

STATA BASICS

Sonya DeMonner, MPH (July 2005)

Using directories to organize your work

The first thing to notice when you launch Stata is your current working directory. Stata displays this information in the lower left hand corner of the program window. Here's what you'll probably see:

C:\data

What you do not want to see is the official Stata directory **C:\Stata** (where Stata itself is installed). You don't ever want to work in (i.e. do data analysis) the official Stata directory or any subdirectory of it. If you do you can accidentally erase important files. Even if you never erase anything, you do not want to store your personal files there because the next time Stata is updated, files in these directories can be erased or copied over.

So which directory should you use when you run Stata? This is a matter of personal preference. I like organizing my work around projects, so I do not have a single directory for Stata. Instead I have one for each project.

For example, you might want to make a directory for the Summer Session. To do this you can use Stata's **mkdir** command.

mkdir c:\summer

and then you'll change into this directory. The term "cd" means change directory.

cd c:\summer

An advantage of changing into this directory for your Stata session is that you will be able to refer to files by just their file name. That is

use demographics

instead of by the entire path

use c:\summer\demographics

Once I have created the working directory I will be using for a project, I save all of my Stata files to that directory. I keep my datasets, log files and do files (programs) in this directory. This makes it much easier to find files when I need them, helps avoid having multiple versions of a file, and keeps individual projects or phases of projects separate.

Technical Note

Modern operating systems such as Windows XP and Macintosh allow embedded blanks in file and directory names. Stata can deal with blanks but you must enclose the name in double quotes. For instance, if you had a directory named "my projects" you might be tempted to type:

cd c:\my projects ←but that will not work. You'll get an invalid syntax message.

Instead, you must type

cd "c:\my projects"

The same is true when you want to use a dataset or run a program. Files names with blanks must be enclosed in quotes. So if your dataset is called *round one*, you'd need to type

use "round one"

Personally, I find it easiest just to avoid blanks in any file or directory names I am using in Stata. Often I'll use underscores instead of blanks so I'd name the above file *round_one*.

Using the do-file editor


Stata can be used interactively--that is you submit commands and capture output without saving any of the commands. This can come in handy when you just want to take a quick look at a dataset but it is extremely bad research practice when you need to do substantial data management or analysis. In these cases you will want to create a program (called a "do" file in Stata) which contains all of the necessary commands. This allows you to document your methods and replicate your analyses.

Even when working from a program, it is possible to work "interactively" with Stata. In order to do this you will keep a text editor window open while you work. Stata version 6.0 contains a built in text editor called the "do-file editor." It is accessed from within Stata by clicking on:



This is where you will write your commands and begin to assemble your program. Once you have your commands written you can submit them ("do" in Stata syntax) by clicking on:



Alternatively you can save the file (click the  icon) and then type

do round_one

in the command window.

In the results window you will see your commands and the output they have produced. If you want to change a command or add to the program simply type into the do-file editor and again submit the program. At the end of your session when you close the do-file editor it will prompt you to save your program if you haven't already.

Keeping a log file

In order to keep a copy of your output it is necessary to keep a log file. The log file is a simple text file. It will capture everything that appears in the Stata results window (all of your commands and Stata's responses). The use of a log file can be built into your program by placing the command

capture log using filename, replace

near the beginning of your program and

capture log close

at the end of your program.

Both of these commands begin with the optional "capture" command. Capture tells Stata to suppress any error messages that might arise from the command. The capture command is mainly important when you are working interactively and want your do-file to continue to run even if the command appears to be redundant. So if the log file is already open when we ask Stata to open it we want our program to keep going instead of giving us the "log file already open" error message.

The optional "replace" is used here because we don't want Stata to give an error message every time we write over the log file. In this case we just want to go ahead and write over the log file each time we run the program so that it always reflects the latest version of our program.

Technical Note

When working with LOG and DO files, there's a naming convention you'll want to adopt right away. Name your do files and the log files they create with the same name. That is, if your do-file is called

summer_stats.do

you'll want the log file it creates to be called

summer_stats.log

This will be enormously helpful when you share log files with colleagues and then need to be able to find the code that created the results.

Writing do-files

We'll cover do-files in more detail in the class exercises. Here are a couple of key points about the syntax of the Stata commands.

Stata distinguishes between equal signs which indicate equality and those which assign a value. To express equality in Stata double equal signs must be used ==. To assign value, a single = is used. For example:

replace patgrp10=1 if patgrp10==10

Not equal to is represented by ~= or !=. Greater than or equal to and less than or equal to are expressed with >= and <= as expected.

Missing values also deserve special attention in Stata. Missing numeric data is represented with a period ".". While this represents missing data, in some operations (notably sort and if, although not in statistical calculations such as means or correlations) a missing value in Stata is considered to have the largest possible positive value for that variable. So for example if you have missing values in your age variable and you want to create an indicator variable for people over age 65 you have to be careful or all of your missing values will be included as over age 65. Instead of:

replace over65=1 if age>=65

You'll want to say

replace over65=1 if age>=65 & age~.

Top 50 commands

The following fifty commands represent the core of Stata's data and file management capabilities. Most of what you will need to do to organize files and clean and manage your data can be accomplished with these commands. Additionally Stata includes almost 500 specific statistical commands which may be applied to your specific analysis plan.

Category	Stata Commands
Getting on-line help	search, help
Operating system interface	pwd, cd, sysdir, mkdir, dir, erase, copy, type
Using and saving data from disk	use, save, append, merge, compress
*Inputting data into Stata	input, edit, infile, infix, insheet
The Internet and Updating Stata	update, net, ado, news
Basic data reporting	describe, codebook, list, browse, count, inspect, summarize, table, tabulate
Data manipulation	generate, replace, egen, rename, drop, keep, sort, encode, decode, order, by, reshape
Formatting	format, label
Keeping track of your work	log, notes
Convenience	Display

* If StatTransfer is used these commands are rarely needed.

Portions from Stata Netcourse 101 Copyright StataCorp. and "Statistics with Stata 5.0" by Lawrence Hamilton.