Programming with Stata 14.1

For more info see Stata's reference manual (stata.com)

Scalars both r- and e-class results contain scalars

scalar x1 = 3create a scalar x1 storing the number 3 scalar a1 = "I am a string scalar" create a scalar a1 storing a string

Matrices e-class results are stored as matrices

 $\underline{\mathsf{mat}}\mathsf{rix} \ \mathsf{a} = (4 \setminus 5 \setminus 6)$ create a 3 x 1 matrix

matrix b = (7, 8, 9)create a 1 x 3 matrix

matrix d = b' transpose matrix b; store in d

 $matrix ad1 = a \ d =$ row bind matrices

matrix ad2 = a, d column bind matrices

matselrc b x, c(13) findit matselro select columns 1 & 3 of matrix b & store in new matrix x

mat2txt, matrix(ad1) saving(textfile.txt) replace export a matrix to a text file

DISPLAYING & DELETING BUILDING BLOCKS

[scalar | matrix | macro | estimates] [list | drop] b list contents of object b or drop (delete) object b

[scalar | matrix | macro | estimates] dir list all defined objects for that class

matrix dir scalar drop x1 list contents of matrix b list all matrices delete scalar x1

Macros public or private variables storing text

GLOBALS available through Stata sessions

PUBLIC

global pathdata "C:/Users/SantasLittleHelper/Stata" define a global variable called pathdata

cd \$pathdata — add a \$ before calling a global macro

change working directory by calling global macro

<u>alobal</u> myGlobal price mpg length **summarize** \$myGlobal

summarize price mpg length using global

♦ Locals available only in programs, loops, or .do files **Private**

local myLocal price mpg length create local variable called myLocal with the

strings price mpg and length

summarize myLocal add a before and a after local macro name to call summarize contents of local myLocal

levelsof rep78, **local**(levels)

create a sorted list of distinct values of rep78, store results in a local macro called levels

local varLab: variable label foreign can also do with value labels store the variable label for foreign in the local varLab

○ TEMPVARS & TEMPFILES special locals for loops/programs

tempvar temp1 — initialize a new temporary variable called temp1 **generate** 'temp1' = mpg^2 — save squared mpg values in temp1 **summarize** 'temp1' — summarize the temporary variable temp1

tempfile myAuto create a temporary file to see also **save** 'myAuto' be used within a program

Building Blocks basic components of programming

R- AND E-CLASS: Stata stores calculation results in two* main classes:

such as **summary** or **tabulate**

return results from general commands return results from estimation commands such as regress or mean

mean price

To assign values to individual variables use:

SCALARS In individual numbers or strings

MATRICES

rectangular array of quantities or expressions

MACROS pointers that store text (global or local)

Access & Save Stored r- and e-class Objects

Many Stata commands store results in types of lists. To access these, use **return** or ereturn commands. Stored results can be scalars, macros, matrices or functions.

summarize price, detail return list

scalars: r(N)

returns a list of scalars

r(mean) = 6165.25...

each time an r-class = 86995225.97...

matrices and functions scalars: $e(N_{over}) = 1$ = 73 e(N)e(k_eq) = 1 e(rank)

ereturn list returns list of șcalars, macros,

generate p mean = r(mean) create a new variable equal to average of price

= 2949.49...

generate meanN = e(N)create a new variable equal to obs. in estimation command

Use **estimates store**

preserve create a temporary copy of active dataframe

restore restore temporary copy to original point set restore points to test code that changes data

Accessing Estimation Results

After you run any estimation command, the results of the estimates are stored in a structure that you can save, view, compare, and export

regress price weight

estimates store est1

store previous estimation results est1 in memory

eststo est2: regress price weight mpg

eststo est3: **regress** price weight mpg foreign estimate two regression models and store estimation results

estimates table est1 est2 est3

print a table of the two estimation results est1 and est2

EXPORTING RESULTS

The estout and outreg2 packages provide numerous, flexible options for making tables after estimation commands. See also putexcel command.

esttab est1 est2, se star(* 0.10 ** 0.05 *** 0.01) label

create summary table with standard errors and labels

esttab using "auto_reg.txt", replace plain se export summary table to a text file, include standard errors

outreg2 [est1 est2] using "auto_reg2.txt", see replace export summary table to a text file using outreg2 syntax

Additional Programming Resources

O bit.ly/statacode

download all examples from this cheat sheet in a .do file

m adoupdate Update user-written .ado files adolist ssc install adolist List/copy user-written .ado files

net install package, from (https://raw.githubusercontent.com/username/repo/master) install a package from a Github repository

Shttps://github.com/andrewheiss/SublimeStataEnhanced configure Sublime text for Stata 11-14

Loops: Automate Repetitive Tasks

Anatomy of a Loop

see also while

Stata has three options for repeating commands over lists or values: foreach, forvalues, and while. Though each has a different first line. the syntax is consistent:



Foreach: Repeat Commands over Strings, Lists, or Variables

```
foreach x in of [local, global, var list, new list, num list] {
   Stata commands referring to 'x'
                                                     loops repeat the same command
STRINGS
                                                   sysuse "auto.dta", clear
foreach x in auto.dta auto2.dta {
                                         same as
                                                   tab rep78, missing
   sysuse "`x'", clear tab rep78, missing
                                                   sysuse "auto2.dta", clear
                                                   tab rep78, missing
foreach x in "Dr. Nick" "Dr. Hibbert" {
                                            display length("Dr. Nick")
   display length ( x )
                                             display length("Dr. Hibbert")
```

VARIABLES

```
foreach x in mpg weight {

    foreach in takes anv list

   summarize 'x'
                                          as an argument with
                                          elements separated by
                                          spaces
foreach x of varilist mpg weight {

    foreach of requires you

                                          to state the list type,
```

summarize `x' which makes it faster

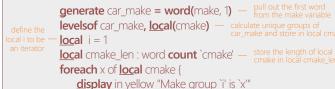
ForValues: Repeat Commands over Lists of Numbers



set trace on (off)

see also capture and scalar _rc trace the execution of programs for error checking

PUTTING IT ALL TOGETHER



_ if `i' == `cmake len' { display "The total number of groups is 'i" **local** i = ++i' — increment iterator by one

summarize mpg

summarize weight