S650: Lab Guide for Stata

Jun Xu, Scott Long, and Simon Cheng

04lhs_guide.doc – 5/27/2004

This *Guide* will structure your work in labs. The amount of time that you spend on each step of the *Guide* will depend on your past experience using Stata, your familiarity with the method being discussed, and your interest in a given topic.

The *Guide* is divided into sections corresponding to the class lectures, and in lab you should work through the section of the *Guide* that corresponds to the day's lecture. Feel free to experiment along the way. Note that the guide does not include all of the output for each command. To see the full output, you can run the do files associated with each section of the guide (e.g., lhs_guide01.do).

When you have worked through a section of the *Guide*, you should start with the assignment that will be handed out in lab.

Here are several rules to keep in mind when reading this lab guide:

a. In this *Guide*, all Stata commands start with a period (.). If we extend a command to multiple lines, we put a > sign at the very beginning of those lines. In your do file or at the command line, do NOT type the period "**.**" or ">". In each section, we have a summary of key commands. In these summaries, words *italicized* within a command are those you can change or modify. Usually they are new variable, file, or object names.
b. Results produced by Stata are in `this font.`
c. If part of a command is in **bold** and the rest is not bold, this means that the bold portion of the command can be used as an abbreviation.
d. Some commands were written by Scott Long and Jeremy Freese; these are indicated by **SPost**. See Chapter 2 of Long and Freese (2003) for details.
e. Other commands are from the Stata Technical Bulletin which can be obtained from www.stata.com. These commands are marked **STB**.

3. Occasionally, this *Guide* will use **L&F** to reference relevant discussions in Long and Freese's *Regression Models for Categorical Dependent Variables Using Stata*. For example, "see L&F: 2.7.1" means you can go to Chapter 2, Section 2.7, and Subsection 2.7.1 for detailed information.

1: Introduction to Stata

The commands from this section are in the file `lhs_guide01.do`.

There are three ways that you can enter commands using Stata. First, the command line. This is the box at the bottom of the screen. Type the command and press enter and the command is executed. Second, use the GUI (graphical user interface) to pull down dialog boxes. Fill in the blank and away you go. The resulting command appears in the output window along with the output. Third, type the commands into a text file with a name such as lhs_guide01.do. Then, from the command line, type: `do lhs_guide01` . We'll have you start by entering commands from the command line. Then, we'll tell you how to use a do files. Methods 1 and 2 are often useful, but do files should be your primary method of working. Why? Because you can easily modify and redo your work in the future. For details on interactive versus command files use of Stata, see the Appendix of this guide.

Trying the Commands from the Command Line

1) Set up a Working Directory. It is a good idea to create a working directory, place your data in the subdirectory, and perform your computation in this subdirectory. This avoids needing to specify the drive path. For now, we assume that you are working on the subdirectory c:\temp and this is also where your data are located. You can change directories with the `cd` command. For example, `cd c:\work` changes to directory work on drive c. Let us assume our current working direcotry is c:\work and want to reset to c:\temp, we then type:

```
. cd c:\temp
```

2) Open a Log. The first step is to open a log file for recording your results (see L&F: 2.6). Remember that all commands are case sensitive.

```
. capture log close
. log using lhs_guide01.log
```

3) Load the Data. We use the data set containing information on the careers of 308 Ph.D. biochemists. A comma demarcates the end of a main command and signals the start of an option string. The `clear` option tells Stata to "clear out" any existing data from memory before loading the new data set. The extension `.dta` is assumed when the `use` command is used. For details, see L&F:2.7. In the command window, enter:

```
. use science2, clear
```

4) Examine the Data Set. The `describe` command gives information about the data set.

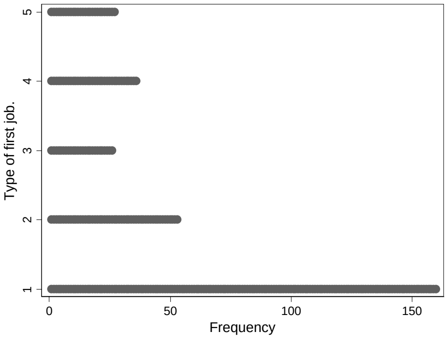
```
. describe

Contains data from science2.dta
  obs:           308                          Some of the variables have been
                                                artificially constructed.
 vars:            35                          19 Jun 2002 12:23
 size:        18,172 (99.9% of memory free)   (_dta has notes)
------------------------------------------------------------------------------
              storage  display     value
variable name   type   format      label      variable label
------------------------------------------------------------------------------
id              float  %9.0g                  ID Number.
cit1            int    %9.0g                  Citations: PhD yr -1 to 1.
::: output deleted :::
faculty         byte   %9.0g       faclbl     1=Faculty in University
jobrank         byte   %9.0g       joblbl     Rankings of University Job.
totpub          byte   %9.0g                  Total Pubs in 9 Yrs post-Ph.D.
                                              * indicated variables have notes
------------------------------------------------------------------------------
Sorted by:
```

5) Examine Individual Variables. A series of commands tell us about individual variables. You can use whichever command you prefer. For details about the `summarize` command, see L&F: 2.11.2. For details about the `tabulate` command, see L&F: 2.11, 2.12.2, 2.13.4

```
. sum work

    Variable |     Obs        Mean   Std. Dev.       Min        Max
-------------+-----------------------------------------------------
        work |     302    2.062914    1.37829          1          5

. tab work

    Type of |
 first job. |      Freq.     Percent        Cum.
------------+-----------------------------------
    FacUniv |        160       52.98       52.98
    ResUniv |         53       17.55       70.53
     ColTch |         26        8.61       79.14
     IndRes |         36       11.92       91.06
      Admin |         27        8.94      100.00
------------+-----------------------------------
      Total |        302      100.00
```

6) Graph Variables. Graphing is also a useful tool for examining data. Note that it takes a long time in Stata to create your first graph. Later graphs will appear much more quickly.

```
. dotplot work
```



7) Save Graphs. There are two formats for graphs. The `gph` format can only be used by Stata. We will use the `wmf` (Window metafile) format that can be used by other programs. To save a graph in wmf format you must have the graph window active. Then, go to the menu bar, click *file* and then select *save graph*. Change the graph type to Windows Metafile and enter a file name, ending in .wmf. An alternative method for saving graphs is the `graph export` command, which can be included in do files:

```
. graph export myname.wmf, replace
```

For details on graphing, see L&F: 2.16.

8) Add Comments. To add comments to your output, which allows you to document your command files, type * at the beginning of each comment line. The comment will be listed in the log file:

```
. * saving a graph
```

Another method is to put `//` between your command and your comments in one line like:

```
. tab work   //  tab is the abbreviation for tabulate
```

However, this method works only in the do-file editor (described later).

9) Create a Dummy Variable. Let's make a dummy variable with faculty in universities coded 1, all others coded 0. The command `gen isfac = work==1 if work<.` combines two statements: (1) generate faculty as a dummy variable; and (2) if work = 1, then faculty = 1, all others 0. The statement `if` `work<.` makes sure that missing values are kept as missing in the new variable. Note that there are various kinds of missing codings in Stata 8 (e.g., .a, .b). All missing codings are considered to be greater than any numeric values, and the system missing, coded as a period (.), is the smallest missing value. Therefore, to specify that a variable is not equal to any type of missing, we simply specify that variable is smaller than the value ".".Therefore,

```
. gen isfac = work==1 if work<.
(6 missing values generated)
```

Six missing values were generated since work contained 6 missing observations. For details on missing values, type `help missing`. For details about how Stata handles missing cases, please read L&F: 2.12.3.

10) Check Transformations. Transformations can be checked with a table. There are 302 cases, not 308, since 6 cases have missing values.

```
. tab isfac work

           |                   Type of first job.
     isfac |   FacUniv    ResUniv     ColTch     IndRes      Admin |     Total
-----------+-------------------------------------------------------+----------
         0 |         0         53         26         36         27 |       142
         1 |       160          0          0          0          0 |       160
-----------+-------------------------------------------------------+----------
     Total |       160         53         26         36         27 |       302
```

11) Label Variables and Values. It is very important to add variable labels and value labels. For many of the regression commands, value labels for the dependent variable are essential. The variable name `isfacfmt` is required to store the value labels. For details, see L&F: 2.14.1 and 2.14.2.

```
. label variable isfac "1=Faculty in University"
. label define isfacfmt 0 "NotFac" 1 "Faculty"
. label values isfac isfacfmt
. tab isfac

  1=Faculty |
         in |
 University |      Freq.     Percent        Cum.
------------+-----------------------------------
     NotFac |        142       47.02       47.02
    Faculty |        160       52.98      100.00
------------+-----------------------------------
      Total |        302      100.00
```

12) Create an Ordinal Variable. The prestige of graduate programs is often referred to in the categories of adequate, good, strong and distinguished. Here we create such an ordinal variable from the continuous variable for the prestige of the first job. The `missing` option tells Stata to show cases with missing values.

```
. tab job, missing

       job. |      Freq.     Percent        Cum.
------------+-----------------------------------
       1.01 |          1        0.32        0.32
        1.2 |          1        0.32        0.65
::: output deleted :::
        4.5 |          6        1.95       51.30
       4.69 |          5        1.62       52.92
          . |        145       47.08      100.00
------------+-----------------------------------
      Total |        308      100.00
```

The `recode` command makes it easy to create binary, ordinal and nominal variables.

```
. gen jobprst=job
. recode jobprst .=. 1/1.99=1 2/2.99=2 3/3.99=3 4/5=4
(162 changes made)
```

162 changes were made, not the total of 308. This occurs because the values of some cases were not changed. For details about using the `recode` command, see L&F: 2.13.3.

13) Label Variables and Label Values. Next we label the new variable:

```
. label variable jobprst "Rankings of University Job."
```

To label the values, we begin by defining the categories:

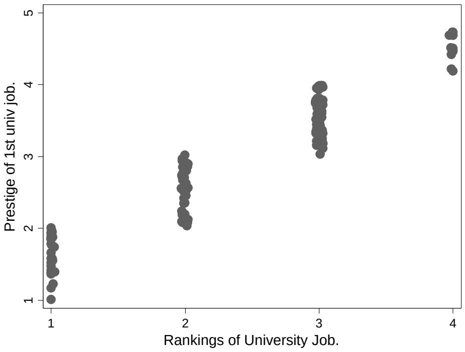
```
. label define jobprstfmt 1 "Adeq" 2 "Good" 3 "Strg" 4 "Dist"
```

The value 1 is given the label “Adeq”, and so on. Then we tell Stata that the label jobprstfmt goes with the variable jobprst:

```
. label values jobprst jobprstfmt
. tab jobprst

Rankings of |
 University |
       Job. |      Freq.     Percent        Cum.
------------+-----------------------------------
       Adeq |         31       19.02       19.02
       Good |         47       28.83       47.85
       Strg |         71       43.56       91.41
       Dist |         14        8.59      100.00
------------+-----------------------------------
      Total |        163      100.00
```

14) Check the Transformation. A graph is an easy way to check if chopping a continuous variable into categories was done correctly. The `jitter` option adds some random noise to give you a sense of how many cases are located in each cluster. The larger the number within the parenthesis, the more noise. The range of this number is from 0 to 30.

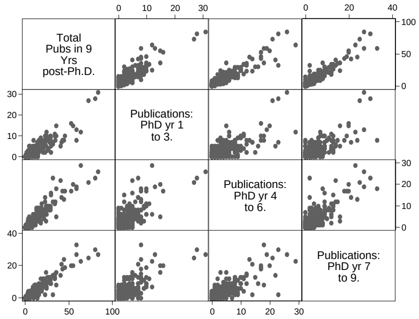
```
. scatter job jobprst, jitter(2)
```



15) Combine Variables. Now we create a new variable by summing existing variables. We add `pub3`, `pub6` and `pub9` to compute the total number of publications. The scatter plot matrix shows how all of the variables are related. For more discussion about the `gen` command, see L&F: 2.13.1.

```
. gen pubsum=pub3+pub6+pub9
. label variable pubsum "Total Pubs in 9 Yrs post-Ph.D."
. sum pub3 pub6 pub9 pubsum
```

```
    Variable |     Obs        Mean   Std. Dev.       Min        Max
-------------+-----------------------------------------------------
        pub3 |     308    3.185065   3.908752          0         31
        pub6 |     308    4.165584   4.780714          0         29
        pub9 |     308    4.512987   5.315134          0         33
      pubsum |     308    11.86364   12.77623          0         84

. graph matrix pubsum pub3 pub6 pub9
```



16) Save the New Data. After you make changes to your data set, it is a good idea to save the data with *a new file name*. Since we will use the variables we created in this exercise tomorrow, be sure to save your file. Here we saved a new data file scitemp.dta, and we will use this data file for later sessions. In case scitemp.dta is not correctly saved, we created a data file sciwork.dta, which is exactly the same as scitemp.dta and we will use sciwork.dta for later sessions. For details about using and saving data in Stata format, see L&F: 2.7.

```
. save scitemp, replace
file scitemp.dta saved
```

17) Close Log File. Last, we need to close the log file so that we can refer to it in the future.

```
. log close
```

Trying the Commands using the Stata menu system

The best way to understand how to do this is to explore the menus at the top of Stata. We'll illustrate a graphic command since we find that the menus help keep track of all of the many graphic options.

Press alt-g to open the graphics menu. Press t to make the cursor go to "Twoway graph". Press enter to open the dialog box. That's it, except for filling in the boxes.

Trying the Commands using a do file

Do files are simply text files containing Stata commands. To open a Stata do file, click the Stata do-file editor icon (the fifth icon from the right). In the Stata do-file editor window, you first go to the menu bar and make the following clicks: File → Open… and then open "*yourdofile*.do" from where it is saved.

To execute your do file, you simply need to click the icon (the second icon from right in the Stata Do-file Editor). If you do not have a do file, you need to type in the commands first. For example, to run all of the commands you entered in the command window, you can type them in the do-file editor (remember NOT to type out the period at the very beginning of each command line!):

```
. log using lhs_guide01.log
. use science2, clear
. describe
. sum work
. tab work
. dotplot work
. graph export myname.wmf, replace
. tab work   //  tab is the abbreviation for tabulate
. gen isfac = work==1 if work<.
. tab isfac work
. scatter work isfac, jitter(2)
. label variable isfac "1=Faculty in University"
. label define isfacfmt 0 "NotFac" 1 "Faculty"
. label values isfac isfacfmt
. tab isfac
. gen jobprst=job
. recode jobprst .=1/1.99=1 2/2.99=2 3/3.99=3 4/5=4
. label define jobprstfmt 1 "Adeq" 2 "Good" 3 "Strg" 4 "Dist"
. label values jobprst jobprstfmt
. tab jobprst
. scatter job jobprst, jitter(2)
. gen pubsum=pub3+pub6+pub9
. label variable pubsum "Total Pubs in 9 Yrs post-Ph.D."
. sum pub3 pub6 pub9 pubsum
. graph matrix pubsum pub3 pub6 pub9
. save scitemp, replace // Note: this file is used in later sections.
. log close
```

and click the icon. To save your do file, you go to File→Save as… in the do-file editor window and save your do file under some directory. Next time, you simply open this do file in the do-editor window and you can rerun what you did today (and easily make changes as needed).

In do files, you can easily extend commands over more than a single line. Normally, each Stata command ends when you enter a carriage return. But, if you end a line with a `///` before the carriage return, Stata lets you continue your line. For example:

```
. label define jobprstfmt 1 "Adeq" 2 "Good" 3 "Strg" 4 "Dist"
```

is a bit hard to read, and would be much harder if there were 10 or 20 categories. But, you can extend a command across lines using the `///` sign:

```
. label define jobprstfmt 1 "Adeq" ///
>                         2 "Good" ///
>                         3 "Strg" ///
>                         4 "Dist"
```

Note that the > sign is an indication of unfinished command in *log* files. So in your do file, please do NOT type the > sign. For details about alternative ways to deal with long lines, see L&F: 2.9.2. For details about data management, read Chapter 2 in L&F.

2: Linear Regression

The commands from this section are in the file `lhs_guide02.do`.

Summary of Key Commands

keep — Specify the variables or observations to be kept (see L&F: 2.12.4 and 2.12.5).

tabulate *varname(s)*, **m**issing — Check frequencies and the number of cases with missing values

	(see L&F: 2.11 and 2.12.2).
regress *depvar indepvarlist*	Estimate a linear regression of a dependent variable (*depvar)* on independent variables (*indepvarlist*).
listcoef, help	List the estimated coefficients; the `help` command lists the guidelines for interpretation and is *optional*. **SPost** (see L&F: 3.1.8, A.3).

Trying the Commands

1) Open a Log File. Always open a log file for recording your results.

```
. log using lhs_guide02.log, replace

------------------------------------------------------------------------------
       log:  lhs_guide02.log
  log type:  text
 opened on:  10 Jun 2004, 14:12:11
```

2) Load the Data. Let's use the data we created earlier. If you did not save a copy of sciwork.dta, you can load the file (also called sciwork.dta) we created for you.

```
. use sciwork, clear
```

3) Select Variables and Examine the Data. Use `keep` to select the dependent variable `pubsum` and the three independent variables, `faculty`, `enrol`, and `phd`, which we will use to construct regression models later. What the `drop` command does is to drop cases with missing values for any variables in the variable list, `pubsum, faculty`, `enroll,` and `phd.`

```
. keep pubsum faculty enrol phd
. tabulate faculty, m

  1=Faculty |
         in |
 University |      Freq.     Percent        Cum.
------------+-----------------------------------
     NotFac |        142       46.10       46.10
    Faculty |        160       51.95       98.05
          . |          6        1.95      100.00
------------+-----------------------------------
      Total |        308      100.00

. tab enrol, m
::: output deleted :::

. tab phd, m
::: output deleted :::
```

If we translate the following command into plain English, it reads "drop any cases if `pubsum` is missing, or if `faculty` is missing, or `enrol` is missing, or `phd is` missing." Note that `|` means "or".

```
. drop if pubsum>=. | faculty>=. | enrol>=. | phd>=.
(36 observations deleted)
```

4) Regression. Specifying a model is simple. The main thing to remember is that the dependent variable is listed *first*, followed by independent variables. Notice that the number of observations is 272, not 308. This is because cases with missing values for any of the variables in this regression analysis have been deleted by the `drop` command.

```
. regress pubsum faculty enrol phd
```

```
      Source |       SS       df       MS              Number of obs =     272
-------------+------------------------------           F(  3,   268) =   11.28
       Model |  3673.99731     3  1224.66577           Prob > F      =  0.0000
    Residual |   29095.885   268  108.566735           R-squared     =  0.1121
-------------+------------------------------           Adj R-squared =  0.1022
       Total |  32769.8824   271  120.922075           Root MSE      =   10.42

------------------------------------------------------------------------------
      pubsum |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
     faculty |   5.102683   1.278623     3.99   0.000     2.585259    7.620107
       enrol |  -1.207163   .4343317    -2.78   0.006    -2.062299   -.3520267
         phd |   1.470497   .6361169     2.31   0.022     .2180752    2.722919
       _cons |   10.40784   3.306921     3.15   0.002      3.89699    16.91869
```

5) Standardized Coefficients. `listcoef` lists the estimated coefficients for a variety of regression models. The `help` option includes details on the meaning of each coefficient. **SPost** (see L&F: 3.1.7, A.3)

```
. listcoef, help

regress (N=272): Unstandardized and Standardized Estimates

 Observed SD: 10.996457
 SD of Error: 10.419536

      pubsum |      b         t     P>|t|    bStdX    bStdY   bStdXY      SDofX
-------------+-----------------------------------------------------------------
     faculty |   5.10268    3.991   0.000   2.5504   0.4640   0.2319     0.4998
       enrol |  -1.20716   -2.779   0.006  -1.7649  -0.1098  -0.1605     1.4620
         phd |   1.47050    2.312   0.022   1.4741   0.1337   0.1341     1.0025
-------------------------------------------------------------------------------
       b = raw coefficient
       t = t-score for test of b=0
   P>|t| = p-value for t-test
   bStdX = x-standardized coefficient
   bStdY = y-standardized coefficient
  bStdXY = fully standardized coefficient
   SDofX = standard deviation of X
```

6) Close Log File. Always close your log file:

```
. log close
```

2.5: Introduction to Graph

The commands from this section are in the file `lhs_guide025.do`. For detailed discussion about graphing with Stata, see L&F: 2.16. The graph commands were changed completely in Stata 8. Each graph can be highly customized, but to do this requires the use of a lot of sometimes confusing options. Here we describe the basic options. Creating graphs is one place where the GUI can be very helpful. Notice that if you use the GUI to create a command, the resulting command is listed in the output window. You can copy this command into a do file for later use.

Summary of Key Commands

dotplot *varlist*	Plot the frequency distribution of one or more variables.
graph bar *varlist*	Graph a bar chart.
twoway scatter *varlist*	Draw a scatter plot.

Trying the Commands

1) Open a Log

```
. log using lhs_guide025.log, replace
--------------------------------------------------------------------------------
       log:  lhs_guide025.log
  log type:  text
 opened on:  20 Jun 2004, 00:24:46
```

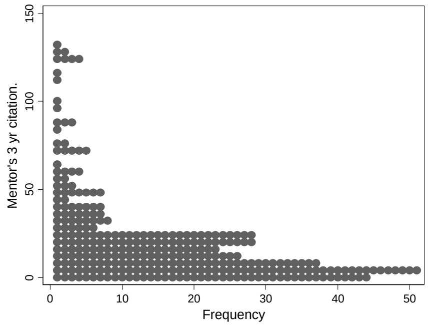
2) Load the Data

```
. use science2, clear
(Note that some of the variables have been artificially constructed.)
```

3) Descriptive Plot. `dotplot` produce a frequency distribution of a single or multiple variables. The x-axis denotes the frequency and y-axis corresponds to the variable of our interest.

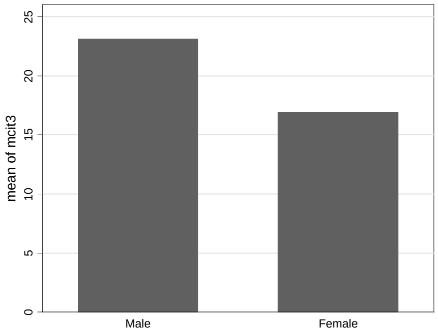
```
. sum mcit3

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
       mcit3 |       306    20.95098    26.26862          0        133
```

```
. dotplot mcit3
```



4) Bivarite Plot. By default, `graph bar` graphs the mean value of a variable conditional on the level of the variable specified with the `over` option. In this case, each bar is the mean of `mcit3` by respondent's gender.

```
. graph bar mcit3, over(female)
```

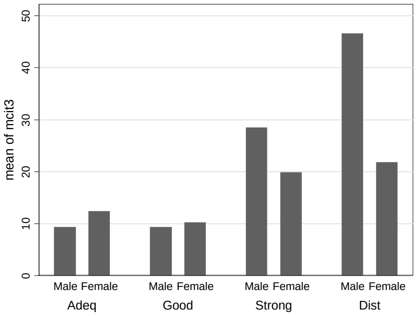


To plot the median values of mcit3 by gender:

```
. graph bar (median) mcit3, over(female)
```

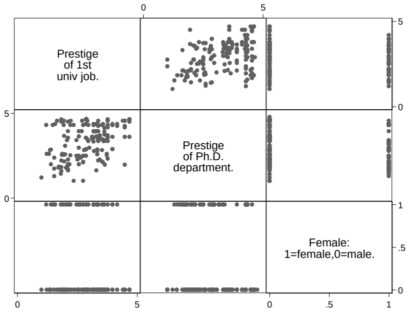
In the following graph, the bars are drawn over both the respondent's gender and the prestige of his/her fellowship or Ph.D program. Since the command is long, we use the "`///`" sign to continue the command across two lines (Note: you cannot use /// in the command window; it is intended for use in a do file).

```
. graph bar mcit3,  ///
>     over(female) over(felclass)
```



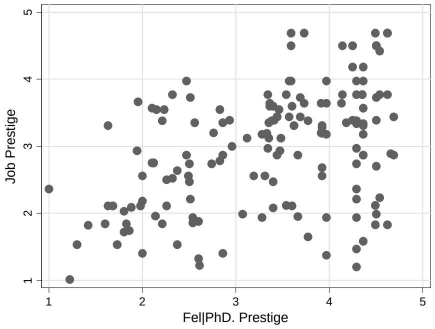
`graph matrix` draws pairwise scatterplots for variables in the variable list.

```
. graph matrix job phd female
```



5) Advanced Plot. `graph twoway scatter` is the command for drawing scatterplots. It has lots of options that can be used. The `ylabel` option denotes which tic marks for the y variable should be labeled. For example, the `ylabel(1(1)5, grid)`tells Stata 8 that numbers from 1 to 5 with an interval of 1 should be marked along the y-axis and that a grid will be superimposed. `ytitle("Job Prestige")` labels the y-axis. The same logic applies to label the x-axis. Note that you can abbreviate the command as either `twoway scatter` or `scatter` instead of `graph twoway scatter.`

```
. graph twoway scatter job fel, ///
>     ylabel(1(1)5, grid) ytitle("Job Prestige") ///
>     xlabel(1(1)5, grid) xtitle("Fel|PhD. Prestige")
```

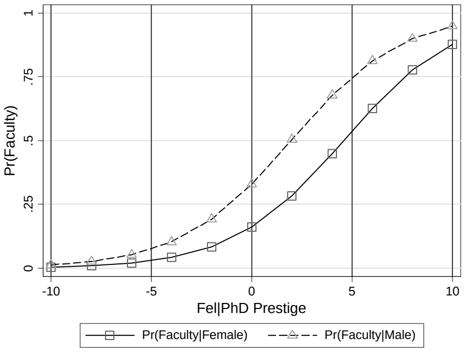


6) More Advanced Plot. A powerful feature of `scatter` is that it can produce multiple scatterplots in one graph. Here is an example from the lecture that estimate a binary logit model and produce six new variables containing predicted probabilities as `fel` varies and `female` set to one (being female) or zero (being male). If we have not discussed these commands in lecture yet, you can skip this section, or just try these commands and see what happens.

```
. logit faculty female fel, nolog
::: output deleted :::

. prgen fel, from(-10) to(10) generate(women) x(female=1) rest(mean)
::: output deleted :::
. label var womenp1 "Pr(Faculty|Female)"

. prgen fel, from(-10) to(10) generate(men) x(female=0) rest(mean)
::: output deleted :::
. label var menp1 "Pr(Faculty|Male)"
```

***Graph Options*:** The following scatter command draws two scatterplots in one graph. Each option is discussed below:

```
. scatter womenp1 womenx, ///
>     ylabel(0(.25)1., grid) ytitle("Pr(Faculty)") ///
>     xlabel(-10(5)10) xtitle("Fel|PhD Prestige") ///
>     msymbol(Sh) ///
>     connect(l) xline(-10 -5 0 5 10) ///
>     legend(lab(1 "Pr(UnivFaculty|Female)")) ///
> || scatter menp1 womenx, ///
>     msymbol(Th) ///
>     connect(l)  ///
>     legend(lab(2 "Pr(UnivFaculty|Male)"))
```



First, notice that there are two scatter commands that are combined using `||`. One command is for the female curve; the other for the male curve. For the first pair, `womenp1` corresponds to the y variable (y-axis) and `womenx` the x variable (x-axis).

`ylabel(0(.25)1., grid)` labels numbers from 0 to 1 with an interval of .25 along the y axis, with the `grid` option adding grids parallel to the x-axis. The `ytitle()` option writes a title along the y axis.

The `xlabel` and `xtitle` do the same thing for the x-axis.

`msymbol( )` specifies the marker symbol used for the first curve. In this case, `msymbol(Sh)`tells Stata to draw the bivariate scatterplot of `womenp1` against `womenx` using the hollow square symbol. For other marker symbols, type `help symbolstyle` in the command window.

`connect( )` allows you to connect each data point of the scatter plot of `womenp1` against `womenx`; the first observation is connected to the second, the second to the third, and so on. The suboption `l` requests connections using straight lines. For other options, type `help connectstyle` in the command window.

The `legend( )` option controls the legend. `legend(lab(1 "Pr(UnivFaculty|Female)"))` instructs Stata to label the first variable in the legend as “`Pr(UnivFaculty|Female)`”. For details, type `help legend_option`.

Well, we are done with the first scatter plot. When we want one plot type overlaid on another, we combine the commands, putting `||` in between the two sets of command. The options for the second graph are just like those for the first, with changes to reflect the different variables.

The following command is an alternative way to draw the same graph using only one `scatter` command. The options should be clear from the example above:

```
. scatter womenp1 menp1 womenx, ///
>     ylabel(0(.25)1., grid) ytitle("Pr(Faculty)") ///
>     xlabel(-10(5)10) xtitle("Fel|PhD Prestige") ///
>     msymbol(Sh Th) connect(l l)  ///
>     legend(lab(1 "Pr(UnivFaculty|Female)") lab(2 "Pr(UnivFaculty|Male)"))
```

There are many other types of graphs that you can use, but for now this is all we need.

7) Close Log File

```
. log close
```

3: Models for Binary Outcomes

The file `lhs_guide03.do` contains these Stata commands. For details about binary models and related Stata commands, see Chapter 4 of L&F (2003).

Summary of Key Commands

logit depvar indvar1 indvar2 — Estimate a logit model (see L&F: 3.1.4, 4.2).

probit depvar indvar1 indvar2 — Estimate a probit model (see L&F: 4.2).

listcoef, help — List the estimated coefficients; the `help` command lists the guidelines for interpretation and is *optional*. **SPost** (see L&F: 3.1.7, 4.7, A.3)

prvalue — Compute the predicted values at specific values of the independent variables. **SPost** (see L&F: 3.5.3, 3.5.7, 4.6.2, A.13).

prchange — Compute discrete and marginal change for regression models for categorical and count variables. **SPost** (see L&F: 3.5.3, 3.5.4, 4.6.5, A.9)

Trying the Commands

1) Open a Log. Remember to open a log file for recording your results.

```
. log using lhs_guide03.log, replace
-------------------------------------------------------------------------------
 log:   lhs_guide03.log
  log type:  text
 opened on:  20 Jun 2004, 00:24:46
```

2) Load the Data. Let’s use the data we created earlier.

```
. use sciwork, clear
```

3) Selecting a Sample. Assume that we only want to look at men. To do this, we drop all female cases.

```
. tab female

    Female: |
1=female,0= |
      male. |      Freq.     Percent        Cum.
------------+-----------------------------------
       Male |        201       65.26       65.26
     Female |        107       34.74      100.00
------------+-----------------------------------
      Total |        308      100.00

. drop if female==1
(107 observations deleted)
```

We also want to drop any cases with missing values for the variables we will be analyzing.

```
. sum female isfac fellow phd mcit3 mnas

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
      female |       201           0           0          0          0
       isfac |       198    .6010101     .490932          0          1
      fellow |       201    .4676617     .500199          0          1
         phd |       201    3.069403    1.010882          1       4.77
       mcit3 |       200      23.115     29.6611          0        133
-------------+--------------------------------------------------------
        mnas |       199    .0854271    .2802212          0          1

. drop if isfac>=. | mcit3>=. | mnas>=.
(6 observations deleted)

. sum isfac fellow phd mcit3 mnas

    Variable |     Obs        Mean   Std. Dev.       Min        Max
-------------+-----------------------------------------------------
       isfac |     195          .6   .4911589          0          1
      fellow |     195    .4769231   .5007528          0          1
         phd |     195    3.079026   1.020399          1       4.77
       mcit3 |     195    23.24615   29.90812          0        133
        mnas |     195    .0820513   .2751493          0          1
```

4) Save the Subset as a New File

```
. save scimaletemp, replace
file scimaletemp.dta saved
```

5) Binary Probit Model. As with regression, the dependent variable is listed first:

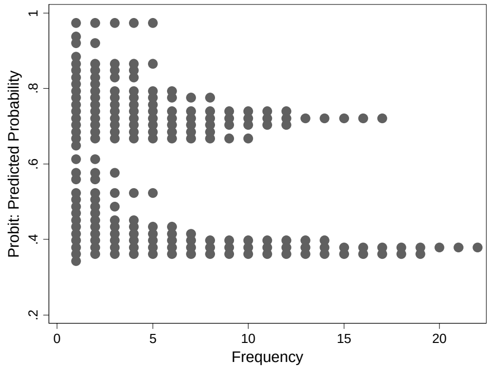
```
. probit isfac fellow phd mcit3 mnas

Probit estimates                                  Number of obs   =        195
                                                  LR chi2(4)      =      30.57
                                                  Prob > chi2     =     0.0000
Log likelihood = -115.95404                       Pseudo R2       =     0.1165

------------------------------------------------------------------------------
       isfac |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      fellow |   .8357913   .1997373     4.18   0.000     .4443133    1.227269
         phd |  -.0326242   .1135348    -0.29   0.774    -.2551483       .1899
       mcit3 |   .0116429   .0043634     2.67   0.008     .0030909     .020195
        mnas |   .0664127   .3899884     0.17   0.865    -.6979506     .830776
       _cons |  -.2740709   .3155085    -0.87   0.385    -.8924562    .3443144
------------------------------------------------------------------------------
```

6) Standardized Coefficients. Use `listcoef` to list standardized coefficients:

```
. listcoef, help

probit (N=195): Unstandardized and Standardized Estimates

 Observed SD: .49115895
   Latent SD: 1.1422799

-------------------------------------------------------------------------------
       isfac |      b         z     P>|z|    bStdX    bStdY   bStdXY      SDofX
-------------+-----------------------------------------------------------------
      fellow |   0.83579    4.184   0.000   0.4185   0.7317   0.3664     0.5008
         phd |  -0.03262   -0.287   0.774  -0.0333  -0.0286  -0.0291     1.0204
       mcit3 |   0.01164    2.668   0.008   0.3482   0.0102   0.3048    29.9081
        mnas |   0.06641    0.170   0.865   0.0183   0.0581   0.0160     0.2751
-------------------------------------------------------------------------------
       b = raw coefficient
       z = z-score for test of b=0
   P>|z| = p-value for z-test
   bStdX = x-standardized coefficient
   bStdY = y-standardized coefficient
  bStdXY = fully standardized coefficient
   SDofX = standard deviation of X
```

7) Predicted Probabilities. We can compute and plot predicted probabilities for our observed data. `prprobit` is a name that you pick for a variable that contains predicted values.

```
. predict prprobit
(option p assumed; Pr(isfac))

. label var prprobit "Probit: Predicted Probability"

. sum prprobit

    Variable |     Obs        Mean   Std. Dev.       Min        Max
-------------+-----------------------------------------------------
    prprobit |     195     .599605    .185754   .3438576   .9749245

. dotplot prprobit
```



For a detailed discussion of `predict`, see L&F: 3.5.2, 4.6.1.

8) Predict Specific Probabilities. `prvalue` [**SPost** (see L&F: 3.5.3, 3.5.7, 4.6.2, A.13)] computes the predicted value of our dependent variable given a set of values for our independent variables. Use the `x(variables=values)` to set the values at which the variables will be examined. Use `rest(mean)` to set the other independent variables at their means. Series of predicted probabilities can be combined into tables later, or use `prtab` [**SPost** (see L&F: 3.5.3, 3.5.6, 4.6.3, A.12)].

```
. prvalue , x(fellow=1 phd=4 mnas=1) rest(mean)

probit: Predictions for LHS variable isfac

  Pr(y=Isfac|x):    0.7788    95% ci: (0.5135,0.9336)
  Pr(y=NotFac|x):   0.2212    95% ci: (0.0664,0.4865)

       fellow        phd      mcit3       mnas
x=          1          4  23.246154          1
```

9) Marginal and Discrete Change. `prchange` [**SPost** (see L&F: 3.5.3, 3.5.4, 4.6.5, A.9)] computes the marginal and discrete change at specific values of the independent variables. By default, discrete and marginal changes are calculated holding all other variables at their mean. Values for specific independent variables can be set using the `x()` and `rest()` options after `prchange`.

```
. prchange, rest(mean) help

probit: Changes in Predicted Probabilities for isfac

        min->max      0->1     -+1/2    -+sd/2  MargEfct
fellow    0.3088    0.3088    0.3105    0.1586    0.3187
   phd   -0.0468   -0.0121   -0.0124   -0.0127   -0.0124
 mcit3    0.4309    0.0046    0.0044    0.1322    0.0044
  mnas    0.0251    0.0251    0.0253    0.0070    0.0253

          NotFac  Faculty
Pr(y|x)   0.3820   0.6180

         fellow      phd    mcit3     mnas
    x=  .476923  3.07903  23.2462  .082051
sd(x)=  .500753   1.0204  29.9081  .275149

 Pr(y|x): probability of observing each y for specified x values
Avg|Chg|: average of absolute value of the change across categories
Min->Max: change in predicted probability as x changes from its minimum to
          its maximum
    0->1: change in predicted probability as x changes from 0 to 1
   -+1/2: change in predicted probability as x changes from 1/2 unit below
          base value to 1/2 unit above
  -+sd/2: change in predicted probability as x changes from 1/2 standard
          dev below base to 1/2 standard dev above
MargEfct: the partial derivative of the predicted probability/rate with
          respect to a given independent variable
```

10) Binary Logit Model. The format is the same for the logit model.

```
. logit isfac fellow phd mcit3 mnas, nolog

Logit estimates                                   Number of obs   =        195
                                                  LR chi2(4)      =      31.04
                                                  Prob > chi2     =     0.0000
Log likelihood = -115.71789                       Pseudo R2       =     0.1183

------------------------------------------------------------------------------
       isfac |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      fellow |   1.387629   .3351305     4.14   0.000     .7307849    2.044472
         phd |  -.0587957    .187301    -0.31   0.754     -.425899    .3083076
       mcit3 |   .0209859   .0081047     2.59   0.010     .0051009    .0368709
        mnas |   .0431684   .6634993     0.07   0.948    -1.257266    1.343603
       _cons |   -.464029   .5153051    -0.90   0.368    -1.474008    .5459503
------------------------------------------------------------------------------
```

11) Computing Odds Ratios. All the commands above work exactly the same with `logit` as they do with `probit`. In addition, for logit models, `listcoef` [**SPost** (see L&F: 3.1.7, 4.7, A.3)] computes the factor change in the odds. Again, the `help` option helps interpret what each column corresponds to.

```
. listcoef, help

logit (N=195): Factor Change in Odds

  Odds of: Faculty vs NotFac

----------------------------------------------------------------------
       isfac |      b         z     P>|z|    e^b    e^bStdX      SDofX
-------------+--------------------------------------------------------
      fellow |   1.38763    4.141   0.000   4.0053   2.0034     0.5008
         phd |  -0.05880   -0.314   0.754   0.9429   0.9418     1.0204
       mcit3 |   0.02099    2.589   0.010   1.0212   1.8732    29.9081
        mnas |   0.04317    0.065   0.948   1.0441   1.0119     0.2751
----------------------------------------------------------------------
       b = raw coefficient
       z = z-score for test of b=0
   P>|z| = p-value for z-test
     e^b = exp(b) = factor change in odds for unit increase in X
 e^bStdX = exp(b*SD of X) = change in odds for SD increase in X
   SDofX = standard deviation of X
```

12) Close Log File.

```
. log close
```

4: Testing and Assessing Fit

The file `lhs_guide04.do` contains these Stata commands. For a fuller discussion of testing and assessing fit, see Chapter 3 of L&F (2003).

Summary of Key Commands

test Compute Wald test of linear hypotheses (see L&F: 3.3.1, 4.3).

test *varname* Test whether the effect of the variable is equal to zero.

test *varname varname* Test whether the effects of the variables are simultaneously zero.

test *varname=varname* Test whether the effects of the two variables are equal.

estimates dir *estnamelist* List the estimation results saved under the names *estname1 estname2*, etc..

estimates store *estname* Store the active estimation results under *estname* for log likelihood ratio test. The stored estimation results can also be used for other purposes (see L&F 3.1.9, 3.3.2, 4.3.2).

estimates table *estnamelist* Display a table with coefficients and statistics for one or more estimation results in parallel columns. In addition, standard errors, t-statistics, p-values and scalar measures of model fit statistics may be listed. The *estname's* in the *estnamelist* should have been stored by the `estimates store` command.

lrtest *estname1 estname2* Conduct log likelihood ratio tests between pairs of nested models. To conduct the test, both the unrestricted and the restricted models have to be fitted using the maximum-likelihood method (or some equivalent method), and the results of at least one of them stored using `estimates store` (see L&F: 3.3.2, 4.3.2).

predict *varname*, **dbeta** Generate a new variable *varname* that contains Cook's distance statistic for each observation (see L&F: 3.5.2, 4.4, 4.6.1)

fitstat Compute fit statistics. The `saving()` option saves the measures in a matrix for subsequent comparisons. `using()` compares the fit measures of the current model with those of the using model. **SPost** (see L&F: 3.4, 4.5, A.2)

Trying the Commands

1) Open a Log.

```
. log using lhs_guide04.log, replace
--------------------------------------------------------------------------------
 log:   lhs_guide04.log
  log type:  text
 opened on:  20 Jun 2004, 00:24:46
```

2) Load and Clean the Data. Let's use the data we created earlier, and drop any cases that have missing data for the variables we will be using. Remember to verify that you have selected the correct number of cases (See Exercise 2 for more details).

```
. use sciwork, clear
. keep isfac fellow phd mcit3 mnas female
. tab isfac, m

  1=Faculty |
         in |
 University |      Freq.     Percent        Cum.
------------+-----------------------------------
     NotFac |        142       46.10       46.10
    Faculty |        160       51.95       98.05
          . |          6        1.95      100.00
------------+-----------------------------------
      Total |        308      100.00

. tab fellow, m
::: output deleted :::
. tab female, m
::: output deleted :::

. drop if isfac>=. | fellow>=. | phd>=. ///
>           | mcit3>=. | mnas>=.   | female>=.
(11 observations deleted)

. tab1 isfac fellow phd mcit3 mnas female, m
-> tabulation of isfac

  1=Faculty |
         in |
 University |      Freq.     Percent        Cum.
------------+-----------------------------------
     NotFac |        140       47.14       47.14
    Faculty |        157       52.86      100.00
------------+-----------------------------------
      Total |        297      100.00
```

```
::: output deleted :::

. sum isfac fellow phd mcit3 mnas female

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
       isfac |       297    .5286195    .5000227          0          1
      fellow |       297    .4006734     .490862          0          1
         phd |       297    3.182963    1.024396          1       4.77
       mcit3 |       297    21.14141    26.53287          0        133
        mnas |       297    .0774411    .2677411          0          1
-------------+--------------------------------------------------------
      female |       297    .3434343    .4756564          0          1
```

3) Computing a z-test. z-scores are produced with the standard estimation commands. For example, let's do a simple logit estimation.

```
. logit isfac female fellow phd mcit3 mnas, nolog

Logit estimates                                   Number of obs   =        297
                                                  LR chi2(5)      =      50.58
                                                  Prob > chi2     =     0.0000
Log likelihood =    -180.09                       Pseudo R2       =     0.1231

------------------------------------------------------------------------------
       isfac |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      female |  -.6083388   .2775948    -2.19   0.028    -1.152415    -.064263
      fellow |   1.066298   .2699367     3.95   0.000     .5372322    1.595365
         phd |   .0508171   .1437644     0.35   0.724    -.2309559    .3325901
       mcit3 |   .0220479   .0070491     3.13   0.002     .0082319     .035864
        mnas |   .3733006   .5551182     0.67   0.501     -.714711    1.461312
       _cons |  -.6958342   .4202993    -1.66   0.098    -1.519606    .1279372
------------------------------------------------------------------------------
```

4) Single Coefficient Wald Test. After estimation, `test` can compute a Wald test that a single coefficient is equal to zero.

```
. test female

 ( 1)  female = 0.0

           chi2(  1) =    4.80
         Prob > chi2 =    0.0284
```

5) Multiple Coefficients Wald Test. We can also test if multiple coefficients are equal to zero.

```
. test mcit3 mnas

 ( 1)  mcit3 = 0.0
 ( 2)  mnas = 0.0

           chi2(  2) =   10.79
         Prob > chi2 =    0.0045
```

6) Equal Coefficients Wald Test. We can test that multiple coefficients are equal:

```
. test mcit3 = mnas

 ( 1)  mcit3 - mnas = 0.0

           chi2(  1) =    0.40
         Prob > chi2 =    0.5275
```

7) Store the Estimation Results. After each estimation, we can use the `estimates store` *estname* command to store the estimation results (coefficients, test statistics, and model fit, etc.) in the memory and we can recall them later for model comparisons.

```
. logit isfac female fellow phd mcit3 mnas
::: output deleted :::

. estimates store base
```

8) Single Coefficient LR Test. To test that the effect of female is zero, run the comparison model without female and then compare with the full model. To do that, we can use `lrtst` *`estname1 estname2`* (see L&F: 3.3.2, 4.3.2) to conduct log likelihood ratio test. Note that the `lrtest` command can automatically detect the restricted and unrestricted model.

```
. logit isfac fellow phd mcit3 mnas
::: output deleted :::

. estimates store nofemale

. lrtest base nofemale
likelihood-ratio test                                  LR chi2(1)  =      4.84
(Assumption: nofemale nested in base)                  Prob > chi2 =    0.0278
```

9) Multiple Coefficients LR Test. To test if effects of mcit3 and mnas are jointly equal to zero, run the comparison model without these variables, store the estimation results, and then compare models using the `lrtest` command.

```
. logit isfac female fellow phd
::: output deleted :::

. estimates store nomcit3mnas

. lrtest base nomcit3mnas
likelihood-ratio test                                  LR chi2(2)  =     13.24
(Assumption: nomcit3mnas nested in base)               Prob > chi2 =    0.0013
```

10) LR Test All Coefficients are Zero. To test that all of the regressors have no effect, we estimate the model with only an intercept, store the estimation results again, and compare the models using the `lrtest` command.

```
. logit isfac
::: output deleted :::

. estimates store intercept

. lrtest base intercept
likelihood-ratio test                                  LR chi2(5)  =     50.58
(Assumption: intercept nested in base)                 Prob > chi2 =    0.0000
```

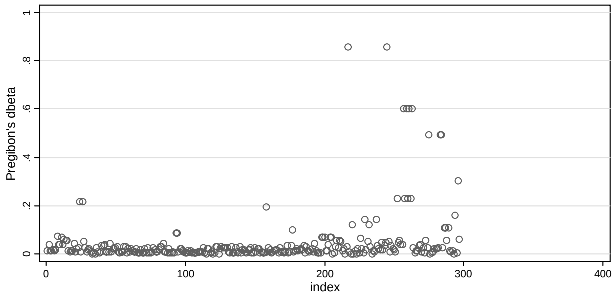
11) Fit Statistics. `fitstat` [**SPost** (see L&F: 3.4, 4.5, A.2)] computes measures of fit for your model. The `save` option saves the computed measures in a matrix for subsequent comparisons. `dif` compares the fit measures of the current model with those of the saved model.

```
. logit isfac female fellow phd mcit3 mnas
::: output deleted :::

. fitstat, save

Measures of Fit for logit of isfac
```

```
Log-Lik Intercept Only:     -205.378     Log-Lik Full Model:         -180.090
D(291):                      360.180     LR(5):                        50.576
                                         Prob > LR:                     0.000
McFadden's R2:                 0.123     McFadden's Adj R2:             0.094
Maximum Likelihood R2:         0.157     Cragg & Uhler's R2:            0.209
McKelvey and Zavoina's R2:     0.224     Efron's R2:                    0.162
Variance of y*:                4.238     Variance of error:             3.290
Count R2:                      0.687     Adj Count R2:                  0.336
AIC:                           1.253     AIC*n:                       372.180
BIC:                       -1296.696     BIC':                        -22.107

(Indices saved in matrix fs_0)

. gen phd2 = phd*phd
. logit isfac female fellow phd phd2
::: output deleted :::

. fitstat, dif

Measures of Fit for logit of isfac

                             Current          Saved       Difference
Model:                         logit          logit
N:                               297            297                0
Log-Lik Intercept Only:     -205.378       -205.378            0.000
Log-Lik Full Model:         -186.439       -180.090           -6.349
D:                           372.877(292)   360.180(291)      12.697(1)
LR:                           37.878(4)      50.576(5)       -12.697(-1)
Prob > LR:                     0.000          0.000            0.000
McFadden's R2:                 0.092          0.123           -0.031
McFadden's Adj R2:             0.068          0.094           -0.026
Maximum Likelihood R2:         0.120          0.157           -0.037
Cragg & Uhler's R2:            0.160          0.209           -0.049
McKelvey and Zavoina's R2:     0.153          0.224           -0.071
Efron's R2:                    0.124          0.162           -0.038
Variance of y*:                3.883          4.238           -0.355
Variance of error:             3.290          3.290            0.000
Count R2:                      0.660          0.687           -0.027
Adj Count R2:                  0.279          0.336           -0.057
AIC:                           1.289          1.253            0.036
AIC*n:                       382.877        372.180           10.697
BIC:                       -1289.692      -1296.696            7.004
BIC':                        -15.103        -22.107            7.004

Difference of    7.004 in BIC' provides strong support for saved model.
```

12) Plotting Outliers Using Cook’s Distance. We first sort our data by phd. The value of index corresponds to the order of observation after such sorting. So the value of index is the rank of each observation in terms of their phd value. The Cook’s distance is therefore plotted against the rank order of phd’s value.

```
. quietly logit isfac female fellow phd mcit3 mnas
. predict cook,dbeta
. sort phd
. gen index = _n
. twoway scatter cook index, ysize(1) xsize(2)  ///
>   xlabel(0(100)400) ylabel(0(.2)1., grid) ///
>   xscale(range(0, 400)) yscale(range(0, 1.))  ///
>   msymbol(Oh)
```



13) Close the Log File.

```
. log close
```

5: Models for Ordinal Outcomes

The file `lhs_guide05.do` contains these Stata commands. For a fuller discussion of models for ordinal outcomes, see Chapter 5 of L&F (2003).

Summary of Key Commands

ologit depvar indvars
Estimate the ordered logit model (see L&F: 5.2, 5.2.1).

oprobit depvar indvars
Estimate the ordered probit model (see L&F: 5.2, 5.2.1).

brant
Perform a Brant test of the parallel regressions assumptions for the ordered logit model estimated by `ologit`. **SPost** (see L&F: 5.6, A.1).

listcoef, help
List the estimated coefficients; the `help` command lists the guidelines for interpretation and is *optional*. Type `help listcoef` for options. **SPost** (see L&F: 5.8.8, A.3)

prvalue
Computes the predicted values at specific values of the independent variables Run `help prvalue` for details. **SPost** (see L&F: 5.8.4, A.13)

prchange
Computes discrete and marginal change for regression models for categorical and count variables type `help prchange` for details. **SPost** (see L&F: 5.8.7, A.9)

prgen
Generate predicted values for regression models. Type `help prgen` for details. **SPost** (see L&F: 5.8.6, A.11)

Trying the Commands

1) Open a Log.

```
. log using lhs_guide05.log, replace
-------------------------------------------------------------------------------
  log:  lhs_guide05.log
  log type:  text
 opened on:  20 Jun 2004, 00:24:46
```

2) Load and Clean the Data. Let's use the data we created earlier, and drop any cases that have missing data for the variables we will be using. Remember to verify that you have selected the correct number of cases.

```
. use sciwork, clear
. keep jobprst pub1 phd female
. tab jobprst, m
::: output deleted :::

. tab female, m

    Female: |      Freq.     Percent        Cum.
------------+-----------------------------------
       Male |        201       65.26       65.26
     Female |        107       34.74      100.00
------------+-----------------------------------
      Total |        308      100.00

. drop if jobprst>=. | pub1>=. | phd>=. | female>=.
(145 observations deleted)

. tab jobprst, m
::: output deleted :::
. tab phd, m
::: output deleted :::
. tab female, m

    Female: |
1=female,0= |
      male. |      Freq.     Percent        Cum.
------------+-----------------------------------
       Male |        123       75.46       75.46
     Female |         40       24.54      100.00
------------+-----------------------------------
      Total |        163      100.00

. sum

    Variable |     Obs        Mean   Std. Dev.       Min        Max
-------------+-----------------------------------------------------
      female |     163    .2453988   .4316495          0          1
         phd |     163    3.329141    .983948          1       4.66
        pub1 |     163    3.208589   3.221176          0         19
     jobprst |     163    2.417178   .8945373          1          4
```

3) Ordered Probit and Ordered Logit. `ologit` (see L&F: 5.2, 5.2.1) and `oprobit` (see L&F: 5.2, 5.2.1) work in the same way. We only show `ologit`, but you might want to try what follows using `oprobit`.

```
. ologit  jobprst pub1 phd female, nolog

Ordered logit estimates                           Number of obs   =        163
                                                  LR chi2(3)      =      31.11
                                                  Prob > chi2     =     0.0000
Log likelihood = -187.72111                       Pseudo R2       =     0.0765

     jobprst |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
        pub1 |   .1031471   .0459393     2.25   0.025     .0131077    .1931865
         phd |   .7812238   .1630057     4.79   0.000     .4617385    1.100709
      female |  -.3888728   .3408996    -1.14   0.254    -1.057024    .2792781
-------------+----------------------------------------------------------------
       _cut1 |   1.134329   .5407607          (Ancillary parameters)
       _cut2 |   2.691888   .5738618
       _cut3 |   5.442969   .6868612
------------------------------------------------------------------------------
```

4) Standardized Coefficients.

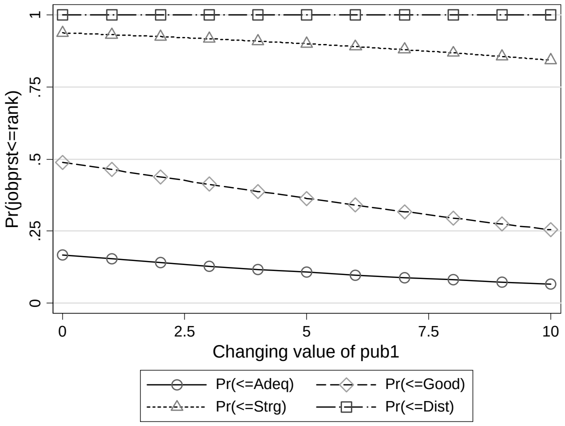
```
. listcoef, help

ologit (N=163): Factor Change in Odds

  Odds of: >m vs <=m

----------------------------------------------------------------------
     jobprst |      b         z     P>|z|    e^b    e^bStdX      SDofX
-------------+--------------------------------------------------------
        pub1 |   0.10315    2.245   0.025   1.1087   1.3941     3.2212
         phd |   0.78122    4.793   0.000   2.1841   2.1569     0.9839
      female |  -0.38887   -1.141   0.254   0.6778   0.8455     0.4316
----------------------------------------------------------------------
       b = raw coefficient
       z = z-score for test of b=0
   P>|z| = p-value for z-test
     e^b = exp(b) = factor change in odds for unit increase in X
 e^bStdX = exp(b*SD of X) = change in odds for SD increase in X
   SDofX = standard deviation of X
```

5) Predicted Probabilities in Sample. Use `predict` (see L&F: 5.8.3) to compute predicted probabilities after `ologit` (`oprobit`). The first new variable contains the probability associated with the lowest outcome; the second, the probability associated with the second outcome; and so on. There are three steps. First, estimate the model. Second, use `predict` (see L&F: 5.8.3) to compute the probabilities associated with each category and store them as new variables. Third, label each of the new variables and display the results with `sum`.

```
. predict lpad lpgo lpst lpdi
. label var lpad "OLM Pr(Adeq)"
. label var lpgo "OLM Pr(Good)"
. label var lpst "OLM Pr(Strg)"
. label var lpdi "OLM Pr(Dist)"
. sum lpad lpgo lpst lpdi

    Variable |     Obs        Mean   Std. Dev.       Min        Max
-------------+-----------------------------------------------------
        lpad |     163    .1855267       .1287   .0200777   .5678892
        lpgo |     163    .2816571    .0747627   .0685664   .3708335
        lpst |     163    .4435312      .13487   .1280107   .5965453
        lpdi |     163    .0892849    .0674725   .0101319   .3963308
```

6) Plot Probabilities. The following commands compute the predicted probabilities of various outcomes for women from distinguished programs using `prgen` [**SPost** (see L&F: 5.8.6, A.11)] and `scatter`. You can make the graphs fancier by using the various options for graph.

```
. prgen pub1, x(female=1 phd=4) rest (mean) from (0) to (10) gen (pubpr)
. label var pubprs1 "Pr(<=Adeq)"
. label var pubprs2 "Pr(<=Good)"
. label var pubprs3 "Pr(<=Strg)"
. label var pubprs4 "Pr(<=Dist)"
. scatter pubprs1 pubprx, ///
>     ylabel(0(.25)1., grid) yscale(range(0 1)) ///
>     ytitle("Pr(jobprst<=rank)") xlabel(0(2.5)10) ///
>     connect(l) msymbol(Oh) ///
> || scatter pubprs2 pubprx, ///
>     connect(l) msymbol(Dh) ///
> || scatter pubprs3 pubprx, ///
>      connect(l) msymbol(Th) ///
> || scatter pubprs4 pubprx, ///
>      connect(l) msymbol(Sh)
```



7) Predict Specific Probabilities. `prvalue` [**SPost** (see L&F: 5.8.4, A.13)] computes the predicted value of our dependent variable given a set of values for our independent variables. Use the `x()` option to set the values at which the variables will be examined. Use the `rest()` option to set the other independent variables at their means.

```
. quietly ologit jobprst pub1 phd female
. prvalue, x(female=1 pub1=0) rest(mean)

ologit: Predictions for jobprst

  Pr(y=Adeq|x):       0.2540
  Pr(y=Good|x):       0.3638
  Pr(y=Strong|x):     0.3442
  Pr(y=Dist|x):       0.0380

         pub1        phd     female
x=          0  3.3291411          1

. prvalue, x(female=0 pub1=0) rest(mean)

ologit: Predictions for jobprst

  Pr(y=Adeq|x):       0.1875
  Pr(y=Good|x):       0.3353
  Pr(y=Strong|x):     0.4222
  Pr(y=Dist|x):       0.0551

         pub1        phd     female
x=          0  3.3291411          0
```

8) Compute the Marginal and Discrete Change. `prchange` [**SPost** (see L&F: 5.8.7, A.9)] computes marginal and discrete change at specific values of the independent variables. By default, the discrete and marginal change is calculated holding all other variables at their mean. Values for specific independent variables can be set using the `x()` and `rest()` options. The `help` option presents guide to interpreting headings of output.

```
. quietly ologit jobprst pub1 phd female

. prchange, rest(mean)

ologit: Changes in Predicted Probabilities for jobprst

pub1
            Avg|Chg|         Adeq         Good       Strong         Dist
Min->Max   .20067943  -.16792272  -.23343614   .17835629   .22300257
```

```
   -+1/2   .01282357  -.01345426  -.01219288   .01904485   .00660229
  -+sd/2    .0412227  -.04337747  -.03906792   .06112385   .02132154
MargEfct   .05130547  -.01345297  -.01219977   .01905224   .00660049

phd
            Avg|Chg|        Adeq        Good      Strong        Dist
Min->Max   .30397561  -.46880359  -.13914762   .44711629   .16083494
   -+1/2    .0959473   -.1024375  -.08945709   .14112428   .05077031
  -+sd/2   .09444418  -.10077657  -.08811179   .13895729   .04993106
MargEfct   .38858152  -.10189116   -.0923996   .14429941   .04999135

female
            Avg|Chg|        Adeq        Good      Strong        Dist
    0->1   .04843176   .05429748   .04256606  -.07393691  -.02292661

              Adeq       Good     Strong       Dist
Pr(y|x)  .15420391  .30974784  .46733576  .06871249

           pub1      phd   female
    x=  3.20859  3.32914  .245399
sd(x)=  3.22118  .983948  .431649
```

9) Odds Ratios. The above commands work for `oprobit` and `ologit`. For logit models only, `listcoef` [**SPost** (see L&F: 5.8.8, A.3)] computes the factor change in the odds. Again, the `help` option presents guide to interpreting headings of output.

```
ologit (N=163): Factor Change in Odds

  Odds of: >m vs <=m

----------------------------------------------------------------------
     jobprst |      b         z     P>|z|    e^b    e^bStdX      SDofX
-------------+--------------------------------------------------------
        pub1 |   0.10315    2.245   0.025   1.1087   1.3941     3.2212
         phd |   0.78122    4.793   0.000   2.1841   2.1569     0.9839
      female |  -0.38887   -1.141   0.254   0.6778   0.8455     0.4316
----------------------------------------------------------------------
       b = raw coefficient
       z = z-score for test of b=0
   P>|z| = p-value for z-test
     e^b = exp(b) = factor change in odds for unit increase in X
 e^bStdX = exp(b*SD of X) = change in odds for SD increase in X
   SDofX = standard deviation of X
```

10) Testing the Parallel Regressions Assumption. `brant` [**SPost** (see L&F: 5.6, A.1)] performs a Brant test of the parallel regressions assumptions for the ordered logit model estimated by `ologit`.

```
. ologit jobprst pub1 phd female
::: output deleted :::
. brant, detail

Estimated coefficients from j-1 binary regressions

              y>1         y>2         y>3
  pub1   .12331639   .09068503    .1046466
   phd   .54755133    .7410395   1.5841797
female  -.09488208  -.34979349  -1.6819005
 _cons  -.60652917  -2.5821879  -8.5100335

Brant Test of Parallel Regression Assumption

    Variable |      chi2   p>chi2    df
-------------+--------------------------
         All |      5.26    0.511     6
-------------+--------------------------
```

```
        pub1 |      0.18    0.915     2
         phd |      3.64    0.162     2
      female |      1.85    0.396     2
----------------------------------------

A significant test statistic provides evidence that the parallel
regression assumption has been violated.
```

11) Close the Log File:

```
. log close
```

6: Models for Nominal Outcomes with Stata

The file `lhs_guide06.do` contains these Stata commands. For details about models for multinomial outcomes and associated Stata commands, please read Chapter 6 of L&F (2003).

Summary of Key Commands

mlogit depvar indvars, **b**asecategory **(#)**	Estimate the multinomial logit model. **b**asecategory**(#)** specifies the value of depvar to use as the base category. By default, Stata uses the most frequent category (see L&F: 6.1).
listcoef, help	List all possible odds ratios. **SPost** (see L&F: 6.6.8, A.3).
mlogview	Opens a command box for constructing discrete change and odds ratio plots. Must be run after `mlogit` and `prchange`**.** This command must either be run from the command line or as the last command in a do file. **SPost** (see L&F: 6.6.7, 6.6.8, A.6).
mlogtest	Makes it simple to compute useful tests for the multinomial logit model. **SPost** (see L&F: 6.3.1, A.5)

Trying the Commands

1) Open a Log.

```
. log using lhs_guide06.log, replace
--------------------------------------------------------------------------------
  log:  lhs_guide06.log
  log type:  text
 opened on:  20 Jun 2004, 00:24:46
```

2) Load and Clean the Data.

```
. use sciwork, clear
. keep jobprst pub1 phd female
. tab jobprst, m
::: output deleted :::

. drop if jobprst>=. | pub1>=. | phd>=. | female>=.
(145 observations deleted)

. sum
::: output deleted :::
```

3) Multinomial Logit. `mlogit` estimates the multinomial logit model. Set `basecategory` to any category you want. Here, the estimates store command after mlogit is used to store estimation results for model comparison in Step 6.

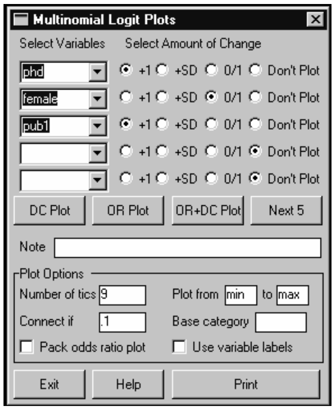
```
. mlogit jobprst pub1 phd female, basecategory(4) nolog

Multinomial regression                            Number of obs   =        163
                                                  LR chi2(9)      =      36.59
                                                  Prob > chi2     =     0.0000
Log likelihood = -184.98037                       Pseudo R2       =     0.0900

------------------------------------------------------------------------------
     jobprst |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
Adeq         |
        pub1 |  -.2138213   .1136144    -1.88   0.060    -.4365014    .0088589
         phd |  -2.052231   .5698192    -3.60   0.000    -3.169056   -.9354056
      female |   1.746598   1.173675     1.49   0.137    -.5537626    4.046959
       _cons |   8.723882    2.32848     3.75   0.000     4.160144    13.28762
-------------+----------------------------------------------------------------
Good         |
        pub1 |  -.1292708   .0942216    -1.37   0.170    -.3139418    .0554002
         phd |  -1.876966   .5550082    -3.38   0.001    -2.964762   -.7891699
      female |   1.843799   1.138741     1.62   0.105    -.3880916     4.07569
       _cons |   8.373428   2.296153     3.65   0.000      3.87305    12.87381
-------------+----------------------------------------------------------------
Strong       |
        pub1 |  -.0799114   .0820373    -0.97   0.330    -.2407016    .0808788
         phd |   -1.32085   .5385631    -2.45   0.014    -2.376414   -.2652859
      female |   1.600885   1.109419     1.44   0.149    -.5735352    3.775306
       _cons |   6.853277    2.25648     3.04   0.002     2.430657     11.2759
------------------------------------------------------------------------------
(Outcome jobprst==Dist is the comparison group)

.      estimates store base
```

4) Odds Ratios. `listcoef` [**SPost** (see L&F: 6.6.8, A.3)] computes coefficients and z-values for all comparisons among outcomes:

```
. listcoef, help

mlogit (N=163): Factor Change in the Odds of jobprst

Variable: pub1 (sd= 3.22118)

    Odds comparing|
Group 1 vs Group 2|      b         z     P>|z|     e^b   e^bStdX
------------------+---------------------------------------------
Adeq    -Good     |  -0.08455   -0.919   0.358   0.9189   0.7616
Adeq    -Strong   |  -0.13391   -1.509   0.131   0.8747   0.6496
Adeq    -Dist     |  -0.21382   -1.882   0.060   0.8075   0.5022
Good    -Adeq     |   0.08455    0.919   0.358   1.0882   1.3130
Good    -Strong   |  -0.04936   -0.775   0.438   0.9518   0.8530
Good    -Dist     |  -0.12927   -1.372   0.170   0.8787   0.6594
Strong  -Adeq     |   0.13391    1.509   0.131   1.1433   1.5393
Strong  -Good     |   0.04936    0.775   0.438   1.0506   1.1723
Strong  -Dist     |  -0.07991   -0.974   0.330   0.9232   0.7731
Dist    -Adeq     |   0.21382    1.882   0.060   1.2384   1.9912
Dist    -Good     |   0.12927    1.372   0.170   1.1380   1.5165
Dist    -Strong   |   0.07991    0.974   0.330   1.0832   1.2936
----------------------------------------------------------------
::: output deleted :::
       b = raw coefficient
       z = z-score for test of b=0
   P>|z| = p-value for z-test
     e^b = exp(b) = factor change in odds for unit increase in X
 e^bStdX = exp(b*SD of X) = change in odds for SD increase in X
```

5) Plot Factor Changes. Enter the command: `mlogview` [**SPost** (see L&F: 6.6.7, 6.6.8, A.6)] which produces a dialog box that looks this:

```
. mlogview
```



Select Variables Select Amount of Change

phd ● +1 ○ +SD ○ 0/1 ○ Don't Plot

female ○ +1 ○ +SD ● 0/1 ○ Don't Plot

pub1 ● +1 ○ +SD ○ 0/1 ○ Don't Plot

○ +1 ○ +SD ○ 0/1 ● Don't Plot

○ +1 ○ +SD ○ 0/1 ● Don't Plot

Note

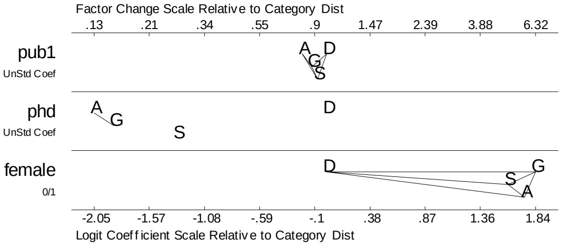
Plot Options

Number of tics 9 Plot from min to max

Connect if .1 Base category

□ Pack odds ratio plot □ Use variable labels

Which lets you make a plot like this:



You can click different radio buttons to choose desired amount of change or to choose between the DC Plot, OR Plot, and OR+DC Plot to have plots of factor change and/or discrete change. A solid line indicates that the coefficient corresponds to this panel cannot differentiate the two outcomes connected.

6) Single Variable LR Test. The effect of female involves three coefficients. We can use an LR test to test that all are simultaneously equal to zero. First, we need to save the base model (see Step 3) and store estimation results; second, we estimate the model without female and store the estimation results; and third, we compare the two models using the `lrtst` *`estname1 estname2`* command (see L&F: 3.3.2, 4.3.2, 6.3.2) to conduct log likelihood ratio test:

```
. quietly mlogit jobprst pub1 phd, basecategory(4)
. estimates store nofemale

. lrtest base nofemale
likelihood-ratio test                                  LR chi2(3)  =      3.66
(Assumption: nofemale nested in base)                  Prob > chi2 =    0.3009
```

And so on eliminating one variable at a time. Or, you can use the command `mlogtest` [**SPost** (see L&F: 6.3.1, A.5)]:

```
. mlogtest, lr

**** Likelihood-ratio tests for independent variables

 Ho: All coefficients associated with given variable(s) are 0.


     jobprst |       chi2   df   P>chi2
-------------+-------------------------
```

```
        pub1 |      4.265    3    0.234
         phd |     26.921    3    0.000
      female |      3.657    3    0.301
---------------------------------------
```

7) Single Coefficient Wald Test. The Wald test that female has no effect can be computed as follows:

```
. mlogit jobprst pub1 phd female, basecategory(4)
::: selected output :::

Multinomial regression                            Number of obs   =        163

------------------------------------------------------------------------------
     jobprst |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
Adeq         |
      female |   1.746598   1.173675     1.49   0.137    -.5537626    4.046959
-------------+----------------------------------------------------------------
Good         |
      female |   1.843799   1.138741     1.62   0.105    -.3880916     4.07569
-------------+----------------------------------------------------------------
Strong       |
      female |   1.600885   1.109419     1.44   0.149    -.5735352    3.775306
------------------------------------------------------------------------------
(Outcome jobprst==Dist is the comparison group)

. test female

 ( 1)  [Adeq]female = 0.0
 ( 2)  [Good]female = 0.0
 ( 3)  [Strong]female = 0.0

           chi2(  3) =    2.66
         Prob > chi2 =    0.4467
```

And, so on for each variable. Or, use the command `mlogtest` [**SPost** (see L&F: 6.3.1, A.5)]:

```
. mlogtest, wald

**** Wald tests for independent variables

 Ho: All coefficients associated with given variable(s) are 0.

     jobprst |       chi2   df   P>chi2
-------------+-------------------------
        pub1 |      3.882    3    0.275
         phd |     19.721    3    0.000
      female |      2.662    3    0.447
---------------------------------------
```

8) Combining Outcomes Test. `test` can also compute a Wald test that two outcomes can be combined. Recall, that the coefficients for category Adeq were in comparison to the category Dist. Therefore, we are testing whether we can combine Adeq and Dist. Note that [Adeq] is necessary in specifying the test across categories, and Adeq does not equal to adeq since syntax in Stata is case sensitive.

```
. test [Adeq]

 ( 1)  [Adeq]pub1 = 0.0
 ( 2)  [Adeq]phd = 0.0
 ( 3)  [Adeq]female = 0.0

           chi2(  3) =   18.52
         Prob > chi2 =    0.0003
```

This could be done for combining other categories with Dist. Then, the model can be estimated with other base categories for tests of combining other categories. The easier way is `mlogtest`:

```
. mlogtest, combine

**** Wald tests for combining outcome categories
 Ho: All coefficients except intercepts associated with given pair
     of outcomes are 0 (i.e., categories can be collapsed).

Categories tested |      chi2   df   P>chi2
------------------+------------------------
    Adeq-    Good |     1.469    3    0.689
    Adeq-  Strong |    11.800    3    0.008
    Adeq-    Dist |    18.517    3    0.000
    Good-  Strong |     7.788    3    0.051
    Good-    Dist |    15.668    3    0.001
  Strong-    Dist |     9.124    3    0.028
-------------------------------------------
```

9) Predicted Values. `prvalue` [**SPost** (see L&F: 6.6.3, A.13)] can be used as before to compute predicted values for a given set of values of the independent variables. By default, `prvalue` holds all variables at their means.

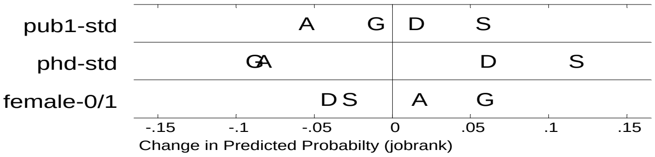
```
. prvalue

mlogit: Predictions for jobprst

Predicted probabilities for each category:
  Pr(y=Adeq|x):       0.1819
  Pr(y=Good|x):       0.3085
  Pr(y=Strong|x):     0.4742
  Pr(y=Dist|x):       0.0355

         pub1        phd     female
x=   3.208589  3.3291411  .24539877

. prvalue, x(female=1) rest(mean)

mlogit: Predictions for jobprst

Predicted probabilities for each category:
  Pr(y=Adeq|x):       0.1918
  Pr(y=Good|x):       0.3501
  Pr(y=Strong|x):     0.4480
  Pr(y=Dist|x):       0.0100

         pub1        phd     female
x=   3.208589  3.3291411          1
```

10) Marginal and Discrete Change. After the `mlogit` command, we can use `prchange` [**SPost** (see L&F: 6.6.6, A.9)] to calculate the marginal and discrete change.

```
. prchange, rest(mean)

mlogit: Changes in Predicted Probabilities for jobprst

pub1
            Avg|Chg|         Adeq         Good       Strong         Dist
Min->Max   .17337222  -.21587602  -.13086843   .20648962   .14025478
   -+1/2   .01078053  -.01767299  -.00388807   .01742038   .00414068
  -+sd/2   .03468679  -.05692466  -.01244891   .05600607   .01336751
MargEfct   .04312737  -.01767309   -.0038906   .01742398    .0041397

phd
            Avg|Chg|         Adeq         Good       Strong         Dist
Min->Max   .28806599  -.30736637   -.2687656   .34701608    .2291159
```

```
   -+1/2   .08879882   -.0858417  -.09175596   .11727661   .06032103
  -+sd/2    .0873833  -.08447389  -.09029269   .11554819   .05921843
MargEfct   .35641503  -.08615865  -.09204887    .1221914   .05601612

female
         Avg|Chg|        Adeq        Good      Strong        Dist
    0->1   .03623489   .01507694   .05739284  -.02954715  -.04292263

             Adeq        Good      Strong        Dist
Pr(y|x)  .18188974  .30845267  .47417167  .03548593

           pub1       phd    female
    x=  3.20859  3.32914  .245399
sd(x)=  3.22118  .983948  .431649
```

11) Plot Discrete Change. You can use `mlogview` to compute the following plot:



12) Can you generate a plot with both discrete and factor changes?

13) Testing for IIA. The mlogtest [SPost (see L&F: 6.3.1)] command can also be used to test the IIA (independence of irrelevant alternatives) assumtion in multinomial logit models.

```
. mlogtest, iia

**** Hausman tests of IIA assumption

 Ho: Odds(Outcome-J vs Outcome-K) are independent of other alternatives.

 Omitted |      chi2   df   P>chi2   evidence
---------+------------------------------------
    Adeq |     1.149    8    0.997   for Ho
    Good |     0.383    8    1.000   for Ho
  Strong |    -5.332    8     ---    for Ho
----------------------------------------------
 Note: If chi2<0, the estimated model does not
 meet asymptotic assumptions of the test.
```

14) Close the Log File.

```
. log close
```

8: Models for Count Outcomes with Stata

The file `lhs_guide07.do` contains these Stata commands. For more details about models for count outcomes, please read Chapter 7 of L&F (2003).

Summary of Key Commands

poisson depvar indvars	Estimate the Poisson regression model (see L&F: 7.1.1, 7.2.1).
prcounts	Compute the predicted rate and probabilities. **SPost**(see L&F: 7.1.2, 7.1.3, 7.2.4, A.10).
nbreg depvar indvars	Estimate the negative binomial regression model (see L&F: 7.3).

zip depvar indvars**, inf(**indvars**)** Estimated the ZIP model (see L&F: 7.4).

zinb depvar indvars**, inf(**indvars**)** Estimated the ZINB model (see L&F: 7.4).

Trying the Commands

1) Open a Log.

```
. log using lhs_guide07.log, replace
--------------------------------------------------------------------------------
  log:  lhs_guide07.log
  log type:  text
 opened on:  20 Jun 2004, 00:24:46
```

2) Load the Data.

```
. use science2, clear
. drop if pub6>=. | female>=. | phd>=. | enroll>=.
(30 observations deleted)

. sum pub6 female phd enrol

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
        pub6 |       278    3.845324    4.237681          0         29
      female |       278    .3453237    .4763312          0          1
         phd |       278    3.166331    .9961592          1       4.66
       enrol |       278    5.564748    1.467253          3         14
```

3) Estimate the Poisson Regression Model. Again, the dependent variable is listed first.

```
. poisson pub6 female phd enrol, nolog

Poisson regression                                Number of obs   =        278
                                                  LR chi2(3)      =      85.33
                                                  Prob > chi2     =     0.0000
Log likelihood = -870.50576                       Pseudo R2       =     0.0467

------------------------------------------------------------------------------
        pub6 |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      female |  -.2521173   .0674324    -3.74   0.000    -.3842824   -.1199521
         phd |    .189403   .0318128     5.95   0.000     .1270511     .251755
       enrol |  -.1327736   .0231447    -5.74   0.000    -.1781363   -.0874108
       _cons |   1.531677   .1679706     9.12   0.000      1.20246    1.860893
------------------------------------------------------------------------------
```

4) Factor Changes. `listcoef` [**SPost** (see L&F: 7.2.3, 7.3.4, A.3)] computes the factor change coefficients.

```
. listcoef, help

poisson (N=278): Factor Change in Expected Count

  Observed SD: 4.2376808

----------------------------------------------------------------------
        pub6 |      b         z     P>|z|    e^b    e^bStdX      SDofX
-------------+--------------------------------------------------------
      female |  -0.25212   -3.739   0.000   0.7772   0.8868     0.4763
         phd |   0.18940    5.954   0.000   1.2085   1.2076     0.9962
       enrol |  -0.13277   -5.737   0.000   0.8757   0.8230     1.4673
----------------------------------------------------------------------
       b = raw coefficient
       z = z-score for test of b=0
   P>|z| = p-value for z-test
     e^b = exp(b) = factor change in expected count for unit increase in X
 e^bStdX = exp(b*SD of X) = change in expected count for SD increase in X
   SDofX = standard deviation of X
```

5) Marginal and Discrete Changes. It is also possible to use the `prchange` command [**SPost** (see L&F: 7.2.3, A.9)] to compute the discrete change in the expected count/rate.

```
. prchange

poisson: Changes in Predicted Rate for pub6

        min->max      0->1     -+1/2    -+sd/2  MargEfct
female   -0.8970   -0.8970   -0.9326   -0.4433   -0.9302
   phd    2.4481    0.4224    0.6998    0.6971    0.6988
 enrol   -3.9824   -0.9604   -0.4902   -0.7199   -0.4899

exp(xb):   3.6894

         female      phd    enrol
    x=  .345324  3.16633  5.56475
sd(x)=  .476331  .996159  1.46725
```

6) Predicted Rate and Probabilities. We can use the `prvalue` command [**SPost** (see L&F: 7.2.4, A.13)]
to compute the predicted rate and probabilities at specific values of the independent variables.

```
. prvalue, x(female=0) rest(mean)

poisson: Predictions for pub6

Predicted rate: 4.03     95% CI [3.74   ,   4.33]

Predicted probabilities:

  Pr(y=0|x):   0.0179  Pr(y=1|x):   0.0719
  Pr(y=2|x):   0.1447  Pr(y=3|x):   0.1941
  Pr(y=4|x):   0.1954  Pr(y=5|x):   0.1573
  Pr(y=6|x):   0.1055  Pr(y=7|x):   0.0607
  Pr(y=8|x):   0.0305  Pr(y=9|x):   0.0136

       female        phd      enrol
x=          0  3.1663309  5.5647482

. prvalue, x(female=1) rest(mean)

poisson: Predictions for pub6

Predicted rate: 3.13     95% CI [2.8    ,    3.5]

Predicted probabilities:

  Pr(y=0|x):   0.0438  Pr(y=1|x):   0.1370
  Pr(y=2|x):   0.2143  Pr(y=3|x):   0.2234
  Pr(y=4|x):   0.1747  Pr(y=5|x):   0.1093
  Pr(y=6|x):   0.0570  Pr(y=7|x):   0.0255
  Pr(y=8|x):   0.0100  Pr(y=9|x):   0.0035

       female        phd      enrol
x=          1  3.1663309  5.5647482
```

7) Predicted Probabilities for the Sample. `prcounts` [**SPost**(see L&F: 7.1.2, 7.1.3, 7.2.4, A.10)] computes predicted probabilities of counts 0 to 9 for each observation in the sample. You can pick up to 4 letters to label the variables that are created. `max(#)` is the maximum count for which predicted probabilities should be provided. `plot` specifies that variables for plotting expected counts should be generated. We use `prm` for the Poisson regression model.

```
. prcounts prm, plot max(9)

. sum prm*

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
     prmrate |       278    3.845324    1.091294   1.118805   7.339315
      prmpr0 |       278    .0354102    .0382139   .0006495   .3266698
      prmpr1 |       278    .1030496    .0690706   .0047669   .3654799
::: and so on :::
-------------+--------------------------------------------------------
      prmpr9 |       278    .0173669     .020694   2.47e-06   .1105892
      prmcu0 |       278    .0354102    .0382139   .0006495   .3266698
      prmcu1 |       278    .1384598    .1057366   .0054163   .6921498
::: and so on :::
-------------+--------------------------------------------------------
      prmcu9 |       278    .9837673    .0280122   .7944917   .9999996
     prmprgt |       278    .0162327    .0280122   3.58e-07   .2055083
      prmval |        10         4.5     3.02765          0          9
     prmobeq |        10    .0920863    .0561172   .0215827   .1942446
     prmpreq |        10    .0983767     .061986   .0173669   .1853696
-------------+--------------------------------------------------------
     prmoble |        10    .6528777    .2495748   .1942446   .9208633
     prmprle |        10    .6167697    .3557846   .0354102   .9837673
```

The variables `prmval`, `prmobeq`, and `prmpreq` are used for plotting. These contain the information for the counts 0 through 9. `prmobeq` has the observed probabilities; `prmpreq` has the average prediction for the regression model; `prmval` has the value of the count from 0 to 9. We'll use these below.

8) The Negative Binomial Regression Model. We can use the same types of commands for the NBRM. Notice that the model can takes a long time to converge. For details about the `nbreg` command, please see L&F: 7.3.

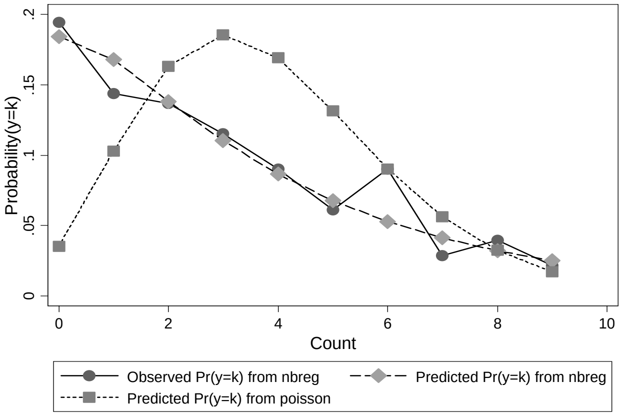
```
. nbreg pub6 female phd enrol, nolog

Negative binomial regression                      Number of obs   =        278
                                                  LR chi2(3)      =      22.94
                                                  Prob > chi2     =     0.0000
Log likelihood = -673.83849                       Pseudo R2       =     0.0167

------------------------------------------------------------------------------
        pub6 |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
      female |  -.2938846   .1334392    -2.20   0.028    -.5554206   -.0323486
         phd |   .2003828    .063463     3.16   0.002     .0759977    .3247679
       enrol |  -.1513925   .0455889    -3.32   0.001     -.240745   -.0620399
       _cons |    1.60997   .3298255     4.88   0.000     .9635242    2.256417
-------------+----------------------------------------------------------------
    /lnalpha |   -.238717   .1237295                     -.4812223    .0037883
-------------+----------------------------------------------------------------
       alpha |   .7876378    .097454                      .6180275    1.003796
------------------------------------------------------------------------------
Likelihood-ratio test of alpha=0:  chibar2(01) =  393.33 Prob>=chibar2 = 0.000
```

9) Predicted Probabilites for the Sample. `prcounts` computes the predicted rate and probabilities for the NBRM:

```
. prcounts nbr, plot max(9)
. sum nbr*

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+-----------------------------------------------------
     nbrrate |       278    3.862753    1.214629   .9563452   7.889484
      nbrpr0 |       278    .1840512     .057882   .0813652   .4902415
      nbrpr1 |       278    .1678654     .035035   .0889832   .2674114
::: and so on :::
-------------+-----------------------------------------------------
      nbrpr9 |       278    .0251105    .0095959    .000499    .043354
      nbrcu0 |       278    .1840512     .057882   .0813652   .4902415
      nbrcu1 |       278    .3519166    .0923878   .1703484   .7576529
::: and so on :::
-------------+-----------------------------------------------------
      nbrcu9 |       278    .9058141     .062153    .690679   .9996071
     nbrprgt |       278    .0941859     .062153   .0003929    .309321
      nbrval |        10         4.5     3.02765          0          9
     nbrobeq |        10    .0920863    .0561172   .0215827   .1942446
     nbrpreq |        10    .0905814    .0572139   .0251105   .1840512
-------------+-----------------------------------------------------
     nbroble |        10    .6528777    .2495748   .1942446   .9208633
     nbrprle |        10    .6510095    .2424767   .1840512   .9058141
```

10) Plot Predicted Sample Probabilities. We can now plot the values computed by `prcounts`:

```
. scatter nbrobeq nbrpreq prmpreq nbrval, ///
>   c(l l l) ytitle("Probability(y=k)")
```



Which model fits better?

11) ZIP Model. `zip` (see L&F: 7.4.1) has been added as Stata commands in Version 6. The `zip` command alone estimates a Poisson model. The subcommand **inf(**indvars**)** adds the zero inflated specification.

```
. zip pub6 female phd enrol, inf(female phd enrol) nolog

Zero-inflated poisson regression                  Number of obs   =        278
                                                  Nonzero obs     =        224
                                                  Zero obs        =         54

Inflation model = logit                           LR chi2(3)      =      49.76
Log likelihood  = -788.3856                       Prob > chi2     =     0.0000

------------------------------------------------------------------------------
        pub6 |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
pub6         |
      female |  -.1404363   .0690717    -2.03   0.042    -.2758143   -.0050582
         phd |   .1434903   .0330619     4.34   0.000     .0786902    .2082905
       enrol |  -.1219442   .0244722    -4.98   0.000    -.1699087   -.0739796
       _cons |   1.780094   .1744264    10.21   0.000     1.438224    2.121963
-------------+----------------------------------------------------------------
inflate      |
      female |   .6079487   .3364265     1.81   0.071    -.0514352    1.267332
         phd |  -.2665731   .1655813    -1.61   0.107    -.5911065    .0579602
       enrol |   .0770141   .1130511     0.68   0.496    -.1445618    .2985901
       _cons |  -1.347693   .8364893    -1.61   0.107    -2.987182    .2917961
------------------------------------------------------------------------------
```

12) Factor, Marginal, and Discrete Changes. `prchange` and `listcoef` can compute marginal/discrete change coefficients and factor change in the predicted rate for zip models, respectively.

```
. listcoef, help

zip (N=278): Factor Change in Expected Count

 Observed SD: 4.2376808

Count Equation: Factor Change in Expected Count for Those Not Always 0

----------------------------------------------------------------------
        pub6 |      b         z     P>|z|    e^b    e^bStdX      SDofX
-------------+--------------------------------------------------------
      female |  -0.14044   -2.033   0.042   0.8690   0.9353     0.4763
         phd |   0.14349    4.340   0.000   1.1543   1.1537     0.9962
       enrol |  -0.12194   -4.983   0.000   0.8852   0.8362     1.4673
----------------------------------------------------------------------
       b = raw coefficient
       z = z-score for test of b=0
   P>|z| = p-value for z-test
     e^b = exp(b) = factor change in expected count for unit increase in X
 e^bStdX = exp(b*SD of X) = change in expected count for SD increase in X
   SDofX = standard deviation of X

Binary Equation: Factor Change in Odds of Always 0

----------------------------------------------------------------------
     Always0 |      b         z     P>|z|    e^b    e^bStdX      SDofX
-------------+--------------------------------------------------------
      female |   0.60795    1.807   0.071   1.8367   1.3359     0.4763
         phd |  -0.26657   -1.610   0.107   0.7660   0.7668     0.9962
       enrol |   0.07701    0.681   0.496   1.0801   1.1196     1.4673
----------------------------------------------------------------------
       b = raw coefficient
       z = z-score for test of b=0
   P>|z| = p-value for z-test
     e^b = exp(b) = factor change in odds for unit increase in X
 e^bStdX = exp(b*SD of X) = change in odds for SD increase in X
   SDofX = standard deviation of X

. prchange, rest(mean)

zip: Changes in Predicted Rate for pub6

        min->max      0->1     -+1/2    -+sd/2
female   -0.9136   -0.9136   -0.9184   -0.4376
   phd    2.4951    0.4818    0.7086    0.7059
 enrol   -4.1106   -0.9644   -0.5049   -0.7413

exp(xb):   3.7264

base x values for count equation:

         female      phd    enrol
```

```
    x=  .345324  3.16633  5.56475
sd(x)=  .476331  .996159  1.46725

base x values for binary equation:

         female      phd    enrol
    x=  .345324  3.16633  5.56475
sd(x)=  .476331  .996159  1.46725
```

13) The ZINB Model. We can use the same types of commands for the ZINB. For details about the `zinb` command, please see L&F: 7.4.

```
. zinb pub6 female phd enrol, inf(female phd enrol) nolog

Zero-inflated negative binomial regression        Number of obs   =        278
                                                  Nonzero obs     =        224
                                                  Zero obs        =         54

Inflation model = logit                           LR chi2(3)      =      19.84
Log likelihood  =  -672.595                       Prob > chi2     =     0.0002

------------------------------------------------------------------------------
        pub6 |      Coef.   Std. Err.      z    P>|z|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
pub6         |
      female |  -.2148377   .1469343    -1.46   0.144    -.5028236    .0731483
         phd |   .2098783   .0635345     3.30   0.001     .0853531    .3344036
       enrol |    -.13814   .0464219    -2.98   0.003    -.2291252   -.0471547
       _cons |   1.507456   .3305816     4.56   0.000     .8595279    2.155384
-------------+----------------------------------------------------------------
inflate      |
      female |   13.16564   372.3562     0.04   0.972    -716.6391    742.9704
         phd |   .5209843   .9629577     0.54   0.588    -1.366378    2.408347
       enrol |   .4379132   .5616997     0.78   0.436    -.6629981    1.538824
       _cons |  -20.08333   372.4266    -0.05   0.957     -750.026    709.8594
-------------+----------------------------------------------------------------
    /lnalpha |  -.3209474    .143072    -2.24   0.025    -.6013635   -.0405314
-------------+----------------------------------------------------------------
       alpha |   .7254614   .1037932                          .5480638     .960279
------------------------------------------------------------------------------
```

14) Factor, Marginal, and Discrete Changes. The `listcoef` and `prchange` commands can also apply to the zinb models.

```
. listcoef, help

zinb (N=278): Factor Change in Expected Count

 Observed SD: 4.2376808

Count Equation: Factor Change in Expected Count for Those Not Always 0

----------------------------------------------------------------------
        pub6 |      b         z     P>|z|    e^b    e^bStdX      SDofX
-------------+--------------------------------------------------------
      female |  -0.21484   -1.462   0.144   0.8067   0.9027     0.4763
         phd |   0.20988    3.303   0.001   1.2335   1.2325     0.9962
       enrol |  -0.13814   -2.976   0.003   0.8710   0.8165     1.4673
-------------+--------------------------------------------------------
    ln alpha |  -0.32095
       alpha |   0.72546   SE(alpha) = 0.10379
----------------------------------------------------------------------
       b = raw coefficient
       z = z-score for test of b=0
   P>|z| = p-value for z-test
     e^b = exp(b) = factor change in expected count for unit increase in X
 e^bStdX = exp(b*SD of X) = change in expected count for SD increase in X
   SDofX = standard deviation of X

Binary Equation: Factor Change in Odds of Always 0

----------------------------------------------------------------------
     Always0 |      b         z     P>|z|    e^b    e^bStdX      SDofX
-------------+--------------------------------------------------------
      female |  13.16564    0.035   0.972 5.22e+05 529.1149     0.4763
         phd |   0.52098    0.541   0.588   1.6837   1.6803     0.9962
       enrol |   0.43791    0.780   0.436   1.5495   1.9013     1.4673
----------------------------------------------------------------------
       b = raw coefficient
       z = z-score for test of b=0
   P>|z| = p-value for z-test
     e^b = exp(b) = factor change in odds for unit increase in X
 e^bStdX = exp(b*SD of X) = change in odds for SD increase in X
   SDofX = standard deviation of X

. prchange, rest(mean)

zinb: Changes in Predicted Rate for pub6

        min->max      0->1     -+1/2    -+sd/2
female   -0.9692   -0.9692   -0.8390   -0.3876
   phd    2.7709    0.4539    0.7943    0.7912
 enrol   -4.2062   -1.0513   -0.5223   -0.7670

exp(xb):   3.7776

base x values for count equation:

         female      phd    enrol
    x=  .345324  3.16633  5.56475
sd(x)=  .476331  .996159  1.46725

base x values for binary equation:

         female      phd    enrol
    x=  .345324  3.16633  5.56475
sd(x)=  .476331  .996159  1.46725
```

15) Close the Log File.

```
. log close
```

Appendix: Stata Commands and Working Mode

Overview of Stata Commands

Getting Help: Type *help <command name>* to learn more about any command.

findit *word*	Get information that contains this word from web resources
help *command*	Get help for a specific command.
lookup *word*	Get help for a more general topic.
display *expression*	A simple calculator function. For example,

```
      display 2+2
      di sqrt(2)/2
      di normprob(-1.1)
```

Stopping execution	Click on *break* to stop a command that is currently running.

File Suffixes

*name***.dta**	Stata data file.
*name***.log**	Log file with commands and results saved as text, except graphs.
*name***.smcl**	Log file is saved in smcl format (can only be read in Stata).
*name***.do**	Command file, often created in the Stata do.file editor.
*name***.gph**	Stata graphics file; suffix automatically inserted by Stata.
*name***.wmf**	Graph in Windows Metafile format for use in other programs.
*name***.ado**	Automatically loaded ado files.

Loading and Saving Data

use *filename*, **clear**	Load a new data file, discarding any data currently being used.
save *filename*, **replace**	Save data and replace file if it already exists.
Data browser	Click on 🔍 to view data.
Data editor	Click on ▦ to edit data. This is dangerous! Don't use it.
Do-file editor	Click on ✎ to edit and/or run a Stata .do file.
dir	Lists files in the current directory.
cd	Change directory. For example, `cd a:\`

Logging Output

capture log close	Close a log file if it exists; ignore any error if it occurs.
log using *filename*, **replace**	Open a log file and allow overwriting of an existing file.
log close	Close a log file**.**

Descriptive Statistics

describe	Display summary of the contents of a data file.
summarize	Compute descriptive statistics.
tabulate	Compute frequencies.

Selecting Observations

drop *varlist*	Drop specific variables.
mark and **markout**	These commands make it simple to explicitly exclude missing data. `mark` *`markvar`* generates a new variable *`markvar`* that equals 1 for all cases. `markout` *`markvar varlist`* changes the values of *`markvar`* from 1 to 0 for any cases in which values of any of the variables in *`varlist`* are missing.

Creating Variables

generate *newvar=expression*	New variable created as algebraic expression of other variables.

gen dens=pop/area	New variable as the ratio of two old variables.
gen sqrtage=**sqrt(**age**)**	New variable equal to square root of old variable.
gen logwg=**ln(**wages**)**	New variable equal to the log of the old variable.
gen *newvar* = *oldvar*	New variable equal to an existing variable.
recode	Recodes the values of an existing variables. Always create a new variable equal to the original one you want to modify. Then recode the new equivalent variable. **recode** is useful for creating categorical and indicator (dummy) variables.
recode *varname* 1=2 3=4	Change 1 to 2; and 3 to 4.
recode 2=1 *=0	Change 2 to 1; all other to 0

Note: Use of "*" will change missing values to the "*=" value!! Add an if statement, "if varname<." to exclude missing values from the changes like the following.

recode *varname* 2=1 *=0 **if** *varname*<**.**	2 changed to 1, all else 0, but missings are not bothered.
recode *varname* 1/4=2	Change 1 through 4 to 2.
recode *varname* 1 3 4 5 = 7	Change 1,3,4, and 5 to 7.
recode *varname* 1 3/5 =7	Change 1 and 3 through 5 to 7.
recode *varname* **min**/5=**min**	Recode the minimum through 5 to minimum.
recode *varname* **.**=9	Change missing value to 9.
recode *varname* 9=**.**	Change 9 to a missing value.

Other Useful Commands

graph export *filename*.wmf, replace	Redirect the graph output to filename. The extension .wmf tells Stata to create a Window Metafile.
delimit { cr | ; }	Resets the character that marks the end of a command to a carriage return or the character ";" *In a command file, if you do not have a carriage return after your last line, Stata will ignore it!*

Mathematical and Logical Expressions

+	add	**-**	subtract	/	divide
*	multiply	^	power	ln()	natural log
exp()	exponential	sqrt()	square root	~	not
&	and	|	or	>	greater than
>=	greater than or equal	<	less than	<=	less than or equal
==	equal	~=	not equal		

Other Useful Expressions

`*` if you put it at the very beginning of a line in a do file, Stata 8 will take this line as comment.

`//` this sign indicates that Stata 8 will take anything after this sign as comment.

`///` this sign is used when the syntax for a command is too long for one line and it allows Stata to search for the rest of the syntax in the next line. Once Stata hits this sign, it will be bumped to next line. Therefore you can also put some comments after this sign. Note that in Stata, by default it is one line per command, otherwise you will receive error message.

Interactive and Batch Mode

In Stata, you can execute commands either from the command line or running a text file (called a do file) that contains the commands. With few exceptions, anything you can run from the command line you can execute by "doing" a do file.

1) Interactive Mode: To run commands interactively, you type one command at a time in the window labeled *Stata Command*. Press Enter when you have finished the command.

- Retrieving prior commands with PageUp and PageDown: You can bring already executed commands into the command window by pressing the PageUp and PageDown keys. You can edit these commands to make changes.
- Retrieving prior commands from the Review window. The window labeled "Review" lists all commands that have been executed in the current sessions. You can click once to bring a command from the Review window to the Command window. Click it twice to execute the command immediately.

2) Batch Mode: To run commands in batch mode, you type the commands in a text file that has the suffix `.do`. You can create these command files with Stata's Do-file editor, or with any editor that saves a text file. You can do this from Word Perfect or Word, but it is awkward. We prefer TextPad (www.textpad.com). To execute a batch program you:

- Create the file in a text editor:
- Save it to the directory where you are working. For example, save it in `c:\mywork\myfile.do`
- Assuming you are in directory `c:\mywork` in Stata (use cd c:\mywork), run the do file by entering: `do myfile`

3) Using the Stata Do-File editor: If you click on , you will be in the Stata do-file editor. This editor works like most text editors; use help to get more details. After you enter your program, click to do the file. When you do the file, the results are sent to the *Stata Results* window and the log file. Click to run the commands, without sending them to the *Stata Results* window.

4) Structure of a Do File: Here is an example of what you want to put in a do file. Note that anything following a * is not executed but is printed. For details see L&F(2003), 24-28.

```
capture log close
* close any log file that may be open
set more off
* don't pause when output scrolls off the page
set scheme smanual
* set graph schemes
log using myfile.log, replace
* log results to file myfile.log
* your commands go here
log close
```