.

From a do-file or from the command line, the syntax for a kernel density plot of the values for a single continuous variable is:

`kdensity` *`variable`*

To produce a histogram using drop-down menus, select "Graphics," then "Smoothing and densities," and then "Kernel density estimation." Doing this pops up a new menu containing a drop-down menu labeled "Variable." Once you have selected your variable, click the "OK" button at the bottom of the pop-up menu.

5.3 Putting descriptive statistics into tables

When we produce descriptive statistics in Stata using either the "tabulate" or "summarize" commands, we get a lot of output. Usually this output is much more than what we need to present in a paper that describes our variables one at a time. We therefore suggest making your own tables in whatever word processing program you are working with.

5.4 Putting Stata graphs into documents

Once you have produced a Stata graph that you want to include in a document, one of the easiest ways to do so is to right-click on the graph in Stata and select "Copy" and then right-click on the location where you want to place the graph in your word processing program and select "paste."

6 Bivariate hypothesis tests in Stata

In this section we go through the four different types of bivariate hypothesis tests discussed in Chapters 8 and 9 and discuss how to conduct these analyses in Stata.

6.1 Tabular analysis

In tabular analysis, we are testing the null hypothesis that the column variable and row variable are unrelated to each other. We will review the basics of producing a table in which the rows and columns are defined by the values of two different variables, generating hypothesis-testing statistics, and then presenting what you have found.

From a do-file or from the command line, the syntax for producing a two-variable table is:

```
tab2 rowvariable colvariable, column
```

where "*rowvariable*" is usually the dependent variable (with its values displayed across rows in the table) and "*colvariable*" is usually the independent variable (with its values displayed down the columns in the table). The option "column" tells Stata that, in addition to the frequency of values (or number of cases) being displayed in each cell, this command produces column percentages for each row beneath the frequencies. As detailed in Chapter 8 of *FPSR*, these column percentages allow for the comparison of interest—they tell us how the independent variable values differ in terms of their distribution across values of the dependent variable. It is crucial, when working with tables of this nature, to put the appropriate variables across the rows and columns of the table and then to present the column frequencies.

To produce a two-variable table using drop-down menus, select "Statistics," then "Summaries, tables, & tests," then "Tables," and then "Two-way tables with measures of association." Doing this pops up a new menu containing a drop-down menus labeled "Row variable" and "Column variable." Once you have selected your variables, click on the box labeled "Within-column relative frequencies" and then click the "OK" button at the bottom of the pop-up menu.

6.1.1 Generating test statistics

In Chapter 8 we discuss in detail the logic of Pearson's chi-squared test statistic which we use to test the null hypothesis that the row and column variables are not related. To get this test statistic and the associated p-value for a two-variable table in Stata from a do-file or from the command line, the syntax for producing a two-variable table is:

```
tab2 rowvariable colvariable, column chi2
```

where "*rowvariable*" is usually the dependent variable (with its values displayed across rows in the table) and "*colvariable*" is usually the independent variable (with its values displayed down the columns in the table). The option "column" tells Stata that, in addition to the frequency of values (or number of cases) being displayed in each cell, this command produces column percentages for each row beneath the frequencies. The option "chi2" tells Stata to report a chi-squared test statistic and the associated p-value.

To produce a two-variable table using drop-down menus, select "Statistics," then "Summaries, tables, & tests," then "Tables," and then "Two-way tables with measures of association." Doing this pops up a new menu containing a drop-down menus labeled "Row variable" and "Column variable." Once you have selected your variables, click on the boxes labeled "Within-column relative frequencies" and "Pearson's chi-squared" and then click the "OK" button at the bottom of the pop-up menu.

6.1.2 Putting tabular results into papers

We recommend that you make your own tables in whatever word processing program you choose to use instead of copying and pasting the tables that you make in Stata . The first reason for doing so is that you will think about your results more closely when you are producing your own tables. This will help you to catch any mistakes that you might have made and to write more effectively about what you have found. Another reason for doing so is that tables constructed by you will tend to look better. By controlling how the tables are constructed, you will be able to communicate with maximum clarity.

As a part of making your own tables, you should have the goal in mind that your table communicates something on its own. In other words, if someone *only* looked at your table, would they be able to figure out what was going on? If the answer is "yes," then you have constructed an effective table. We offer the following advice ideas for making useful tables:

- Give your tables a title that conveys the essential result in your table
- Make your column and row headings as clear as possible
- Put notes at the bottom of your tables to explain the table's contents

6.2 Difference of means

Difference of means tests are conducted when we have a continuous dependent variable and a limited independent variable.

6.2.1 Examining differences graphically

When we use graphs to assess a difference of means, we are graphing the distribution of the continuous dependent variable for two or more values of the limited independent variable. Figure 8.1 shows how this is done with a box-whisker plot. The syntax for doing this is:

`graph box` *`depvariable`*`, over(`*`indvariable`*`)`

Where *depvariable* is the name of the continuous dependent variable and *indvariable* is the name of the limited independent variable.

To produce a box-whisker plot using drop-down menus, select "Graphics," then "Box plot." Doing this pops up a new menu containing a drop-down menu labeled "Variables." Once you have selected your dependent variable, click the tab labeled "By" and select the independent variable, then click the "OK" button at the bottom of the pop-up menu.

In Figure 8.2 we produced a kernel density plot of the distribution of our continuous dependent variable for the two values of our limited independent variable. The syntax for producing this figure involves Stata creating two separate plots and overlaying them. The syntax for doing this is:

As before, *depvariable* is the name of the continuous dependent variable and *indvariable* is the name of the limited independent variable. The *condition* in this case is that the independent variable is equal to a particular value. The "||" in the command above tells Stata that you are going to overlay a second twoway plot.

6.2.2 Generating test statistics

To conduct a difference of means t-test such as the one discussed on pages 146-150 of *FPSR*, the syntax is:

```
ttest depvariable, by(indvariable)
```

To conduct a difference of means t-test using drop-down menus, select "Statistics," then "Summaries, tables, & tests," then "Classical tests of hypotheses," and then "Two-group mean-comparison test." Doing this pops up a new menu containing a drop-down menus labeled "Variable name" and "Group variable name." Select your continuous dependent variable from the drop-down menu labeled "Variable name" and select your limited independent variable from the drop-down menu labeled "Group variable name." Once you have selected your variables, click the "OK" button at the bottom of the pop-up menu.

6.3 Correlation coefficients

Correlation coefficients summarize the relationship between two continuous variables.

6.3.1 Producing scatter plots

We can examine the relationship between two continuous variables in a scatter plot such as Figure 8.3 in *FPSR*. The syntax for producing such a figure is:

```
twoway scatter depvariable indvariable
```

Using a drop-down menu select "Graphics" then "Twoway graph." On the resulting menu click "Create" on the "Plots" tab. Click the button labeled "Basic Plots" and select "Scatter" under the menu labeled "Basic plots: (select type)." Then use the pull-down menus to select your dependent variable as the "Y variable" and your independent variable as the "X variable." Once these selections have all been made, click "Submit."

6.3.2 Generating test statistics

To generate a correlation coefficient with the associated p-value, the syntax is:

```
pwcorr depvariable indvariable, sig
```

The command "pwcorr" is short for "pairwise correlations" and can be used to obtain the correlation coefficients for more than two pairs of variables.

Using a drop-down menu select "Statistics" then "Summaries, tables, and tests," then "Summary and descriptive statistics," and then "Pairwise correlations." On the resulting menu click the box next to "Print significance level for each entry." Under the drop-down menu labeled "Variables" select the two (or more) variables for which you want a correlation coefficient. Once these selections have all been made, click "Submit."

6.4 Bivariate regression

The estimation of a bivariate regression model, as discussed in Chapter 9 of *FPSR*, is fairly straightforward. The syntax is:

```
reg depvariable indvariable
```

Using a drop-down menu select "Statistics" then "Linear models and related," then "Linear regression." On the resulting menu select your dependent variable from the drop-down menu labeled "Dependent variable" and select your independent variable from the drop-down menu labeled "Independent variables." Once these selections have all been made, click "Submit."

This will produce output such as that pictured Table 9.4.

7 Multiple regression

The estimation of a multiple regression model, as discussed in Chapters 10 and 11 of *FPSR*, is just an extension of the command used for estimating a bivariate regression. For example, if we have three independent variables the syntax is:

`reg` *`depvariable indvariable1 indvariable2 indvariable3`*

Using a drop-down menu select "Statistics" then "Linear models and related," then "Linear regression." On the resulting menu select your dependent variable from the drop-down menu labeled "Dependent variable" and select each of your independent variables from the drop-down menu labeled "Independent variables." Once these selections have all been made, click "Submit."

7.1 Standardized coefficients

In order to obtain standardized coefficients, as discussed on pages 196 and 197, you can add a comma and the word "beta" to the syntax:

`reg` *`depvariable indvariable1 indvariable2 indvariable3`*`, beta`

To do this using the drop down menu, follow all of the instructions above and then click on the tab titled "Reporting" before clicking on "Submit." On the resulting menu check the box labeled "Standardized beta coefficients" and the click "Submit."

7.2 Post-estimation diagnostics in Stata for OLS

In Chapter 11 we discussed a number of diagnostic procedures that can be carried out once an OLS model has been estimated. These procedures are all available in Stata . It is important to keep in mind that when asked to conduct such an analysis, Stata will always do so using information from the last regression model that was estimated.

7.2.1 Identifying outliers and influential cases in OLS

Figure 11.5 shows the results from a lvr2plot which is short for "leverage-versus-residual-squared plot." The syntax for producing this type of figure on the last estimated regression model is:

```
lvr2plot
```

Using a drop-down menu select "Statistics" then "Linear models and related," then "Regression Diagnostics" and then "Leverage-versus-residual-squared plot." This will produce a new menu with a series of options. If you just want this plot, you can left-click on "Submit."

Table 11.9 shows the five largest DFBETA scores from a regression model. The syntax for estimating DFBETA scores is:

`predict` *`newvariable`*`, dfbeta(`*`indvariable`*`)`

Where *newvariable* is the name of a new variable that will contain the DFBETA value for each observation and *indvariable* is the name of the independent variable for which you want to have the DFBETA calculations made.

Using a drop-down menu select "Statistics" then "Linear models and related," then "Regression Diagnostics" and then "DFBETAs." This will produce a new menu from which you can select the independent variable(s) for which you want to have DFBETA scores calculated. Once you have made your selection, you should left-click "Submit."

7.2.2 Detecting multicollinearity in OLS

As we discussed on pages 227 and 228 of *FPSR*, one way to detect multicollinearity is to estimate a Variance Inflation Factor (or "VIF") for each independent variable after you have estimated your regression model. The syntax for producing this type of figure on the last estimated regression model is:

`vif`

Using a drop-down menu select "Statistics" then "Linear models and related," then "Regression Diagnostics" and then "Specification tests, etc.." This will produce a new menu with a series of options. Under "Reports and Statistics (subcommand)" select "Variance inflation factors for the independent variables (vif)" and left-click on "Submit."

8 Dealing with time series data before estimating OLS

As we discussed on pages 233-244 of *FPSR*, time series data require careful treatment. In Stata it is quite helpful to identify a data set as a time series (unless told otherwise, Stata assumes that the data it is working with are from a cross-sectional data set). This is done with the following syntax:

`tsset` *`timevariable`*

Where "tsset" is short for "time series set" and *timevariable* is the name of the variable that

identifies the time period to which each observation corresponds.

To use a drop down menu to identify a time series data set, select "Statistics" then "Time series," then "Setup and utilities" followed by "Declare dataset to be time-series data." From the drop-down menu labeled "Time variable" select the name of the variable that identifies the time period to which each observation corresponds.

Once you have identified a data set as a time series data set, you can then use a series of time series operators to analyze your data. These operators are as follows:

- In order to create a one period lagged variable value (for example changing from X_t to X_{t-1}) you can write L.*variable*.
- In order to create a one period lead variable value (for example changing from X_{t-1} to X_t) you can write F.*variable*.
- In order to create a one period differenced variable value (for example changing from Y_t to $\triangle Y_t = Y_t - Y_{t-1}$) you can write D.*variable*.

9 Binomial logit and binomial probit

The commands for estimating binomial logit and binomial probit models in Stata are fairly similar to the commands for estimating OLS models. For binomial logit, the syntax is:

`logit` *`depvariable indvariable1 indvariable2 indvariable3`*

Using a drop-down menu select "Statistics" then "Binary Outcomes," then "Logistic regression." On the resulting menu select your dependent variable from the drop-down menu labeled "Dependent variable" and select each of your independent variables from the drop-down menu labeled "Independent variables." Once these selections have all been made, click "Submit."

For binomial probit, the syntax is:

`probit` *`depvariable indvariable1 indvariable2 indvariable3`*

Using a drop-down menu select "Statistics" then "Binary Outcomes," then "Probit regression." On the resulting menu select your dependent variable from the drop-down menu labeled "Dependent variable" and select each of your independent variables from the drop-down menu labeled "Independent variables." Once these selections have all been made, click "Submit."

9.1 Obtaining predicted probabilities for binomial logit and binomial probit models

As discussed in Chapter 11 of *FPSR*, the results from models with dichotomous dependent variables can be interpreted in terms of predicted probabilities. To obtain predicted probabilities after a binomial logit or binomial probit model has been estimated, the syntax is:

```
predict(newvariable), p
```

Where *newvariable* is the name of the new variable that will be constructed in the data set containing the predicted probabilities from the last model (logit or probit) that was estimated.