ECON 360 - Econometrics

Useful STATA Commands for In-Class Exercises

This file provides commands that are used to perform in-class exercises. We will be using the dataset "WAGE2.dta" to demonstrate usage of STATA commands.

To open a dataset: File > Open > Navigate to the folder where you have saved the data file (in this example, "WAGE2.dta")

You can implement a command (i.e., tell STATA to do something) by typing your code in the *Command* window (the rectangular window at the bottom of the screen) and press "Enter". The results appear in the *Output* window (the rectangular window above the Command window).

I. EXAMINING A DATA SET: THE BASICS

- To get a quick glimpse at the data **browse** the dataset: `br`
- To see the structure of the dataset **describe** its contents: `des`
- To **tabulate** categories of a specific variable, "married": `tab married`
- To get **summary statistics** of all the variables in the dataset: `sum`
  - This will report the number of observations, mean, standard deviation, minimum value and maximum value of the variables.
- To get **summary statistics** of a specific variable, "wage": `sum wage`
- To get additional statistics, such as skewness, kurtosis, the four smallest and largest values, and various percentile, add the "detail" option after "sum":
  `sum wage, detail`
- To get **correlation** between two variables, age and wage: `corr age wage`
- **Use of a conditional operator** (Using `if`):
  Suppose we want to know mean "wage" for married individuals:

```
sum wage if married == 1
```

Suppose we want to know mean “wage” for unmarried individuals:

```
sum wage if married == 0
```

Here we use a double equal sign (==) for testing equality.

Following are some useful operators in Stata:

+ plus

− minus

∗ multiply

/ divide

ˆ exponent

= equal

& and

| or

> greater than

>= greater than or equal to

<= less than or equal to

< less than

! not (also ˜)

! = not equal to (also ˜ =)

== logical test for equality (usually follows “if”)

II. DATA CLEANING: CREATING VARIABLES

We can create new variables using `gen` command. We can change the value of an existing variable using `replace` command.

Following are some examples:

- Generate log of "hours": `gen lhours = log(hours)` Note: While creating a new variable, pick a name (in this example, `lhours`) for the new variable. The `gen` command is followed by "new variable name", which is followed by "= mathematical expression to create the variable".
- Generate square of "age": `gen agesq = age*age`
- Generate an interaction term (interact married with education):
  `gen marriededuc = married*educ`
- Generate an indicator variable "fulltime" which takes value 1 if individuals work 40 hours or more per week, otherwise the variable takes value 0.
  `gen fulltime = 1 if hours>=40`
  `replace fulltime = 0 if hours<40`
- Generate a random variable: `gen randomvar = uniform()`
  The `uniform()` function generates random draws from a uniform distribution between 0 and 1.

III. DATA VISUALIZATION: CREATING PLOTS

*SCATTER PLOT

To create a scatterplot, use the scatter command, then list the variables you want to plot. The first variable you list will be the Y variable and the second will be the X variable.

Suppose we want to plot "wage" in the y-axis and "age" in the x-axis:

```
scatter wage age
```

*SCATTER PLOT WITH LINEAR PREDICTION

We want to make a scatterplot, and add a linear prediction-based line of best fit. To do that, first specify the scatterplot, and next add the command for linear fit plot

```
scatter wage age || lfit wage age
```

*HISTOGRAM

Plot the distribution of "hours":

```
hist hours
```

IV. DATA ANALYSIS: RUN REGRESSION

*SIMPLE REGRESSION

Let's run a simple regression: $wage = \beta_0 + \beta_1 educ + u$

`reg wage educ`

Note: We are regressing wage(y) on educ(x). The `reg` command is followed by dependent or y variable first, and then by explanatory or x variable.

- Linear Model: $wage = \beta_0 + \beta_1 educ + u$

  `reg wage educ`

- Log-linear Model: $lwage = \beta_0 + \beta_1 educ + u$

  `reg lwage educ`

  Note: "lwage" variable is already there in the dataset, we don't need to generate it.

- Linear-log Model: $wage = \beta_0 + \beta_1 leduc + u$

  First, generate log of education: `gen leduc = log(educ)`

  Then, run regression: `reg wage leduc`

- Log-log Model: $lwage = \beta_0 + \beta_1 leduc + u$

  `reg lwage leduc`

*MULTIPLE REGRESSION

Next we run a multiple regression, where we have one dependent (wage) variable and 2 explanatory variables (age and educ): $wage = \beta_0 + \beta_1 educ + \beta_2 age + u$

`reg wage educ age`

Next we run the multiple regression for those who have more than 12 years of education:

`reg wage educ age if educ>12`

Next we run the multiple regression by **adding** more explanatory variables (tenure, square of tenure, experience, and square of experience):

$wage = \beta_0 + \beta_1 educ + \beta_2 age + \beta_3 tenure + \beta_4 tenuresq + \beta_5 exper + \beta_6 expersq + u$

Note: We first need to create the variables "square of tenure" and "square of experience":

`gen tenuresq = tenure*tenure`

`gen expersq = exper*exper`

Then, run the regression: `reg wage educ age tenure tenursq exper expersq`

Next we re-run the multiple regression by **dropping** some explanatory variables (experience, and square of experience):

$wage = \beta_0 + \beta_1 educ + \beta_2 age + \beta_3 tenure + \beta_4 tenuresq + u$

`reg wage educ age tenure tenursq`

*CONTROL FOR HETEROSKEDASTICITY

To control for heteroskedasticity, run the regression with **"robust"** option

`reg wage age tenure tenursq, robust`

This reports robust standard errors.

*ADD INTERACTION TERM

Let's add an interaction term for being married and living in urban area.

First, generate the interaction term:

`gen marriedurban = married*urban`

Then run the regression:

$wage = \beta_0 + \beta_1 educ + \beta_2 age + \beta_3 tenure + \beta_4 tenuresq + \beta_5 marriedurban + u$

`reg wage educ age tenure tenursq marriedurban`

*LINEAR PROBABILITY MODEL

In this case, the dependent variable is a **binary variable**. Consider the following model: $Fulltime = \beta_0 + \beta_1 age + u$. Here, β_1 tells us the change in the probability of working full time from a one year increase in age, holding everything else constant.

Note that we already created the indicator variable "fulltime" which takes value 1 if individuals work 40 hours or more per week, otherwise the variable takes value 0.

```
gen fulltime = 1 if hours>=40
replace fulltime = 0 if hours<40
```

We will now run the regression: `reg fulltime age`

V. POST-ESTIMATION ANALYSIS: AFTER RUNNING THE REGRESSION

*PREDICT FITTED VALUES AND RESIDUALS

The `predict` command can be used to generate predicted (fitted) values and the residuals after running `reg`.

First, run the regression: $wage = \beta_0 + \beta_1 educ + u$

```
reg wage educ
```

Next, generate the predicted values of wage using `predict` immediately after running the regression:

```
predict wagehat, xb
```

Similarly, generate the residual for each observation:

```
predict uhat, resid
```

Next, we can perform more analysis using these newly generated variables, "wagehat" and "uhat". For example:

```
sum wagehat
scatter wagehat educ
sum uhat
gen uhatsq = uhat*uhat
```

*HYPOTHESIS TESTING

First, run the log-linear model:

```
reg lwage educ age exper married motheduc fatheduc
```

We will now test the hypothesis that experience has no effect on log(wage).

```
test exper = 0
```

The t-statistic can be obtained by running the following code:

```
scalar tvalue = (_b[exper]-0)/_se[exper]
```

We can get the p-value by running the following code:

```
scalar pvalue = ttail(928,tvalue)
```

Note: The degrees of freedom = No. of observations - No. of explanatory variables - 1 = 935 - 6 - 1 = 928.

The following code will display the t-statistic and the p-value on Output window:

```
display "T-value:  " tvalue ", P-value:  " pvalue
```

Next, we will test the hypothesis that the rate of return to an extra year of experience is 10%.

```
test exper = 0.10
scalar tvalue2 = (_b[exper]-0.10)/_se[exper]
display "T-value:  " tvalue2
```

Next, we will conduct a test for joint significance of motheduc and fatheduc:

```
test motheduc fatheduc
```

Next, we test the hypothesis that another year of experience has the same effect on log(wage) as another year of education:

```
test educ = exper
```

Alternative code:

```
lincom educ-exper
```

Next, we investigate whether the effect of education on log(wage) differ by marital status. We will first generate an interaction term:

```
gen marreduc = married*educ
```

Then, we will run the regression by including the interaction term:

```
reg lwage educ age exper married motheduc fatheduc marreduc
```

Then, we will test whether the coefficient on marreduc is zero or not.

```
test marreduc = 0
```