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 auto, clear
load system data (Auto data)

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

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

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

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**:**]	*command*	[varlist2]	[=*exp*]	[*if exp*]	[**in** range]	[weight]	[**using** filename]	[,*options*]
apply the *command* across each group in *varlist1*	function: what are you going to **do** to *varlists*?	column to apply *command* to	save output as a new variable	condition: only apply the function *if* something is true	apply to specific rows	apply weights	pull data from a file (if not loaded)	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) combine (strings)
−	subtract
*****	multiply
/	divide
^	raise to a power

Logic

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

if foreign != 1 | price >= 10000

make	foreign	price
Chevy Colt	0	3,984
Buick Riviera	0	10,372
Honda Civic	1	4,499
Volvo 260	1	11,995

if foreign != 1 & price >= 10000

make	foreign	price
Chevy Colt	0	3,984
Buick Riviera	0	10,372
Honda Civic	1	4,499
Volvo 260	1	11,995

Change Data Types

Stata has 7 data types:

no data	true/false	words	numbers	
missing	**byte**	**string**	**int**	**float**
			long	**double**

Often, you have to convert between numbers & strings:

1 — **tostring** foreign, **gen**(foreign_string) → "1"
1 ← **destring** foreign_string, **gen**(foreign_numeric) — "1"

1 — **decode** foreign , **generate**(foreign_string) → "foreign"
1 ← **encode foreign_string, gen(foreign_numeric)** — "foreign"

Explore Data

View How Data are Organized

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

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

ds, has(type string)
list variables matching name patterns or other characteristics

isid mpg
check if mpg uniquely identifies the data

mdesc
view how many observations are missing for each variable

See How Data Are Distributed

codebook make price
overview of variable type, stats, number of missing/unique values

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

inspect mpg
show histogram of data, number of missing/zero observations

Browse Observations within the Data

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

display price[4]
display the 4th observation in price; only works on single values

list make price if price > 10000
list the make and price for observations with price > $10,000

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

Summarize Data

tabulate rep78, **mi gen**(repairRecord)
include missing values
save binary variables for each category in a new variable, 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 combination of rep78 and foreign

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 & max mpg by car type. Replaces all data.

Create New Variables

generate mpgSquared = mpg * mpg
generate byte lowPrice = price < 4000
create or change contents of a variable. Useful also for creating binary (Boolean) variables based on a condition (generate byte).

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

Tim Essam (tessam@usaid.gov) • Laura Hughes (lhughes@usaid.gov)
inspired by RStudio's awesome Cheat Sheets (rstudio.com/resources/cheatsheets) geocenter.github.io/StataTraining updated January 2016

Transform Data

Select Parts of Data

Select Specific Columns

drop make
remove the 'make' variable

keep make price
opposite of drop;
keep only columns 'make' and 'price'

Filter Specific Rows

drop if mpg < 20   **drop in** 1/4
drop observations based on a condition (left) or rows 1-4 (right)

keep in 1/30
opposite of drop; keep only rows 1-30

sample 25
sample 25% of the observations in the dataset (set seed before using)

Combining Data

Adding (Appending) New Data

Create Individual Numbers or Strings

Merging Two Datasets Together

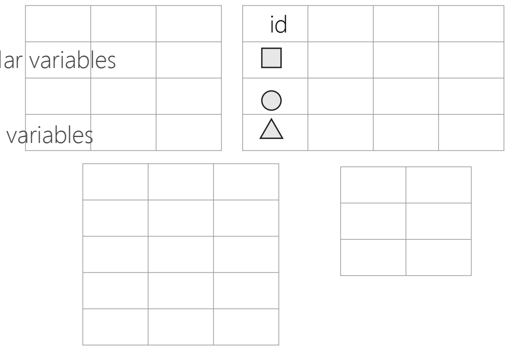
scalar a = 3
define a scalar called 'a' and store the value 3 in it

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

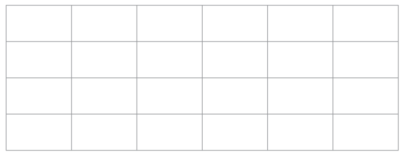
one-to-one

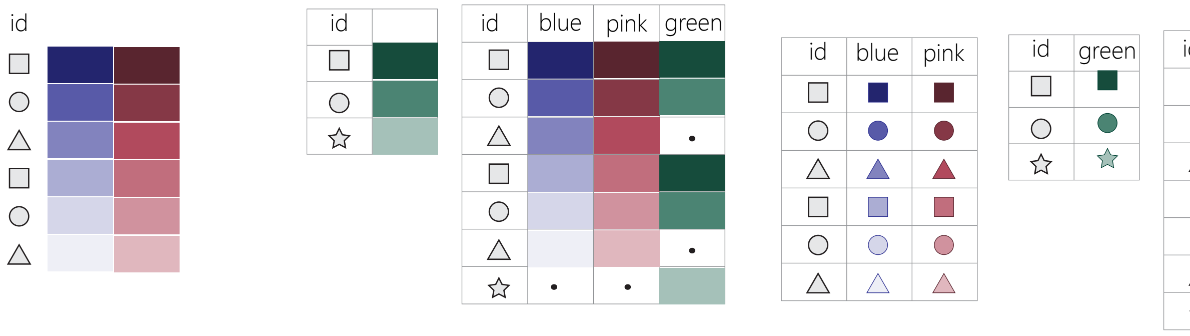
scalar dir
display scalar variables

scalar drop
drop scalar variables



Reshape Data





Strings

charlist
display all characters within a string

Missing Data

Factors

Labels

rename (rep78 foreign) **(**repairRecord carType**)**
rename one or multiple variables

mvencode _all, **mv(**9999**)**
replace missing values with the numeric value 9999 for all variables

mvdecode _all, **mv(**9999**)**
replace numeric value 9999 with missing value