

mistake in the file, *Stata* will stop at that point issuing an error message. Re-open your do-file by typing

```
. doedit birthday.do
```

Fix your mistake, and try again.

Creating a Log-File

Do-files are great if you know ahead of time that you will want to save all the commands you issue, but often we proceed with an analysis and then wish that we could save the commands. Well, that is possible too. A log-file allows us to recall all the commands we have typed and save them into a file.

Again, we want to make sure we are in our home working directory. Change your working directory by clicking on the “File” menu and selecting “Set Working Folder.” The default is your “Documents” folder; click the “Choose” button.

Suppose I was working on the *Constructing Confidence Intervals* lab. After completing the in-class portion, I realize that the assignment is to basically do exactly the same thing with a few minor changes. It would be great if I didn’t have to type all those commands again, if I could just change a few things and re-run the code. We can do this utilizing a log-file

We open a log file by typing **log using** and then the name we want to call the log file. Suppose I want to name my file *cilab.log*, then I would type the following command:

```
. log using cilab.log
```

Now you can ask Stata to reprint the last 50 (or however many you want) commands you typed in.

```
. #review 50
```

Close your log-file with

```
. log close
```

The log-file you have created won't work immediately as a do-file. It must be cleaned up to resemble a do-file. You can modify your log-file using the *Stata* Do-file Editor.

```
. doedit cilab.log
```

Here you change the log-file to be a do-file. Remember, a proper do-file has exactly one correct *Stata* command per line. Delete everything that does not look like a *Stata* command or part of a *Stata* command.

In this example, part of my log-file looks like this:

```
13 count if lower68 <= -12.567 & upper68 >= -12.567
12 generate num = _n
11 graph upper68 lower68 bs1 num, connect(||.) symbol(iio) yline(-12.567) ylabel
> ti("100 68\% confidence intervals from samples n=16")
10 generate lower90 = bs1 - 1.645*23.66127/sqrt(16)
```

To correctly run as a do-file I must remove the numbers at the beginning of each line and although the log-file split the graph command onto two lines, I have to put this back onto one line or *Stata* will produce an error. This section of my log-file should be changed to look like this:

```
count if lower68 <= -12.567 & upper68 >= -12.567
generate num = _n
graph upper68 lower68 bs1 num, connect(||.) symbol(iio) yline(-12.567) ylabel ti("...
generate lower90 = bs1 - 1.645*23.66127/sqrt(16)
```

To save your file, simply click on the "File" menu and select the "Save As..." option. Change the extension of your file from *.log* to *.do*. I now should have saved the file *cilab.do*.

Back in the command line, type **do cilab.do** and you will get the same information as when you originally did the lab. If the do-file produces an error, that means some command in your file is not correct. Get back into the doedit window and fix your file.

```
. doedit cilab.do
```

Another example of how this proves useful is that I can now edit my *cilab.do* file to automatically proceed with the assignment. I can change the few commands that need to be changed and rerun the do-file.