Data Processing with Stata Cheat Sheet

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

Useful Shortcuts

clear
delete data set in memory

Ctrl + **9**
open a new .do file

Ctrl + **D** — keyboard buttons
highlight text in .do file, then ctrl + d executes it in the command line

Ctrl + **8**
open the data editor

At Command Prompt

PgUp **PgDn** scroll through previous commands

Tab autocompletes variable name after typing part

cls clear the console (where results are displayed)

Set up

pwd
print current (working) directory

cd "C:\Program Files (x86)\Stata13"
change working drive

dir
display filenames in working directory

fs *.dta
List all Stata files in working directory

capture log close
close the log on any existing do files

log using "$pathlog/myDoFile.do", **replace**
create a new log file to record your work and results

findit estout
find the package estout to install

ssc install estout
install the package estout; needs to be done once

packages contain extra functions that expand Stata's toolkit

Import Data

sysuse auto2, clear
load system data (Auto data)

For many of these examples, we use the auto2 dataset.

use "auto2.dta", **clear**
load the auto dataset from the current directory

import excel "yourSpreadsheet.xlsx", **/***
***/ sheet**("Sheet1") **cellrange**(A2:H11) **firstrow**
import an Excel spreadsheet

use first row in the cellrange as the column names

import delimited"yourFile.csv", **/***
***/ rowrange**(2:11) **colrange**(1:8) **varnames**(2)
import a .csv file

webuse "auto2.dta"
load data from the web

Basic Syntax

All Stata functions have the same format (syntax):

[by varlist1**:]** *command* [varlist2] **[=***exp***]** **[***if exp***]** **[in** range**]** [weight] **[using** filename**]** **[,***options***]**

- **[by** varlist1**:]**: apply the *command* across each group in *varlist1*
- *command*: function: what are you going to **do** to *varlists*?
- [varlist2]: column to apply *command* to
- **[=***exp***]**: save output as a new variable
- **[***if exp***]**: condition: only apply the function *if* something is true
- **[in** range**]**: apply to specific rows
- [weight]: apply weights
- **[using** filename**]**: pull data from a file (if not loaded)
- **[,***options***]**: special options for *command*

bysort rep78 : summarize price if foreign == 0 & price <= 9000, detail

In this example, we want a ***detailed*** summary with stats like kurtosis, plus mean and median

To find out more about any command – like what options it takes – type **help** *command*

Basic Data Operations

Arithmetic

+	add (numbers) concatenate (strings)
−	subtract
*	multiply
/	divide
^	raise to a power

Logic

&	and	<	less than
! or ~	not	<=	less than or equal
|	or	>	greater than
==	equal	>=	greater or equal
!= or ~=	not equal		

Data Types

	ex.	range	checking type	converting
byte	1	0 or 1	checking type	converting
int	−1	−32,767 - 32,740	R: is.int(x)	R: int(x)
long	62,134	~−2B - +2B	checking type	converting
float	5.14	range	checking type	converting
double	5.14	range	checking type	converting
string	"hello"		checking type	converting
missing	.		checking type	converting

Explore Data

View How Data are Organized

desc make price
display variable type, format, and any value/variable labels

count **count if** price > 5000
number of rows (observations). Can be combined with logic

ds
List variables matching name patterns or other characteristics

isid mpg
check if mpg uniquely identifies the data

confirm numeric variable price make
check that price and make are numeric

mdesc
view how many observations are missing for each variable

See How Data Are Distributed

codebook make price
overview of variable type, stats (range, mean, stdev, percentiles), number of missing or unique values

summarize make price mpg
Print summary statistics (mean, stdev, min, max) for selected variables

inspect mpg
display histogram of data and number of missing/zero/positive/negative observations

Browse Observations within the Data

browse or **Ctrl** + **8**
open the data editor

display price[4]
display the fourth observation in price.
doesn't work with vectors (more than one value)

Summarize Data

include missing values

save binary variables for each categ in a new variable, repairRecord

tabulate rep78, **mi gen(**repairRecord**)**
one-way table: number of observations with each value of rep78

tabulate rep78 foreign, mi
two-way table: cross-tabulate number of observations for each combi

bysort rep78**:** tabulate foreign
for each value of rep78, apply the command tabulate foreign

tabstat price weight mpg**, by(**foreign**) stat(mean sd min max n)**
Create compact table of summary statistics

collapse (mean) price (max) mpg**, by(**foreign**)**
calculate mean price and max mpg by car type. * Replaces all the dat

Create New Variables

Transform Existing Columns

generate mpgSquared = mpg * mpg
generate byte lowPrice = price < 4000
create or change contents of a variable. Useful also for creating binary

egen unique_ID = **group(**var1 var2...**)**
create a unique id from a combination of variables

pctile mpgQuartile= mpg**, nq** = 4
create quartiles of the mpg data

clonevar mpg2 = mpg
create a unique id from a combination of variables

rename (rep78 foreign) (repairRecord carType)

Tim Essam (tessam@usaid.gov) • Laura Hughes (lhughes@usaid.gov)

Create Individual Numbers or Strings

scalar a = 3
define a scalar called 'a' and store the value 3 in it

scalar s = "hello world"
define a scala called s and store a string value in it

scalar dir
display scalar variables

scalar drop
drop scalar variables

```
              storage   display    value
variable name   type    format     label      variable label
------------------------------------------------------------------------------
make            str18   %-18s                 Make and Model
price           int     %8.0gc                Price
```

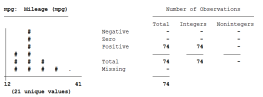
```
price
-------------------------------------------------------------
      type:  numeric (int)

     range:  [3291,15906]                 units:  1
unique values:  74                    missing .:  0/74

      mean:   6165.26
   std. dev:   2949.5

  percentiles:        10%       25%       50%       75%       90%
                     3895      4195    5006.5      6342     11385
```

Variable	Obs	Mean	Std. Dev.	Min	Max
make	0				
price	74	6165.257	2949.496	3291	15906
mpg	74	21.2973	5.785503	12	41



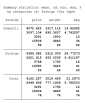
Number of Observations

	Total	Integers	Noninteger
Negative	-	-	-
Zero	-	-	-
Positive	74	74	-
Total	74	74	-
Missing	-		
	74		

gsort price mpg **gsort –**price **–**mpg
sort in ascending (left) or descending (right) order, first by price then miles per gallon

Repair Record 1978	Freq.	Percent	Cum.
Poor	2	2.70	2.70
Fair	8	10.81	13.51
Average	30	40.54	54.05
Good	18	24.32	78.38
Excellent	11	14.86	93.24
.	5	6.76	100.00
Total	74	100.00	

Repair Record 1978	Car type Domestic	Foreign	Total
Poor	2	0	2
Fair	8	0	8
Average	27	3	30
Good	9	9	18
Excellent	2	9	11
.	4	1	5
Total	52	22	74



-> rep78 = Poor

Car type	Freq.	Percent	Cum.
Domestic	2	100.00	100.00
Total	2	100.00	

-> rep78 = Fair

Car type	Freq.	Percent	Cum.
Domestic	8	100.00	100.00
Total	8	100.00	