Stata Programing Introduction for EE426 2/2014

Typical structure of a .do file:

```
cd "…directory…"

capture log close
clear
set more off
log using filename.log, replace

use datafilename, clear
.
.
.
log close
```

capture log close: close any opening log file, either there is one opening or none.
→ -capture- in front of a command line tells Stata to ignore the line if there is an error associated with it.
set more off: let the whole program runs without stop
log: save results window (output). It can be saved into 2 formats: log file or smcl file

```
*display a description of the data set
describe

*display summary statistics
summarize

*display detailed summary statistics of select variables
summarize varlist, detail

*display correlation matrix of select variables
corr varlist

*display covariance matrix of select variables
corr varlist, cov

*display frequency breakdown of a "sufficiently discrete" variable
tab varname

*display cross-tabulation two "sufficiently discrete" variables
tab varname1 varname2

*additional summary commands: tabstat, table (see help)

*generate new variable in data set
generate newvarname = log(varname)

*generate new variable with more complex commands XX (see –help egen- )
egen newvarname = XX(varname)

*delete a variable(s) from the data set
drop varlist
```

*instead of dropping all variables not needed, you can list the ones you want to *keep, and the remainder are deleted
keep *varlist*

*rearrange the order of the observations (e.g., arrange a time series data set in order by year)
sort *varname*

*After sort, you can use –by *varname*– with stata command
by country: summarize *varname*

Stata commands can be executed on a sub-set of observations. Use with “if”
gen white = 1 if race == 4
replace white = 0 if race ~= 4

Alternative:
gen white = (race == 4)
gen asian = (race == 2)

Alternative, if you want to create dummy variables from categorical variable:
capture tab race, gen(race)

After Stata commands, results are temporarily stored in “locals”. These are very useful for reference to, but are over-written as soon as another command is executed. So, to keep them for reference later in your .do file, they should be stored in locals for which you give a name to.

To see what results are saved after a particular command, see from help or type the following after a command has been executed.

return list
or
ereturn list

Example:
```
summarize varname
return list
local m=r(mean)
local sd=r(sd)
display "mean =" `m'
gen newvarname = (varname - `m')/`sd'
```

Use ` ' to refer to local name that you specify previously
Use " " to refer to text format

Locals can also be used to store repeatedly types phrases, such as variable names.

Example:
```
local vars "varname1 varname2 varname3 varname4"
summarize `vars' if white == 1
summarize `vars' if black == 1
```

Globals can be used to refer to phrases as well. Once you define global, Stata will remember it until you define the new phrase. For local, you

need to select the line with local definition when execute commands referring to local.

Example:

```
      global varsx "varx1 varx2 varx3 varx4"
       reg y $varsx if varx1 >= 100000
```

Loops

To perform repeated operations, you can use loop to do so.

For example:

```
sort nr year
by nr: gen l_lwage = lwage[_n-1]
by nr: gen l_educ = educ[_n-1]
by nr: gen l_black = black[_n-1]
by nr: gen l_hisp = hisp[_n-1]
by nr: gen l_exper = exper[_n-1]
by nr: gen l_married = married[_n-1]
by nr: gen l_union = union[_n-1]
```

We can use –foreach- to operate the loop above:

```
foreach x in lwage educ black hisp exper married union {
      by nr: gen l_`x' = `x'[_n-1]
      }
```

If you have consecutive numbers, you can use –forvalues-
For example:

```
generate smath=1 if math<=50
replace smath=2 if math>50 & math <=65
replace smath=3 if math>65 & math <.
forvalues i=1/3 {
      summarize science if smath==`i'
      local m`i'=round(r(mean), 0.01)
      }
display "Average science score when math score is below mean: `m1', mean:
`m2', top: `m3'"
```

Data merging

```
*One-to-one merge on specified key variable(s)
merge 1:1 varlist using filename

*Many-to-one merge
merge m:1 varlist using filename

*One-to-many merge
merge 1:m varlist using filename

Many-to-many merge
merge m:m varlist using filename
```

For example:

```
use dollars, clear
```

```
list
```

```
     +----------------------------+
     | region      sales      cost |
     |----------------------------|
  1. | N Cntrl   419,472   227,677 |
  2. | NE        360,523   138,097 |
  3. | South     532,399   330,499 |
  4. | West      310,565   165,348 |
     +----------------------------+
```

```
use sforce, clear
list
```

```
     +---------------------------+
     | region        name   price |
     |---------------------------|
  1. | N Cntrl     Krantz    3500 |
  2. | N Cntrl     Phipps    3500 |
  3. | N Cntrl     Willis    3500 |
  4. | NE         Ecklund    4000 |
  5. | NE          Franks    4000 |
     |---------------------------|
  6. | South     Anderson    3200 |
  7. | South      Dubnoff    3200 |
  8. | South          Lee    3200 |
  9. | South       McNeil    3200 |
 10. | West       Charles    3800 |
     |---------------------------|
 11. | West          Cobb    3800 |
 12. | West         Grant    3800 |
     +---------------------------+
```

```
*Set dollars as 'master' data, then merge sforce (as 'using')
use dollars, clear
merge 1:m region using sforce
```

```
*Set sforce as 'master' data, then merge dollars (as 'using')
use sforce, clear
merge m:1 region using dollars

*Result:
list
```

```
     +-------------------------------------------------------------------+
     | region        name   price     sales      cost        _merge |
     |-------------------------------------------------------------------|
  1. | N Cntrl     Krantz    3500   419,472   227,677   matched (3) |
  2. | N Cntrl     Phipps    3500   419,472   227,677   matched (3) |
  3. | N Cntrl     Willis    3500   419,472   227,677   matched (3) |
  4. | NE         Ecklund    4000   360,523   138,097   matched (3) |
  5. | NE          Franks    4000   360,523   138,097   matched (3) |
     |-------------------------------------------------------------------|
```

Basic Econometrics

∗OLS
```
regress y x1 x2
regress y x1 x2, vce(cluster groupvar)
```

∗Least-squres dummy-variables (LSDV) regression
```
areg y x1 x2, absorb(group/id variable)

predict yhat, xb
predict e, resid

test
testparm
testnl

∗Fitted graph
twoway (scatter y x1) (lfit y x1)

∗xi:and i.varname are used to create dummies variables for the regression
xi: reg y x1 x2 i.varname
```

∗First-differences estimator
```
regress D.(y x1 x2 x3), noconstant
```

∗Panel
- FE by hand
```
bysort firm: egen m_y = mean(y)
gen demean_y = y - m_y
bysort firm: egen m_x1 = mean(x1)
gen demean_x1 = x1 - m_x1
bysort firm: egen m_x2 = mean(x2)
gen demean_x2 = x2 - m_x2

reg demean_y demean_x1 demean_x2
```

- FE and RE Stata commands
```
xtset idvar yearvar

xtreg y x1 x2, fe
est sto fe
xtreg y x1 x2, re
est sto re

hausman fe re
```

∗Making table
```
estout fe re, cells(b(star fmt(4)) se(par fmt(4))) stats(r2 N, fmt(3 0))
starlevels(∗ 0.10 ∗∗ 0.05 ∗∗∗ 0.01)
```

∗IV - 2SLS
```
ivregress 2sls y (x1 = z1 z2 z3) x2 x3, first
      option ‘first’ for the first-stage regression to be displayed
ivregress 2sls y (x1 x2 = z1 z2 z3) x3 x4

∗IV postestimation
estat endogenous
estat firststage
estat overid
```

```
*SUR
sureg (read female ses socst) (math female ses science)
*hypothesis testing: this correlation is zero
sureg, notable noheader corr

sureg (read math = female ses socst science), corr
```

Matrix

The matrix monadic operators are

```
        -B      negation
        B'      transpose
```

The matrix dyadic operators are

```
        B \ C   add rows of C below rows of B (row join)
        B , C   add columns of C to the right of B (column join)
        B + C   addition
        B - C   subtraction
        B * C   multiplication (including mult. by scalar)
        B / z   division by scalar
        B # C   Kronecker product
```

Parentheses may be used to control order of evaluation. The default order of precedence for the matrix operators (from highest to lowest) is

Operator	Symbol
parentheses	()
transpose	'
negation	-
Kronecker product	#
division by scalar	/
multiplication	*
subtraction	-
addition	+
column join	,
row join	\

```
*Set maximum number of columns/rows for matrix
set matsize 800

matrix A = (1,2\3,4)
matrix B = (5,7\9,2)
matrix C = A+B
mat list C

*Kronecker product:
matrix D=A#C
mat list D

matrix G=A',B\D
matrix list G
```

```
* , column join \ row join


*mkmat: Convert variables to matrix and vice versa
gen cons = 1

mkmat write math socst cons, matrix(x_read)
mkmat read, matrix(y1)

mat list x_read
mat list y1

mat b = invsym(x_read'*x_read)*x_read'*y1
mat list b

reg read write math socst
ereturn list
mat beta = e(b)'
mat list beta


regress read write math socst
predict r_read, resid

regress science female math
predict r_sci, resid

help corr
*Option for covariances
corr r_read r_sci, cov

return list

mat s = r(C)
mat list s

*Define identity matrix with dimension = 200
mat i = I(200)
*Create covariance matrix of error terms(from residuals)
mat v = s#i


mkmat math write socst cons, matrix(x_read)
mat x_read = x_read, J(200,3,0)

mkmat female math cons, matrix(x_sci)
mat x_sci = J(200,4,0), x_sci

*join row
mat x = x_read \ x_sci

mkmat read, matrix(y_read)
mkmat science, matrix(y_sci)
mat y = y_read \ y_sci

mat b = inv(x'*inv(v)*x)*x'*inv(v)*y
mat list b
```