

Stata Assignments

[stata_assignments]

1. Estimates table

See *view help estimates* & *view help est_table*.

*Estimates table is useful in comparing a set of regression models & for displaying/printing the results in publishable form. (See **view help outreg** for a more powerful way to display/print regression results for publication.)*

```
. use hsb2, clear
```

1. Estimate increasingly complex regression models & display the results via "estimates table."

```
. reg read math science socst
. est store Model3, title(Most complex model)
. est t
. est t, star
. est t, se

. reg read math science
. est store Model2, title(Intermediate model)
. est table
. est table, star

. reg read math
. estimates store Model1, title(Simplest model)
. est table
. est table, star

. est replay Model3
. est query [to confirm which model is 'active']

. est replay Model2
. est query

. est dir _all
. est t Model1 Model2 Model3, coded
. est t Model1 Model 2 Model3
. est t Model1 Model2 Model3, se
. est t Model1 Model2 Model3, star(.01 .05 .10)

. xi:reg read math science female i.race
. est store Model4
. est t Model4, se
. est t Model4, star(.01 .05 .10)
. est for Model4: test female
. est for Model4: test _Irace_2 _Irace_3 _Irace_4
```

2. *Specify particular model-fit statistics to compare & display.*

```
. ereturn list [list matrix coefficients for current regression model]
. est t Model1 Model2 Model3, stats(N r2 r2_a) [specify selected model statistics]
```

3. *Clear the models from memory.*

```
. est clear
```

2. Log & Cmdlog: how to log Stata sessions. From now on you must log the results and commands for all Stata assignments (but turn in only the 'log' and 'cmdlog' commands, not the logs themselves).

```
. log using practice, replace [or: append]
. cmdlog using practice, replace [or: append]
```

[Note: The downloadable command 'log2do' turns logfiles into command files.]

* I will practice logging a Stata session's results and commands.

```
. use hsb2, clear
. d
. su
. su math, d
. codebook race
. insp race
```

* I will now practice turning log and command log off and on during a session.

```
. log off
. cmdlog off
. log on
. cmdlog on
```

* I will now practice making wrap-around lines during a session. I will ///
now practice making wrap-around lines. I will now practice making ///
wrap-around lines. I will stop practicing wrap-around lines because ///
I am tired of this.

```
. d race ses
```

* I will stop logging the results and commands. I will stop logging the ///
results and commands.

```
. log close
. cmdlog close
```

```
. view practice.log [or: Click log icon]
. view practice.txt
```

[Note: You can use your word processor to select log or cmdlog from the Windows-Explorer folder, edit, save, and print, but always save in ASCII or Stata format.]

3. Create variables

- Categorizing a quantitative variable can be problematic if the variable has pronounced skewness, but doing so can be useful in key ways. Here are various ways to categorize variables. Compare the results. In the long run you'll want to learn how to tailor the procedures to your particular needs.

```
. gr7 science, bin(8) norm
. su science, d
```

* Categorize where you specify:

```
. gen rscience=recode(science, 44, 53, 58, 74)
. tab rscience
. bys rscience: su science [or: tab rscience, su(science)]
. la var rscience "Quintiles science"
. note rscience: TS-- gen rscience=recode(science, 44, 49, 54, 58, 74)
```

* Five approximately equal intervals

```
. gen ascience=autocode(science, 5, 26, 74)
. bys ascience: su science
. la var ascience "Autocode-quintiles science"
. note ascience: TS-- gen ascience=autocode(science, 5, 26, 74)
```

```
. xtile xscience=science, nq(5)
. bys xscience: su science
. la var xscience "Xtile-quintiles science"
. note xscience: TS-- xtile xscience=science, nq(5)
```

* Five approximately equal groups of observations

```
. sort science
. gen gscience=group(5)
. bys gscience: su science [or: tab gscience, su(science)]
. la var gscience "Group-quintiles science"
. note gscience: TS-- gen gscience=group(5)
```

* "Manual," divided according to your specifications

```
. su science, d
. gen qscience=science
. recode qscience 0/44=1 45/49=2 50/54=3 55/58=4, 59/74=5
. tab qscience
. bys qscience: su science
. la var qscience "Quintiles science"
. note qscience: TS-- gen qscience=science, recode qscience 0/44=1 45/53=2
54/58=3 59/74=4
```

```
. su science, d
. gen qsci=recode(science, 44, 49, 54, 58, 74)
```

```
. su science, d
. recode science (0/53=0) (54/max=1) if science<., gen(scid)
. tab scid
. bys scid: su science
```

```

. la def scid 0 "low" 1 "high"
. la val scid scid
. tab scid
. bys scid: su science
. la var scid "low=0, high=1"
. note scid: TS-- recode science (0/53=0) (54/max=1 if max<.), gen(scid)

. compress [stores variables more efficiently to save memory]

. tab1 gscience-scid [How do the results differ for gscience-scid?]

```

- Creating squared variables (second-order polynomials)

```

. su read
. gen readsq=read^2
. su read readsq, d
. for varlist read readsq: gr7 X, bin(8) norm \ more ['bin()' changes # bins]
. for varlist read readsq: histogram X, norm \ more
. la var readsq "read squared"
. note readsq: TS--gen readsq=read^2
. compress

. xi:reg science female i.ses read readsq math

```

- Creating interaction variables

```

. su female science
. gen femXsci=female*science
. su fem science femXsci, d
. tab femXsci
. for varlist science femXsci: gr7 X, bin(8) norm \ more
. for varlist science femXsci: histogram X, norm \ more
. la var femXsci "female*science"
. compress

. xi:reg read female i.ses science femXsci

```

Stata shortcut for interaction variables:

```

. xi:reg read i.ses i.female*science [Note the difference in equation format versus
the standard equation.]

```

- On your own: How would you create a new quantitative or categorical variable by combining one or more other variables (e.g., `gen readwrite=read + write`; or create the variable 'ethnic' by combining, say, variables for 'black,' 'hispanic,' 'asian,' and 'white')? Regarding the new quantitative variable, what formulas give rise to the new mean and standard deviation (see Moore & McCabe)?

4. Scatterplots

```
. use WAGE, clear
. view help spar1
. for varlist wage educ: gr7 X, bin(8) norm \ more
. su wage educ, d

. spar1 wage educ
. spar1 wage educ, logy
. spar1 wage educ, logx
. spar1 wage educ, power [estimates both logy & logx at once]
. spar1 wage educ, quad

. gr matrix educ exper wage, half
. gr7 educ exper wage, matrix half c(s) bands(8)

. scatter wage educ, ml(id) || lfit wage educ
. twoway lfitci wage educ
. gr7 wage educ, twoway

. scatter wage educ, ml(id) || qfit wage educ
. twoway qfitci wage educ
. gr7 wage educ, twoway c(s) bands(8)

. scatter wage educ, ml(id) || fffit wage educ
. twoway fffitci wage educ

. twoway fffitci wage educ, bc(teal)
. twoway fffitci wage educ, bc(teal) scheme(economist)
. twoway fffitci wage educ, plotr(c(black)) bc(teal)
. twoway fffitci wage educ, plotr(c(black)) bc(teal) scheme(economist)

. scatter wage educ || lfit wage educ || fffit wage educ
. scatter wage educ || lfit wage educ || mbands wage educ
. scatter wage educ || lfit wage educ || lowess wage educ
```

5. Outreg: command for displaying regression output in publication format (see Allison, chap. 2; and *ASR*, *ASJ*, or *Social Forces* for professional publication format in sociology. See also *American Political Science Review* and *American Economic Review*).

Part A

```
. view help outreg
. use WAGE, clear
. for varlist wage educ exper: gr7 X, bin(8) norm \ more
. su wage educ exper, d
. gr7 educ exper wage, matrix half
. corr educ exper wage
```

```
. reg wage educ exper
. outreg using wage, se 3aster replace
. reg wage educ exper tenure
. outreg using wage, se 3aster append [then use word processor to select the
outreg file, format, and save as a word-processed document by, e.g., choosing the
appropriate options for "save as Word document."]
```

[See view help outreg concerning titles, notes, etc. for regression output tables.]
[See also *view help estimate* for "estimate tables."]

Part B

Repeat the outreg command for the first model above, but this time include the following output in professional publication format: R2=...; Adj R2=... To obtain the appropriate commands, after estimating each regression model type:

```
. reg wage educ exper
. ereturn list
. outreg using wage, se 3aster addstat(e(r2), e(r2_a)) replace
. outreg using wage, se 3aster addstat(e(r2), e(r2_a)) append [to add other models]
```

6. Log transformations

Part A

Under what circumstances is it valid to carry out a log transformation of a variable? Under what circumstances is it valid to perform a quadratic transformation? What are the advantages of a quadratic transformation versus a log transformation? Under what circumstances might it be advantageous to categorize an explanatory variable rather than perform a log or quadratic transformation? (See Allison on these issues.)

Part B

Outcome variable is log transformed

```
reg lwage educ exper
```

lwage	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
educ	.0979356	.0076224	12.85	0.000	.0829613	.1129099
exper	.0103469	.0015551	6.65	0.000	.0072919	.013402
_cons	.2168544	.108595	2.00	0.046	.0035184	.4301904

Interpretation #1

- Every unit increase in educ increases lwage by .098 on average, holding the other variable constant.
- Every unit increase in exper increases lwage by .01 on average, holding the other variable constant.

Interpretation #2: recommended

[take exponent of educ]

di exp(.0979356)

1.1028918

[take exponent of exper]

. di exp(.0103469)

1.0104006

- Every unit increase in educ multiplies wage (i.e. the original, not the log transformed, variable 'wage') by 1.103 on average, holding the other variable constant.
- Every unit increase in exper multiplies wage (i.e. the original, not the log transformed, variable 'wage') by 1.01 on average, holding the other variable constant.

Interpretation #3: recommended

di 100*(exp(.0979356) - 1) [express 1.10289 as %]

10.29

di 100*(exp(.0103469) - 1) [express 1.0104 as %]

1.04

- Every unit increase in educ increases wage (i.e. the original, not the log transformed, variable 'wage') by 10.3% on average, holding the other variable constant.
- Every unit increase in educ increases wage (i.e. the original, not the log transformed, variable 'wage') by 1.04% on average, holding the other variable constant.

Explanatory variable is log transformed

reg educ lwage

educ	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
lwage	2.245548	.2053479	10.94	0.000	1.842142	2.648955
_cons	8.91761	.350718	25.43	0.000	8.228624	9.606596

Interpretation #1

- Every unit increase in lwage increases education by 2.25, on average.

Interpretation #2

di exp(2.245548)

9.4455903

- Multiplying wage (i.e. the original, not the log transformed, version of 'wage') by 2.718 (the unit increment of natural log) increases educ by 9.45, on average.

Interpretation #3: recommended

Express lwage in terms of 10% change: $\log([100+10]/100)=\log(1.1)$

According to this formula: 10% change in lwage= $\log(1.1)$
 20% change in lwage= $\log(1.2)$
 30% change in lwage= $\log(1.3)$
 etc.

```
di log(1.1)*2.245548
.21402358
```

- Every 10% increase in wage (i.e. in the original, not log transformed, variable 'wage') multiplies educ by 0.214, on average.

Outcome variable & explanatory variable are log transformed

```
reg lwage leduc
```

lwage	Coef.	Std. Err.	t	P> t	[95% Conf. Interval]	
leduc	.8182079	.091271	8.96	0.000	.6389059	.99751
_cons	-.5493537	.2433165	-2.26	0.024	-1.027349	-.071358

Interpretation #1

- Every unit increase in leduc increases lwage by .82, on average.

Interpretation #2: recommended

- Every 1% increase in educ (the original, not the log transformed, variable 'educ') increases wage (the original, not the log transformed, variable 'wage') by .818%, on average. [This—perhaps the most common way of interpreting $\log y/\log x$ —is called an 'elasticity': the percent change in one variable given a percent change in another variable.]

Interpretation #3

```
di exp(.8182079)
2.2664345
```

- Multiplying educ (the original, not the transformed, variable 'educ') by 2.718 (i.e. the unit increment of natural log) multiplies wage (the original, not the transformed, variable 'age') by 2.72.

Interpretation #4: recommended

```
di exp((.8182079)*ln(1.1))
1.0811049
```

- Every 10% increase in educ (the original, not the transformed, variable 'educ') multiplies wage (the original, not the transformed, variable 'wage') by 1.08, on average.

Interpret the following log-transformed variables in the various recommended ways:

```
. use WAGE, clear
. gen lwage=ln(wage)           [or log(wage); if not already in data set]
. gen leduc=ln(exper)         [or log(wage); if not already in data set]

. su educ exper lexper wage lwage, d
. for varlist educ exper wage lwage: gr7 X, bin(8) norm \ more
. gr7 educ exper lexper wage lwage, matrix half
. corr educ exper lexper wage lwage

. reg lwage educ exper
. reg wage lexper
. reg lwage educ lexper
```

7. Do-files

See *Getting Started with Stata* and Long/Freese on do-files.

Part A

```
. doed           [then type the following]
```

```
version 8.2
```

```
* lwage.do: this a do-file for regressing lwage in WAGE.dta.
```

```
. capture log close
. set more off
. log using wage, text replace
. use WAGE1, clear
```

```
. d
. compress
. su
. su, d
```

```
* Create an id variable and move it to the top of the variable list.
```

```
. gen id=_n
. l id in 1/10
. l id in -10/1
```

```

. move id wage

. count      [does not permit listing of variables]
. nmissing   [works with or without variables listed]

. tab1 female smsa

. reg lwage educ exper female smsa
. outreg using lwage, se 3aster addstat(e(r2), e(r2_a)) replace

. clear
. log close          [then save as "lwage" before exiting]

```

Part B

To see and perhaps edit the do-file, type the following in the Command Window:

```
. doed lwage
```

To run the do-file, type the following in the Command Window:

```
. do lwage
```

Then open the do-file, edit in a word processor, & save **not as, e.g., a Word or WordPerfect file but rather as a text file**, so that it works properly in Stata.

Note: To automatically turn a log file into a do-file, download the command 'log2do' and follow the instructions.

8. Diagnostics A

```

. use WAGE, clear
. for varlist educ exper wage lwage: gr7 X, bin(8) norm \ more
. su educ exper wage lwage, d
. gr7 educ exper wage lwage, matrix half c(s) bands(8)
. corr educ exper wage lwage

. reg lwage educ exper

```

Model specification diagnostics

```

. linktest
. ovttest
. rvfplot, yline(0)

```

Normality of residuals diagnostics

```
. predict rstu if e(sample), rstu    [to obtain unstandardized residuals:  
                                   predict resid, resid]  
. kdensity rstu, norm  
. su rstu, d  
. imtest
```

Heteroscedasticity (non-constant variance)

```
. hettest, rhs mt(sidak)  
. szroeter, rhs mt(sidak)  
. imtest                                [but hettest &/or szroeter are preferable]  
. rvfplot, yline(0)  
. rvppplot educ, yline(0)  
. rvppplot exper, yline(0)  
. hettest educ                          [then use any graphs that help you out]  
. qladder educ  
. ladder educ  
. hettest exper  
. qladder exper  
. ladder exper
```

[One possible solution for non-constant variance is to incorporate previously omitted variables. If transformations—another possible solution—are advisable, consider not only the ones suggested by qladder & ladder but also the possibility of using interaction terms or categorizing the explanatory variables in question. Regarding this particular example, do not transform the dependent variable because it is already in proper functional form. See Allison & Mendenhall/Sincich on how to address heteroscedasticity, etc.]

[Continue in “Diagnostics B.”]

9. Diagnostics B

[Continue where you left off in “Diagnostics A.”]

```
. gen id=_n                            [create an id variable]  
. move id wage  
. save, replace
```

Outliers/influence diagnostics

```
. lvr2plot, ml(id)  
  
. set textsize 170  
. avplots  
. avplot educ, ml(id)  
. avplot exper, ml(id)
```

```
. predict rstu if e(sample), rstu           [actually this was generated earlier]
. predict h if e(sample), hat
. predict d if e(sample), cooks d
. dfbeta                                   [this command is correct]
. su rstu h d DFeduc DFexper, detail
```

Scan the summarized values to see if any of them indicate problems according to the following thresholds:

```
rstu: abs(rstu)>3
h: >3*K/_N      display 3*K/(_N)    K=#explanatory vars. _N yields sample # obs.
d: >1 or, for large samples, >4/_N  display 4/(_N)
DF: abs(DF)>1 or, for large samples, abs(DF)>2/sqrt(_N)  display 2/sqrt(_N)
```

Here's an example of scatterplots to graphically assess the possible problems:

```
. scatter id d, yline(-1 1) ml(id)
. scatter id d, (yline 4/_N), ml(id)    [for large sample]
```

To explore possible problems:

```
. list id d wage exper educ if abs(rstu)>=3
. list id d wage exper educ if hat>=(3*K)/(_N)
. list id d wage exper educ if d>1      [or d>4/_N: large sample]
. list id d wage exper educ if abs(DFeduc)>1 [or abs(DFeduc>2/sqrt(_N)): large sample]
```

[Include other variables that might provide insight into the problems, such as gender, race-ethnicity, ses, urban-rural.]

Multicollinearity

```
. corr, _coef
. vif           [VIF>10 or 1/VIF (called "tolerance")<.10]
```

[Other indicators of multicollinearity: F-test is significant but none of the individual explanatory variables is significant; highly inflated standard errors; highly inflated coefficients in opposite direction of what's expected; coefficients and standard error or unstable when other variables are deleted from or added to the model]

[See Allison on how to address multicollinearity]

[If the model included one or more key categorical explanatory variables, at this point we would test the validity of the assumption of uniform slope coefficients. We would do so by interacting one or more of the key categorical explanatory with the other explanatory variables and conducting nested model F-tests.]

10. Categorical dependent variables, part 1

```
. use binlfp2, clear [Long/Freese data: follow the steps for binlfp2.dta on pp. 114-22]
. logistic lfp k5 k618 age wc hc lwg inc, nolog
```

[After you've arrived at your most parsimonious model, do the following diagnostics according to the Power Point slides.]

```
. linktest, nolog  
. ldev [download: ldev lfit]  
. lfit  
. lfit, g(#) table [Interpret the results.]
```

[Predict the probabilities if e(sample) and predict the following outlier-influence diagnostics: db, dd, dx, hat, and n. Graphically and numerically summarize them, and interpret the results.]

[Make any necessary adjustments to the model. Test the equality of slopes assumption via one or more interaction variables. Estimate the final model using the *robust* option and interpret the final model]

[Do the exercises on pp. 131-39 (middle of page), 141(bottom of page)-49. Simply follow the instructions.]

11. Categorical dependent variables, part 2

```
. use ordwarm2, clear [Long/Freese data: follow the steps for ordwarm2.dta on pp.  
157-163 ("Testing multiple coefficients"), 165-69, 171-77, 181-84]
```

```
. use nomintro2, clear [Long/Freese data: follow the steps for nomintro2.dta on pp.  
196-205, 211-29]
```

12. Helpful Commands, Data Checking/Cleaning, & Data Manipulation

Helpful commands

- See Long/Freese on the command "profile" to automate various features in Stata.
- To create a table of specific statistics for specific variables & to limit the decimals to two places:

```
tabstat read write math science, stats(mean med sd min max) format(%9.2f)
```

```
tabstat read write math science, stats(mean med sd min max) format(%9.2f)  
col(stats)
```

```
tabstat read write math science, stats(mean med sd min max) format(%9.2f)  
by(female) col(stats)
```

- Here's another way to make such a table:

```
makematrix, from(r(mean) r(p50) r(sd) r(min) r(max)) format(%9.2f): su read-  
science, detail
```

```
makematrix, from(r(rho)): corr read-socst
makematrix, from(r(rho)) cols(socst) format(%9.2f): corr read-socst
```

Other such useful, downloadable commands: statsmat, tabout (contingency tables),
mkcorr (for correlation matrices), estout (for regression tables)

The above commands are useful to create tables of descriptive statistics or other results (see articles in, e.g., *American Sociological Review*, *American Journal of Sociology*, or *Social Forces*; and see *view help tabstat* or *view help makematrix*).

- To sort data in descending order:

```
sort math [ascending order]
list math
```

```
gsort -math [descending order]
list math
```

- To list lowest & highest values (or download the 'hilo' command):

```
sort math
list math in 1/10 [list 10 lowest values]
list math in -10/1 [list 10 highest values]
```

Data cleaning & checking exercises

Note: Most of the following is best done as a do-file, tailored to your particular data set.

```
. log using check_hsb2, text replace
. cmdlog using check_hsb2, text replace
. u hsb2_miss, clear
```

*** Part I**

* Point of departure: Do any of the summary values (#obs, min, max, mean, median, sd) look questionable in terms of what you expect to find? See Allison, pages 28-29.

```
. d
. su
```

*** *Are there duplicate observations?***

* Check id:

```
. sort id
. duplicates list id
. duplicates drop in ... [enter obs #'s of duplicate cases to drop]
```

*** *Check for duplicates in general:***

```
. duplicates report
```

. duplicates list

* If a duplicate includes not just id but other variables as well, delete the observation's entire row of variables, e.g.: drop in 27 [27 refers not to the id# but to the observation#]

* **Are there missing values?** See Allison, page 29.

. count

. nmissing

. mvpat id-socst, skip

. egen mvals=rmiss(_all)

. tab mvals

. list id female ses race if mvals==...

* **Is there an id &/or socioeconomic profile for each level of missing values, & for missing values in general?** Explore other variables, too, including test scores.

. gen mv1=(mvals==1)

. tab mv1

. gen mv2=(mvals==2)

tab mv2

* Do for each level of missing values.

. tab mv1 mv2, chi2

. tab mv1 mv..., chi2

* Do any statistically significant patterns of missing values emerge?

* **Are there coding errors?**

* **Check variable labels:**

labelbook, problem

* **Check numeric coding of categorical variables.** (Note: *assert* is best suited for use in do-files):

. assert id>=1 & id<=200

. assert female==0 | female==1 if female<.

. codebook female

assert race>=1 & race<=4 if race<.

. codebook race

. assert ses>=1 & ses<=3 if ses<.

. codebook ses

. assert schtyp==0 | schtyp==1 if schtyp<.

. codebook schtyp

. assert prog>=1 & prog<=3 if prog<.

. codebook prog

* **For quantitative variables with less than about 20 values, check that their distributions make sense.** E.g., for years of schooling in the U.S., check that the variable's frequencies are greatest at 12(high school), 14(associate's degree), and 16(college degree): tab educ [this variable is not in the data set].

* **For quantitative variables with more than about 15-20 values, check that there are no extremely unusual or impossible values,** such as negative

achievement test scores. Recall that 'assert' works best in a do-file. Let's say that on these achievement scores it's impossible to score less than 20 or more than 80:

```
. assert read>=20 & read<=80 if read<.
. codebook read, mv
. assert write>=20 & write<=80 if write<.
. codebook write, mv
. assert math>=20 & math<=80 if math<.
. codebook math, mv
. assert science>=20 & science<=80 if science<.
. codebook science, mv
. assert socst>=20 & socst<=80 if science<.
. codebook socst, mv
```

* Part II

1. Quantitative variables

univariate: do for each quantitative variable

```
. for varlist read-socst: kdensity X, norm \ more
. for varlist read-socst: gr7 X, box \ more
. su read-socst, d
```

bivariate:

```
. gr7 read-socst, matrix half c(s) bands(8) [best to list outcome variable last]
. corr read-socst [or: pwcorr read-socst, obs bonf sig star(.05)]
```

2. Categorical variables

univariate: do for each categorical variable

```
. tab1 female-prog, miss plot
```

bivariate: cross-check each categorical variable with other key categorical variables, including create cross-check variables where appropriate (e.g., hhead with adult, married with adult; emp with labor force participation, emp with unemp):

```
. tab2 female-prog, miss chi2
```

The following are not in this data set but provide examples. Create a variable that=1 if there's a coding error (i.e. that indicates a coding error):

* Confirm that all married observations are adults:

```
. gen errormarr=1 if marr==1 & adult==0
. tab errormarr
. l id age if errormarr==1
```

or: tab marr if marr==1 & adult==0
. l id marr age if marr==1 & adult==0

* Confirm that, according to a questionnaire's instructions, only hhead (household heads) answered the question about hinc (household income). In this example, we check if a hhead has a non-missing answer for hinc:

```
. gen errorhinc=1 if hhead==0 & hinc<.  
. tab errorhinc  
. l id hhead hinc if errorhinc==1
```

3. Quantitative & categorical variables

bivariate: cross-check each quantitative variable with key categorical variables.

```
. tabstat read-socst, by(female) stats(mean med sd min max)
```

* **Do the following for each quantitative variable with key categorical variables:**

```
. sort female  
. by female: gr7 read, box  
or: gr box read, over(female, total)
```

```
. ttest read, by(female) unequal  
. oneway read ses, table bonf [because categorical variable is multilevel]
```

Multivariate: do the following for each quantitative variable with a key categorical variable:

```
. sort female  
. for varlist read-socst: gr7 X, box by(female) total \ more  
or: scatter read write || lpfif read write, by(female, total)
```

```
. log close  
. cmdlog close
```

Check for possible interactions: do three-way tables& graphs to explore possible interaction terms or other issues of causality (see Agresti & Finlay). Let's say that 'science' score is the outcome variable:

```
. scatter science math, ml(id) || fpfif science math, by(white)  
. bys white: tab female, su(science) [A linear trend or not?]  
. xtile qsci=science, nq(4)  
. tab qsci, su(science)  
. bys white: tab qsci white, col chi2 [A linear trend or not?]
```

Data manipulation

*** Part I: How to “match merge” two data sets having same id-variable**

```
. use odd, clear [two data sets: odd.dta & even.dta]
. list
. sort number [sort number, which is the id-variable]
. save, replace

. use even, clear
. list
. sort number [sort number, which is the id-variable]

. merge number using odd, unique [i.e. merge using the id-variable & 'unique';
                                see Manual, page 435, on 'unique'.]

. duplicates number [See Manual, pages 445-46: check for id duplicates.]
. sort number
. list
. describe [see newly created variable ' _merge'.]
. tab _merge [See Manual, page 443. Do the following if tab
             _merge=3 only. If otherwise, see page 443.]
. drop _merge
```

[See instructions for 'append data.']

*** Part II: How to convert a data set from wide to long format or vice versa**

```
. use wide, clear
. describe
. list

. reshape long lrn, i(id) j(year)
. describe
. list
. table year, c(n lrn mean lrn sd lrn)

. reshape wide
. describe
. list

. reshape long [Stata remembers how you did it the first time.]
. describe
. list
```

*** Note: How can you create a summary dummy variable (e.g., female=1 male=0) from 'wide format' data from 'gender' for person1, person2, person3, etc., without converting from wide to long format?**

12. Survey Statistics: for future reference

See UCLA/ATS, Stat 130 Class Notes, "Survey Sampling" for an introduction with practice data; and see Levy & Lemeshow (1999), which is the principal source of the following notes. The commands changed in Stata-8.

Stata survey-statistics format commands:

```
. svydescribe
. svyset, clear

. svyset [pw=weightvar], stata(stratvar) psu(psuvar) [last term is name of variable]
. svydes
```

- See Stata instructions for the command *sampsi*.
- svy commands with "if" give the wrong CI; use *subpop* or *by* instead of "if" (the same applies to bootstrapping)
- do not use LR-test with svystats, or otherwise with pweight. Instead use either *svytest* (if # clusters < ~100) or *test* (which in this instance gives a Wald test)
- *Finite population correction (fpc)*: computes adjusted N for se estimates ($N - n/N$); used only in simple random sampling without replacement. That is, it accounts for the reduction in variance that occurs when sampling **without** replacement from a finite population, as compared to sampling **with** replacement. Use *fpc()* option for cases of simple random sampling or stratified random sampling without replacement of psu's within each stratum with no subsampling within psu's. **Including fpc() reduces the variance estimate, but minimally if N-psu's is large relative to sampled-n-psu's.** To use, set *fpc()* to *Nh*, the var representing the total number of psu's per stratum in the population (e.g., *hid* in datafile comprised of individual household members). **Caution: you must know the total population-parameter of the pertinent var (e.g., total pop-N in each stratum or cluster) in order to use fpc, so thus we will rarely if ever use fpc**
- Recall that that unequal nonresponse rates across strata &/or clusters must be adjusted for

How to set survey-stats in Stata for various samples

simple random sample

random sample of one hospital from a population (hospno) of 25 hospitals; enumeration unit (i.e. population) is #births (N=773) in the sampled hospital during the previous year; in sum, what Stata needs for a simple random sample is (1) weighting var & (2) total enumeration units of population (N) from which sample is drawn

```
. svyset [pw=weight1], fpc(birth)          momsag.dta; sampling weight=N/n
```


- . svyset [pw=weighta], strata(oblevel) identifies the total # of strata (1-3) in the pop
- . svyset fpc tothosp identifies total population-N within each obs's stratum; yields corrected pop size
- [psu—not entered—#obs]
- . svytotal births births: #births in each hospital during previous yr
- . svytotal births, by(oblevel)

in sum: use this approach (fpc tothosp) only when you know the total population-N within each stratum & you are drawing a sample from this total-N; this is not applicable to, e.g., the World Bank surveys because we don't know the total population-N in each city for each stratum (in such cases use only pweight & strata, or, if applicable, use pweight, strata & psu [clusters]; see below)

stratification after sampling

- . svyset [pw=sampwt], strata(stratum) fpc(npop) jacktwn2.dta
 identifies the # of strata in the pop identifies total pop-N within each obs's stratum; yields corrected pop size
 [psu—not entered—yields #obs]
- . svytotal twin
- . svytotal twin, by(quart1)

in sum: use fpc npop only if you know total population-N for each stratum, which is not the case for the World Bank surveys (see notes under stratified random sampling)

one-stage clustered sampling

- . svyset [pw=wt1], fpc(M) psu(devlpmnt) tab9_la.dta
 identifies the total # of clusters (1-5) in the pop
 identifies the sampled clusters (2, 5) only
- . svytotal nvstnrs nge66
- . svymean nvstrs hhneedvn
- . svyratio nvstnrs nge65
- . svymean nge65dv

in sum: use this approach (fpc M) only if you are drawing a sample of clusters from a total population-N of clusters, which is not the case in the World Bank surveys

two-stage clustered sampling

- . svyset [pw=w], psu(hospno) il10pt2.dta
 hospno=total # of clusters in the pop
- . svytotal dxdead dxdead=admitted or discharged dead
- . svyratio dxdead lifethrt lifethrt=admission of patient with life-threatening illness

in sum: in this configuration, Stata cannot use both psu & fpc, so the latter is not used (see example for il10pt2.dta in Levy & Lemeshow)

probability proportional to size sampling

```
. svyset [pw=wstar], psu(drawing)      hospslet.dta
                                       drawing=# clusters

. svytotal lifethrt dxdead
. svyratio dxdead lifethrt
```

in sum: in this configuration, Stata cannot use both psu & fpc, so the latter is not used

Stata survey-set & describe procedures

```
. svyset [pw=wtvar], strata(res_ses) psu(hid)    example from World Bank Teguc &
                                                Salv: individual-level data
                                                hid: for individuals clustered by
                                                households

. svydes
. svyset, clear                               to delete svyset
```

in sum: use fpc only if you know the total pop-parameter (e.g., total pop-N for each stratum or cluster)

Stata survey statistics

```
. svytab xcat1 xcat2: two-way table
. svytab xcat1 xcat2, col
. svytab xcat1 xcat2, row per
. svytest x1 x2: hypothesis test              or test if cluster size>100
. svylc: estimate linear combinations such as differences of means and of regression
coefficient
. svymean
. svyprop
. svyratio
. svytotal

. svyreg
. svylogit
. svyolog    eform
. svymlog    rrr
. svyprobt
. svyoprob
. svyintrg
. svypois
```

Examples

```
. svydes
. svymean dwelf [note: var must not be weighted separately/individually,
but rather only via the svy-weighting procedure]
. svymean dwelf, subpop(lte730)              var: lte730=1
. svymean dwelf, subpop(lte730) in f/50 (complete) (95) [or: (available) (99)]
. svyprop id, subpop(lte730)                [note: first sort id]
. svyset, clear
```

If necessary to collapse psu's & re-set survey stats:

```
. gen newstr=stratid  
  gen newpsu=psuid  
  replace newpsu=psuid + 2 if stratid==1  
  replace newstr=2 if stratid==1  
. svyset strat newstr  
  svyset psu newpsu  
  svydes x1, bypsu
```

```
. svymean x1 x2  
  svymean x1 x2, obs  
  svymean x1 x2, com ci deff  
  svytest x1 x2
```

complete: nonmissing only
no test with com, subpop or
by() in command
svylc requires equal # obs,
so must run com
beforehand

```
  svymean x1 x2, com ci deff  
  svylc x1 - x2, deff
```

```
. svymean x1 x2, subpop(female)  
  svymean x1 x2, subpop(female) obs  
  svymean x1 x2, subpop(female) com ci deff
```

no test with com, subpop
or by() in command

Note: "subpop" only works for a binary var's 1-level. So convert any multilevel categorical var into a series of 0/1 dummy vars; or for a dummy var create another, complementary dummy var: male 0=female 1=male; female 0=male 1=female. Then specify the 1-level of any dummy var as "subpop." Alternatively, simply specify, e.g., either "by(male)" or "by(female)" to yield the data for male & female.

```
. svymean x1, by(female)  
  svymean x1, by(female black)
```

```
. svyprop x1  
  svyprop, by(female) deff  
  svyprop, subpop(black) deff
```

no obs or com options

```
. svyratio x1 x2  
  svyratio x1 x2, obs  
  svyratio x1 x2, com ci deff
```

provides more accurate mean-estimate

```
. svytotal x1 x2  
  svytotal x1 x2, obs  
  svytotal x1 x2, com ci deff  
  svylc x1 - x2, deff  
  svytotal, by(female)  
  svytotal, by(female) obs  
  svytotal, by(female) com ci deff  
  svytotal, subpop(hispanic)  
  svytotal, subpop(hispanic) com ci deff
```

must have equal obs; run
com beforehand

```
. svytab x1 x2
```

row totals in small format

svytab x1 x2, row se ci svytab x1 x2, row se ci format(%7.4f) svytab, row se ci nomarg	row totals in large format no row totals
. svytab x1 x2 x3, tab(x4) row svytab gender race, tab(income) row	computes proportions relative to a specified var
. svyreg y x1 x2 x3 svyreg y x1 x2 x3, deff linktest svytest x1 x2 svylc x1 – x2, deff	
. svylogit y x1 x2 x3 linktest svylogit y x1 x2 x3, deff svylogit, or di(or – 1)*100 svytest x1 x2 svylc x1 – x2, deff	displays previous results as odds ratio; display % change
<i>To combine subpop() with by():</i> . gen black=(race==1) if race!=. svymean x1, subpop(black) by(marital age20)	<i>do not use if x1==</i>
. svymean, ci deff deff meff meff obs size	deff & ci are default
. svytotal x1, by(female)	same syntax as svymean
. svyratio x1 x2 svyratio x1/x2	computes x1/x2 ratio no obs or com
<i>[alternatively: svymean x1, subpop(x2)]</i>	perhaps easier to do if x2 is set up appropriately
. svyprop x1 x2 svyprop x1 x2, subpop(female) svyprop x1, by(white female)	
. svyset fpc hid svyset svymean x1	see Levy & Lemeshow
. svyreg y x1 x2 x3 x4 svytest x1 x2 svytest x1 x2, b	
. svymlogit health female black age age2 svytest [good] female=[excellent] female, notest svytest [good]black=[excellent]black, accum	

How to do a design-based analysis (Levy & Lemeshow, chap. 16)

1. Identify the following elements of the sample design: stratification; clustering vars used; pop sizes required to determine fpc's
2. Using the above info, determine the sampling weight for each sample object
3. Determine for each sample record a final sampling weight that takes into consideration any nonresponse & poststratification adjustments that are desired
4. Ensure that all stratification, clustering, & pop size data required for an appropriate design-based analysis are identified on each sample record
5. Determine the procedure & set of commands for performing the required analysis for the particular software package used
6. Run the analysis & interpret the findings

How to incorporate stratification on several vars simultaneously [Stata digest v4 #872]: the following *does not work* if unequal sampling was done at the various levels of stratification; in that case, use SUDAAN)

1. egen stratvar=group(industry region size)
2. do crosstabs of psu with each individual stratification var to see if there's overlap

How to deal with the following design: persons clustered at various sites, which in turn are stratified by geographic region--use pweight & cluster options, & include fixed effects for the regions [Stata digest v4 #872]

How to combine two weighted data set:

1. Prepare data set 1:
 - a. Compute new weight for data set 1:
 - i. $newwt1 = n1(n1+n2)$, tab newwt1, la var newwt1 "n1(n1+n2)"
 - ii. gen wtcombine1=newwt1*wtoriginal1, la var wtcombine1 "newwt1*wtoriginal", note wtcombine1: newwt1=n1(n1+n2), gen wtcombine1=newwt1*wtoriginal1, tab wtcombine1, list wtcombine1 wtoriginal1
 - b. Create new id #'s for data set 1: gen id1=id, la var id1 "id #'s for data set 1"
 - c. Create value=1 of a new dummy variable, 1=data set 1: gen dataset1=1, la var dataset1 "1=data set 1, 0=data set 2", list dataset1
 - d. sort id id1
 - e. save, replace
2. Prepare data set 2:
 - a. Compute new weight for data set 2:
 - i. $newwt2 = n2(n1+n2)$, tab newwt2, la var newwt2 "n2(n1+n2)"
 - ii. gen wtcombin21=newwt2*wtoriginal2, la var wtcombine2 "newwt2*wtoriginal2", note wtcombine2: newwt2=n1(n1+n2),

- ```

gen wtcombine2=newwt2*wtoriginal2, tab wtcombine2, list
wtcombine2 wtoriginal2

```
- b. Create new id #'s for data set 2: gen id2=id, la var id2 "id #'s for data set 2"
  - c. Create value=0 of the new dummy variable, 1=data set 1, 0=data set 2: gen dataset1=0, la var dataset1 "1=data set 1, 0=data set 2", list dataset1
  - d. sort id, id2
  - e. save, replace
3. Append dataset 2 to dataset1 (see *view help append*)
    - a. tab dataset1, list dataset1 if dataset1==1, list dataset1 if dataset1==0, su if dataset1==1, su if dataset1==0, tab1 wtoriginal1 wtoriginal2 wtcombine1 wtcombine2
    - b. Create new combined weight: gen wtcomb=wtcombine1 if dataset1==1, gen wtcomb=wtcombine2 if dataset1==0, tab1 wtcombine1 wtcombine2 wtcomb, list wtcombine1 wtcomb if dataset1==1, list wtcombine2 wtcom if dataset1==0, la var wtcomb "Weights for combined data sets", note wtcom: gen wtcomb=wtcombine1 if dataset1==1, gen wtcomb=wtcombine2 if dataset1==0
    - c. Check further that data sets have been correctly combined, e.g.: su if dataset1==1, su if dataset1==0.
    - d. save combined datasets1\_2