Results Based Financing for Health Impact Evaluation Workshop

Tunis, Tunisia
October 2010

Stata 2

Willa Friedman

Outline of Presentation

- Importing data from other sources
- IDs
- Merging and Appending multiple datasets
- Difference in means and T-tests
- Graphs

Importing data from other sources

- Copy and paste
  - If your data is in Excel (or can be opened in Excel), one of the easiest ways to open it in Stata is just to copy and paste it.
  - Open the data (*data_day2.xls*) in Excel, select the part you want and copy it (Ctrl+C)
  - Then in Stata, type:

```
edit
```

  - and a new window will open.
  - Click in this window and paste the data (Ctrl+V)
  - Close this window. Did it work?

```
count
tab sex
```

Importing data from other sources (2)

- Many programs will let you save your data as a text file (.txt) or as comma-separated values (.csv)
- These can be opened in Stata using "insheet."

insheet ***filename.csv***, clear

```
insheet data_day2.csv, clear
```

- Is it there?

```
browse
```

Importing data from other sources (3)

- Hints:
  - You can also use a program called StatTransfer to convert files in other formats directly into Stata files (.dta)
  - Whatever method you choose, once it is in Stata, you can save it in Stata format and use that the next time you need it:
  - save **[data_file_name]**.dta, replace

IDs

- You will often want a unique identifier in your data. This is a number that is unique for each observation in your data.
- This identifier will become important in later steps and can be useful in checking for issues with the data. You can check for duplicates:

duplicates report ***id_1 id_2 …***

duplicates tag ***id_1 id_2 …*** , gen(***newvar***)

```
use data_day2a.dta, clear
duplicates report id household healthcenter
  district
duplicates tag pid household healthcenter
  district, generate(dup)
duplicates report after pid household
  healthcenter district
```

- Now we know that these three variables together uniquely identify observations in the dataset.

IDs (2)

- Sometimes you may need to create a single unique identifier
- Create any unique ID:

```
gen id=_n
```

  - (creates a new variable equal to the line number)
- Create an ID based on other information:
  - Sometimes each observation is uniquely identified by a combination of IDs. For example, each individual might have a household ID and an ID within the household. For example:

hh_id	person_id
1	1
1	2
2	1
2	2
2	3
3	1
3	2

IDs (3)

- Example: Creating a unique ID based on other IDs:

dist	health	house	pid	after	new_id
1	1	8	1	1	11811
1	1	8	2	1	11821
1	1	9	1	1	11911
1	1	9	2	1	11921
1	1	9	3	1	11931

```
egen new_id = concat(district healthcenter
  household pid after)
```

  - (creates a string variable that matches new_id)

```
gen new_id2= after+pid*10+household*1000
+healthcenter*100000+district*10000000
```

  - (creates a numeric variable that matches new_id)

IDs (4)

- Can be more complicated:
  - If IDs have different numbers of digits:

dist	health	house	pid	after	new_id_bad	new_id_good
1	1	8	1	1	11811	118011
1	1	8	12	1	118121	118121
1	1	81	2	1	118121	1181021

  - Why is the bad one bad?

```
duplicates report new_id
duplicates report new_id2
```

  - Can fix it by adding extra digits:

```
set type double
gen new_id3=hrbf_id1*10000+hrbf_id2*100+a1_pid
```

  - Always leave enough space for the largest number. If your ID will have more than 7 digits, include "set type double" before it.

IDs (5)

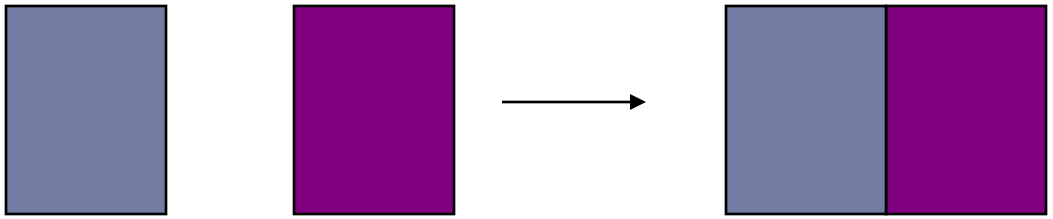
- Hints:
  - It is easier to work with IDs that all have the same number of digits. Rather than starting with 1, it can be useful (later on) to start with 11 or 101 or 1001 when assigning IDs.
  - The method above works only with numeric IDs. You can switch between numeric and strings:

    egen **newvar**=concatenate **numeric_variable(s)**
    - (This creates a new string variable that is equal to the string equivalent of the numeric variable)

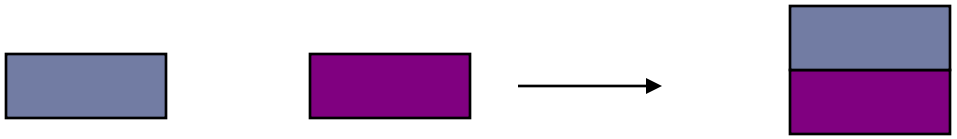
    destring **stringvar**, replace force
    - (This replaces a string variable with a numeric version. Any observations that contain non-numeric values are turned to missing. It may be helpful to save an old version of the string variable before you destring it, in case anything goes wrong)

Merging and Appending

- Sometimes you will have two datasets that you want to combine. This can be done in two ways:
  - Merging: This is done if you want to add more variables for the same observations.
    - Eg: You have surveyed the same individuals a second time and want to combine their answers from both surveys.
    - Graphically:



  - Appending: This is done to add more observations with the same variables.
    - Eg: You have used the same survey with a new group of individuals and you want to add the new group to the original sample
    - Graphically:



Appending

- Make sure that all variables are in the same format (string or numeric) in both datasets or values can be deleted. Useful commands:

destring ***varname***, replace force

gen ***new_string_var***=concatenate(***old_numeric_var***)

- Hints:
  - Before appending, you may want to create a new variable to identify which file an observation came from. Eg:

```
gen surveyround=1
append using data_day2b.dta
replace surveyround=2 if surveyround==.
count
```

Merging

- Each file must be sorted by the same unique identifier or by the same set of variables that jointly uniquely identify each observation. Syntax:

  sort ***id1 id2 …***

  save ***file_1.dta***, replace

  use ***file_2.dta***, clear

  sort ***id1 id2 …***

  merge ***id1 id2 …*** using ***file_1.dta***, _merge(***new_name***)

  Note: In new versions of Stata, the syntax is:

  merge 1:1 ***id1 id2 …*** using ***file_1.dta***, gen(***newname***)

Merging (2)

- Try it:

```
sort district healthcenter household pid after
save to_merge.dta, replace
use data_day2c.dta, clear
sort district healthcenter household pid after
merge district healthcenter household pid
after using to_merge.dta, _merge(merge1)
*OR:
merge 1:1 district healthcenter household pid
after using to_merge.dta, gen(merge1)
tab merge1
```

Merging (3)

- Hints:
  - If you have variables with the same names in both files (besides the identifiers), one may be deleted when merged. Don't do this without looking into the options that go with merge.
  - If you have already created a unique_id in each file, using a method that will be consistent between files, you can merge on this id, rather than a set of variables.
    - Be very careful that the method used to generate IDs is the same for each dataset. When in doubt, it might be better to use the set of variables.

Merging (one to many)

- Sometimes you want to match multiple observations in one dataset with a single observation in another dataset.
  - Eg: One dataset has information on many children and the other has information on mothers
  - This can be done in the same way, merging on the variable that is unique in one dataset and repeated in the other. For example:

```
use mothers.dta
sort mother_id
save mothers.dta, replace
use child.dta, clear
sort mother_id {note that this will not be unique in this dataset}
merge mother_id using mothers.dta, replace
```

  - This will add the mother's information to each of her children.

Difference in means

- The first step in analysis of a good randomized trial is just to compare the mean outcomes in the treatment and control groups:
- tab ***category***, sum(***outcome)***

  tab treat if after==1, sum(prev)

```
. tab treat if after==1, sum(prev)

           |           Summary of prev
     treat |        Mean   Std. Dev.       Freq.
-----------+------------------------------------
         0 |   .11598952   .32031732        1526
         1 |   .17849613   .38306446        1423
-----------+------------------------------------
     Total |   .14615124   .35331768        2949
```

Difference in means (2)

- Test the significance of the difference in means by treatment group:
  - ttest ***outcome***, by(***treatment***)

ttest prev if after==1, by(treat)

```
Two-sample t test with equal variances
------------------------------------------------------------------------------
   Group |     Obs        Mean    Std. Err.   Std. Dev.   [95% Conf. Interval]
---------+--------------------------------------------------------------------
       0 |    1526    .1159895    .0081998    .3203173    .0999054    .1320736
       1 |    1423    .1784961    .0101548    .3830645    .1585762     .198416
---------+--------------------------------------------------------------------
combined |    2949    .1461512    .0065062    .3533177    .1333941    .1589084
---------+--------------------------------------------------------------------
    diff |           -.0625066    .0129716               -.0879409   -.0370724
------------------------------------------------------------------------------
    diff = mean(0) - mean(1)                                      t =  -4.8187
Ho: diff = 0                                     degrees of freedom =     2947

    Ha: diff < 0                 Ha: diff != 0                 Ha: diff > 0
 Pr(T < t) = 0.0000         Pr(|T| > |t|) = 0.0000          Pr(T > t) = 1.0000
```

Difference in means (3)

- The command ttest can be used to test the significance of other differences. These next two are useful for comparing outcomes before and after:
- Test whether a mean is significantly different from a value
  - ttest ***varname*** == #
  - eg: ttest **change_malaria**==0
- OR: Test whether two variables are significantly different from each other:
  - ttest ***varname1*** ==***varname2***
  - eg: ttest ***malaria_before*** == ***malaria_after***

Graphing Basics

- Scatter plot - relationship between two variables
  - graph twoway scatter *dependent independent*
  - eg: graph twoway scatter birthweight gest_age
  - (graphing works best when both variables are continuous)

- Histogram - general distribution
  - hist *variable*
  - eg: hist malaria

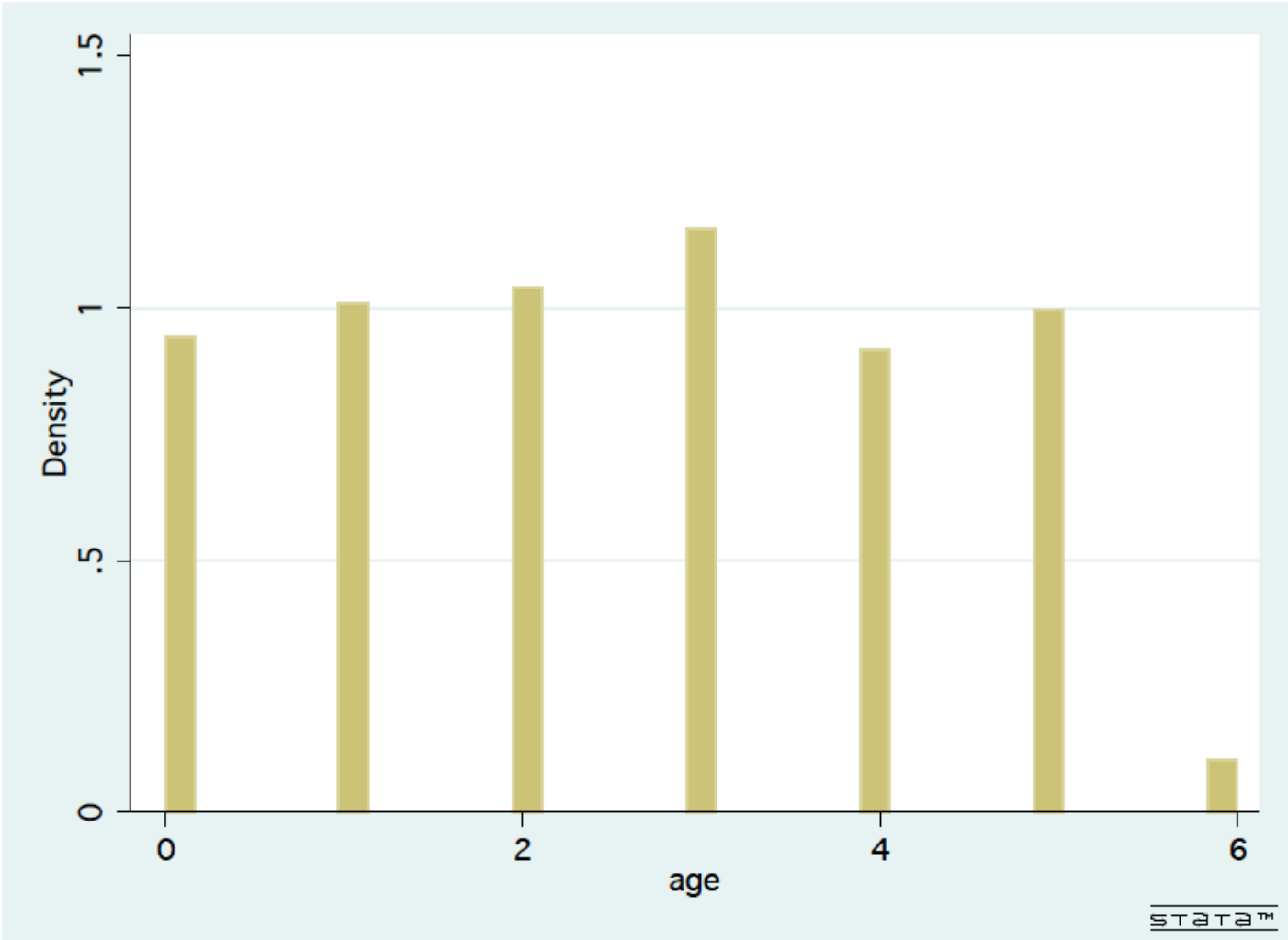
Graphing Basics (2)

Example: Let's look at the relationship between age and days sick in the last 4 weeks:

- First we will look individually at each variable and then the two together

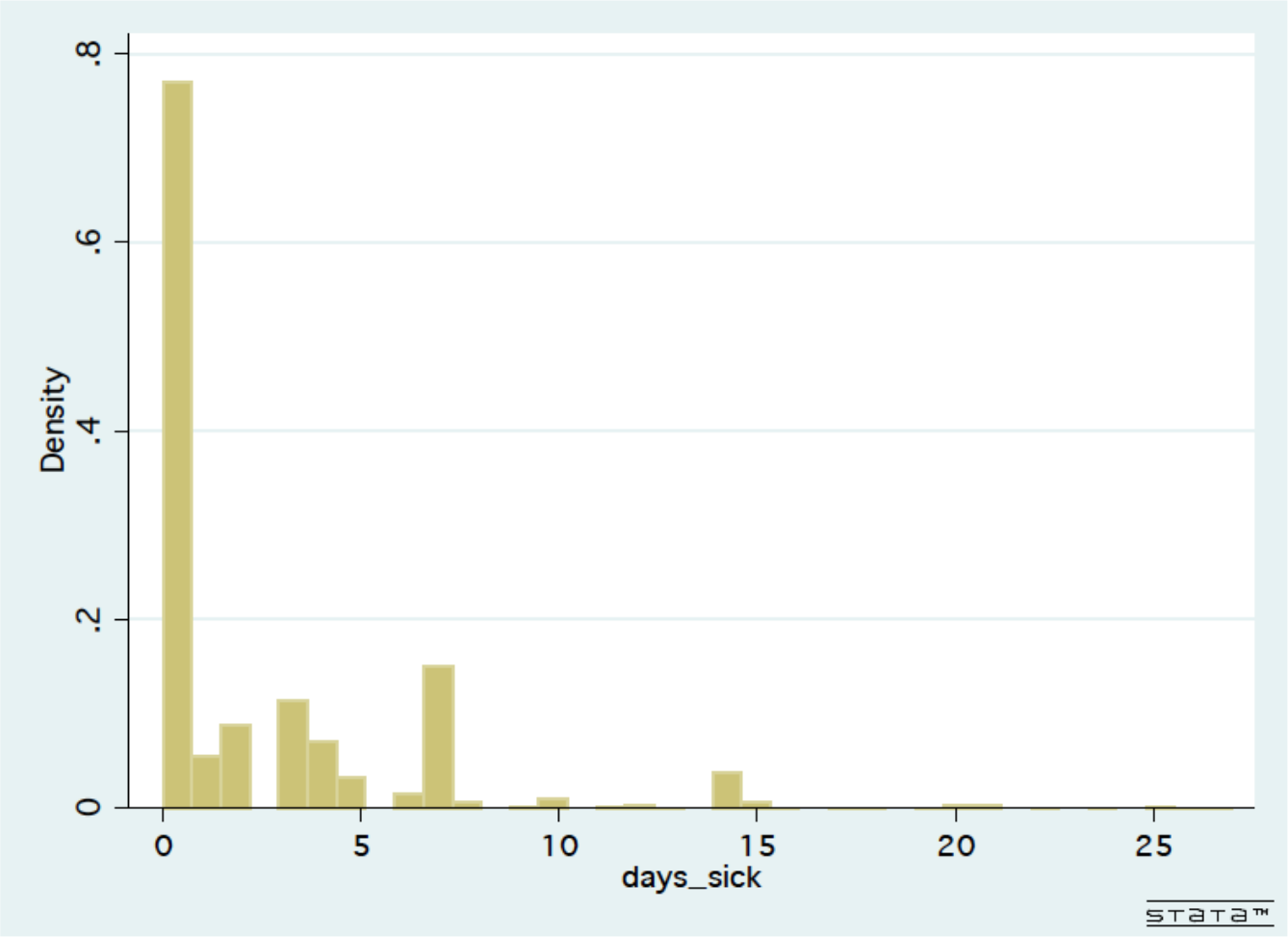
Graphing basics(3)

hist age



Graphing basics (4)

hist days_sick



Graphing basics (5)

graph twoway scatter days_sick age