L3: BIG STATA CONCEPTS

Getting started with Stata

Angela Ambroz

May 2015

Today

- Homework review and questions
- How do we work with **multiple datasets**?
- What is a **loop**? How do we loop?
- What are **locals**, and why are they helpful?
- Which commands can I use for some **inferential statistics**?

Review

- Homework solution video **will be up this weekend**: www.angelaambroz.com/stata.html
- Lecture 2's key messages:
  - Organizing a .do file
  - Importing data `(use file.dta, clear)`
  - Cleaning
  - Basic summary statistics `(tab q1)`
- Questions?

Big concepts in Stata

- First-level benefit of Stata: replicable .do files, full suite of statistical tests
- But the real power:
  1. Combining datasets (`merge, append`)
  2. Loops (`foreach`)
  3. Macros
- Understanding these concepts will make you work **faster** and **more reliably**

Multiple datasets

- Sometimes your data is spread out across 2+ files
- Stata has two commands for combining datasets:
  1. `append` – More people (going long)
  2. `merge` – Same people, more questions (going wide)

The character “Data” (an android) from Star Trek: The Next Generation.

The append command

- When you use append, you are **adding** a dataset to the bottom of another one

id	name	age	gender	urban_rural
001	Elvis	42	Male	urban
002	Youdi	19	Male	rural
003	Nellin	22	Female	rural

file1.dta

id	name	age	gender	urban_rural
675	Miriam	50	Female	urban
918	Angela	41	Female	rural

file2.dta

The append command

```
use file1.dta, clear

append using file2.dta
```

	id	name	age	gender	urban_rural
file1.dta	001	Elvis	42	Male	urban
	002	Youdi	19	Male	rural
	003	Nellin	22	Female	rural
file2.dta	675	Miriam	50	Female	urban
	918	Angela	41	Female	rural

The `merge` command

id	name	age	gender	urban_rural
001	Elvis	42	Male	urban
002	Youdi	19	Male	rural
003	Nellin	22	Female	rural

`file1.dta`

id	name	asset_quint	education
421	Angela	2	1
002	Youdi	3	2
110	Natalie	5	3

`file2.dta`

- When you use `merge`, you are **combining** datasets about the same people/households/observations.

The merge command

id	name	age	gender	urban_rural
001	Elvis	42	Male	urban
002	Youdi	19	Male	rural
003	Nellin	22	Female	rural

file1.dta

id	name	asset_quint	education
421	Angela	2	1
002	Youdi	3	2
110	Natalie	5	3

file2.dta

The merge command

id	name	age	gender	urban_rural
001	Elvis	42	Male	urban
002	Youdi	19	Male	rural
003	Nellin	22	Female	rural

file1.dta

id	name	asset_quint	education
421	Angela	2	1
002	Youdi	3	2
110	Natalie	5	3

file2.dta

The `merge` command

id	name	age	gender	urban_rural	asset_quint	education	_merge
001	Elvis	42	Male	urban	.	.	1
002	Youdi	19	Male	rural	3	2	3
003	Nellin	22	Female	rural	.	.	1
110	Natalie	.	.	.	5	3	2
421	Angela	.	.	.	2	1	2

`file1.dta` merged with `file4.dta`
(Note the gaps!)

The `merge` command

id	name	age	gender	urban_rural
001	Elvis	42	Male	urban
002	Youdi	19	Male	rural
003	Nellin	22	Female	rural

When you `merge`, you combine two datasets about the same people. You're adding more questions/variables.

The merge command

id	name	age	gender	urban_rural	asset_quint	education
001	Elvis	42	Male	urban	3	5
002	Youdi	19	Male	rural	1	3
003	Nellin	22	Female	rural	1	2

file1.dta file2.dta

```
use file1.dta, clear

merge 1:1 id using file2.dta
```

The `merge` command

id	name	age	gender	urban_rural	asset_quint	education
001	Elvis	42	Male	urban	3	5
002	Youdi	19	Male	rural	1	3
003	Nellin	22	Female	rural	1	2

To merge, you need to use a unique identifier. (You can check if a variable is a unique identifier by using the command `isid`.)

Here, `id` is a unique identifier.

The `merge` command

- GOOD identifiers: long, complicated numbers
- BAD identifiers: `strings` (e.g. names!)
- Always review your merge report:

```
. mer 1:1 uhn using szw-baseline.dta

    Result                           # of obs.
    -----------------------------------------
    not matched                         1,001
        from master                         0  (_merge==1)
        from using                      1,001  (_merge==2)

    matched                             1,399  (_merge==3)
    -----------------------------------------
```

The `merge` command

- GOOD identifiers: long, complicated numbers
- BAD identifiers: `strings` (e.g. names!)
- Always review your merge report:

_merge is an automatically-created variable that tells you which dataset each observation comes from

```
. mer 1:1 uhn using szw-baseline.dta

    Result                           # of obs.
    -----------------------------------------
    not matched                         1,001
        from master                         0   (_merge==1)
        from using                      1,001   (_merge==2)

    matched                             1,399   (_merge==3)
    -----------------------------------------
```

The `merge` command

id	name	age	gender	urban_rural	asset_quint	education	_merge
001	Elvis	42	Male	urban	.	.	1
002	Youdi	19	Male	rural	3	2	3
003	Nellin	22	Female	rural	.	.	1
110	Natalie	.	.	.	5	3	2
421	Angela	.	.	.	2	1	2

`file1.dta` merged with `file2.dta`
(Note the gaps!)

Big Stata concept #2: Loops

Dizzee Rascal, "I Don't Need a Reason" (2013)

This is a loop.

Loops

- A loop: Repeating the same action over and over… and over… again.
- ***This is what computers were made for.***

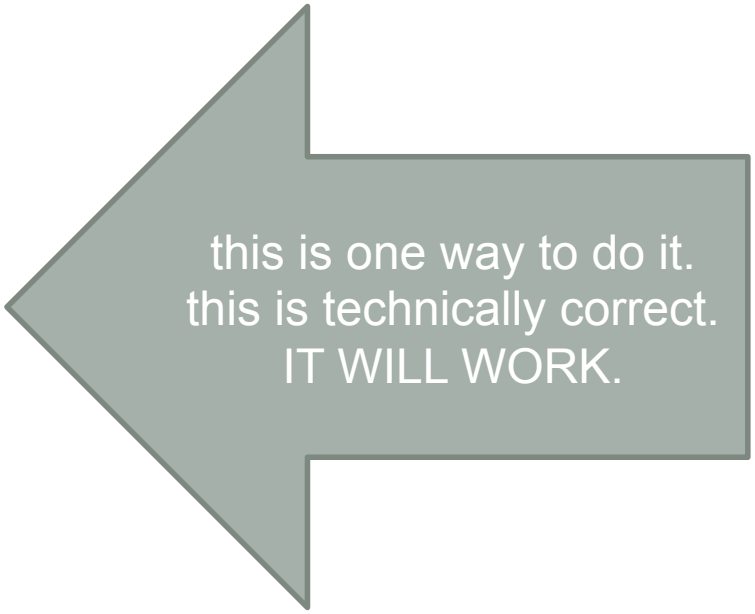
This is another loop.

The “mind blown” Internet GIF meme.

Loops: An example

- Suppose you have 10 variables (`q1, q2,...,q10`) that all need to be cleaned.
- You want to replace a value with `.` (*missing*) whenever you see `-888` (*no response* in the survey).

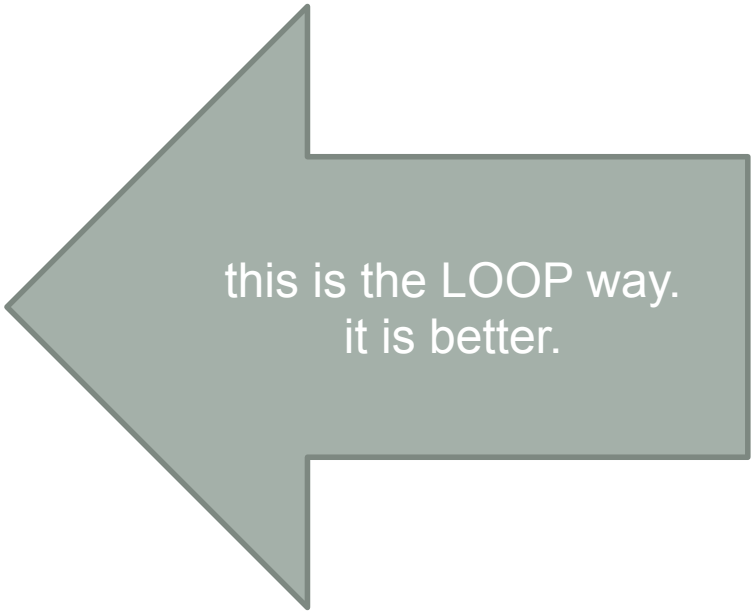
```
replace q1=. if q1==-888
replace q2=. if q2==-888
replace q3=. if q3==-888
replace q4=. if q4==-888
replace q5=. if q5==-888
replace q6=. if q6==-888
replace q7=. if q7==-888
replace q8=. if q8==-888
replace q9=. if q9==-888
replace q10=. if q10==-888
```



Loops: An example

- Suppose you have 10 variables (`q1, q2,...,q10`) that all need to be cleaned.
- You want to replace a value with `.` (*missing*) whenever you see `-888` (*no response* in the survey).

```
foreach i of varlist q1-q10 {
    replace `i'=. if `i'==-888
}
```



Dissecting the loop

```
replace q1=. if q1==-888
replace q2=. if q2==-888
replace q3=. if q3==-888
replace q4=. if q4==-888
replace q5=. if q5==-888
replace q6=. if q6==-888
replace q7=. if q7==-888
replace q8=. if q8==-888
replace q9=. if q9==-888
replace q10=. if q10==-888
```

=

```
foreach i of varlist q1-q10 {
    replace `i'=. if `i'==-888
}
```

- Are you copy+pasting? **Then you should use a loop.**
- Which part of the command repeats? Which changes?

Dissecting the loop

```
replace q1=. if q1==-888
replace q2=. if q2==-888
replace q3=. if q3==-888
replace q4=. if q4==-888
replace q5=. if q5==-888
replace q6=. if q6==-888
replace q7=. if q7==-888
replace q8=. if q8==-888
replace q9=. if q9==-888
replace q10=. if q10==-888
```

=

```
foreach i of varlist q1-q10 {
    replace `i'=. if `i'==-888
}
```

- Are you copy+pasting? **Then you should use a loop.**
- Which part of the command repeats? Which changes?

Dissecting the loop

```
replace q1=. if q1==-888
replace q2=. if q2==-888
replace q3=. if q3==-888
replace q4=. if q4==-888
replace q5=. if q5==-888
replace q6=. if q6==-888
replace q7=. if q7==-888
replace q8=. if q8==-888
replace q9=. if q9==-888
replace q10=. if q10==-888
```

```
foreach i of varlist q1-q10 {
    replace `i'=. if `i'==-888
}
```

- Tell Stata that you're starting a loop: `foreach`
- Tell Stata to use a placeholder: `i`
- Replace every placeholder (`` `i' ``) with a value from the variable list, q1 to q10 (`varlist q1-q10`)
- …and run the command on each value

Dissecting the loop

```
replace q1=. if q1==-888
replace q2=. if q2==-888
replace q3=. if q3==-888
replace q4=. if q4==-888
replace q5=. if q5==-888
replace q6=. if q6==-888
replace q7=. if q7==-888
replace q8=. if q8==-888
replace q9=. if q9==-888
replace q10=. if q10==-888
```

```
foreach i of varlist q1-q10 {
    replace `i'=. if `i'==-888
}
```

- Tell Stata that you're starting a loop: `foreach`
- Tell Stata to use a placeholder: `i`
- Replace every placeholder (`` `i' ``) with a value from the variable list, q1 to q10 (`varlist q1-q10`)
- …and run the command on each value

Dissecting the loop

```
replace q1=. if q1==-888
replace q2=. if q2==-888
replace q3=. if q3==-888
replace q4=. if q4==-888
replace q5=. if q5==-888
replace q6=. if q6==-888
replace q7=. if q7==-888
replace q8=. if q8==-888
replace q9=. if q9==-888
replace q10=. if q10==-888
```

```
foreach i of varlist q1-q10 {
    replace `i'=. if `i'==-888
}
```

- Tell Stata that you're starting a loop: `foreach`
- Tell Stata to use a placeholder: `i`
- Replace every placeholder (`` `i' ``) with a value from the variable list, q1 to q10 (`varlist q1-q10`)
- …and run the command on each value

Dissecting the loop

```
replace q1=. if q1==-888
replace q2=. if q2==-888
replace q3=. if q3==-888
replace q4=. if q4==-888
replace q5=. if q5==-888
replace q6=. if q6==-888
replace q7=. if q7==-888
replace q8=. if q8==-888
replace q9=. if q9==-888
replace q10=. if q10==-888
```

```
foreach i of varlist q1-q10 {
    replace `i'=. if `i'==-888
}
```

- Tell Stata that you're starting a loop: `foreach`
- Tell Stata to use a placeholder: `i`
- Replace every placeholder (`` `i' ``) with a value from the variable list, q1 to q10 (`varlist q1-q10`)
- …and run the command on each value

Dissecting the loop

```
replace q1=. if q1==-888
replace q2=. if q2==-888
replace q3=. if q3==-888
replace q4=. if q4==-888
replace q5=. if q5==-888
replace q6=. if q6==-888
replace q7=. if q7==-888
replace q8=. if q8==-888
replace q9=. if q9==-888
replace q10=. if q10==-888
```

```
foreach i of varlist q1-q10 {
    replace `i'=. if `i'==-888
}
```

- Tell Stata that you're starting a loop: `foreach`
- Tell Stata to use a placeholder: `i`
- Replace every placeholder (`` `i' ``) with a value from the variable list, q1 to q10 (`varlist q1-q10`)
- …and run the command on each value

You can loop almost anything

- Loop over the numbers from 1 to 100:

```
forval i=1/100
```

- Loop over a group of names:

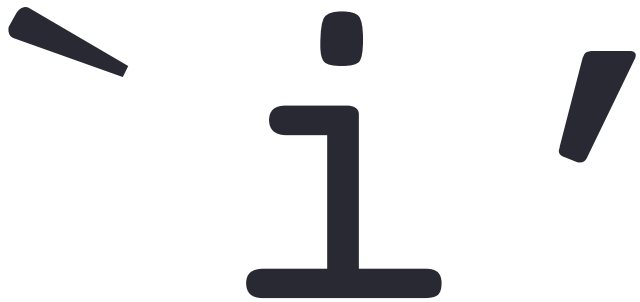
```
foreach i in “Angela” “Jean-Luc” “Ada”
```

- **NOTE:** You can call your placeholder anything you want (`i, j, variable, stuff, thingie, x`, etc…)
- **NOTE:** The syntax for loops is tricky, use `help foreach` a lot! (I still use `help foreach` every time I run a loop. And it’s been YEARS.)

Loop syntax: A note about quotes

- And please don't forget: when using your placeholder in your command, you need to use a weird quote mark (`) and a non-weird one (').

```
foreach i of varlist q1-q10 {
    replace `i'=. if `i'==-888
}
```



Loop syntax: A note about quotes

- And please don't forget: when using your placeholder in your command, you need to use a weird quote mark (`) and a non-weird one (').

```
foreach i of varlist q1-q10 {
    replace `i'=. if `i'==-888
}
```

Just one of those things.

Big Stata concept #3: Macros

- A convenient transition!
- This placeholder is actually our next big Stata concept
- Macros – AKA `locals` and `globals`

What is a macro?

- A macro is a placeholder for some text
- It only works in a `.do file` (not in the Command Editor)
- Rather than typing some text again and again, you save it as a macro
- Some example macros:

```
local   population_mean_age = 18

ttest sample_age = `population_mean_age'
```

What is a macro?

- A macro is a placeholder for some text
- It only works in a `.do file` (not in the Command Editor)
- Rather than typing some text again and again, you save it as a macro
- Some example macros:

```
global  my_folder  "C:/MyDocuments/Stata"

use "$my_folder/file1.dta", clear
```

What is a macro?

- A macro is a placeholder for some text
- It only works in a `.do file` (not in the Command Editor)
- Rather than typing some text again and again, you save it as a macro
- Some example macros:

```
local   control_vars   q1 q2 q3 q4

foreach var of local control_vars {
        rename `var' `var'_control
}
```

What is a macro?

- A macro is a placeholder for some text
- It only works in a `.do file` (not in the Command Editor)
- Rather than typing some text again and again, you save it as a macro
- Some example macros:

```
summarize resp_farmer

local avg_rural      `r(mean)'
```

Two types of macros

Two types of macros

GLOBALS

- Stata remembers them for as long as it's open
- Call them like `$this`

LOCALS

- Stata only remembers them for the duration of the .do file
- Call them like `` `this' ``

Stata has its own (hidden) macros

- With many commands, Stata generates a list of macros
- See what's saved via `return list` and `ereturn list`
- These are handy if you want to use them for something in a loop

```
. summ resp_farmer

    Variable |        Obs        Mean    Std. Dev.       Min        Max
-------------+---------------------------------------------------------
 resp_farmer |       2400       .6475    .4778482         0          1

. return list

scalars:
                  r(N) =  2400
              r(sum_w) =  2400
               r(mean) =  .6475
                r(Var) =  .2283388912046686
                 r(sd) =  .477848188449709
                r(min) =  0
                r(max) =  1
                r(sum) =  1554

.
```

Stata has its own (hidden) macros

- With many commands, Stata generates a list of macros
- See what's saved via `return list` and `ereturn list`
- These are handy if you want to use them for something in a loop

```
. summ resp_farmer

    Variable |        Obs        Mean    Std. Dev.       Min        Max
-------------+---------------------------------------------------------
 resp_farmer |       2400       .6475    .4778482         0          1

. return list

scalars:
                 r(N) =  2400
             r(sum_w) =  2400
              r(mean) =  .6475
               r(Var) =  .2283388912046686
                r(sd) =  .477848188449709
               r(min) =  0
               r(max) =  1
               r(sum) =  1554

.
```

Stata has its own (hidden) macros

```
. macro list
uwezo:          C:/Documents and Settings/aambroz/My
                Documents/TWAWEZA/Visualizations/Uwezo
baseline:       C:/Documents and Settings/aambroz/My Documents/TWAWEZA/P_Sauti
                za Wananchi/Comparison analysis
SZW:            //Htz-win-srv-01/Twaweza_Common_Files/Twa13/1. Programs/1.5
                Uwazi/1.5.1. Data/1.5.1.1 Sauti za Wananchi/2 - CATI rounds/2014
work:           C:/Documents and Settings/aambroz/My Documents/TWAWEZA/P_Sauti
                za Wananchi/Stata
S_level:        95
F1:             help advice;
F2:             describe;
F7:             save
F8:             use
S_ADO:          UPDATES;BASE;SITE;.;PERSONAL;PLUS;OLDPLACE
S_StataSE:      SE
S_FLAVOR:       Intercooled
S_OS:           Windows
S_MACH:         PC
```

How can I see which `globals` are active during this session of Stata? `macro list`

Inferential statistics

- This mini-course will **not** cover statistics
- Great resources: Khan Academy, Coursera, EdX, Udacity
- Many built-in commands for inferential statistics:
  - `pwcorr`
  - `ttest`
  - `anova`
  - `sampsi`
  - `sampclus`
  - `reg`
  - `logit`
- Use `help` to learn more

Some examples

Say you want to run a naïve OLS regression to see the effect of `x` on `y`:

```
reg y x
```

…with multiple control variables (`x1, x2, x3`):

```
reg y x1 x2 x3
```

…with fixed effects for regions (`region`):

```
xi: reg y x1 x2 x3 i.region OR
areg y x1 x2 x3, absorb(region)
```

…with clustering of standard errors at the school-level:

```
reg y x, cluster(school) OR
reg y x, vce(cluster school)
```

Homework

- Homework 3 – Merge some data, do some simple stats. Optional: check statistical significance with a t-test.
- Instructions are on the course website: www.angelaambroz.com/stata.html
- Stuck? E-mail/Skype.
- Finished? E-mail your completed `.do file` to me. **DEADLINE: Wednesday, 27 May 2015**