PUBH 501 Biostatistics

STATA GRAPHS AND RANDOM SAMPLING

Tip of the day

- Copy and paste from the Stata results window using a fixed-width font
- Courier new is a common fixed width font. All characters have the same width, so table formatting is preserved

Pasted in calibri

age in three categories	Freq.	Percent	Cum.
age 14-24	120	63.49	63.49
age 25-29	42	22.22	85.71
age 30-45	27	14.29	100.00
Total	189	100.00	

Pasted in courier new

age in three categories	Freq.	Percent	Cum.
age 14-24	120	63.49	63.49
age 25-29	42	22.22	85.71
age 30-45	27	14.29	100.00
Total	189	100.00	

Reminder: Importing excel data

- Know your data first
  - Check variable names and formats
- Clear old data, either by the command –clear- or by including –clear- as an option in your import command
- Tell Stata if the first row is variables names using the –firstrow- option
- Use –list- to check that import went okay

```
import excel "computer path\LAB3LOSdata500", firstrow clear
list in 1/10  //list the first 10 observations for all variables
list in -10/l //list the last 10 observations for all variables
```

Graphs

Violin plot

- Combination of density and box plots
  - See distribution and percentiles
- Before we go over the command…we have to install this user-written command into Stata
- The command is –vioplot-, which you can find through googling “Stata violin plot”
- A user-written command is one that is not natively installed in Stata, but Stata allows them to be installed by users after the fact.
  - Often, really good or useful user-written commands are integrated into the next version of Stata

Installing a user-written command

- Option 1:
  - Type the command: `help vioplot`
  - The help file will link you to the command, and you can install it from the help file
- Option 2:
  - Install directly from the do file with the following code
  - `ssc install vioplot`
  - If –vioplot- is already installed, Stata will tell you this

Violin plot

- Command: -vioplot *variable*
- Options
  - Can stratify by another variable with an additional–over(*variable*)- option
  - Add a title with –title- “title” option

- `vioplot bwt`
- `vioplot bwt, over(smokepreg) title(“Birthweight by smoking status”)`

More graph options

- Options that work in most graph types, we'll look at a histogram
- Options bin(), width(), if *var condition*
  - bin(n) tells Stata how many bars you want in your histogram
  - width(n) tells Stata how wide each bar will be, in relation to the scale of your variable. Width(1) for age says each bar should be one year wide
  - If age<45 makes a histogram only including observations where age is less than 45. This goes *before* the comma as it is not an option but part of the original request
- `histogram age, freq bin(5)`
- `histogram age, freq width(1)`
- `histogram age if age<45, freq`

Naming graphs to combine then

- Add option, -name*(“name”)*- to the command of one or more graphs
- Use the –graph combine- command to display two graphs on one output
- Save or copy the graph
- Use the –graph drop- command to drop the names you created. You won’t be able to use those names again unless you use this command.

```
•vioplot bwt, name(g1)
•vioplot bwt, over(smokepreg) name(g2)
•graph combine g1 g2
•graph drop g1 g2
```

Creating a variable

Categorical age using –generate-

- Use –gen- (which is a shortened version of –generate- and either is acceptable) to create a new variable where all values =.
  - . denotes missing
- Use –replace- to fill in the missing values
- Use –if- to assign a value to the new variable based on values on an old variable
- Use the operators <, >, ==, >=, <=, !=
  - You must use == if you want to set the new value equal to the old value, not just a single =
  - != stands for does not equal

```
gen agecat=.
replace agecat=1 if age<25
replace agecat=2 if age<30 & age>=25
replace agecat=3 if age>=30
```

Check variable was created as intended

- Use the –tab- command to perform a two-way tabulation. This command is –tab- *var1 var2* where *var1* will be the variable in the rows, and *var2* will be the variables in the columns
- Use –bysort- and –summ- to summarize the old variable over levels of the new variable
- You are checking to make sure you assigned the categorical new variable *agecat* the values of the old variable *age* that you intended.

```
tab age agecat, m
bysort agecat: summ age
```

Labeling variables

- Give the variable a label with –label var-
- Tell Stata what values of your variable mean, called a value label, with –label def-
- Assign the value label you created to your new variable, with –label val-

```
label var agecat "age in three categories"
label def cat 1 "age 14-24" 2 "age 25-29" 3 "age 30-45"
label val agecat cat
```

Rename variables

- Use the –rename- command to change the name of your variable.
- The order is –rename- *oldname newname*
- You can change more than one name at a time by place all old names followed by all new names in parentheses
- `rename racenew race2`
- `rename (racenew hbphx bwt) (race2 hpb bweight)`

Random sampling

Get a random sample

- Use the command –sample- to draw a random sample
- Default is to sample a percent of the original data, denoted by the number after the –sample- command
- Draw a 50% sample of the original data with the following:
  - `sample 50`

- Can also draw a sample of a certain number instead of percent by adding the option –count-.
- Draw a sample of n=50 with the following
  - `sample 50, count`

Examine random sample

- Examine your sample by using the –tabstat- or –summ- commands
- `tabstat LOS, stats(mean median var)`
- `summ LOS, detail`

Repeat random sample

- Redraw your sample and you will get different observations and different summary statistics
- Stata approximates a random sample by using a complex algorithm. The algorithm starts at a unique value each time, resulting in a new sample each time.
- The starting point is called the seed. You can tell Stata where to start by using the command –set seed-
- Setting the seed allows you to get the same random sample each time you draw, which can be useful if you want to reproduce your results.
- The very first time Stata runs a random function (such as sample), it will use the seed 123456789. Therefore some of you may get the same results if you've never run a random function in Stata. The next seed will be randomly generated so this won't happen after the first time.
- `set seed 1298767`