

Your first Stata project

This guide will take you through the steps of starting a Stata .do file, importing data, and doing some analysis. Along the way, several useful functions will be introduced.

1. In your internet browser, go to <http://goo.gl/DUNcW>. This should bring you to a page that offers you the chance to download a file called "ACC Basketball Players.csv."
2. Download the file. My suggestion is to right-click on the link, choose the "Save Link As..." option from the context-menu, and save it to a folder you'll remember, perhaps even the Desktop.
3. Open Stata
4. From the menu, go to Window > Do-file Editor > New Do-file. Or, hit Ctrl-8.
 - a. This is a Do-file, in which you will type commands for Stata to execute.
5. I suggest starting all Do-files with a description of what the file does. Anything on a line that begins with an "*" character will not be read and executed by Stata. Thus, the asterisk can be called a comment character. You may want to begin your Do-file with something like:

```
* This file is my first Do-file, in which I load data and run a few analyses.
```

6. Next, I suggest you type the clear command, which will empty Stata's memory every time you run the Do-file from the beginning. You will also want to set memory to something larger (so Stata can handle more data and more complicated operations:

```
clear  
set mem 400m
```

7. Now, find the file path that points to where you saved "ACC Basketball Players.csv" on your hard drive. It might be something like "C:\Users\dbs9\Desktop\ACC Basketball Players.csv"
8. You load this into Stata with the command:

```
insheet using "C:\Users\dbs9\Desktop\ACC Basketball Players.csv", comma
```

9. Here, `insheet` is the command for loading data, and `comma` tells Stata that the file is a .csv, or comma-separated value file.
10. At this point, you'll want to run your Do-file. You can do this in many ways:
 - a. On the Do-file editor menu, go to Tools > Do
 - b. Select all of the code you've written (Ctrl-A), and then hit the keyboard shortcut for "do," Ctrl-D.
 - c. You can also run one line at a time. If your cursor is on a line of code, or selecting part of a line or several lines of code, you can "do" that selection alone.
 - d. For now, you'll want to run everything you've typed, through the `insheet` command.

11. Now, add a line with the command `list` to your Do-file and run that line alone. In the output window of Stata, you'll see a table with five columns and 75 observations (you may have to click "more" at the bottom of the output window, or hit space to advance to the end of the table).
 - a. As you can see, the data lists a subset of ACC Men's basketball players from the 2010-11 season, along with their team, height, weight, and class.
12. Now, add and Do the `describe` command in your Do-file.
13. The output here tells you the nature of the variables in memory. In this case, there are four string variables (as in, containing letters/words/phrases/text), and one integer variable, `wt`.
14. Now, we'll convert our `ht` variable, in a "ft-in" string format, to an integer variable indicating a height in inches.
15. There are several ways of doing this, but we're going to use a command called `split`, which will create new variables from the `ht` variable:

```
split ht, parse("-") gen(height)
```

16. Here, `split` is the command, `ht` is the variable to switch, "-" is the character at which the `ht` string will be split, and `height1` and `height2` are the new variables created in the process.
17. Add, and Do, a line in your Do-file that says:

```
list ht height1 height2
```

18. This will show the original height variable, plus the new variables you just generated with the original number of feet (`height1`), and the original number of inches (`height2`).
19. These new variables are still strings, as you can see in your Variables window, or with the `describe` command.
20. We need to convert them to another variable type, so we can add them together. One way to do this is with the `destring` command. You can either use two lines in your Do-file that say:

```
destring height1, replace  
destring height2, replace
```

Wherein `replace` tells Stata that you want to replace the variables you're changing, rather than generate new variables with new names. Alternatively, you can use a single line, that says:

```
destring height*, replace
```

Here, the asterisk serves as a wild-card character, telling Stata to run the `destring` command on any variable that matches the pattern of beginning with "height."

21. Now, we'll generate a new variable, which you will be doing a lot in Stata. This variable will convert our feet and inches variables to a total cumulative number of inches.

```
gen heightinches = height1 * 12 + height2
```

22. Where `gen` or `generate` is our command, `heightinches` is our new variable, which we define as equal to 12 times our measure of feet, plus extra inches. Here, of course, the asterisk is a multiplication operator.
23. Now we can do some analysis. First, let's calculate some summary statistics for heights.

```
sum heightinches
```

24. The `summary` command gives you a summary of any variables you list. Here, we are told that the mean height is 77.68 inches, or just under six-and-a-half feet.
25. Now, let's make a variable that indicates whether a player is greater than the average height.

```
gen overmean = 0  
replace overmean = 1 if heightinches > 77.68
```

26. First, we generated a new variable called `overmean` that consists entirely of zeroes.
27. Then, we replaced the zero with a one if and only if the `heightinches` variable is greater than the mean we saw in the summary.
28. Now, we can explore this new variable:

```
sum overmean  
table overmean
```

29. The mean we see in the summary is 0.52, which we can interpret to say that 52% of players in this file are greater than the mean. This is reinforced in the table of values shown, which indicates that 39 players are taller than the mean, and 36 are not.
30. How does this variable break down by class? We can show a cross-tab with the command:

```
table overmean yr
```

31. The output here shows the breakdown of those over the average height (and not) by class. In all cases, except for Juniors, there are more players above the average height than not.
32. Now, what is the distribution of weights? First, we can summarize player weights with the now-familiar `sum` command. Then, we can have Stata draw a histogram of weights, which gives us a more complete understanding of the distribution:

```
sum wt  
hist wt
```

33. As the histogram shows, the modal weight is around 200 pounds, and there is a longer tail to the upper end of the weight range.
34. Now for some science. Let's say you have a daring new hypothesis that taller players will weigh more, due to their larger mass. Let's explore this hypothesis in several ways.

35. First, have Stata show the average weight of players taller than the mean height, and not taller than the mean height:

```
mean wt, over(overmean)
```

36. As you can see, the mean weight of the taller group of players is greater than the mean weight of the shorter group (and the output suggests that the 95% confidence intervals of the mean estimates do not overlap.) Now, can we observe a linear relationship between height and weight? First, let's examine the data visually:

```
twoway scatter wt heightinches
```

37. The scatter plot puts height in inches on the x-axis as the independent variable, and weight on the y-axis as a response variable, because our hypothesis posits that height influences weight. There looks to be a clear and positive relationship between the two. How well do they correlate?

```
corr wt heightinches
```

38. Stata tells us that the correlation is positive and large, at about 0.80. In a similar vein, we can run a regression predicting weight from height:

```
regress wt heightinches
```

39. With the `regress` command, the response variable is always listed first, followed by any independent/explanatory variables. The output here tells us that our model explains about 64% of the variance in weight, and that every additional inch of height is associated with, on average, an increase in weight of 6.27 pounds.

40. There is much more that can be done in Stata; this tutorial has only scratched the surface. For further information, merely run the command "`help [command]`" where "`[command]`" is replaced by the function you want to know about, as in "`help regress.`" This will bring up Stata's on-line help, which details syntax, use, and examples.

41. I can also recommend several online quick references on Stata:
<http://stuff.mit.edu/afs/athena/course/17/17.842/www/statacheat.pdf>
<http://www-personal.umich.edu/~agrogan/stata/TwoPageStata.pdf>

42. Of course, if you run into a problem, Google almost certainly has an answer, if you are willing to look and make analogies from situations faced by others in the past.

Here is what your Do-file might look like at the end of this tutorial:

```
* This file is my first Do-file, in which I load data and run a few analyses.
```

```
clear
```

```
set mem 400m
insheet using "C:\Users\dbs9\Desktop\ACC Basketball Players.csv", comma
list
describe
split ht, parse("-") gen(height)
destring height*, replace
gen heightinches = height1 * 12 + height2
sum heightinches
gen overmean = 0
replace overmean = 1 if heightinches > 77.68
sum overmean
table overmean
table overmean yr
sum wt
hist wt
mean wt, over(overmean)
twoway scatter wt heightinches
corr wt heightinches
regress wt heightinches
```