Longitudinal Data Analysis: Stata Tutorial

Part A: Overview of Stata

I. Reading Data:

- use
  Read data that have been saved in Stata format.
- infile
  Read raw data and "dictionary" files.
- insheet
  Read spreadsheets saved as "CSV" files from a package such as Excel.

II. Do Files

- **What is a do file?**
  A "do" file is a set of commands just as you would type them in one-by-one during a regular Stata session. Any command you use in Stata can be part of a do file. Do files are very useful, particularly when you have many commands to issue repeatedly, or to reproduce results with minor or no changes.

  Example: cd4-readin.do

```
* Read in cd4.raw data and create stata data set

log using cd4-readin , replace
set memory 40m
infile time cd4 age packs drugs sexpart cesd id using cd4

gen timedays = round(time*365.25,1)
compress

label var id      "subject ID"
label var time    "years since seroconversion"
label var timedays      "days since seroconversion"
label var cd4     "CD4 Count"
label var age     "age (yrs) relative to arbitrary origin"
label var packs   "packs of cigarettes smoked per day"
label var drugs   "recreational drug use yes/no"
label var sexpart "number of sexual partners"
label var cesd    "depression score relative to arbitrary origin"

save cd4 , replace

clear
log close
```

  You can edit a do file anywhere then save as a file with the extension ".do". In Windows or Mac, you can type *doedit* in Stata to edit any do files.

- **Where to put a do file?**
  Put the do file in the working directory of Stata.
- **How to run a do file?**
  do mydofile
  Example: do cd4-readin

III. Ado files

- **What is an ado file?**
  An ado file is just a Stata program. You can use it as a command.
  A *.ado file usually contains a program called * in it.
  For example, the first non-comment line "autocor.ado" is
  ```
  program define autocor
  ```

- **Where to put an ado file?**
  Put them in your current directory, in your stata "ado" directory, or in a directory where Stata will know where to look for them.
  Use "**adopath**" to find out where Stata is looking for ado files.
  Here is an example in a Windows PC (Ado directory may be different among different platforms).
  ```
  . adopath
    [1]  (UPDATES)   "C:\STATA\ado\updates/"
    [2]  (BASE)      "C:\STATA\ado\base/"
    [3]  (SITE)      "C:\STATA\ado\site/"
    [4]              "."
    [5]  (PERSONAL)  "c:\ado\personal/"
    [6]  (STBPLUS)   "c:\ado\stbplus/"
    [7]  (OLDPLACE)  "c:\ado/"
  ```

- **How to run an ado file?**
  Use the name of the program as a command as you use other default Stata commands.
  For example:
  ```
  . autocor cd4res timeyrs id
  ```

IV. Convert data from wide to long or vice versa

- **Two forms of data: wide and long**
  Different models may require different forms of data in Stata. For instance, "logit" or "logistic" model in Stata prefers a wide format.

(wide form)

id	sex	inc80	inc81	inc82
1	0	5000	5500	6000
2	1	2000	2200	3300
3	0	3000	2000	1000

(long form)

id	year	sex	inc
1	80	0	5000
1	81	0	5500
1	82	0	6000
2	80	1	2000
2	81	1	2200
2	82	1	3300
3	80	0	3000
3	81	0	2000
3	82	0	1000

- **reshape converts data from one form to the other:**
  - **From Wide to Long**

```
. reshape long inc, i(id) j(year)
```

  - **From Long to Wide**

```
. reshape wide inc, i(id) j(year)
```

- **Examples: Cows Data**

```
. infile prot1-prot19 using cowslupins
. gen id = _n
. order id
. list in 1/2
Observation 1
          id            1       prot1         3.69       prot2         3.38
       prot3            3       prot4          3.5       prot5         3.09
       prot6          3.3       prot7         3.07       prot8         3.22
       prot9         2.97      prot10          3.6      prot11         3.42
      prot12         3.59      prot13         3.77      prot14         3.74
      prot15          3.7      prot16         3.78      prot17         3.78
      prot18         3.77      prot19         3.53
Observation 2
          id            2       prot1          4.2       prot2         3.35
       prot3         3.37       prot4         3.07       prot5         2.82
       prot6         3.05       prot7         3.12       prot8         2.85
       prot9          3.2      prot10         3.38      prot11         3.25
      prot12         3.26      prot13          3.3      prot14         3.17
      prot15          3.4      prot16         3.41      prot17         3.28
      prot18         3.42      prot19         3.25
. reshape long prot , i(id) j(week)
(note:  j = 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19)
Data                               wide   ->  long
-----------------------------------------------------------------------------
Number of obs.                      513   ->      27
Number of variables                   3   ->      20
j variable (19 values)             week   ->   (dropped)
xij variables:
                                   prot   ->   prot1 prot2 ... prot19
-----------------------------------------------------------------------------
. list in 1/20
           id       week       prot
```

```
  1.         1          1       3.69
  2.         1          2       3.38
  3.         1          3          3
  4.         1          4        3.5
  5.         1          5       3.09
  6.         1          6        3.3
  7.         1          7       3.07
  8.         1          8       3.22
  9.         1          9       2.97
 10.         1         10        3.6
 11.         1         11       3.42
 12.         1         12       3.59
 13.         1         13       3.77
 14.         1         14       3.74
 15.         1         15        3.7
 16.         1         16       3.78
 17.         1         17       3.78
 18.         1         18       3.77
 19.         1         19       3.53
 20.         2          1        4.2

. reshape wide prot, i(id) j(week)
(note:  j = 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19)

Data                               wide   ->  long
-----------------------------------------------------------------------------
Number of obs.                      513   ->      27
Number of variables                   3   ->      20
j variable (19 values)             week   ->   (dropped)
xij variables:
                                   prot   ->   prot1 prot2 ... prot19
-----------------------------------------------------------------------------
. list in 1/2
Observation 1
          id            1       prot1         3.69       prot2         3.38
       prot3            3       prot4          3.5       prot5         3.09
       prot6          3.3       prot7         3.07       prot8         3.22
       prot9         2.97      prot10          3.6      prot11         3.42
      prot12         3.59      prot13         3.77      prot14         3.74
      prot15          3.7      prot16         3.78      prot17         3.78
      prot18         3.77      prot19         3.53
Observation 2
          id            2       prot1          4.2       prot2         3.35
       prot3         3.37       prot4         3.07       prot5         2.82
       prot6         3.05       prot7         3.12       prot8         2.85
       prot9          3.2      prot10         3.38      prot11         3.25
      prot12         3.26      prot13          3.3      prot14         3.17
      prot15          3.4      prot16         3.41      prot17         3.28
      prot18         3.42      prot19         3.25
```

Part B: Longitudinal data analysis in Stata

I. Convert an ordinary dataset into a longitudinal dataset (cross-sectional time-series data): use tsset vs. iis, tis

- "tsset" declares ordinary data to be time-series data,
  - Simple time-series data: one panel
  - Cross-sectional time-series data: multi-panel
    - Each observation in a cross-sectional time-series (xt) dataset is an observation on x for unit i (panel) at time t.
- For this course, we use cross-sectional time-series data.
- Syntax for "tsset" for cross-sectional time-series data:

```
. tsset panel timevar
```

Example:

```
. infile time cd4 age packs drugs sexpart cesd id using cd4
(2376 observations read)
. iis
(i() has not been defined)
. tis
(t() has not been defined)
. tsset id time
time variable must contain only integer values
r(451);
. list time in 1/10
          time
  1.  -.741958
  2.  -.246407
  3.   .243669
  4. -2.729637
  5. -2.250513
  6.  -.221766
  7.   .221766
  8.   .774812
  9.  1.256673
 10. -1.240246

. gen timedays=round(time*365.25,1)

. list time timedays in 1/10

          time   timedays
  1.  -.741958       -271
  2.  -.246407        -90
  3.   .243669         89
  4. -2.729637       -997
  5. -2.250513       -822
  6.  -.221766        -81
  7.   .221766         81
  8.   .774812        283
```

```
  9.  1.256673        459
 10. -1.240246       -453

. tsset id timedays
       panel variable:  id, 10002 to 41844
        time variable:  timedays, -1092 to 1994, but with gaps

. iis
i() is id

. tis
t() is timedays
```

- Alternative Way: iis & tis

```
iis id
tis timedays
```

- Some commands require tsset (built-in xt commands), others require iis and tis. For this course, mostly we are using iis and tis.

II. xt commands

The xt series of commands provide tools for analyzing cross-sectional time-series (panel) datasets:

- xtdes Describe pattern of xt data

Example: Cows data

```
. use cows
. keep if (diet=="barley")
. drop if (prot==.)

. xtdes, patterns(0)

      id:  1, 2, ..., 25                                     n =         25
    week:  1, 2, ..., 19                                     T =         19
           Delta(week) = 1; (19-1)+1 = 19
           (id*week uniquely identifies each observation)

Distribution of T_i:   min      5%     25%       50%       75%     95%     max
                        12      14      15        18        19      19      19
. xtdes, patterns(5)

      id:  1, 2, ..., 25                                     n =         25
    week:  1, 2, ..., 19                                     T =         19
           Delta(week) = 1; (19-1)+1 = 19
           (id*week uniquely identifies each observation)

Distribution of T_i:   min      5%     25%       50%       75%     95%     max
                        12      14      15        18        19      19      19

     Freq.  Percent    Cum. |  Pattern
```

```
 --------------------------+---------------------
       11     44.00   44.00 |  11111111111111111
        5     20.00   64.00 |  11111111111111.....
        2      8.00   72.00 |  11111111111111111.
        2      8.00   80.00 |  111111111111111....
        2      8.00   88.00 |  111111111111111...
        3     12.00  100.00 | (other patterns)
 --------------------------+---------------------
       25    100.00         |  XXXXXXXXXXXXXXXXXXX


. xtdes   //default number of patterns is 9

      id:  1, 2, ..., 25                                      n =         25
    week:  1, 2, ..., 19                                      T =         19
           Delta(week) = 1; (19-1)+1 = 19
           (id*week uniquely identifies each observation)

Distribution of T_i:   min      5%     25%       50%       75%     95%     max
                        12      14      15        18        19      19      19

     Freq.  Percent    Cum. |  Pattern
 --------------------------+---------------------
       11     44.00   44.00 |  11111111111111111
        5     20.00   64.00 |  11111111111111.....
        2      8.00   72.00 |  111111111111111...
        2      8.00   80.00 |  11111111111111111.
        2      8.00   88.00 |  111111111111111....
        1      4.00   92.00 |  11111111.1.111.....
        1      4.00   96.00 |  1.11111111111111111
        1      4.00  100.00 |  11111111.1111111111
 --------------------------+---------------------
       25    100.00         |  XXXXXXXXXXXXXXXXXXX
```

Other xt commands:

- xtsum Summarize xt data
  Paul has a improved version: xtsumcorr.

- xttab Tabulate xt data
- xtreg Fixed-, between- and random-effects, and population-averaged linear models
- xtdata Faster specification searches with xt data
- xtlogit Fixed-effects, random-effects, & population-averaged logit models
- xtprobit Random-effects and population-averaged probit models
- xttobit Random-effects tobit models
- xtpois Fixed-effects, random-effects, & population-averaged Poisson models
- xtnbreg Fixed-effects, random-effects, & population-averaged negative binomial models
- xtclog Random-effects and population-averaged cloglog models

- xtintreg Random-effects interval data regression models
- xtrchh Hildreth-Houck random coefficients models
- xtgls Panel-data models using GLS
- xtgee Population-averaged panel-data models using GEE

Look "help xt" in Stata

III. Graphs for longitudinal data

- **xtgraph**
  A new command for summary graphs of xt data (cross-sectional time series data).
  Download the xtgraph.ado file from course website.
  Syntax:

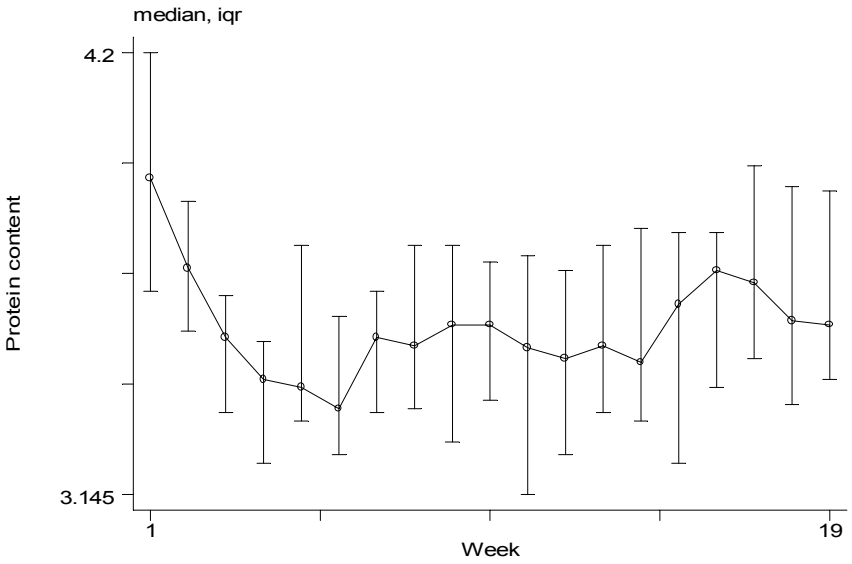
```
xtgraph varname [if] [in] , group(groupvar) av(avtype) bar(bartype)
graph options xt options
```

  Choice of average
  xtgraph , av(avtype)
  The average types are
  - am - arithmetic mean, the default
  - gm - geometric mean
  - hm - harmonic mean
  - median - only with bars ci - default, iqr or rr.

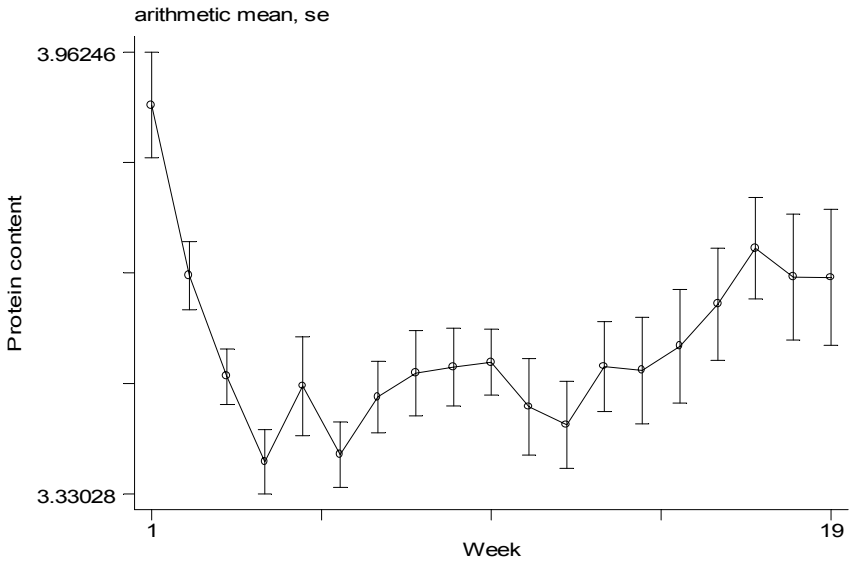
  Choice of error bars
  xtgraph , bar(bar type)
  level(significance level)
  The bar types are
  - ci - the default, significance set by level()
  - se - standard error
  - sd - standard deviation
  - rr - reference range, level set by level()
  - iqr -same as bar(rr) level(50)
  - no - no bars

Examples:

```
. xtgraph prot, av(median) bar(iqr) t1("median, iqr")
```




```
. xtgraph prot, av(am) bar(se) t1("arithmetic mean, se")
```



Refer to xtgraph.pdf or xtgraph.hlp for help.

- **How to graph trajectories**

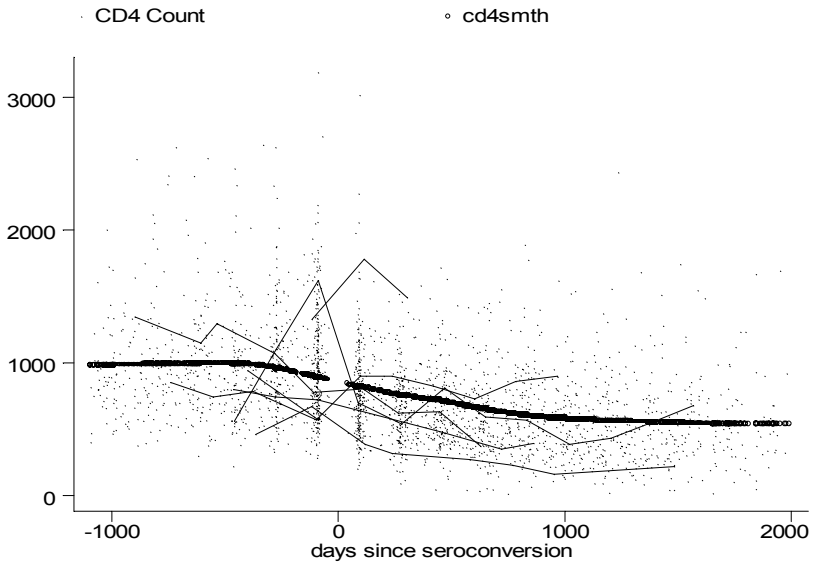
In the lectures notes, Paul gave an example to draw trajectories using subjects picked based on ranking of within-subject statistics (the difference in the medians before and after HIV seroconversion).

Other examples:

- A random set (trajectory1.do)

```
*trajectory.do file for Stata 6.0
clear
use cd4
egen newid=group(id)
sum newid
drop id
ren newid id
sort id timedays
gen pick = 0
local i=1
while `i' < 8{
  set seed `i'
  local r = round(1+uniform()*369,1)
  gen cd4l`i' = cd4 if (id == `r')
  local i=`i'+1
}

ksm cd4 timedays, lowess gen(cd4smth) nograph
graph cd4 cd4l1-cd4l7 cd4smth timedays, c(.LLLLLLL.) s(.iiiiiiio)
pen(233333334) xlab ylab
```



- **Ranking with the individual mean CD4 counts (trajectory2.do)**

```
*trajectory.do file for Stata 6.0
clear
use cd4
egen newid=group(id)
sum newid
drop id
ren newid id
egen cd4mean = mean(cd4), by(id)
list id cd4 cd4mean in 1/10
sort id
quietly by id: replace cd4mean=. if (_n > 1)
egen rnk=rank(cd4mean)
local i = 1
while `i' <= 7{
  gen sub`i' =(rnk == `i'*25)
  sort id timedays
  quietly by id: replace sub`i'=sub`i'[1]
  gen cd4l`i' = cd4 if (sub`i')
  drop sub`i'
  local i=`i'+1
}
ksm cd4 timedays, lowess gen(cd4smth) nograph
graph cd4 cd4l1-cd4l7 cd4smth timedays, c(.LLLLLLL.) s(.iiiiiiio)
pen(233333334) xlab ylab
```