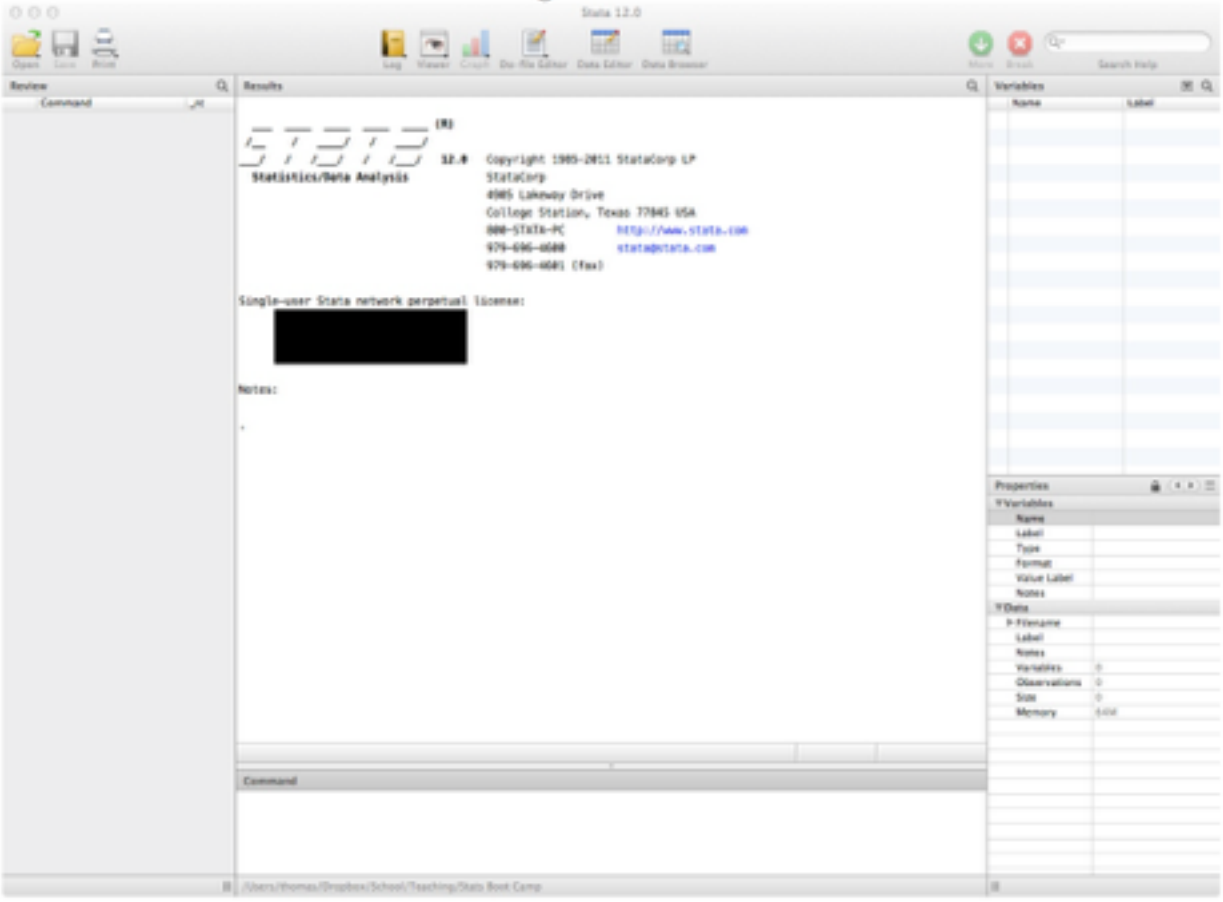
Stats Refresher Day 4 - Introduction to Stata
(Adapted from write-ups by Thomas Elliot)

Stata is one of a few statistical analysis programs that social scientists use. Stata is in the mid-range of how easy it is to use. Other options include SPSS, considered easier to use, but clunky if performing many commands, SAS, and R.

Stata features a primarily command line interface. What that means is you type in com- mands into the command field, and when you press enter the results of the command appear in the results field (as opposed to using menu options, the primarily interface for SPSS). This requires learning what the commands are and how to use them, making the initial learning curve a bit steep. But once you've learned the basic commands, it becomes easy to work with your dataset.

For this introduction, we will be using a sample of data from the General Social Survey, a popular dataset based on a national survey given approximately every two years.

Figure 1: Stata



1 Starting Stata

When you first start up Stata, you will be presented with the main window. Figure 1 shows what this looks like. On the left is the command history, which will list all the previous commands given during the current session. On the right is the variable list. When you load up a dataset, the available variables will be listed in the top, and the bottom will contain information about the variable currently selected in the list. In between is the results and command fields. The command field, located at the bottom, is where you type commands to tell Stata what to do. The results of those commands appear as plain text in the results field.

A few things to note:

1. Stata's variable names are case sensitive. So the variable AGE is different than the variable Age, which is different than the variable age.

2. Stata allows commands and options to be shortened so you don't need to type the full name of the command - just enough for Stata to know which one you are referring to. In Stata's help pages, the part of the command underlined is the minimum you need to type for Stata to recognize the command.

3. This shortening also applies to variable names. As long as you type enough of a variable name to distinguish it from the other variables, Stata will know which one you are talking about.

1.1 Opening and Saving Datasets

To open a dataset, we use the use command. But first, we need to tell Stata where the dataset is saved on the hard drive. We do this by setting our working directory to the directory containing the dataset using the cd command (for change directory):

```
cd "~/Documents/Stats"
cd "C:\Users\Mikaela\Documents\Stats"
```

You can also go to the File menu and select Change Working Directory for a directory menu.

After you have changed the working directory, you should see the current working directory in the status bar along the bottom of the Stata window.

You'll always want a log of what you do in Stata so you can go back to see what you've done before. To tell Stata to save the output of the results window to a text files, type:

```
log using mylog.log, text
```

Notice that we've added an option to this command. Most Stata commands have options that allow you to customize the results. Options are specified after a comma in the command.

You can add replace or append as options to the log command. Replace will replace the log file if a previous one exists. Append will start the new log at the end of the old log file:

```
log using mylog.log, replace
log using mylog.log, append
```

Note that Stata will give you an error if you try to create a log file that already exists unless you specify the replace or append option.

After you've finished using Stata, you close the log with the command

```
log close
```

You can then find the contents of your results window saved in a text file with the name you gave when you started the log. This way, you can keep track of everything you've done and the results of your analysis. It's important to keep track of what you do with your data so it's easy to recreate your results later on.

You can now load up a dataset. Stata datasets are stored as .dta files. Open the dataset using the use command:

```
use gss2010.dta
```

You'll see the command appear in the results window. If there was an error, it will also be reported in the results window. If Stata successfully opened the file, you'll see the variables in the dataset listed on the right hand side.

To save a dataset, you use the save command:

```
save gss2010.dta, replace
```

Notice the `, replace` in the command - this is an option. In Stata, command options are included after a comma and modify the command in specific ways. We'll see more options later. Here, the replace option tells Stata to replace any current files named `gss2010.dta`, if they exist. If you didn't include the replace, Stata would give you an error, saying the filename already exists.

Now that you know how to open and save datasets, let's explore the data.

1.2 Modifying a Variable

We can generate summary statistics for interval and ratio variables using the `sum` command:

```
. sum AGE
```

Variable	Obs	Mean	Std. Dev.	Min	Max
AGE	**2041**	**47.96717**	**17.67799**	**18**	**89**

`sum` is a shortcut for the command `summarize`. As you can see, this gives us the mean, standard deviation, min, and max of the variable. We can ask for even more information by using the `detail` option

```
. sum AGE, d
```

AGE OF RESPONDENT

Percentiles		Smallest		
1%	**19**	**18**		
5%	**22**	**18**		
10%	**25**	**18**	Obs	**2041**
25%	**33**	**18**	Sum of Wgt.	**2041**
50%	**47**		Mean	**47.96717**
		Largest	Std. Dev.	**17.67799**
75%	**61**	**89**		
90%	**72**	**89**	Variance	**312.5112**
95%	**80**	**89**	Skewness	**.2921034**
99%	**88**	**89**	Kurtosis	**2.234348**

Let's say we want a dummy variable for respondents who are of retirement age. We can generate new variables using the generate command:

```
gen retire = AGE > 60
```

After typing `gen` you type the name of the new variable, then an equal sign and what you want the values of the new variable to be. Here, we've defined the variable `retire` to contain the result of the boolean expression `Age>60`. We'll learn more about boolean expressions later, but for know just know that if the expression is true for a case, the variable `retire` is given the value of 1, and if the expression is false then `retire` gets 0. We can see the results if we tab the new variable:

```
. tab retire
```

retire	Freq.	Percent	Cum.
0	**1,528**	**74.76**	**74.76**
1	**516**	**25.24**	**100.00**
Total	**2,044**	**100.00**	

But wait! We've defined retirement age as 60, but normally it's 65. Don't worry, we can easily change values in variables using replace:

```
. replace retire = AGE >= 65
(120 real changes made)
```

Now if we tab our variable we get:

```
. tab retire
```

retire	Freq.	Percent	Cum.
0	1,648	80.63	80.63
1	396	19.37	100.00
Total	2,044	100.00	

Notice, however, that the table is hard to read because of the way we've defined the variable. What does a 0 mean? What does a 1 mean? And what exactly is `retire` measuring? You might know now, since we just created the variable, but what if we come back to the dataset in six months? Will we remember then? Enter labels! Stata allows us to label variables and values to better explain what our variables are measuring. Let's first label our variable.

```
label variable retire "Respondent is of retirement age"
```

The label command actually has many sub commands, which you access with the second word. Here, we are using the `label variable` subcommand. Then we tell Stata which variable we are labeling, and finally the actual label in quotation marks. If you look at the `retire` variable in the variable window, you should see our label next to it. Now let's label our values. Labeling values is a two step process. First, we have to define labels for the actual values, then we have to assign the value labels to the specific variable. This allows us to assign the same value labels to multiple variables (for example, if you have a series of questions with Likert scale answers, you could use the same value labels for all the questions).

```
label define retirelabel 0 "Not Retirement Age" 1 "Retirement Age"
```

We label the label command again, though this time we are using the label define subcommand. We first give the set of value labels a name. The label define subcommand takes value-label pairs, as many of them as you want. So first comes the value you will label, then comes the label for that value in quotation marks, followed by the next value-label pair. Once we've defined value labels, we assign the value labels to a variable:

```
label value retire retirelabel
```

This subcommand assigns the `retirelabel` set of value labels to the variable retire. Now, if we tab our retire variable we get something more readable:

```
. tab retire
```

Respondent is of retirement age	Freq.	Percent	Cum.
Not Retirement Age	1,648	80.63	80.63
Retirement Age	396	19.37	100.00
Total	2,044	100.00	

Now that we've created our own variable, let's save our work. We can save datasets using the `save` command:

```
save gss2010new.dta
```

It's a good idea to keep a copy of your original data in case you accidentally screw up at some point later on and lose some important data (it happens too often). If you save the data with a new name, like above, it will create a new file leaving the hold one untouched.

2 Central Tendency and Variance

2.1 Basic Count Information

To find out how many cases we have, we can type count into the command line and press enter:

```
. count
2044
```

Notice we have a variable named `SEX`. We can see how people answered this question by typing `tab SEX` (note: in the following examples, the command you would type is preceded by a `.` and the results appear below the command)

```
. tab SEX
```

RESPONDENTS SEX	Freq.	Percent	Cum.
MALE	891	43.59	43.59
FEMALE	1,153	56.41	100.00
Total	2,044	100.00	

We've used the tabulate command to create a table of frequencies for the values of the variable `SEX`. Notice that we don't have to type out the entire tabulate command - Stata is smart enough to figure out what we mean as long as we type out enough of a command to differentiate from other commands. Stata help files will underline the part of the command necessary to type for Stata to know what you are talking about. Here, it is enough to type tab for Stata to know we want tabulate. As with commands, you can also type in just part of the name of a variable and Stata will know which one you mean, as long as you type enough to differentiate it from the other variable names.

2.2 Measures of Central Tendency

The first things we learned earlier this week were measures of central tendency, including mean, median, and mode. Stata can easily compute these for us for any interval or ratio level variable in the dataset. To calculate the mean of a variable, we can use the summarize command, which can be shortened to sum

```
. sum AGE

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
         AGE |      2041    47.96717    17.67799        18         89
```

So the mean age in this sample is 47.97 years.
To get the median, we use the same command but include the `details` option:

```
. sum AGE, d

                        AGE OF RESPONDENT
-------------------------------------------------------------
      Percentiles      Smallest
 1%           19            18
 5%           22            18
10%           25            18       Obs                2041
25%           33            18       Sum of Wgt.        2041

50%           47                     Mean           47.96717
                       Largest       Std. Dev.      17.67799
75%           61            89
90%           72            89       Variance       312.5112
95%           80            89       Skewness       .2921034
99%           88            89       Kurtosis       2.234348
```

The 50th percentile is the median, so in this case the median age is 47.

Stata does not have a command to calculate the mode, though rarely do people care about the mode so this usually isn't a problem.

2.3 Variance and Standard Deviation

You may have noticed that the summarize command also calculates the standard deviation, and variance with the details option. Looking at the results above, we see that the standard deviation for AGE is 47.97 years and the variance is 312.51 years squared.

We can list more than one variable at a time in the summarize command:

```
. sum AGE EDUC AGEKDBRN

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
         AGE |      2041    47.96717    17.67799        18         89
        EDUC |      2039    13.46101    3.149267         0         20
    AGEKDBRN |      1470    23.91633    6.022915        12         55
```

This makes generating tables of descriptive statistics super easy.

2.4 Proportions

Proportions can be a little tricky to work with, depending on how they are coded. To use Stata to analyze proportions, we need them coded as 0s and 1s. We can see how variables are coded in Stata using the `codebook` command:

```
. codebook SEX

-------------------------------------------------------------------------------
SEX                                                          RESPONDENTS SEX
-------------------------------------------------------------------------------

                  type:  numeric (int)
                 label:  SEX

                 range:  [1,2]                        units:  1
         unique values:  2                        missing .:  0/2044

            tabulation:  Freq.   Numeric  Label
                          891         1  MALE
                         1153         2  FEMALE
```

So according to the codebook command, the `SEX` variable is coded as 1 for male, 2 for female. If we wanted to work with proportions of males, we need to recode this variable as 1 for male, 0 for female. We can do that with the recode command:

```
. recode SEX (2=0), gen(male)
(1153 differences between SEX and male)

. codebook male

-------------------------------------------------------------------------------
male                                                 RECODE of SEX (RESPONDENTS SEX)
-------------------------------------------------------------------------------

                  type:  numeric (int)

                 range:  [0,1]                        units:  1
          unique values:  2                       missing .:  0/2044

             tabulation:  Freq.  Value
                          1153  0
                           891  1
```

In the recode command, we told Stata we wanted to recode SEX into a new variable called male (the `gen(male)` option), and that we wanted the new variable to transform all the 2s to 0s. The other values will remain the same. The `codebook` command for the new variable male shows that the number of 1s is the same as the SEX variable, and that the number of 0s is the same as the number of 2s in the SEX variable. You'll notice that the new variable does not have value labels (MALE, FEMALE) - we would need to do that manually, but its not necessary.

Now that we have our male variable, we can use the `summarize` command to calculate the proportion male:

```
. sum male

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
        male |      2044      .43591    .4959968         0          1
```

Recall that the mean of a variable coded 0/1 is the proportion of 1s. Here, 1 stands for male, so the mean of the variable is the proportion of males in the dataset, which is .436, or 43.6%. Note, though, that the standard deviation reported here assumes this is a mean - this is not the proportion standard deviation. Recall that the proportion standard deviation is:

$$s = \sqrt{\pi(1-\pi)}$$

So in this case, it would be

$$s = \sqrt{0.43591(1-0.43591)} = \sqrt{0.2559} = 0.495875$$

As you can see, this is very close to the standard deviation reported by the summarize command, but not exactly the same. The command for calculating confidence intervals for proportions will calculate the correct standard error for us.

3 Confidence Intervals

3.1 Means

The command for calculating confidence intervals in Stata is `ci`

```
. ci AGE

    Variable |        Obs        Mean    Std. Err.       [95% Conf. Interval]
-------------+--------------------------------------------------------------
         AGE |       2041    47.96717    .3913013        47.19978    48.73456
```

As you can see, the command calculates the mean and standard error, along with the confidence interval. By default, the command calculates the 95% confidence interval, but we can use the level() option to specify a different level:

```
. ci AGE, level(99)

    Variable |        Obs        Mean    Std. Err.       [99% Conf. Interval]
-------------+--------------------------------------------------------------
         AGE |       2041    47.96717    .3913013         46.9583    48.97604
```

We can be 95% confident that the true population mean age is between 47.19978 and 48.73456 years old. We can be 99% confident that the true population mean age is between 46.9583 and 48.97604 years old.

3.2 Proportions

To tell Stata that we are using proportions instead of means, we include the `binomial` option:

```
. ci male, b

                                                    -- Binomial Exact --
    Variable |        Obs        Mean    Std. Err.       [95% Conf. Interval]
-------------+--------------------------------------------------------------
        male |       2044       .43591    .0109681        .4142732    .4577308
```

This makes sure that Stata calculates the standard error correctly for proportions. We can be 95% confident that the true population proportion of males is between 0.4142732 and 0.4577308.

4 Hypothesis Testing

4.1 Means

Hypothesis testing is very easy in Stata using the `ttest` command. The command take the form:

```
ttest varname = #
```

```
. ttest AGE = 47

One-sample t test
------------------------------------------------------------------------------
Variable |     Obs        Mean    Std. Err.   Std. Dev.   [95% Conf. Interval]
---------+--------------------------------------------------------------------
     AGE |    2041    47.96717    .3913013    17.67799    47.19978    48.73456
------------------------------------------------------------------------------
    mean = mean(AGE)                                              t =   2.4717
Ho: mean = 47                                  degrees of freedom =     2040

    Ha: mean < 47               Ha: mean != 47                 Ha: mean > 47
 Pr(T < t) = 0.9932         Pr(|T| > |t|) = 0.0135          Pr(T > t) = 0.0068
```

We told Stata to test whether the mean age is equal to 47. So in the above, $H_0 = 47$. The command calculates the mean, standard deviation, standard error, a 95% confidence interval, and a t statistic. Above, our t statistic is 2.4717. Stata also calculates p-values for the three possible tests, a one tailed test for the mean less than the null, a two tailed test for the mean not equal to the null, and a one tailed test for the mean greater than the null. We can choose the appropriate p-value for the test we are doing. Here, we see that even the two tailed test is significant at an α of 0.05.

To perform a test of the difference between two means, we include the `by()` option, with a variable that defines the two groups:

```
ttest depvar, by(indvar)
```

Note that `indvar` must define two and only two groups.

```
. ttest EDUC, by(SEX)

Two-sample t test with equal variances
------------------------------------------------------------------------------
   Group |     Obs        Mean    Std. Err.   Std. Dev.   [95% Conf. Interval]
---------+--------------------------------------------------------------------
    MALE |     889    13.44544    .1092575    3.257632    13.23101    13.65988
  FEMALE |    1150    13.47304    .0903597    3.064247    13.29575    13.65033
---------+--------------------------------------------------------------------
combined |    2039    13.46101     .069743    3.149267    13.32424    13.59779
---------+--------------------------------------------------------------------
    diff |           -.0275992    .1406763               -.3034835    .2482852
------------------------------------------------------------------------------
    diff = mean(MALE) - mean(FEMALE)                              t =  -0.1962
Ho: diff = 0                     degrees of freedom =     2037

    Ha: diff < 0                 Ha: diff != 0                 Ha: diff > 0
 Pr(T < t) = 0.4222         Pr(|T| > |t|) = 0.8445          Pr(T > t) = 0.5778
```

Notice that Stata automatically calculates the mean for each group, the overall mean, and the mean difference. So here, we are testing whether males and females have different education levels. `EDUC` here is years of education. According to the p-value calculated for the two tailed test, we fail to reject the null that males and females have the same amount of education.

5 Introduction to Regression

Regression analysis is about exploring linear relationships between a dependent variable and one or more independent variables. Regression models can be represented by graphing a line on a cartesian plane. Think back on your high school geometry and algebra.

Let's work with a truncated dataset, taken from the GSS: `hire771`. Make sure your previous dataset was saved otherwise you'll get an error message. Begin by loading the dataset:

```
use hire771
```

Next we want to find out how age might be related to a respondent's salary. So we type in the regression command as follows:

```
. regress salary age

      Source |       SS       df       MS              Number of obs =    3131
-------------+------------------------------           F(  1,  3129) =  298.30
       Model |  1305182.04     1  1305182.04           Prob > F      =  0.0000
    Residual |  13690681.7  3129  4375.41762           R-squared     =  0.0870
-------------+------------------------------           Adj R-squared =  0.0867
       Total |  14995863.8  3130  4791.01079           Root MSE      =  66.147

------------------------------------------------------------------------------
      salary |      Coef.   Std. Err.      t    P>|t|     [95% Conf. Interval]
-------------+----------------------------------------------------------------
         age |   2.335512   .1352248    17.27   0.000     2.070374    2.600651
       _cons |   93.82819   3.832623    24.48   0.000     86.31348    101.3429
------------------------------------------------------------------------------
```

The first table shows the various sum of squares, degrees of freedom, and such used to calculate the other statistics. (Think back to the sample ANOVA output we talked about in class). In the top table on the right lists some summary statistics of the model including number of observations, R^2 (which reflects model fit) and such. However, the table we will focus most of our attention on is the bottom table. Here we find the coefficients for the variables in the model, as well as standard errors, p-values, and confidence intervals.

From the bottom table, we see that the coefficient on age is 2.34 and the constant is 93.8 giving us an equation of:

$$salary = 93.8 + 2.34age$$

How do we interpret this? For every year older someone is, they are expected to receive another $2.34 a week. A person with age zero is expected to make $93.8 a week. We can find the salary of someone given their age by just plugging in the numbers into the above equation. So a 25 year old is expected to make:

$$salary = 93.8 + 2.34(25) = 152.3$$

Looking back at the results tables, we find more interesting things. We have standard errors for the coefficient and constant because the data are messy, they do not fall exactly on the line, generating some error. If we look at the R^2 term, 0.087, we find that this line is not a very good fit for the data.

5.1 Regression with more than one IV

There are likely other factors that influence salary aside from age. For example, research suggests that the more education an individual has, the higher their salary. To explore this more, we can add another variable into the STATA regression code:

```
. regress salary age educ
```

Source	SS	df	MS		
Model	3745633.26	2	1872816.63	Number of obs =	3131
Residual	11250230.5	3128	3596.62101	F(2, 3128) =	520.72
				Prob > F =	0.0000
				R-squared =	0.2498
				Adj R-squared =	0.2493
Total	14995863.8	3130	4791.01079	Root MSE =	59.972

salary	Coef.	Std. Err.	t	P>|t|	[95% Conf.	Interval]
age	2.129136	.1228567	17.33	0.000	1.888248	2.370024
educ	13.02054	.4998516	26.05	0.000	12.04046	14.00061
_cons	61.53663	3.689336	16.68	0.000	54.30286	68.77039

Like above, we see that the model is significant as well as the independent variables. From these results, we could construct the equation:

$$salary = 61.5 + 2.13age + 13.0educ$$

and plug in values for each of the independent variables to obtain a projected outcome. One piece that's importantly different between the two models is the R^2 term. In this second model, we see a much higher R^2, indicating that this model is better at representing our data than the original. This should make intuitive sense: if the education variable is significant in predicting salary, a model that includes this alongside age will be more complete than a model that includes age only. In performing a full analysis of data, you would continue adding or taking out variables based on the literature, and hopefully end up with a relatively high R^2.

6 Closing out

Before closing, you'll want to make sure to save any data that you may have changed during your session using the "save" command. You'll also want to save your do files, by using the dropdown menu. Finally, closing out your log allows you to access output from the session once you have exited. This is done with the command

```
log close
```