

Lab 1: Introduction to Stata

Simon Halliday¹

ECO311, Fall 2016

Introduction

In this lab – and subsequent labs over the semester – you will go through a sequence of steps to become better informed about survey data, Stata commands, and the measurement of consumption, income, poverty and inequality.

I will regularly ask you to ‘type in commands’ to see what happens when you use them with the data. I will also ask you to do *exercises* (shown in grayed out boxes). You should maintain a document, written in MSWord, Google Doc, L^AT_EX or RMarkdown, to provide your answers to these exercises. I will also ask you to reflect on your answers in discussions with other students. You will be able to submit your answers to these exercises on Moodle.

In most cases, I want to *see that you have done the work* and you will be graded for *completion* of the exercises. You should have enough time to complete these exercises in our lab classes (generally Thursdays). If you need to take time to complete them outside of class (or you miss class), then let me know that’s what you need to do – please keep in contact with me about your work and your progress. The exercises are meant to be both instructive and fun (because you’re discovering things about the world through data).

Survey Data Basics

We will use the National Income Dynamics Study (NIDS) from South Africa. The data set has been constructed from information collected from a *household survey*. We will look at what types of questions a typical household survey might ask before looking at the ‘answers’ as they are stored in the data sets. The questionnaire for the survey, as well as other material explaining how certain variables were created and coded (called the ‘metadata’), can be found in the Resources section of the ECO311 Moodle site.

Open the NIDS Wave 1 household questionnaire by opening the folder and reading NIDS_Web_Household_Wave1.pdf. At the top of the document, you should be able to read: “National Income Dynamics Study Household Questionnaire Cover.” This first page is a Cover Page, and captures information that is *not* usually available in the data set e.g. the name of the respondent and the address of the household.

What types of questions do household surveys ask? Typically, a wide range of topics may be covered in such surveys:

- *Personal Descriptive (demographic) questions* Surveys always include

¹ This lab exercise, and others in ECO311, are based on initial labs developed by Murray Leibbrandt and Callie Ardington and which I have significantly re-tailored to my purposes.

Doing Stata

By the end of this lab, you should have:

- Found where supplementary documents for the data sets are stored (e.g. the survey questionnaires) and looked at some typical questions from the surveys.
- Learned how to open a log-file and save it (and why log files are important).
- Clarified your understanding of relative file paths to help make your work reproducible.
- Found and accessed the data sets, become familiar with the way that survey data can be captured and displayed in Stata.
- Learned how to open and use a do-file and why do-files are important for *replicating* your work.
- Started to learn some of the basic commands for getting around in Stata (e.g. the `help` command)

METADATA A survey’s ‘metadata’ help us to understand a survey’s data. The metadata are contained in document written about *how* the data were captured. See if you can open the metadata file for NIDS now (Wave1Metadata_Final.pdf). You should be able to see how some of the questions have been coded. The metadata is useful when trying to figure out how answers like ‘not specified,’ ‘not applicable,’ and ‘did not reply’ were *coded* (entered as numbers). Many surveys also include a user guide with important information for the analyst. Open up the introduction to NIDS data (Wave1IntroductiontoNIDSdataOct2009.pdf)

questions about personal-social characteristics of respondents such as sex, age, occupation, income, and education. This kind of information helps researchers to understand relationships between variables they may be interested in. For example, we could look at how many people in the age group 25 – 35 have a tertiary education, and what types of jobs they have, compared to others in the same age group with less education.

- *Behavioral questions* Many survey questions relate to people's actions or behavior. For example, many economists have studied patterns of spending and saving.
- *Attitudinal questions* Surveys ask questions concerning people's beliefs, opinions, attitudes, or expectations. For example, there could be questions about how safe people feel in their homes, compared to an earlier period. It is unlikely that attitudinal data would be available from non-survey sources (though social media like Facebook and Twitter are changing that because of #datamining).
- *Living Environment questions* Surveys ask questions about the circumstances in which respondents live, and may include information about the neighborhood, the adequacy of living quarters, membership in groups and organizations, and so on.

Let's have a look at some of the questions asked in the NIDS household survey. Scroll down to page 4 of the questionnaire where the first questions for the household roster appear. You can see the different types of questions that appear: some collect information which will not be released in the data (e.g. name of the person), some collect information that must be coded (e.g. if you answer female to question B4, your answer is recorded as "2"), and still others must be written in as heard (for example: how old is . . . ?).

Many of the questions will have answers that are *coded*: for example, if you have never been married, your 'answer' to question B8 will be recorded, or coded, as a "5." All the codes for marital status can be found in the code sheet document.

There are hundreds of questions in any household survey, and so many codes represent different answers. In the data sets we look at sometimes the data will be labeled clearly (so when you look at whether an observation is male or female, the gender will be clear from how the data is labeled). But, variables will often not be labeled. For example, when we look at data on relationship to the household head, this information may have only been captured as a list of numbers, rather than as "head," "spouse," "daughter," etc.

HOUSEHOLD ROSTER The household roster is the list of all people who are members of the household. The household roster normally details their characteristics, such as age, gender, education, and so on. The definition of a household may be that people are members of the same family, or that they 'regularly eat from a common pot.' The definition of a 'household' depends on the social context.

Exercise 1

- Find a question about each household member's education. Keep a note of where that question is and what it asks for.

- b. Find the section that asks questions about the household's water supply. Which question would you be directed to if you answered "No" to question D4?

Having the survey at hand means that we can always go back to the original question and check what the codes stand for. Before you use a variable, it is also important to clarify how a question has been phrased in the original questionnaire and how it has been coded, because your data may not be labeled as clearly as you would like or be immediately suitable to statistical analysis and inference (e.g. linear regression).

Because questionnaires are coded in particular ways, it is also important to *code* our cleaning and management of the data to make it *reproducible*. We can think of reproducibility having two main virtues:

1. We (the researchers) can return to our data at a later date and reproduce our earlier work in such a way that we can understand it easily, rather than struggling to remember what choices we made or why we made them (we document the choices we make and our motives when we properly comment on our code to document our choices)
2. Other researchers can reproduce what we did if they can read our code. Without our code and comments, our research will not understand what we did or how we did it, impeding the scientific process.


Paying close attention to the challenges of reproducibility, we shall take the opportunity to make our lives easier by following the TIER protocol (outlined later).

Getting Started

Stata is a statistical program that economists use to manage, look at and analyze economic data. It is particularly suited to the analysis of complex survey data – in our case, data from household surveys. It has a library of pre-programmed commands, which are useful for creating summary statistics from the data, and for performing more detailed statistical analyses. We can also write our own programs in Stata to perform a specific task, but we will largely steer clear of such programming in this course.

The Stata User Interface

In Windows, open Stata by navigating to: Start | Programs | StataSE14

In MacOS, open Stata by clicking on the Stata shortcut in your dock:  or by navigating to the Stata shortcut in the Applications folder.

You should now have the Stata window open, as in Figure 1.

There is a set of pull down menus, as well as 5 smaller windows, named: Review, Variables, Stata Command, Stata Results, and Properties. When we

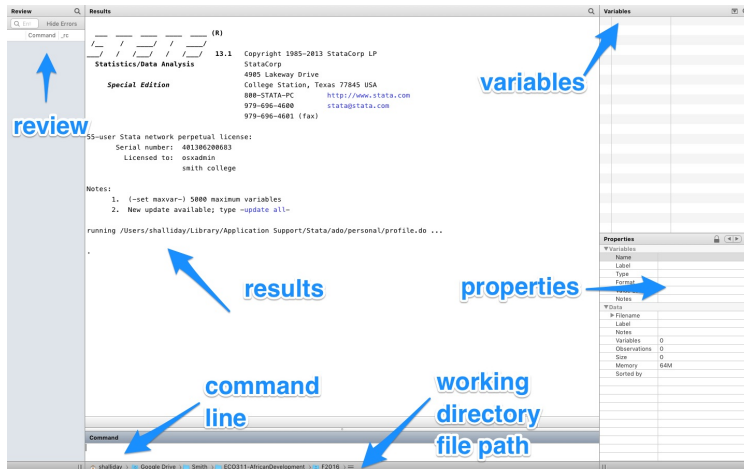


Figure 1: The Stata Graphic User Interface

want to tell Stata to do something (execute some command), we will type it into the Stata Command line window. The results of our command will show up in the Stata Results window. (I often close the Properties window).

In any session, all commands that we give to the program will show up in the Review window once we execute them. The Review window is useful, because it allows us to recall commands quickly (by clicking on the relevant command in the Review window it will reappear in the Stata Command line) and can be saved, for editing later in a do-file.

Right now the Variables window is empty because we have not loaded any data. Whenever you have a data set open, the variable names will show up in this window. If the variables have been labeled, then the labels will appear too, e.g. in the NIDS data one variable is `w1_r_b6` and is labeled as “b6 - Age in Years.”

Setting Your Working Directory

In Figure 1, one of the arrows points to the ‘Working Directory.’ This is the directory in which you are currently doing your work, that is, where Stata will save files you create unless you give it an alternative file path. It is a good practice to navigate the working directory to the set of directories that you will use. I would recommend that you now create a set of directories in your Google drive that you use for your data analysis.

First, download the TIER protocol guidelines on Moodle (TierProtocol2.1.pdf). As we confirm in the protocol, one of the ways to ensure that you have reproducible work is to maintain a consistent directory and file structure. We’re going to set up just that structure for Lab 1.

Now, second, go through the following steps

- Create a folder called ECO311
- In your ECO311 folder create a sub-folder called Labs and a sub-folder in

DO-FILE A Stata do-file is a short program file that runs a set of commands at once. We will use do-files regularly to make our work *reproducible*.

the Labs folder call Lab1.

- In your Lab1 folder create two folders: 'Originals' and 'ProcessingAnalysis'
- In your Originals folder create two folders: 'OriginalData' and 'Metadata'
- In your 'ProcessingAnalysis' folder create two folders: 'Importable Data' and 'CommandFiles'
- In your 'CommandFiles' folder create two folders 'logs' and 'cmdlogs'

You will notice that I've left out spaces in the folder names above. It's often easier for programs to read folder names *without* spaces, which is why I suggest you do this.

I would suggest that you copy/paste the file structure for future labs (2 through 6) now so that you don't need to re-construct the file structures later.

Now click on the area where you have your working directory and navigate to EC0311/Labs/Lab1. This will be your *working directory* for Lab 1.

Log files

If you want to record what you do in a Stata session so that you can look at results later, you need to open a *log file*. A log file is simply a record of all the commands you enter into Stata and the output from those commands. The key is to make sure you have a log file open at the beginning of a Stata session, and to close it once you have finished, and *before* you close Stata.

There are three ways you might open a log file.

On a Windows computer, *if you have not set up your working directory* you will have to use the drive letter you plan to use and a full file path, e.g. C:\MyDocuments\EC0311\Labs\Lab1\ProcessingAnalysis\logs\yourlastname_lab1.log. If you have set up your working directory as I suggested earlier, then you can simply use *relative* file paths, i.e. paths that are *relative* to your current working directory location.

- You can open a log file by typing:

```
log using .\ProcessingAnalysis\CommandFiles\logs\
yourlastname_lab1.log
```

in the Stata Command window.

- If you want to create your log file by *saving over* an old log file, you can type

```
log using .\ProcessingAnalysis\CommandFiles\logs\
yourlastname_lab1.log , replace
```

- If you wanted to *add* to the old file instead of replacing it with a new one, you could write `append` instead of `replace`.

```
log using .\ProcessingAnalysis\CommandFiles\logs\
yourlastname_lab1.log , append
```

On MacOS:

- If you are using a Mac, all the backslashes above should be replaced with *forward slashes*
- If you have *not* set up your working directory, you will not use a drive letter and you will use a *username*.
- On my MacOS computer (running OS X El Capitan) and having set the working directory to Lab1, I would use:

```
log using ./ProcessingAnalysis/CommandFiles/logs/
halliday_2016.06.21.log
```

Be sure to save your log file with the extension `.log` (you define a file's extension at the end of the file name). This will make it easier for you to edit, cut and paste your log in any text editor. If you do *not* add `.log`, then Stata will automatically save the file as an `.smcl` file, which cannot easily be read outside of Stata. To verify that it is a `.log` file and not a `.smcl` file, look at the output that Stata shows you, e.g.

```
log : lab1_2015.log
log type : text
opened on : 29 Jan 2015, 11:11:23
```

If it says "smcl" instead of "text" under log type, then you have done it incorrectly. Close the log file (type `log close`) and do it again with the `log` extension.

If your file path has a space in it you will need to use quotation marks around the file name:

```
log using ‘\folder with spaces\yourlastname_lab1.log’
```

Data in Stata

You will need to download the data from Moodle before you can open it in Stata. Double click on `nids.zip` in the ECO311 resources section, unzip the file and save it in the following folder: `Labs/Lab1/Originals/OriginalData`

You should be able to open the data set by double clicking on the unzipped file `nids.dta`.

If you go to the NIDS website and download the publicly released datasets, you will find child, adult, proxy, household, household roster, individual derived and household derived datasets. The dataset that you have now loaded into

Stata was created for this course by merging these datasets. We have also made the questionnaires available on Moodle.

First let's take a more detailed look at what variables are stored in the data set. In the Stata command line, type:

```
describe
```

You should see a list of what variables are contained in the data, how many observations there are, what kind of variables they are (eg. 'byte', 'float,' 'int,' 'str' or 'long' variables), and any comments which are attached.

A typical variable name in the dataset you are using will look as follows: `w1_a_j1`. The `w1` tells us that the variable is from wave 1 of the NIDS dataset in which data was collected in 2008. The `a` tells us that the information contained in the variable was obtained from an answer to one of the questions in the adult questionnaire. The `j1` tells us that the variable corresponds to question `j1` of the adult questionnaire. Questions from the child, household, household roster and proxy questionnaires have the `w1_c_`, `w1_h_`, `w1_r_` and `w1_p_` prefix respectively.

You can Google what these variable types mean, but the two abbreviations you should understand are 'int' for *integer* and 'str' for *string* (a word or string of words/letters), such as a month, "Male", or similar.

Exercise 2

- a. What is the variable for "What is the highest educational qualification attained?" called?
- b. What is the variable for the question "Is [...]s biological mother listed on this roster?" called?

Now type the following command into the command line:

```
browse
```

This command directs you to a spreadsheet, where the data appears. You should note the following:

- each *observation* (in this case, each person in the household for which there is recorded information) appears on a separate *row* of the spreadsheet.
- each *variable* appears in a separate *column* of the spreadsheet.
- For example, the first person (first observation as a row) has several variables (in columns) `hhid = 103034` (the unique household id number), `pid = 301011` (their unique person id number), `w1_r_b1 = Person 01` (their person number within the household) and `w_r_b6 = 41` (they are 41 years old). If you move along the row, you can see the rest of his data recorded for each question.

Investigating Observations

Using the `browse` command, find the 420th observation. What is this person's relationship to the household head? Their gender? Their education?

A quicker way to do this is to type in:

```
list
```

Exercise 3

Using the `browse` command again, pull up the data for all the people in the household with `hhid 103152`.

- a. How many people are there in the household?
- b. Can you describe the composition of the household?
- c. Which variables would you look at to describe this household?
- d. What is person number 9's marital status? What about person number 13?
- e. How many children under 15 years of age are there in the household?
- f. How old is the household head?

Now consider the questions that we have looked at above, for a *household of your choice*. Use the `|browse|` command to find a household, and jot down a description based on the few variables of interest we have discussed. Make sure you distinguish between information about *individuals*, and information about the *household*. Be prepared to describe your household to the class.

You should see all the data for the first observation being displayed in the Results window. If you continue to press the space bar, you will scroll through the entire data set – which is not what we want to do! A really useful key to use here is the *break* key, the red button with the white cross mark, at the right hand side of the tool bar at the top (see Figure 2). If you click on this button, you interrupt the command that is currently being executed, and return control to yourself.

If you type `list in 420`, you will see only the record for the observation that we want. Note that the person id number (`pid`) for this person is 301510. So, another way to display only the record for this observation is to type:

```
list if pid == 301510
```

Note that we typed in *two* equal signs when we wanted to tell Stata to fetch the observation for which the person id was 'equal to' some number. This is standard Stata syntax. If we wanted to see only this person's data in the spreadsheet format, we would type in

```
browse if pid == 301510
```

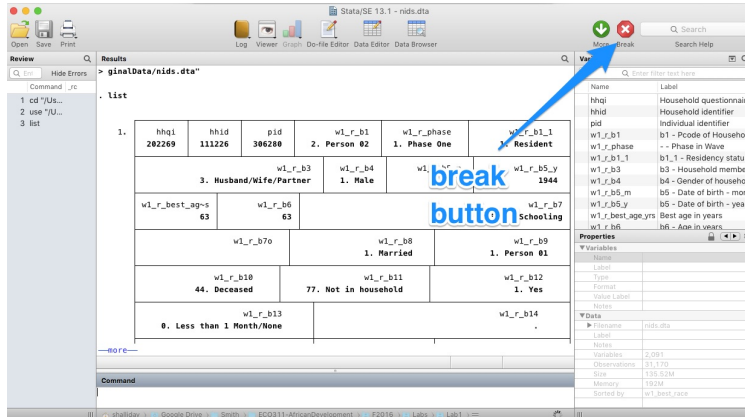



Figure 2: Hitting the Break button!

Examining Variables

We've looked at values of particular variables for a few individuals and a few households. But now we want to look at these values more generally – for the entire data set. Close the spreadsheet and return to the Stata command window. There is an efficient way to find the names of variables that you are interested in. Type in:

```
lookfor educ
```

This should give you a list of all the variables that have 'educ' in their name. Suppose you are interested in the variable `w1_r_b7`.

- get back into the spreadsheet using `browse`, locate the relevant column, and take a look at the entries for this variable. Just looking at the column doesn't tell us much about the variable.
- now get back to the Stata command line, and type in
`codebook w1_r_b7`

The `codebook` command gives us more summarized information about the variable: (among other things), it tells us the range of values captured and the number of missing values.

Go back to the NIDS household questionnaire. Pick any question that you would be interested in finding out answers for. Find out what the variable name associated with this question is and make a note of it. Now suppose you want to look at this variable in conjunction with a few others such as `household_id` and `person_id`. For example, if you type in

```
browse hhid pid w1_r_b6 w1_r_b7 w1_r_b4
thevariableyouchose
```

the spreadsheet will only show you data from these variables.

We have just worked from the questionnaire to the data set to find variables that we are interested in. However, there may be some cases where we need

STATA TIP If you want to scroll through the commands you have run hit "Page Up" and "Page Down" in Windows or "Fn + ↑" and "Fn + ↓" in MacOS. These shortcut keys make it a lot easier to edit a command in which there's a minor error rather than typing out the command again.

Note, don't actually type in the words `thevariableyouchoose` you should substitute that with the name of the variable you wrote down earlier... stranger things have happened though, so don't worry if you type the wrong thing in.

to work in the opposite direction: for example, when we find a variable that has no clear value labels. In this case, you will want to work from the data set back to the questionnaire, to clarify the following:

1. which *question* the variable captures information for, and
2. how the variable was *coded*

Aggregate Properties of Data: Tabulation

A frequency distribution table lists all the observed values for a given variable and the number of observations that take on each of these values.

The command `tabulate` (abbreviated to `tab`) produces one- and two-way tables of frequency counts along with various measures of association (relationships between variables). For example, Table 1 shows the frequencies (counts) and percentages (proportions of the population) of male and female persons in the NIDS data.

	Frequency	Percent	Cumulative Percent
Male	14643	46.98	46.98
Female	16527	53.02	100.00
Total	31170	100.00	

Table 1: A frequency table of gender in the NIDS data

In this first class, we will use the `tab` command to produce summary tables of data, to answer the general question: how much of our data set falls into different variable *categories*. We can see what `tab` does easily by example.

So, if we wanted to know how many people in our data set were classified as 'colored,' then we could type:

```
tab w1_best_race
```

and we would get a table specifying the number of observations of each race-type, and the percentage of the data set taking on each of the variable values. From the table, what is the answer to our question about how many people are 'Colored'? What percentage of the data set is 'African'?

To get a frequency distribution of the highest educational qualification completed in the sample, you could type:

```
tab w1_r_b7
```

When we use the `tabulate` command you will notice that descriptive labels appear for each category. However, the actual data is recorded as a numeric value. For example if we codebook the education variable:

```
codebook w1_r_b7
```

we see that the actual values of this variable are *numeric* and range from -9 to 25 . We also see that a code of 18 is given to individuals who have completed a "Certificate with Grade 12." Stata uses the numerical values, thus

Remember that in South Africa, there are many ethnic groups and people who think of themselves as culturally and ethnically 'colored' conceive of this differently to people thinking of themselves as 'black African.' Historically, colored people were related to Malay slaves, the Khoi-San people of the Cape Area, and people of mixed race. Black African people are more likely to be part of the Nguni and Bantu peoples/language groups, such as the amaXhosa group of which Nelson Mandela was a member.

it is essential to know how a variable has been coded. The `noLabel` option with the `tabulate` command, displays the underlying numeric value rather than the label assigned to that value. To see numbers rather than labeled variables, type

```
tab w1_r_b7, noLabel
```

We can now see that the person who refused to answer the question was assigned a value of `-8` for this variable. There is a very useful command that allows us to see both the underlying numeric value and the description in one table. Type

```
numLabel, add
tab w1_r_b7
```

You only need to type `numLabel, add` once in a Stata session. We recommend that you do so every time you open a dataset, though it may take Stata a little time to process the command.

For how many people is the highest level of education unspecified? NIDS uses `-9` (9999 for years, 99 for months) for “Don’t Know,” `-8` (or 8888, 88) for “Refused,” `-5` for “Not Applicable” and `-3` for “Missing.” In Stata a missing value is usually recorded as a full stop/period. When we start to analyze the data we will need to recode these negative values to missing.

The `tabulate` command can be used to create tables of two variables, and may be restricted by using the qualifier, comparison or logical operators. These operators are in Table 2.

Qualifying operators		Comparison operators		Logical operators	
if	Qualifies <i>when</i> a command is to be executed	==	Equal to		or
		~= and !=	Not equal to	&	and
in	Qualifies <i>which</i> observation should have command applied	>, <	Greater than, Less than,	~	not
		>=, <=	Greater than/equal to Less than/equal to		

To generate a frequency table for education for women only, type:

Table 2: Operators in Stata Commands

```
tab w1_r_b7 if w1_r_b4==2
```

We can also use the `tabulate` command to generate two-way tables. To generate a frequency table for gender and education, type

```
tab w1_r_b7 w1_r_b4
```

To generate a frequency table for population group and education qualification, type

```
tab w1_r_b7 w1_best_race
```

Note that we have a *two-way table* with the number of observations in each category of population group and highest educational qualification, however, the percentages have disappeared. If we are interested in the percentages, we need to specify which type of percentages we want:

```
tab w1_r_b7 w1_best_race , row
```

will give a frequency table which generates the percentage of the *row* observations which are in each *column* category. Thus, 86.95% of the people with no schooling in our sample are African. At the bottom of each column, note that you will see the percentage of the entire data set that fall into that particular column category.

```
tab w1_r_b7 w1_best_race , column
```

will give a frequency table which generates the percentage of the *column* observations which are in each *row* category. So, 19.61% of African people in our sample have no schooling. At the end of each row, you will find the percentage of the entire data set that fall into that particular row category.

```
tab w1_r_b7 w1_best_race , row column
```

will give a frequency table which combines the information in the above two tables. You just need to be careful about which percentages you are reading from this table!

What if we want a frequency table with the percentages of the overall sample in each cell?

```
tab w1_r_b7 w1_best_race , cell
```

will give a frequency table which gives the percentage of the sample in each possible cell. For example, 1.9% of the sample are colored individuals with no schooling. The cell option can be combined with the row and column options.

Exercise 4

Use the commands you have learnt so far to answer the following questions:

- a. What percentage of the sample have no schooling?
- b. What percentage of the sample over 18 years of age have no schooling?
- c. What percentage of married females have no schooling?

Asking Stata for help

The help command in Stata is a useful way to learn about new commands, or remind yourself how to use commands. For example:

```
help log
```

tells you how to open and close logs, and why we use log files

```
help browse
```

reminds you about the browse command

At the end of your Stata session, you generally need to do at least 3 of the following 4 tasks:

- close your log file: type `log close`
- clear your data set: type `clear` or save your data set: using the *File* menu, choose *Save As*, find a space in your IntermediateData folder and give the data file a name, with file extension `'.dta'`
- close Stata, by typing `exit`

Now that you have saved and closed your log file let's take a look at it. In MS Word or any other text editor open your log file. It should look something like the log file I have uploaded to Moodle under the Lab1 folder.

If your log file looks very different to the file on Moodle and is hard to read then you probably saved it as an `scml` file and not a text file. It is important to save it as a text file so that you can look at the output once your Stata session is complete. If you are not sure how to save a log file as a text file read the section on opening log files again. Notice that all commands are preceded by a period. Log files store all the commands *and* the output in a Stata session.

Do files: a quick intro

Open Stata and type the following commands (be sure that your working directory is `/Labs/Lab1`).

```
clear all
use ./Originals/OriginalData/nids.dta
numlabel, add
tab w1r_b7 w1_r_b3
```

Often, we will want to execute commands repetitively in Stata, or we'll want to save the set of commands for a particular procedure so that we can run them on a different data set, or we want to ensure our work is *reproducible* by ourselves and others at a later date. We can write simple programs (do-files) in Stata's do-file editor which will run a set of commands at once.

Suppose we wanted to repeat the process of looking at the variables above. We can save the commands we have already used using the Review window. Select all the commands in the Review window, Right click and select "Send to Do-file Editor." This will open up the do file editor and a do file with all the commands you have used in your Stata session. You can now save this do file by selecting `File>Save` within the do file editor. Save the do file to your directory in a new folder called 'do' in the ProcessingAnalysis folder, e.g. `Labs/Lab1/ProcessingAnalysis/CommandFiles/do` and call the file "Example.do." Close the do file editor.

Open the do file editor, clicking on the envelope-button on the toolbar. Open the file you just created called `example.do`. In the do file editor, click on the button with the arrow that says “Do Current File.” If you just want to run some of the commands in a do file then highlight those commands and click on the “Do Current File” button.

You can copy and paste and write new commands in the do file, and then save the do file to run again on this data set or other data sets. If you made any mistakes in your Stata session the do file will end and an error message will appear. You will need to edit the do file by correcting or deleting invalid commands. You can also create a special log file of all the commands that you type in a Stata session. This has an advantage over saving the contents in the Review window as there is a limit to the number of commands that are stored in the Review window.

Now, when you start your Stata session type the following:

For Windows:

```
cmdlog using .\ProcessingAnalysis\CommandFiles\cmdlogs\
  Yourlastname_lab1.do
```

For MacOS:

```
cmdlog using ./ProcessingAnalysis/CommandFiles/cmdlogs/
  Yourlastname_lab1.do
```

When you end your Stata session type

```
cmdlog close
```

The do-file is just a text file, and can be created, edited, and saved using Word or any editor, as well as the internal Stata do-file editor. If you generate new variables in a do file, you cannot run the do file again without dropping those variables or loading the data set again. The same options that we used for log files – `replace` and `append` – work for `cmdlog` too.

Recording your work

Before you start any lab session or your problem sets remember to open *both* a log file (to record your results) and a `cmdlog` file to record your commands. If you want to continue from where you left off in a previous Stata session follow these steps:

Re-run the previous session by running your command log that you saved in the previous session:

```
do ./ProcessingAnalysis/CommandFiles/cmdlogs/
  yourlastname_lab1.do
```

Then open your existing log and command log files. You will need to specify the `append` option so that all new commands and outputs will be appended to the bottom of the files:

```
log using ./ProcessingAnalysis/CommandFiles/logs/
    yourlastname_lab1.log , append
cmdlog using ./ProcessingAnalysis/CommandFiles/cmdlogs/
    yourlastname_lab1.do , append
```

Now you can continue your session.

Best Practice for Stata

If you're going to write Stata code in the command line, then you should be sure to do the following:

- open a log file with `log using`
- open a cmdlog file with `cmdlog using`
- use the command `numlabel, add`
- *sometimes*, it is useful to use the command `set more off` which we'll refer to in the next lab

I start every session with these commands to ensure I know what I've done.

Getting Help for Stata

There are a couple of ways you can learn more about Stata, or simply refresh your memory about what we have done in class:

- the Stata `help` command
- the Stata website, which is accessible from Stata, if you click on the Help pull-down menu, and choose Stata website. This web site does tend to be somewhat more technical though.

Commands in Lab 1

break	tab,row column	use
clear	browse	browse if
cmdlog	codebook	cmdlog close
using	describe	do
exit	help	list
list if	list in	log close
log using	lookfor	tab
tab, cell	tab,column	tab, row