Stata Class Notes

Ethan Deng

Version 1.0

April 14, 2015

NOTES: All the notes are from the website UCLA Stata Class Notes! The latest version of this notes (Highlighted PDF format) can be found on EthanDeng's Page.

Stata is a powerful and yet easy-to-use statistical package that runs on Windows, Macintosh and Unix platforms. This class is designed for people who are just getting started using Stata. The students in the class will have a hands-on experience using Stata for statistics, graphics and data management. The class notes are the scripts for the class available to the students in the class and to others on the Internet. The Stata class notes do not contain any of the output. The class notes are not meant to be a Stata textbook or a reference manual. However, it is possible for individuals to use the class notes to help in learning Stata even if they don't enroll in the class.

1 Entering Data

1.1 Stata Commands in this section

The Stata commands in this section is list in Table 1.

Table 1: Stata commands in section: Entering Data

Command	Explanation of Command
`cd`	Change directory
`dir` or `ls`	Show files in current directory
`insheet`	Read ASCII (text) data created by a spreadsheet
`infile`	Read unformatted ASCII (text) data
`infix`	Read ASCII (text) data in fixed format
`input`	Enter data from keyboard
`import excel`	Import Excel .xls or .xlsx file
`describe`	Describe contents of data in memory or on disk
`compress`	Compress data in memory
`save`	Store the dataset currently in memory on disk in Stata data format
`use`	Load a Stata-format dataset
`count`	Show the number of observations
`list`	List values of variables
`clear`	Clear the entire dataset and everything else
`memory`	Display a report on memory usage
`set memory`	Set the size of memory

1.2 Demonstration and explanation

1.2.1 Comma-separated file with variable names

We will start with inputting a spreadsheet type of data file into Stata. A spreadsheet type of file is created by programs such as Excel. For example, in Excel, we can save a file into a comma-separated-values format (`.csv`) file. Stata reads in this type of data using the `insheet` command. Let's first get to the directory where the file `hs0.csv` is. This data file has variable names on the first line.

Here is a partial listing from the comma-separated file:

```
gender,id,race,ses,schtyp,prgtype,read,write,math,science,socst
0,70,4,1,1,general,57,52,41,47,57
1,121,4,2,1,vocati,68,59,53,63,61
0,86,4,3,1,general,44,33,54,58,31
0,141,4,3,1,vocati,63,44,47,53,56
0,172,4,2,1,academic,47,52,57,53,61
0,113,4,2,1,academic,44,52,51,63,61
0,50,3,2,1,general,50,59,42,53,61
0,11,1,2,1,academic,34,46,45,39,36
0,84,4,2,1,general,63,57,54,,51
0,48,3,2,1,academic,57,55,52,50,51
```

And here are the Stata commands to read these data.

```
cd d:\stata_data  /* note: directory and path may differ on your computer */
dir
insheet using hs0.csv, clear
describe
```

1.2.2 Comma-separated file without variable names

What if the data file does not have the variable names on the first line? We have a such file called `hs0_noname.csv`. We will also do a count to see if the inputting was successful.

```
insheet gender id race ses schtyp prgtype read write math science socst using hs0_noname.
    csv, clear
count
```

1.2.3 Space-delimited file

To read a space-delimited file we will use `infile` command. The first part of the file `hs0.raw` is shown below.

```
0 70 4 1 1 general 57 52 41 47 57
1 121 4 2 1 vocati 68 59 53 63 61
0 86 4 3 1 general 44 33 54 58 31
0 141 4 3 1 vocati 63 44 47 53 56
0 172 4 2 1 academic 47 52 57 53 61
0 113 4 2 1 academic 44 52 51 63 61
0 50 3 2 1 general 50 59 42 53 61
0 11 1 2 1 academic 34 46 45 39 36
0 84 4 2 1 general 63 57 54 . 51
0 48 3 2 1 academic 57 55 52 50 51
0 75 4 2 1 vocati 60 46 51 53 61
0 60 5 2 1 academic 57 65 51 63 61
```

Notice how we specify a character variable below. The variable *prgtype* is a string variable. We tell Stata that *prgtype* is a string variable which should have a length of 10 by typing `str10` **before** the variable name. We will use the `hs0.raw` data file.

```
infile gender id race ses schtyp str10 prgtype read write math science socst using hs0.
    raw, clear
```

1.2.4 Fixed format file

The other type of commonly used ASCII data format is fixed format. It always requires a codebook to specify which column(s) corresponds to which variable. Here is small example of this type of data with a codebook. Notice how we make use of the codebook in the `infix` command below. We will use the `schdat.fix` data file.

```
 195  094951
 26386161941
 38780081841
 479700  870
 56878163690
 66487182960
 786  069  0
 88194193921
 98979090781
107868180801
```

Table 2: Codebook of the fix width dataset

variable name	column number
id	1-2
a1	3-4
t1	5-6
gender	7
a2	8-9
t2	10-11
tgender	12

```
clear
infix id 1-2 a1 3-4 t1 5-6 gender 7 a2 8-9 t2 10-11 tgender 12 using schdat.fix
```

1.2.5 Excel file

The `import excel` command was introduce in Stata 12. Here is what the file `hsbdemo.xlsx` looks like.

Here is how the spreadsheet is read in.

```
import excel using hsbdemo.xlsx, sheet("hsbdemo") firstrow clear
```

1.2.6 Space delimited data from a do-file

We can also use the do-file editor to input data. The do-file editor is used for writing a sequence of commands and running them all at once. You can copy and paste the following Stata syntax to the do-file editor and run it.

id	female	ses	schtyp	prog	read	write	math	science	socst	honors	awards	cid
45	female	low	public	vocation	34	35	41	29	26	not enrolled	0	1
108	male	middle	public	general	34	33	41	36	36	not enrolled	0	1
15	male	high	public	vocation	39	39	44	26	42	not enrolled	0	1
67	male	low	public	vocation	37	37	42	33	32	not enrolled	0	1
153	male	middle	public	vocation	39	31	40	39	51	not enrolled	0	1
51	female	high	public	general	42	36	42	31	39	not enrolled	0	1
164	male	middle	public	vocation	31	36	46	39	46	not enrolled	0	1
133	male	middle	public	vocation	50	31	40	34	31	not enrolled	0	1
2	female	middle	public	vocation	39	41	33	42	41	not enrolled	0	1
53	male	middle	public	vocation	34	37	46	39	31	not enrolled	0	1
1	female	low	public	vocation	34	44	40	39	41	not enrolled	0	1
128	male	high	public	academic	39	33	38	47	41	not enrolled	0	2
16	male	low	public	vocation	47	31	44	36	36	not enrolled	0	2

Figure 1: Excel datasets sample

```
clear
input id female race ses str3 schtype prog read write math science socst
147 1 1 3 pub 1 47 62 53 53 61
108 0 1 2 pub 2 34 33 41 36 36
 18 0 3 2 pub 3 50 33 49 44 36
153 0 1 2 pub 3 39 31 40 39 51
 50 0 2 2 pub 2 50 59 42 53 61
 51 1 2 1 pub 2 42 36 42 31 39
102 0 1 1 pub 1 52 41 51 53 56
 57 1 1 2 pub 1 71 65 72 66 56
160 1 1 2 pub 1 55 65 55 50 61
136 0 1 2 pub 1 65 59 70 63 51
end
```

After running the above program, we can issue the `describe` command to get a general idea about the data set. The `compress` command reduces the size of the data set. We can save the data set to disk by issuing the `save` command.

```
describe
compress
save hsb10
```

1.2.7 Stata data file on hard drive or flash drive

To read in a Stata data file, we use the `use` command.

```
clear
use hsb10
use "D:\data\hsb10.dta", clear
```

1.2.8 Stata data file via the Internet

The `use` command can also be used to read a data file over the internet.

```
use "http://www.ats.ucla.edu/stat/data/hs0.dta", clear
```

2 Exploring Data

2.1 Stata commands in this section

The Stata commands in this section is list in Table 3.

Table 3: Stata Commands in section: Exploring Data

Command	Explanation of Command
`cd`	Change directory
`use`	Load dataset into memory
`describe`	Describe a dataset
`list`	List the contents of a dataset
`codebook`	Detailed contents of a dataset
`labelbook`	Information on value labels
`log`	Create a log file
`lookfor`	Find variables in large dataset
`summarize`	Descriptive statistics
`tabstat`	Table of descriptive statistics
`table`	Create a table of statistics
`stem`	Stem-and-leaf plot
`graph`	High resolution graphs
`kdensity`	Kernel density plot
`sort`	Sort observations in a dataset
`histogram`	Histogram for continuous and categorical variables
`tabulate`	One- and two-way frequency tables
`correlate`	Correlations
`pwcorr`	Pairwise correlations
`view`	Display file in viewer window

2.2 Demonstration and explanation

We will begin by loading `hs0.dta`, a dataset saved in Stata's format that we encountered in the Entering Data section. Stata data files end with the `dta` extension. Stata data files are loaded into memory using the `use` command. Only one dataset can be loaded at a time.

```
use "http://www.ats.ucla.edu/stat/data/hs0.dta", clear
```

Before we start our statistical exploration we will look at the data using the `describe`, `codebook`, `lookfor`, `labelbook` and `list` commands. Note that the variable *prgtype* is a string variable.

```
describe
codebook
lookfor s
labelbook
list
list gender-read in 1/20
```

Next, we will open a log file which will save all of the commands and the output (except for graphs) in a text file. We use the `text` option so that the log can be read in any text editor, such as NotePad or WordPad.

```
log using unit2.txt, text replace
```

The basic descriptive statistics command in Stata is `summarize`. Along with `summarize`, we also show the `tabstat` and `table` commands for displaying descriptive statistics within groups.

```
summarize
summarize read math science write
display 9.48^2                          /* note: variance is the sd (9.48) squared */
summarize write, detail
sum write if read>=60                   /* note: sum is abbreviation of summarize */
sum write if prgtype=="academic"
sum write in 1/40
tabstat read write math, by(prgtype) stat(n mean sd)
tabstat write, by(prgtype) stat(n mean sd p25 p50 p75)
```

Next, let's use some graphics commands to look at our data. We will begin with `stem` which generates an ASCII stem-and-leaf plot. We will also use the `graph` command with the `histogram` (histogram) and `box` (boxplot) options. We also show the `kdensity` command which produces a smoothed density plot.

```
stem write
stem write, lines(2)
histogram write, normal
histogram write, normal start(30) width(5)
kdensity write, normal
kdensity write, normal width(5)  /* a smoother kdensity plot */
kdensity math, normal
graph box write
graph box write, over(prgtype)
```

The `tabulate` command can produce one-way or two-way frequency tables. The `tab1` command is a convenience command to produce multiple one-way frequency tables. The `histogram` command is used to display histograms for categorical variables.

```
histogram ses
histogram ses, discrete
tabulate ses
tab write       /* note: tab is abbreviation of tabulate */
tab1 gender schtyp prgtype
```

Two-way crosstabulation.

```
tab prgtype ses
```

Two-way crosstabulation with row and column percents.

```
tab prgtype ses, row col
```

There are two commands to create correlation matrices, `correlate` which uses *listwise deletion* of missing data and `pwcorr` which uses *pairwise deletion*. The general purpose `graph` command produces scatter plots using the `twoway` option and an scatterplot matrix using the `matrix` option. The `jitter` option is used to spread apart identical observations.

```
correlate write read science
pwcorr write read science, obs
scatter write read
scatter write read, jitter(2)
graph matrix read science write, half
```

We have completed all of the analyses in this section, so it is time to close the log file.

```
log close
```

Now, let's see what is in our log file.

```
view unit2.txt
```

3 Modifying Data

3.1 Stata commands in this section

The Stata commands in this section is list in Table 4.

Table 4: Stata Commands in section: Modifying Data

Command	Explanation of Command
`codebook`	Show codebook information for file
`order`	Order the variables in a data set
`label data`	Apply a label to a data set
`label variable`	Apply a label to a variable
`label define`	Define value labels for a categorical variable
`label values`	Apply value labels to a variable
`encode`	Create numeric version of a string variable
`list`	Lists the observations
`rename`	Rename a variable
`recode`	Recode the values of a variable
`notes`	Apply notes to the data file
`generate`	Creates a new variable
`replace`	Replaces values for an existing variable
`egen`	Extended generate - has special functions that can be used when creating a new variable

3.2 Demonstration and explanation

```
use "http://www.ats.ucla.edu/stat/data/hs0.dta", clear
```

Let's use the codebook command to see what our variables look like. Because we have not listed any variables after the command, Stata will show us the codebook for all of the variables.

```
codebook
```

First, let's order the variables in a way that makes sense. While there are several possible orderings that are logical, we will put the id variable first, followed by the demographic variables, such as *gender*, *ses* and *prgtype*. We will put the variables regarding the test scores at the end.

```
order id gender
```

Now let's include some variable and value labels so that we know a little more about the variables.

```
label variable schtyp "type of school"
label define scl 1 public 2 private /* also 1"public" 2"private" */
label values schtyp scl
codebook schtyp
list schtyp in 1/10
list schtyp in 1/10, nolabel
```

Now let's create a new numeric version of the string variable prgtype. We will call our new variable *prog*.

```
encode prgtype, gen(prog)
label variable prog "type of program"
codebook prog
list prog in 1/10
list prog in 1/10, nolabel
```

The variable *gender* may give us trouble in the future because it is difficult to know what the 1s and 2s mean.

```
rename gender female
recode female (1=0)(2=1)
label define fm 1 female 0 male
label values female fm
codebook female
list female in 1/10
list female in 1/10, nolabel
```

Let's recode the value 5 in the variable *race* to be missing.

```
list race if race == 5
recode race 5 = .
list race if race == .
```

Now let's create a variable that is a total of some of the test scores.

```
generate total = read + write + math + science
summarize total
```

Note that there are five missing values of *total* because there are five missing values of *science*.

Now let's see if we can assign some letter grades to these test scores.

```
recode total (0/140=0 F) (141/180=1 D) (181/210=2 C) (211/234=3 B) (235/300=4 A), gen(
    grade)
label variable grade "combined grades of read, write, math, science"
codebook grade
list read write math science total grade in 1/10
list read write math science total grade in 1/10, nolabel
```

Let's label the dataset itself so that we will remember what the data are. We can also add some notes to the data set.

```
label data "High School and Beyond"
notes female:  the variable gender was renamed to female
notes race:  values of race coded as 5 were recoded to be missing
notes
```

Stata has another way of generating new variables called `egen` which stands for extended generation. The `egen` command is a useful tool for many of specialized situations.

In our first example, we will use `egen` to create standard scores for the variable *read*.

```
egen zread = std(read)
summarize zread
list read zread in 1/10
```

Next we will a variable that has the mean of read for each level of ses.

```
egen readmean = mean(read), by(ses)
list read ses readmean in 1/10
```

Now we will compute the average of several variables for each observation. Please note that there will be a mean for observation 9 even though it has a missing value for *science*.

```
egen row_mean = rowmean(read write math science)
list read write math science row_mean in 1/10
```

These are just a few of the many useful `egen` functions built-in to Stata.

Finally, we will save our data and continue on to the next unit.

```
save hs1
```

4 Managing Data

4.1 Stata commands in this section

The Stata commands in this section is list in Table 5.

Table 5: Stata Commands in section: Managing Data

Command	Explanation of Command
`pwd`	Show current directory (pwd=print working directory)
`dir` or `ls`	Show files in current directory
`cd`	Change directory
`keep if`	Keep observations if condition is met
`keep`	Keep variables or observations
`drop`	Drop variables or observations
`drop if`	Drop observations if condition is met
`append`	Append a data file to current file
`sort`	Sort observations
`merge`	Merge a data file with current file

4.2 Demonstration and explanation

4.2.1 Example - Subsetting data

Suppose we are undergraduates working on our honors thesis and we wish to analyze just a subset of the `hs1` data file. In fact, we are studying "good readers" and just want to focus on the students who had a reading score of 60 and higher. The following shows how we can take the `hs1` data file and make a separate folder called `honors` and store a copy of our data which just has the students with reading scores of 60 or higher.

```
use hs1, clear
pwd
dir
ls
cd Stata_data
keep if read >= 60
describe
summarize read
save hsgoodread, replace
pwd
```

4.2.2 Example continued - Keeping variables

Further suppose that our data file had many, many variables, say 2000 variables, but we only care about just a handful of them, *id*, *female*, *read* and *write*. We can subset our data file to keep just those variables as shown below.

```
keep id female read write
save hskept, replace
describe
list in 1/20
```

4.2.3 Example continued - Dropping variables

Instead of wanting to keep just a handful of variables, it is possible that we might want to get rid of just a handful of variables in our data file. Below we show how we could get rid of the variables *ses* and *prog*.

```
use hsgoodread, clear
drop ses prog
save hsdropped, replace
describe
list in 1/10
```

4.2.4 Example - Appending data

Now we have moved on to our master's thesis. We have a folder called `masters` and we have been given a file with the data for the males (called `hsmale`) and a file for the females (called `hsfemale`). We need to combine these files together to be able to analyze them, as shown below. In this example, we are adding cases, sometimes called "stacking" datasets.

```
cd masters
dir
use hsmale
tabulate female
append using hsfemale
tabulate female

save hsmasters, replace
cd ..
```

4.2.5 Example - Merging data

Now we are working on our dissertation and, as with our masters, we have been given two files. In this case, we have a file that has the demographic information (called `hsdemo`) and a file with the test scores (called `hstest`) and we wish

to merge these files together. First, we need to open, sort and save each data file. Each data file must be sorted by the same variable. Next, we use the `merge` command to merge the two datasets.

```
cd diss
dir
use hsdem, clear
list
merge 1:1 id using hstest
tab _merge
list
save hsdiss
cd ..
dir
```

5 Analyzing Data

5.1 Stata commands in this section

The Stata commands in this section is list in Table 6.

Table 6: Stata Commands in section: Analyzing Data

Command	Explanation of Command
`ttest`	t-test
`anova`	Analysis of variance
`margins`	Predicted means
`marginsplot`	Plot predicted means
`regress`	Regression
`predict`	Predicts after model estimation
`kdensity`	Kernel density estimates and graphs
`pnorm`	Graphs a standardized normal plot
`qnorm`	Graphs a quantile plot
`rvfplot`	Graphs a residual versus fitted plot
`test`	Test linear hypotheses after model estimation
`logit`	Logistic regression
`tabulate`	Crosstabs with chi-square test
`signtest`	Tests the equality of matched pairs of data
`signrank`	Wilcoxon matched-pairs signed rank test
`ranksum`	Mann-Whitney two-sample test
`kwallis`	Nonparametric analog to the one-way anova

5.2 Demonstration and explanation

We will begin by downloading the dataset for this unit over the internet.

```
use "http://www.ats.ucla.edu/stat/data/hsbdemo.dta", clear
```

5.2.1 chi-square test of frequencies

Here is the `tabulate` command for a crosstabulation with an option to compute chi-square test of independence and measures of association.

```
tabulate prog ses, all
```

Here is the command with an option to display expected frequencies so that one can check for cells with very small expected values.

```
tabulate prog ses, all expected
```

5.2.2 t-tests

This is the one-sample t-test, testing whether the sample of writing scores was drawn from a population with a mean of 50.

```
ttest write = 50
```

This is the paired t-test, testing whether or not the mean of *write* equals the mean of *read*.

```
ttest write = read
```

This is the two-sample independent t-test with pooled (equal) variances.

```
ttest write, by(female)
```

This is the two-sample independent t-test with separate (unequal) variances.

```
ttest write, by(female) unequal
```

5.2.3 Analysis of Variance

The `anova` command, unsurprisingly, performs analysis of variance (ANOVA). Here is an example of a one-way analysis of variance.

```
anova write prog
```

In this example the `anova` command is used to perform a two-way factorial analysis of variance (ANOVA).

`anova write prog female prog#female` or `anova write prog##female`

The `margins` command compute cells means and the `marginsplot` command plots the interaction.

```
margins prog#female

marginsplot
```

Here is an example of an analysis of covariance (ANCOVA) using the `anova` command.

```
anova write prog female prog#female c.read
```

The `margins` command compute cells means and the `marginsplot` command plots the interaction.

```
margins prog#female, asbalanced

marginsplot
```

5.2.4 Regression

Plain vanilla OLS linear regression.

```
regress write read i.female
```

In the example below, we run the regression with robust standard errors. This is very useful when there is heterogeneity of variance. This option does not affect the estimates of the regression coefficients.

```
regress write read i.female, robust
```

The `predict` command calculates predictions, residuals, influence statistics, and the like after an estimation command. The default shown here is to calculate the predicted scores.

```
predict p
```

When using the `resid` option the `predict` command calculates the residual.

```
predict r, resid
```

The `list` command displays the values of the variables that we have generated. The `in 1/20` option stipulates that only the first 20 observations be displayed.

```
list write p r in 1/20
```

The `kdensity` command with the `normal` option displays a density graph of the residuals with an normal distribution superimposed on the graph. This is particularly useful in verifying that the residuals are normally distributed, which is a very important assumption for regression.

```
kdensity r, normal
```

The `pnorm` command produces a normal probability plot and it is another method of testing weather the residuals from the regression are normally distributed.

```
pnorm r
```

The `qnorm` command produces a normal quantile plot. It is yet another method for testing if the residuals are normally distributed. The `qnorm` plot is more sensitive to deviances from normality in the tails of the distribution, whereas the `pnorm` plot is more sensitive to deviances near the mean of the distribution.

```
qnorm r
```

`rvfplot` is a convenience command that generates a plot of the residual versus the fitted values; it is used after `regress` or `anova`.

```
rvfplot
```

The `i.` prefix is use to dummy code categorical variables such as *prog*. The predictor *prog* has three levels and requires two dummy-coded variables. The `test` command is used to test the collective effect of the two dummy-coded variables; in other words, it tests the main effect of *prog*.

```
regress write read i.prog
test 2.prog 3.prog
```

The `i.` prefix along with the `c.` prefix can also be used to code the interaction of a categorical by continuous variable for *prog* and *read*. The `testparm` command tests the overall interaction and the while the `test` command tests the main effect of *prog*.

```
regress write i.prog##c.read

testparm prog#c.read
test 2.prog 3.prog
```

5.2.5 Logistic regression

In order to demonstrate the logistic regression command we will use the binary variable *honors* (short for honors composition) as our response variable.

```
tab honors
```

The default output for the `logit` command is given as coefficients in the log odds metric. To obtain odds ratios, use the `or` option. The `logistic` command, on the other hand, defaults to odds ratio output. The log odds coefficients can be obtained using the `coef`.

```
logit honors read female
logit, or
logistic honors read female
logistic, coef
```

5.2.6 Non-parametric Tests

The `signtest` is the nonparametric analog of the single-sample t-test.

```
signtest write = 50
```

The `signrank` command computes a Wilcoxon sign-ranked test, the nonparametric analog of the paired t-test.

```
signrank write = read
```

The `ranksum` test is the nonparametric analog of the independent two-sample t-test and is know as the Mann-Whitney or Wilcoxon test.

```
ranksum write, by(female)
```

The `kwallis` command computes a Kruskal-Wallis test, the non-parametric analog of the one-way ANOVA.

```
kwallis write, by(prog)
```

6 For more information

6.1 Entering Data

- Data Management Using Stata: A Practical Handbook,*Ch 2*.
- Statistics with Stata 12,*Ch 2*.
- Gentle Introduction to Stata, Revised Third Edition,*Ch 2*.
- Data Analysis Using Stata, Third Edition,*Ch 11*.
- An Introduction to Stata for Health Researchers, Third Edition,*Ch 6*.
- Stata Learning Modules.
  - A sample Stata session.
  - Inputting raw data files into Stata.

- Frequently Asked Questions.
  - How can I convert files among SAS, SPSS and Stata?
  - How can I input a dataset quickly?
  - How can I read Excel files in Stata? (Stata 12)
  - How can I read Stata 12 data files in Stata 11?
  - How do I read a data file that uses commas/tabs as delimiters?
  - How can I handle the No Room to Add Observations Error?

6.2 Exploring Data

- Statistics with Stata 12,*Ch 3 & Ch 5*.
- Gentle Introduction to Stata, Revised Third Edition,*Ch 5*.
- Data Analysis Using Stata, Third Edition,*Ch 7*.
- An Introduction to Stata for Health Researchers, Third Edition,*Ch 11*.
- Stata Learning Modules.
  - Descriptive information and statistics.
  - Using if with Stata commands.

6.3 Modifying Data

- Data Management Using Stata: A Practical Handbook,*Ch 4-5*.
- Statistics with Stata 12,*Ch 2*.
- Gentle Introduction to Stata, Revised Third Edition,*Ch 3*.
- Data Analysis Using Stata, Third Edition,*Ch 5*.
- An Introduction to Stata for Health Researchers, Third Edition,*Ch 7-8*.
- Stata Learning Modules.
  - Labeling data.
  - Creating and recoding variables.
- Frequently Asked Questions.
  - How can I quickly convert many string variables into numeric variables?
  - How can I quickly recode continuous variables into groups?
  - How do I standardize variables in Stata?

6.4 Managing Data

- Data Management Using Stata: A Practical Handbook,*Ch 6-8*.
- Statistics with Stata 12,*Ch 2*.
- Gentle Introduction to Stata, Revised Third Edition,*Ch 3*.
- Data Analysis Using Stata, Third Edition,*Ch 11*.
- An Introduction to Stata for Health Researchers, Third Edition,*Ch 9*.
- Stata Learning Modules.
  - Subsetting variables and observations.
  - Combining Stata data files.
- Frequently Asked Questions.
  - How can I merge multiple files in Stata?

6.5 Analyzing Data

- Statistics with Stata 12,*Ch 4 & Ch 7-13*.
- Gentle Introduction to Stata, Revised Third Edition,*Ch 6-11*.
- Data Analysis Using Stata, Third Edition,*Ch 8-10*.
- An Introduction to Stata for Health Researchers, Third Edition,*Ch 11-15*.
- Interpreting and Visualizing Regression Models Using Stata.
- Regression with Stata Webbook

  Includes such topics as diagnostics, categorical predictors, testing interactions and testing contrasts.
- Choosing the Correct Statistical Test.

  Includes guidelines for choosing the correct non-parametric test.
- Data Analysis Examples.

  Gives examples of common analysis and interpretation of the output.
- Annotated Output.

  Fully annotates the output from common statistical procedures.
- Frequently Asked Questions

  Covers many topics, including ANOVA, linear regression, logistic regression and use of the margins command.