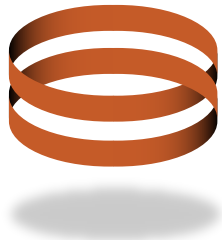
Data Management in Stata

Ista Zahn

Harvard MIT Data Center

April 5th 2013



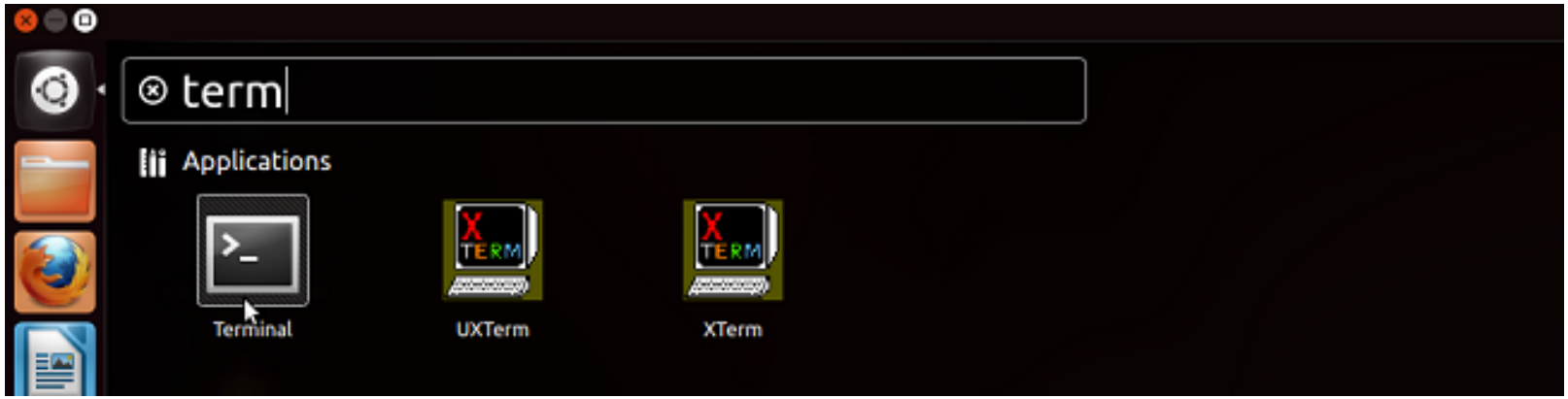
The Institute
for Quantitative Social Science
at Harvard University

Outline

Topic

Copy the workshop materials to your home directory

- Log in to an Athena workstation using your Athena user name and password
- Click on the "Ubuntu" button on the upper-left and type "term" as shown below



- Click on the "Terminal" icon as shown above
- In the terminal, type this line exactly as shown:

```
cd; wget http://tinyurl.com/statadatman-zip; unzip statadatman-zip
```

- If you see "ERROR 404: Not Found", then you mistyped the command – try again, making sure to type the command exactly as shown

Launch Stata on Athena

- To start Stata type these commands in the terminal:

```
add stata
xstata
```

- Open up today's Stata script
  - In Stata, go to Window => New do file => Open
  - Locate and open the `StatDatMan.do` script in the StataDatMan folder in your home directory
- I encourage you to add your own notes to this file!

Workshop Description

- This is an Introduction to data management in Stata
- Assumes basic knowledge of Stata
- Not appropriate for people already well familiar with Stata
- If you are catching on before the rest of the class, experiment with command features described in help files

Organization

- Please feel free to ask questions at any point if they are relevant to the current topic (or if you are lost!)
- There will be a Q&A after class for more specific, personalized questions
- Collaboration with your neighbors is encouraged
- If you are using a laptop, you will need to adjust paths accordingly

Opening Files in Stata

- Look at bottom left hand corner of Stata screen
  - This is the directory Stata is currently reading from
- Files are located in the StataDatMan folder on the Desktop
- Start by telling Stata where to look for these

```
// change directory
cd "C:/Users/dataclass/Desktop/StataDatMan"

// Use dir to see what is in the directory:
dir

// use the gss data set
use gss.dta
```

Topic

Logic Statements Useful Data Manipulation

==	equal to (status quo)
=	used in assigning values
!=	not equal to
>	greater than
>=	greater than or equal to
&	and
|	or

Basic Data Manipulation Commands

- Basic commands you'll use for generating new variables or recoding existing variables:
  - egen
  - replace
  - recode
- Many different means of accomplishing the same thing in Stata
  - Find what is comfortable (and easy) for you

Generate and Replace

- The gen command is often used with logic statements, as in this example:

```
// create "hapnew" variable
gen hapnew = . //set to missing
//set to 1 if happy equals 1
replace hapnew=1 if happy==1
//set to 1 if happy and hapmar = 3
replace hapnew=1 if happy>3 & hapmar>3
//set to 3 if happy or hapmar = 4
replace hapnew=3 if happy>4 | hapmar>4
tab hapnew // tabulate the new variable
```

Recode

- The `recode` command is basically generate and replace combined
- You can recode an existing variable OR use recode to create a new variable

```
// recode the wrkstat variable
recode wrkstat (1=8) (2=7) (3=6) (4=5) (5=4) (6=3) (7=2) (8=1)
// recode wrkstat into a new variable named wrkstat2
recode wrkstat (1=8), gen(wrkstat2)
// tabulate workstat
tab wrkstat
```

Basic Rules for Recode

Rule	Example	Meaning
#=#	3=1	3 recoded to 1
##=#	2. =9	2 and . recoded to 9
#/# = #	1/5=4	1 through 5 recoded to 4
nonmissing=#	nonmiss=8	nonmissing recoded to 8
missing=#	miss=9	missing recoded to 9

egen

- egen means "extension" to generate
- Contains a variety of more sophisticated functions
- Type "help egen" in Stata to get a complete list of functions
- Let's create a new variable that counts the number of "yes" responses on computer, email and internet use:

```
// count number of yes on comp email and interwebs
egen compuser= anycount(usecomp usemail usenet), values(1)
tab compuser
// assess how much missing data each participant has:
egen countmiss = rowmiss(age-wifeft)
codebook countmiss
// compare values on multiple variables
egen ftdiff=diff(wkft//)
codebook ftdiff
```

Topic

The "By" Command

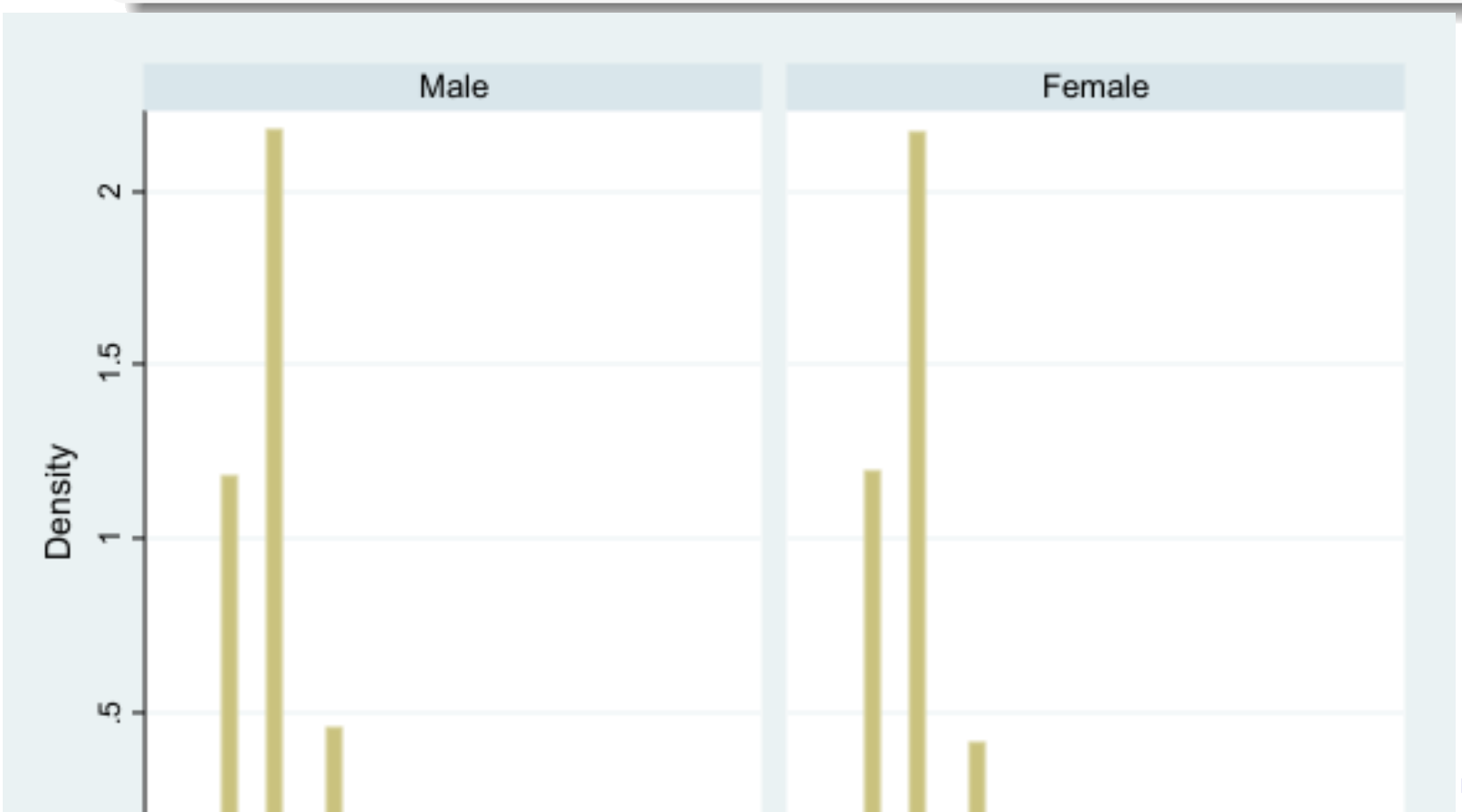
- Sometimes, you'd like to create variables based on different categories of a single variable
  - For example, say you want to look at happiness based on whether an individual is male or female
- The "by" prefix does just this:

```
// tabulate happy separately for male and female
bysort sex: tab happy
// generate summary statistics using bysort
bysort state: egen stateincome = mean(income)
bysort degree: egen degreeincome = mean(income)
bysort marital: egen marincomesd = sd(income)
```

The "By" Command

- Some commands won't work with by prefix, but have by options:

```
// generate separate histograms for female and male
hist happy, by(sex)
```



Topic

Missing Values

- Always need to consider how missing values are coded when recoding variables
- Stata's symbol for a missing value is "."
- Stata interprets "." as a large value
  - What are implications of this?

Missing Values

- If we want to generate a new variable that identifies highly educated women, we might use the command:

```
// generate and replace without considering missing values
gen hi_ed=0
replace hi_ed=1 if wifeduc>15
// What happens to our missing values when we
tab hi_ed, mi nola
// Instead, we might try:
drop hi_ed
// gen hi_ed, but don't set a value if wifeduc is missing
gen hi_ed = 0 if wifeduc != .
// only replace non-missing
replace hi_ed=1 if wifeduc >15 & wifeduc !=.
tab hi_ed, mi //check to see that missingness is preserved
```

- Note that you need the "mi" option to tab to view your missing data values

Missing Values

- What if you used a numeric value originally to code missing data (e.g., "999")?
- The mvdecode command will convert all these values to missing

```
mvdecode _all, mv(999)
```

- The "_all" command tells Stata to do this to all variables
- Use this command carefully!
  - If you have any variables where "999" is a legitimate value, Stata is going to recode it to missing
  - As an alternative, you could list var names separately rather than using "_all" command

Topic

Variable Types

- Stata uses two main types of variables: String and Numeric
- String variables are typically used for text variables
- To be able to perform any mathematical operations, your variables need to be in a numeric format

Variable Types

- Stata's numeric variable types:

```
type                 Minimum                 Maximum    being 0      bytes
-------------------------------------------------------------------------------
byte                    -127                     100    +/-1             1
int                  -32,767                  32,740    +/-1             2
long           -2,147,483,647         2,147,483,620    +/-1             4
float   -1.70141173319*10^38  1.70141173319*10^38      +/-10^-38        4
double -8.9884656743*10^307  8.9884656743*10^307       +/-10^-323       8
-------------------------------------------------------------------------------
Precision for float is 3.795x10^-8.
Precision for double is 1.414x10^-16.
```

Variable Types

- How can I deal with those annoying string variables?
- Sometimes you need to convert to/from string variables

```
// not run; generate "newvar" equal to numeric version of var2
destring var1, gen(newvar)
// not run; convert var1 to string.
tostring var1, gen(newvar)
```

Date and Time Variable Types

- Stata offers several options for date and time variables
- Generally, Stata will read date/time variables as strings
- You'll need to convert string variables in order to perform any mathematical operations
- Once data is in date/time form, Stata uses several symbols to identify these variables
  - %tc, %td, %tw, etc.

Variable Types: Date and Time

```
Format String-to-numeric conversion function
    -------+----------------------------------------
    %tc      clock(string, mask)
    %tC      Clock(string, mask)
    %td      date(string, mask)
    %tw      weekly(string, mask)
    %tm      monthly(string, mask)
    %tq      quarterly(string, mask)
    %th      halfyearly(string, mask)
    %ty      yearly(string, mask)
    %tg      no function necessary; read as numeric
    ------------------------------------------------
```

Variable Types: Date and Time

Format	Meaning	Value = -1	Value = 0	Value = 1
%tc	clock	31dec1959	01jan1960	01jan1960
		23:59:59.999	00:00:00.000	00:00:00.001
%td	days	31dec1959	01jan1960	02jan1960
%tw	weeks	1959w52	1960w1	1960w2
%tm	months	1959m12	1960m1	1960m2
%tq	quarters	1959q4	1960q1	1960q2
%th	half-years	1959h2	1960h1	1960h2
%tg	generic	-1	0	1

Variable Types: Date and Time

- To convert a string variable into date/time format, first select the date/time format you'll be using (e.g., %tc, %td, %tw, etc.)
- Let's say we create a string variable, today's date (today) that we want to format

```
// create string variable and convert to date
gen today = "Feb 18, 2011"
gen date1 = date(today, "MDY")
tab date1
// use the format command to change how the date is displayed
// format so humans can read the date
format date1 %d
tab date1
```

Variable Types

- What if you have a variable "time" formatted as DDMMYYYYhhmmss?

```
// Not run: conceptual example only
generate double time2 = clock(time, "DMYHMS")
tab time2
format time2 %tc
tab time2
```

- "double" command necessary for all clock formats
- basically tells Stata to allow a long string of characters

Exercise 1: Generate, Replace, Recode & Egen

Open the datafile, gss.dta.

1. Generate a new variable that represents the squared value of age.
2. Recode values "99" and "98" on the variable, "hrs1" as "missing."
3. Generate a new variable equal to "1" if income is greater than "19".
4. Recode the marital variable into a "string" variable and then back into a numeric variable.
5. Create a new variable that counts the number of times a respondent answered "don't know" in regard to the following variables: life, richwork, hapmar.
6. Create a new variable that counts the number of missing responses for each respondent.
7. Create a new variable that associates each individual with the average number of hours worked among individuals with matching educational degrees.

Topic

Merging Datasets

- Merge in Stata is for adding new variables from a second dataset to the dataset you're currently working with
  - Current active dataset = master dataset
  - Dataset you'd like to merge with master = using dataset
- If you want to add OBSERVATIONS, you'd use "append" (we'll go over that next)

Merging Datasets

- Several different ways that you might be interested in merging data
  - Two datasets with same participant pool, one row per participant (1:1)
  - A dataset with one participant per row with a dataset with multiple rows per participant (1:many or many:1)

Merging Datasets

- Stata will create a new variable ("$\backslash_{\text{merge}}$") that describes the source of the data
  - Use option, "nogenerate" if you don't want $\backslash_{\text{merge}}$ created
  - Use option, "generate(varname)" to give _merge your own variable name
- Need to add IDs to your dataset?

```
// create a variable "id" equal to the row number
generate id = _n
```

Merging Datasets

- Before you begin:
  - Identify the "ID" that you will use to merge your two datasets
  - Determine which variables you'd like to merge
  - In Stata >= 11, data does NOT have to be sorted
  - Variable types must match across datasets
  - Can use "force" option to get around this, but not recommended

Merging Datasets

- Let's say I want to perform a 1:1 merge using the dataset "data2" and the ID, "participant"

merge 1:1 participant using data2.dta

- Now, let's say that I have one dataset with individual students (master) and another dataset with information about the students' schools called "school"

```
// Not run: conceptual example only. Merge school and student data
merge m:1 schoolID using school.dta
```

Merging Datasets

- What if my school dataset was the master and my student dataset was the merging dataset?

```
// Not run: conceptual example only.
merge 1:m schoolID using student.dta
```

- It is also possible to do a many:many merge
  - Data needs to be sorted in both the master and using datasets

Merging Datasets

- Update and replace options:
  - In standard merge, the master dataset is the authority and WON'T CHANGE
  - What if your master dataset has missing data and some of those values are not missing in your using dataset?
  - Specify "update"- it will fill in missing without changing nonmissing
- What if you want data from your using dataset to overwrite that in your master?
  - Specify "replace update"- it will replace master data with using data UNLESS the value is missing in the using dataset

Appending Datasets

- Sometimes, you'll have observations in two different datasets, or you'd like to add observations to an existing dataset
- Append will simply add observations to the end of the observations in the master

```
// Not run: conceptual example. add rows of data from dataset2
append using dataset2
```

Appending Datasets

- Some options with Append:
  - generate(newvar) will create variable that identifies source of observation

```
// Not run: conceptual example.
append using dataset1, generate(observesource)
```

- "force" will allow for data type mismatches (again, this is not recommended)

Joinby

- Merge will add new observations from using that do not appear in master
- Sometimes, you need to add variables from using but want to be sure the list of participants in your master does not change

```
// Not run: conceptual exampe. Similar to merge but drops non-matches
joinby participant using dataset1
```

- Any observations in using that are NOT in master will be omitted

Topic

Collapse

- Collapse will take master data and create a new dataset of summary statistics
- Useful in hierarchal linear modeling if you'd like to create aggregate, summary statistics
- Can generate group summary data for many descriptive stats
  - Mean, media, sd, sum, min, max, percentiles, standard errors
- Can also attach weights

Collapse

- Before you collapse
  - Save your master dataset and then save it again under a new name
    - This will prevent collapse from writing over your original data
  - Consider issues of missing data. Do you want Stata to use all possible observations? If not:
    - cw (casewise) option will make casewise deletions

Collapse

- Let's say you have a dataset with patient information from multiple hospitals
- You want to generate mean levels of patient satisfaction for EACH hospital

```
// Not run: conceptual example. calculate average ptsatisfaction by hospital
save originaldata
collapse (mean) ptsatisfaction, by(hospital)
save hospitalcollapse
```

Collapse

- You could also generate different statistics for multiple variables

```
// create mean ptsatisfaction, median ptincome, sd ptsatisfaction for each h
collapse (mean) ptsatisfaction (median) ptincome (sd) ptsatisfaction, by(hos
```

- What if you want to rename your new variables in this process?

```
// Same as previous example, but rename variables
collapse (mean) ptsatmean=ptsatisfaction (median) ptincmed=ptincome
 (sd) sdptsat=ptsatisfaction, by(hospital)
```

Exercise 2: Merge, Append, and Joinby

Open the dataset, gss2.dta

1. The gss2 dataset contains only half of the variables that are in the complete gss dataset. Merge dataset gss1 with dataset gss2. The identification variable is "id."
2. Open the dataset, gss.dta
3. Merge in data from the "marital.dta" dataset, which includes income information grouped by individuals' marital status. The marital dataset contains collapsed data regarding average statistics of individuals based on their marital status.
4. Additional observations for the gssAppend.dta dataset can be found in "gssAddObserve.dta." Create a new dataset that combines the observations in gssAppend.dta with those in gssAddObserve.dta.
5. Create a new dataset that summarizes mean and standard deviation of income based on individuals' degree status ("degree"). In the process of creating this new dataset, rename your three new variables.

Topic

Help Us Make This Workshop Better

- Please take a moment to fill out a very short feedback form
- These workshops exist for you–tell us what you need!
- `http://tinyurl.com/StataDatManFeedback`

Additional resources

- training and consulting
  - IQSS workshops: `http://projects.iq.harvard.edu/rtc/filter_by/workshops`
  - IQSS statistical consulting: `http://rtc.iq.harvard.edu`
- Stata resources
  - UCLA website: `http://www.ats.ucla.edu/stat/Stata/`
  - Great for self-study
  - Links to resources
- Stata website: `http://www.stata.com/help.cgi?contents`
- Email list: `http://www.stata.com/statalist/`