EViews 10 Tutorial

by Manfred W. Keil

to Accompany

Introduction to Econometrics

by James H. Stock and Mark W. Watson

---

1. *EViews*: INTRODUCTION

This tutorial will introduce you to a statistical and econometric software package called *EViews*. The most current professional version is *EViews* 10 and all output in this tutorial was created using *EViews* 10. However there is an *EViews Student Version Lite* that is free for university students, with a license that expires after one year. Then there is a *EViews University Edition* for $49.95 with a 6-months license. Both *EViews* 10 and 9 are sufficiently similar that those who have access to *EViews* 9 can comfortably use this tutorial. The difference is only apparent in more advanced techniques that you, as a first time user, will not encounter in a course of econometrics (or at least not in the beginning of the course).

EViews runs on the Windows (Vista, or Windows 7, 8, 10), and on a Mac platform (OS X. 10, OS X. 12 or OS X. 13). It is produced by IHS Inc in Irvine, California. You can read about various product information at the firm's Web site, www.eviews.com . EViews 10 no longer comes with a hard copy of the four manuals, a *User's Guide* (2 books), a *Command Reference*, and an *Object Reference* book. The manuals can be accessed within the program through the **Help** function. You can order *EViews* by calling (949) 856-3368 or writing to sales@eviews.com. The *User's Guide* is better for first-time users.

The difference between the free student version and the full version is in the limitation on the size of data sets ("capacity limitation" is 1,500 observations for each series and no more than 15,000 observations for all series; students can work with larger data sets but will then not be able to save/export the workfile) and the availability of some features such as advanced seasonal adjustment methods (X11, X12, X13). Furthermore, and perhaps most importantly for you right now, the **student version does not allow you to run *EViews* in "batch mode" using Program-Files**. Instead you can only use the interactive use. This tutorial will explain the difference between interactive use and batch mode below. Once you have gone through the first series of commands in interactive mode, you will almost certainly want to run programs in batch mode.

Econometrics deals with three types of data: cross-sectional data, time series data, and panel (longitudinal) data (see Chapter 1 of the Stock and Watson (2018) textbook). In a *time series* you observe the behavior of a single entity over multiple time periods. This can range from high frequency data such as financial data (hours, days); to data observed at somewhat lower (monthly) frequencies, such as industrial production, inflation, and unemployment rates; to quarterly data (GDP) or annual (historical) data. In a *cross-section* you analyze data from multiple entities at a single point in time. One big difference between time series and cross-sectional analysis is that the order of the observation numbers does not matter in cross-sections. With time series, you would lose some of the most interesting features of the data if you shuffled the observations. Finally, *panel data* can be viewed as a combination of time series and cross-sectional data, since multiple entities are observed at multiple time periods. *EViews* allows you to work with all three types of data.

EViews is most commonly used for time series analysis in academics, business, and government, but you can work with it easily when you have cross-sections and/or panel data. *EViews* allows you to save results within a program and to "retrieve" these results for further calculations later.

Remember how you calculated confidence intervals in statistics say for a population mean? Basically you needed the sample mean, the standard deviation, and some value from a statistical table. In *EViews* you can calculate the mean and standard deviation of a sample and then temporarily "store" these. You then work with these numbers in a standard formula for confidence intervals. In addition, *EViews* provides the required numbers from the relevant distribution (normal, χ^2, F, etc.).

While *EViews* is truly interactive, you can also run a program as a "batch" job, i.e., you write a sequence of commands and then execute the program in one go. In the good old days the equivalent was to submit a "batch" of cards, each containing a single command, to a technician, who would use a card reader to enter these into the computer, and the computer would then execute the sequence of statements. (You stored this batch of cards typically in a filing cabinet, and the deck was referred to as a "file.") While you will work at first in interactive mode by clicking on buttons, you will very soon discover the advantage of running your regressions in batch mode. This method allows you to see the history of commands, and you can also analyze where exactly things went wrong if there are problems with any of your commands. This tutorial will initially explain the interactive use of *EViews*, since it is more intuitive. However, we will switch as soon as it makes sense into the batch mode.[1]

While *EViews* produces graphs and charts, these can often be improved upon by saving the data used in these graphs in a spreadsheet or ASCII format, and then to import the data into Excel (or another spreadsheet program you prefer). Even better, since *EViews* works in a Windows format, it allows you to cut and paste the data into any other Windows-based program.

Finally, there is a warning about the limitations of this tutorial. The purpose is to help you gain an initial understanding of how to work with *EViews*. I hope that the tutorial looks less daunting than the manuals. However, it cannot replace the accompanying manuals, which you will have to consult for more detailed questions (alternatively use "Help" in the program). Feel free to provide me with feedback of how we can improve the tutorial for future generations of students (mkeil@cmc.edu). Colleagues of mine and I have decided to set up a "Wiki" run by students but supervised by faculty at my academic institution. We have found that the "wisdom of crowds" often produces very valuable information for those who follow. This is, of course, just a suggestion. Finally you may want to think about working with statistical software as learning a new language: practicing it routinely will result in improvement. If you set it aside for too long, you will only remember the most important lines but will forget the important details. Another danger of tutorials like this is that you simply follow the instructions and when you are done, you don't remember the commands. It is therefore a good idea to keep a separate sheet and to write down commands and examples of them if you think you will use them later on. I will give you short exercises so that you can practice the commands on your own.

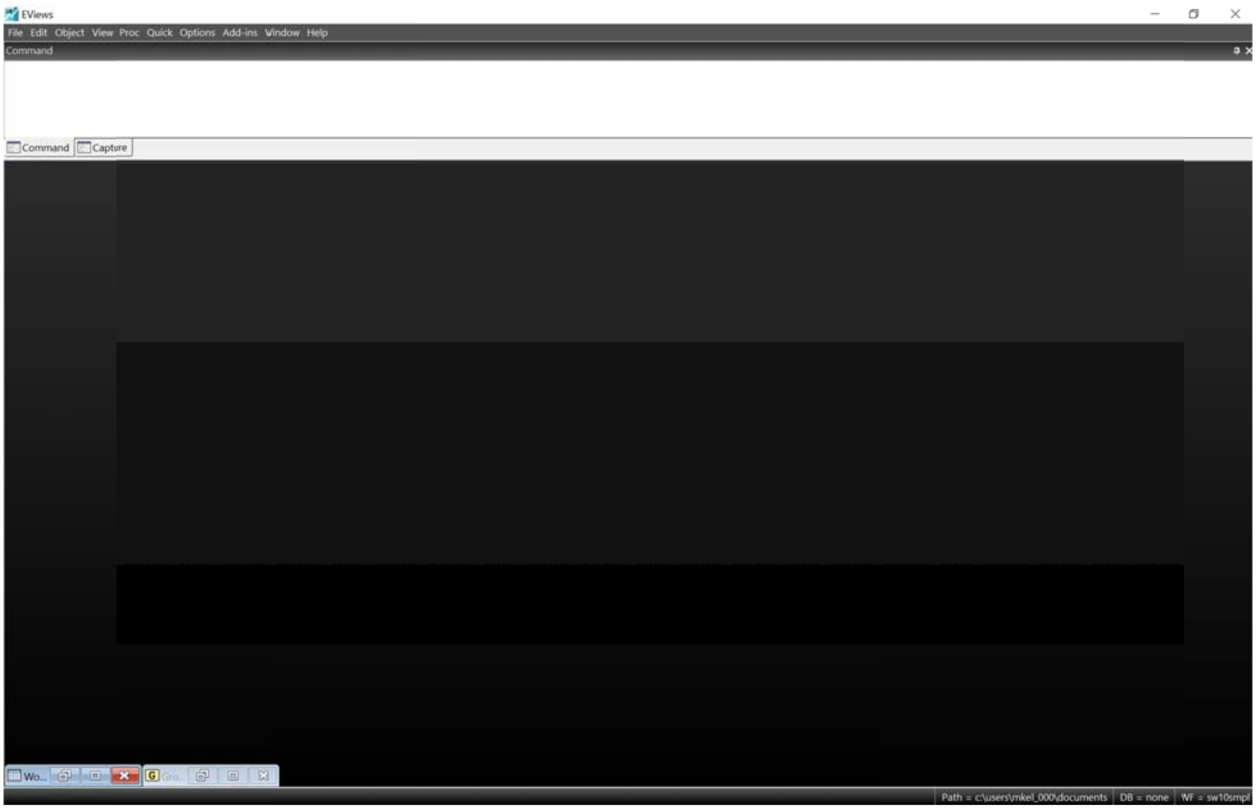
---

1 As mentioned above, the very reasonably priced student version does not run batch files. However, even if you purchased the student version, the academic version may be available to you at your college/university, or you may decide to upgrade on your own.

2. CROSS-SECTIONAL DATA

Interactive Use: Data Input and Simple Data Analysis

Let's get started. Click on the *EViews* icon to begin your session. What you see next is the *EViews* window, with the *title bar* at the top, the *command window* immediately below and the *status line* at the very bottom (ignore my path, etc., below).



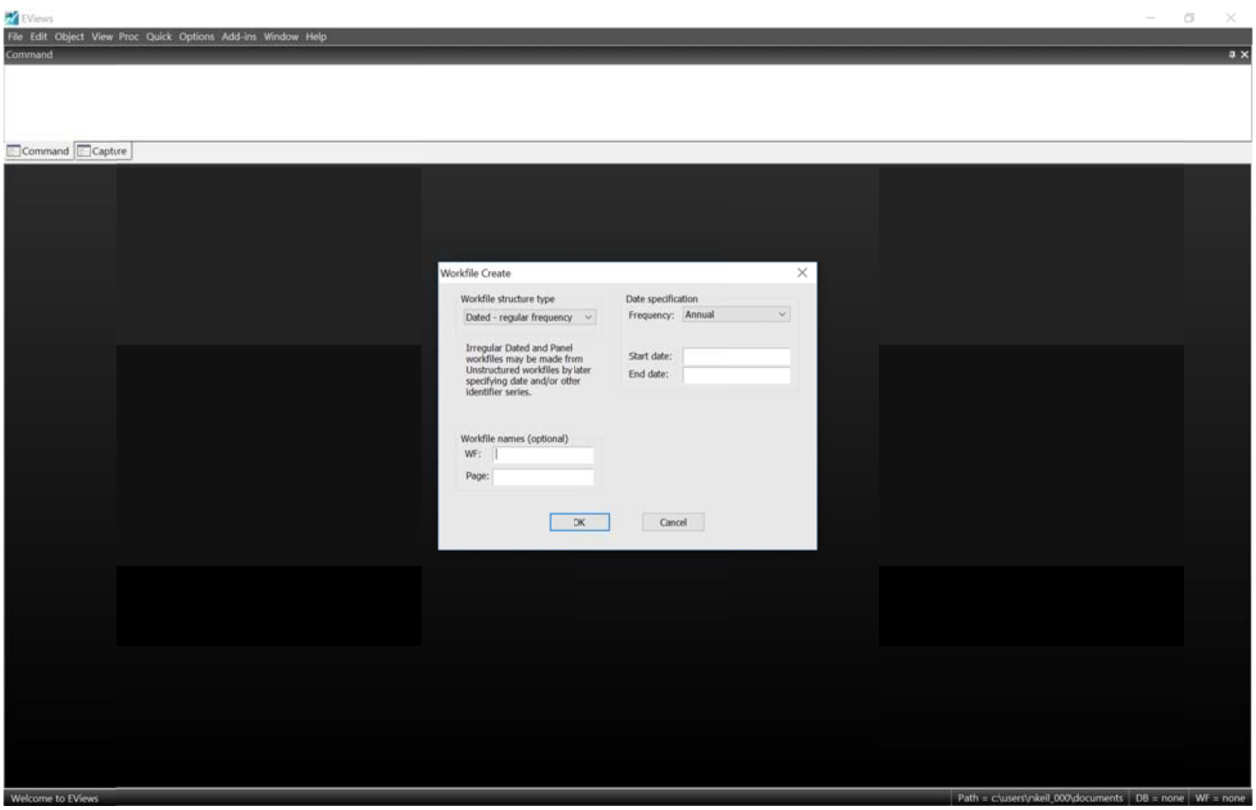
The results of your various operations will be displayed between the command window and the status line in the so-called *work area*. In interactive use, *EViews* allows you to execute commands either by clicking on command buttons or by typing the equivalent command into the *command window*. To view past commands, click on the "capture" button, located to the left of the command button.

In this tutorial, we will work with two data applications, two cross-sectional (California Test Score Data Set used in chapters 4-9; Current Population Survey Data Set used in Chapters 3 and 8), and one time series (U.S. Macro Quarterly Data Set used in Chapter 15).

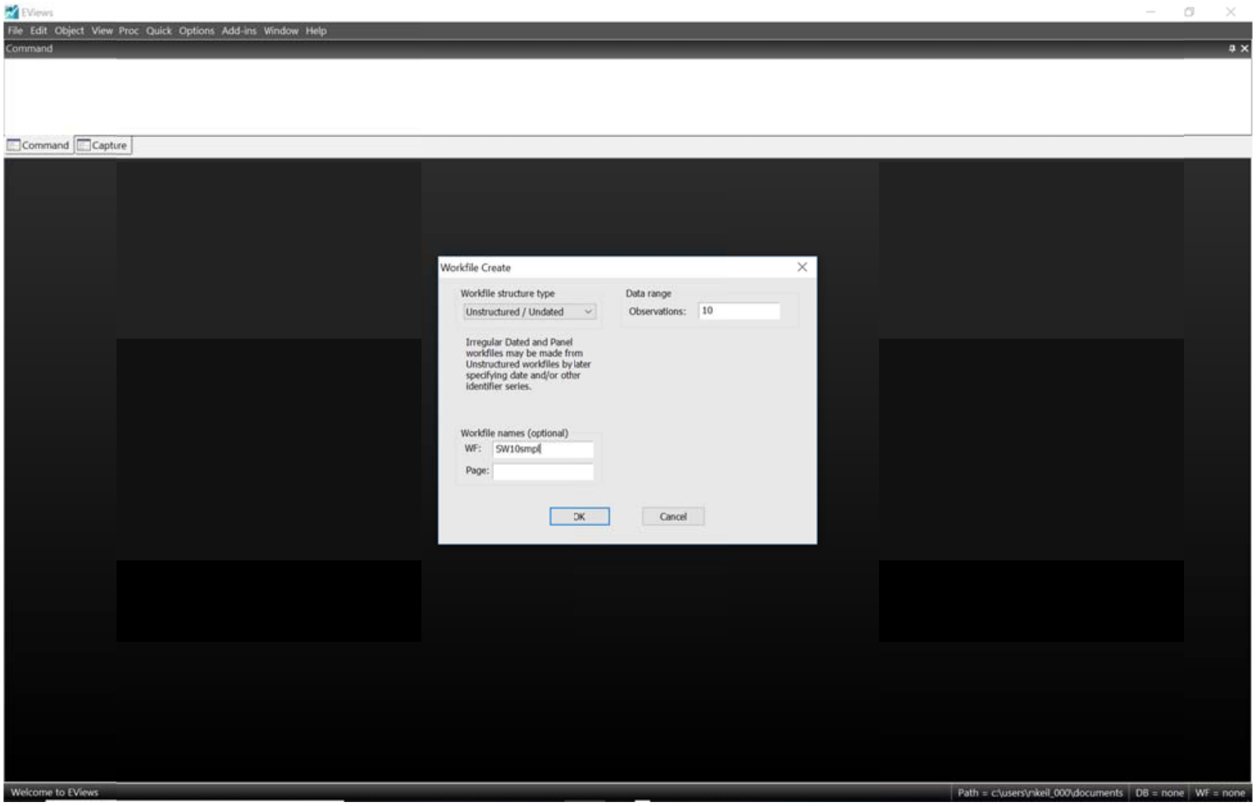
a) The Easy but Tedious Way: Manual Data Entry

In Chapters 4 to 9 you will work with the California Test Score Data Set. These are cross-sectional data, referred to in *EViews* as "*undated or irregular*" data. There are 420 observations from K-6 and K-8 school districts for the years 1998 and 1999. You will not want to enter a large amount of data manually, unless you have collected data by yourself (something that economists are doing more and more). The alternative is to enter the data into a spreadsheet (Excel) and then to cut and paste the data (see below). However, for the purpose of this introduction it will be useful that you become aware of entering, and editing, data. Here I will use a sub-sample of 10 observations from the California Test Score Data Set.

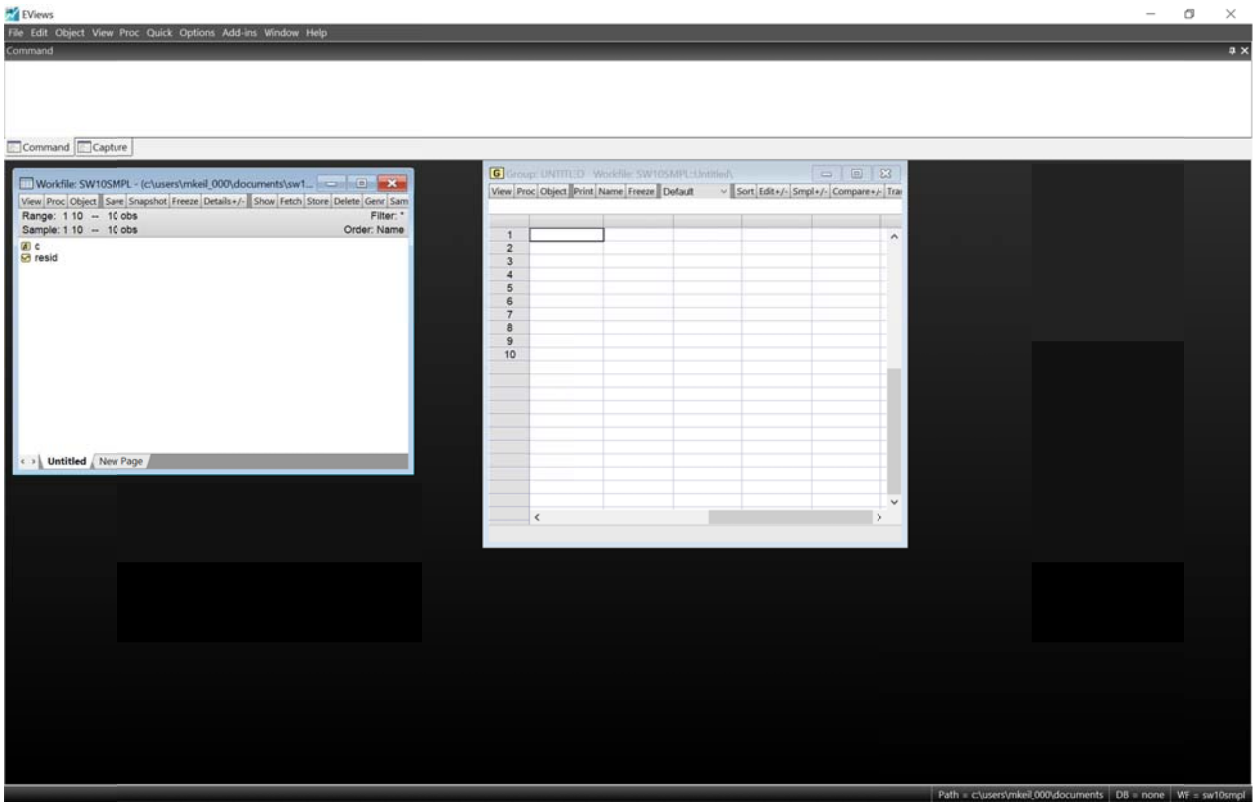
To start, we must create a *workfile* in *EViews*. Click on the **File** pull-down menu, and then on **New** and **Workfile**. As is common in Windows programs, you will see a *dialog box*.



This particular dialog box asks you for the start and end dates of your data set, and for the type of data you are entering. We are working with undated or irregular data (cross sectional data), so use the pull-down menu for **Workfile Structure Type** and select **Unstructured/Undated**. Then enter 10 in the **Observations** box. While you are at it, enter "SW10smpl" into the "WF" field.



You will see a workfile window, which contains two entries (*c*, *resid*). Do not worry about these for the moment. To enter the data into a format similar to the spreadsheets you have become familiar with, click on **Quick** in the title bar, and then on **Empty Group (Edit Series)**.



Next enter the data for two variables (two columns). Here are the 10 observations to enter. (*EViews* will add zeros. You will see later how to get rid of these.)

obs	**TESTSCR**	**STR**
1	**606.8**	**19.5**
2	**631.1**	**20.1**
3	**631.4**	**21.5**
4	**631.8**	**20.1**
5	**631.9**	**20.4**
6	**632.0**	**22.4**
7	**632.0**	**22.9**
8	**638.5**	**19.1**
9	**638.7**	**20.2**
10	**639.3**	**19.7**

Once you have entered the data, close the object (click on "X") (you will be asked "Delete Untitled GROUP ?" Click on "Yes.") You will be able to (re)name the variables. Click on **SER01**, then rightclick and chose "Rename…" and enter "testscr". Do the same to change **SER02** to "str".

Entering data in this way is very tedious, and you will make data input errors frequently. You will see below how to enter data directly from a spreadsheet or an ASCII file, which are the most common forms of data you will receive in the future. Also, you noticed when you entered the test score (*testscr*) first and then the student-teacher ratio (*str*) that you were automatically moved into the test score column after entering each student-teacher data point. This is an unfortunate feature, but there is no alternative unless you enter all the data by observation.

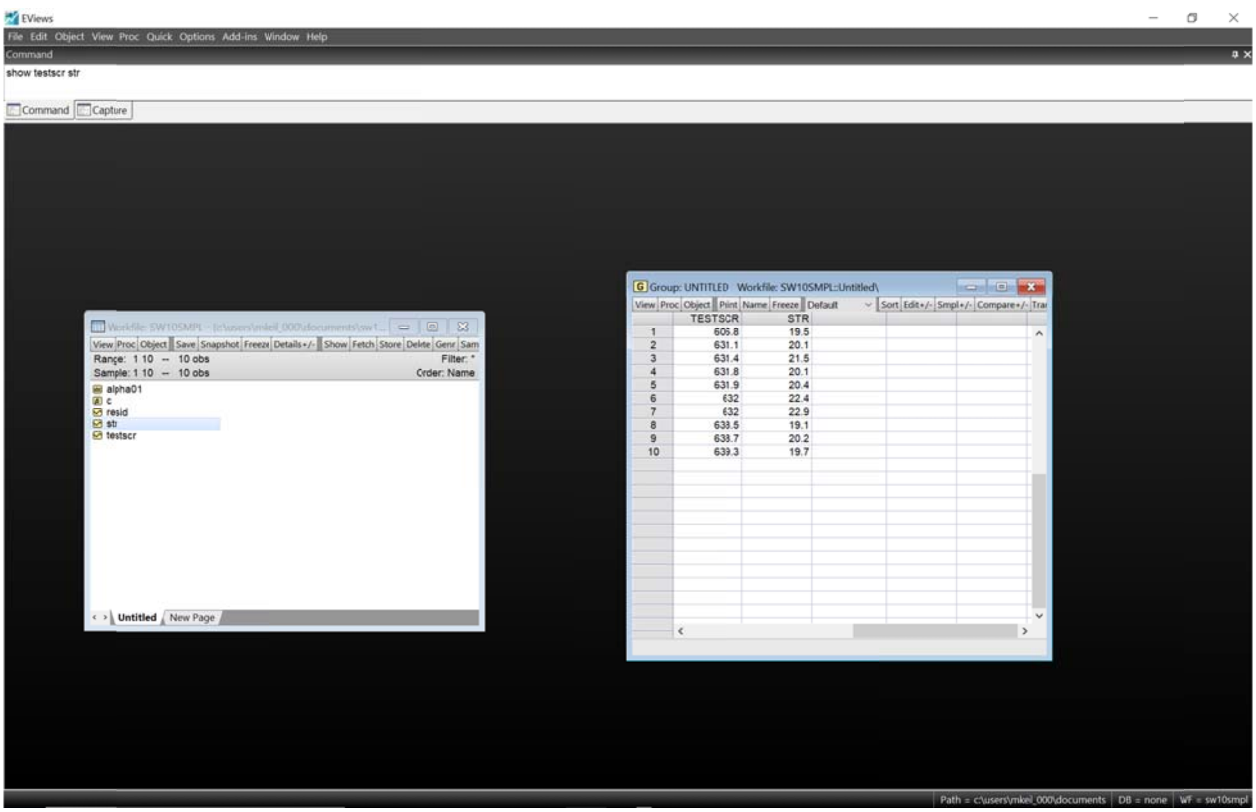
In general, you can look at variables in your workfile by typing in the command

Show varname1, varname2, …

where *varnamei* refers to a variable that exists in your workfile. Try it here by typing

show testscr str

You should see the following:



b) Summary Statistics

For the moment, let’s just see if we are working with the same data set. Locate the **View** button at the upper-left corner of the Group window, click on it, and then click on **Descriptive Stats** and **Common sample**. You should see the following output (instead of using **Prnt Scrn** on my computer, I pressed the **Freeze** button in *EViews*. This allows me to copy and paste output into another Windows based program, a feature that will come in handy down the road when you want to display some of your output):

Date: 11/01/18
Time: 13:35
Sample: 1 10

	TESTSCR	STR
Mean	631.3500	20.59000
Median	631.9500	20.15000
Maximum	639.3000	22.90000
Minimum	606.8000	19.10000
Std. Dev.	9.264418	1.260908
Skewness	-1.992947	0.782889
Kurtosis	6.247292	2.295517
Jarque-Bera	11.01344	1.228314
Probability	0.004059	0.541097
Sum	6313.500	205.9000
Sum Sq. Dev.	772.4650	14.30900
Observations	10	10

The summary statistics are explained in Chapter 2 of your textbook (for example, *Kurtosis* is defined in equation (2.15) on page 22 in Stock and Watson (2018).

If your summary statistics differ, then check the data again. (To return to the data observations, either click on **View** and then choose **Spread Sheet**, or simply click on the **Sheet** button). Once you have located the data problem, click the **Edit+/-** button on the workfile toolbar, move to the observation in question, enter the correct value, and press Enter. You may want to explore some of the other toolbar buttons to see their functions. **CellFmt**, for example, allows you to get rid of unnecessary digits after the decimal point, but appears only after you "**Freeze**" the object and click **Edit** +/-.

Once you have entered the data, there are various things you can do with it. You may want to keep a hard copy of what you just entered. If so, click on the **Print** button.

In general, it is a good idea to save the data and your work frequently in some form. Many of us have learned through painful experiences how easy it is to lose hours of work by not backing up data/results in some fashion. There are two ways to save data in EViews. One is to save an entire workfile (**Save**), and the other is to store individual series (**Store**).

Press the **Save** button in the workfile toolbar (the window displaying the variable names), or click on **File** and then **SaveAs** in the *main menu*. Follow the usual Windows format for saving files (drives, directories, file type, etc.). If you save workfiles in *EViews* readable format, then you should use the extension ".WF1." Once you have saved a workfile, you can call it up the next time you intend to use it by clicking on **File** and then **Open**. Try these operations by saving the current workfile under the name "SW10smpl.wf1."

Alternatively, you may want to just save a few series of the current workfile. The reason is that sometimes you use some of these original series, or transformations of these series, in a different workfile. Let's save the test score and student-teacher series. First mark the two series in the workfile by clicking on *testscr*, then hold down the control or shift button and click on *str*. (Make sure that you are doing this in the Workfile window, not in the Group View window.) After that, press the **Store** button in the workfile toolbar. Once again, a dialog box will pop up. Store the two data series in the *EViews* subdirectory with the extension ".db." Next time you need to retrieve these two series, you can simply click on the **Fetch** button in the workfile toolbar.

c) Graphical Presentations

Most often it is a good idea to generate graphs (figures or "pictures") to get some "feel" for the data. You will be able to detect outliers which may be the result of data entry errors or you will be able to see if the data "makes sense." Although *EViews* offers many graphing options, there are two that you will use most often: line graphs, where one or more variables are plotted across entities (these will become more important in time series analysis when you are plotting over time), and scatterplots (crossplots), where one variable is graphed against another.

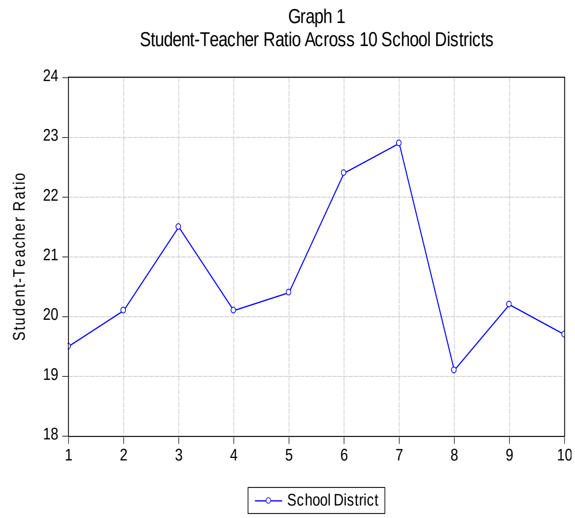
First set the sample to 1-10 either by clicking on the **Sample** button in the workfile toolbar and typing in "1 10" under **Sample range pairs** or by entering "*smpl 1 10*" in the command window. (The command window is the white box located directly under the main menu and is where you will type all commands.[2]) Then type, in the next line, the command "*freeze(graph_str) str.line*" in the command line.[3] This will create the line graph and give it a name (*graph_str* here, but other names, such as graph_1 or mygraph, can be chosen instead). Think of freezing an object as taking a photograph of it and giving it a name. This allows you to locate it easily in your photo album later. You can also edit the photograph later. Most importantly, you can cut and paste it into your word-processing program. "*graph_str*" now appears in the *workfile* window. Double click on it to see the graph you just created. In the future and when in interactive use, you will most often work in the *command window* rather than clicking on buttons.

After the graph appears, either double click on the graph or click on the **Options** button, and alter it until it looks like the one below. Some of the alterations can be made in the resulting dialog box; others, such as text inserted, title of the graph, etc., have to be edited in.

---

[2] Make sure you press enter after each command line for the program to register it. If you make a typo, you can go back and correct your mistake, but be sure to press enter again for the computer to read the command. When typing commands, be careful where you put spaces, because the computer is sensitive to these and "freeze(graph)" is not the same as "freeze (graph)". Oftentimes, the underscore "_" is used in a saved variable name in place of an actual space.

[3] Alternatively the same graph can be generated by marking the variable *str* first and then double clicking on it. In the resulting *Series* window, click on **View** /**Graph/Line&Symbol**. You can then freeze the graph by clicking on the **Freeze** button.



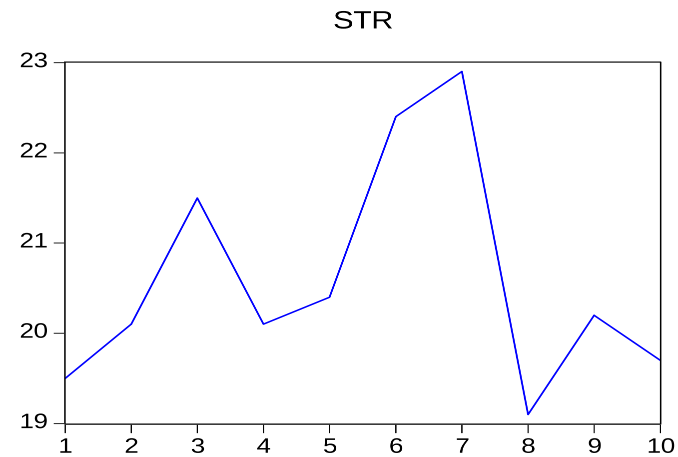
Typically we are interested either in causal relationships between variables or in the ability of one variable to predict (later, in time series, forecast) another, it is a good idea to plot two variables together. Commands, such as *line*, can often be modified by an option in parentheses. In this case, "*m*" means "display multiple graphs." Use the line command to generate the graph below.[4] This will require you first to define or create a "*group*" by giving it a name (here *size_perform* but others, such as *mygroup* are possible). Next you tell the program which series form the group, here *str* and *testscr*. Then "**freeze**" the graph as before.

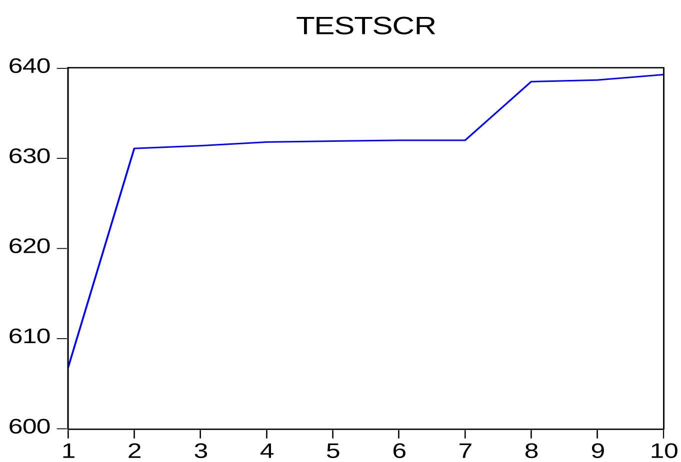
The line commands are

group size_perform str testscr
freeze(two_series_plot) size_perform.line(m)

You should see the following two graphs in your EViews display (I used copy/paste here).

[4] Pushing buttons is relegated to footnotes from here on. You should work with commands now. If you have to, mark *testscr* and *str*, opening the two variables as a group, then select **View/Multiple Graphs/Line**).



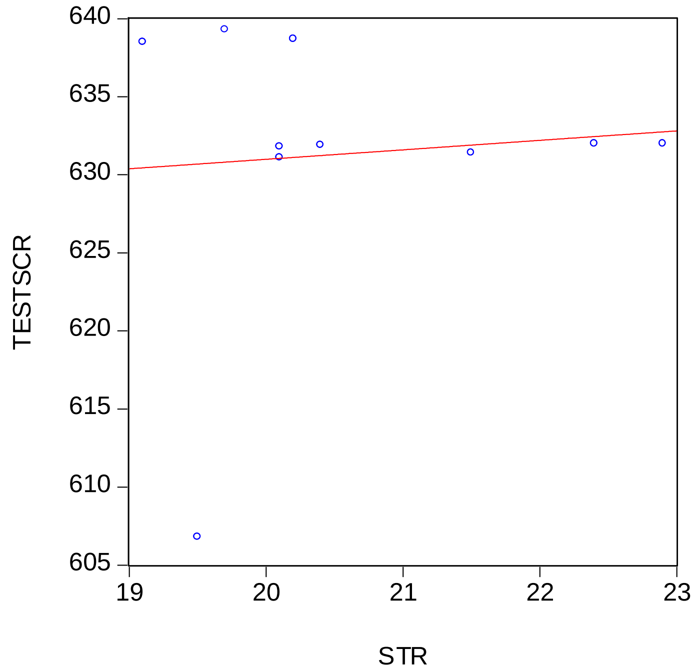


To get an even better idea about the relationship, you can display a two-dimensional relationship in a scatterplot (see p. 106 of your Stock and Watson (2018) textbook). The command is

size_perform.scat linefit

where *size_perform* refers to the name of a previously created group.[5]

[5] Alternatively select **View/Graph/Scatter/Details: Fit Lines: Regression Line**. Choose *None* in the resulting *Global Fit Options* Box.



(Not to worry about the positive slope here. Remember, this is a sample, and a very small one at that. After all, you may get 10 heads in 10 flips of a coin.)

d) Simple Regression

There is a commonly held belief among many parents that lower student-teacher ratios will result in better student performance. Consequently, in California, for example, all K-3 classes now have a maximum student-teacher ratio of 20 ("Class Size Reduction Act" – CSR).

For the 10 school districts in our sample, we seem to have found a *positive* relationship between larger classes and poor student performance. This disturbing result (increase in class size results in higher test scores) will disappear once we work with all 420 observations from the California School Data Set, and we will then find the negative relationship you have seen in the textbook – for now, we are more concerned about learning techniques in *EViews*.

In the previous section, we included a regression line in the scatterplot, something that you should have encountered towards the end of your statistics course. However, the graph of the regression line does not allow you to make exact quantitative statements about the relationship; you want to know the exact values of the slope and the intercept. For example, in general applications, you may want to predict the effect of increase by one in the explanatory variable (here the student-teacher ratio) on the dependent variable (here the test scores).

To answer the questions relating to the more precise nature of the relationship between large classes and poor student performance, you need to estimate the regression intercept and slope. A regression line is little else than fitting a line through the observations in the scatterplot according to some principle. You could, for example, draw a line from the test score for the lowest student-teacher ratio to the test score for the highest student-teacher ratio, ignoring all the observations in between. Or you could sort the data by student-teacher ratio and split the sample in half so that the observations with the lowest ten student-teacher ratios are in one set, and the observations with the highest ten student-teacher ratios are in the other set. For each of the two sets you could calculate the average student-teacher ratio and the corresponding average test score, and then connect the two resulting points. Or you could just eyeball the relationship. Some of these principles have better properties than others to infer the true underlying (population) relationship from the given sample. The principle of ordinary least squares (OLS), for example, will give you desirable properties under certain restrictive assumptions that are discussed in Chapter 4 of the Stock/Watson textbook.

Back to computing. If the dependent variable, *Y*, is only determined by a single explanatory variable *X* in a linear fashion of the type

$$Y_i = \beta_0 + \beta_1 X_i + u_i \qquad i=1,2,\ldots,N$$

with "*u*" representing the error, or random disturbance, not accounted for by the linear equation, then the task is to find some value for β_0 and β_1. If you had values for these coefficients, then β_1 describes the effect of a unit increase in *X* on *Y*. Often a regression line is a linear approximation to an underlying relationship and the intercept β_0 only has a useful meaning if observations around *X*=0 occur in the data. As we have seen in the scatterplot above, there are no observations around the student-teacher ratio of zero, and it is therefore better not to interpret the numerical value of the intercept at all. Your professor most likely will give you a serious penalty in the exam for interpreting the intercept here because with no students present, there is no score to record. (What would be the function of the teacher in that case?)

There are various ways to estimate the regression line. The command for regressing a variable *Y* on a constant (intercept) and another variable *X* is:

ls Y c X

where "*ls*" stands for least squares. Here, working with the *command* window,[6] type

ls(h) testscr c str

[6] If you are working in a Group Window, possibly by having invoked the **Show** option, then click on **Proc**. Next press **Make Equation,** and a dialog box will open. If *EViews* has not suggested a regression of the test score on the student-teacher ratio plus a constant ("c"; this letter is reserved in *EViews* for the constant – actually a vector of ones – and you are not allowed to give another variable this name), then type in the variable names in that order (*EViews* takes the first variable as the dependent variable; it does not matter if you place the constant before the explanatory variable or after). Alternatively, start in the Main menu and click on **Object** and the **New Object** and finally **Equation**. The same dialog box will open.

where the "*h*" in parentheses indicates that you are using heteroskedasticity-robust standard errors ("*c*" stands for the intercept).

The output appears as follows:

Dependent Variable: TESTSCR
Method: Least Squares
Date: 11/01/18 Time: 16:56
Sample: 1 10
Included observations: 10
Huber-White-Hinkley (HC1) heteroskedasticity consistent standard errors
and covariance

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	618.8527	51.06075	12.11993	0.0000
STR	0.606961	2.333492	0.260108	0.8013

R-squared	0.006824	Mean dependent var	631.3500
Adjusted R-squared	-0.117323	S.D. dependent var	9.264418
S.E. of regression	9.792813	Akaike info criterion	7.578031
Sum squared resid	767.1935	Schwarz criterion	7.638548
Log likelihood	-35.89016	Hannan-Quinn criter.	7.511644
F-statistic	0.054969	Durbin-Watson stat	0.853391
Prob(F-statistic)	0.820522	Wald F-statistic	0.067656
Prob(Wald F-statistic)	0.801349		

According to these results, lowering the student-teacher ratio by one student per class results in an decrease of 0.6 points, on average, in the districtwide test score. Using the notation of your textbook, you should display the results as follows:

$$\widehat{TestScore} = \underset{(51.1)}{618.9} + \underset{(2.33)}{0.61} \times STR,\ R^2 = 0.007,\ SER = 9.8$$

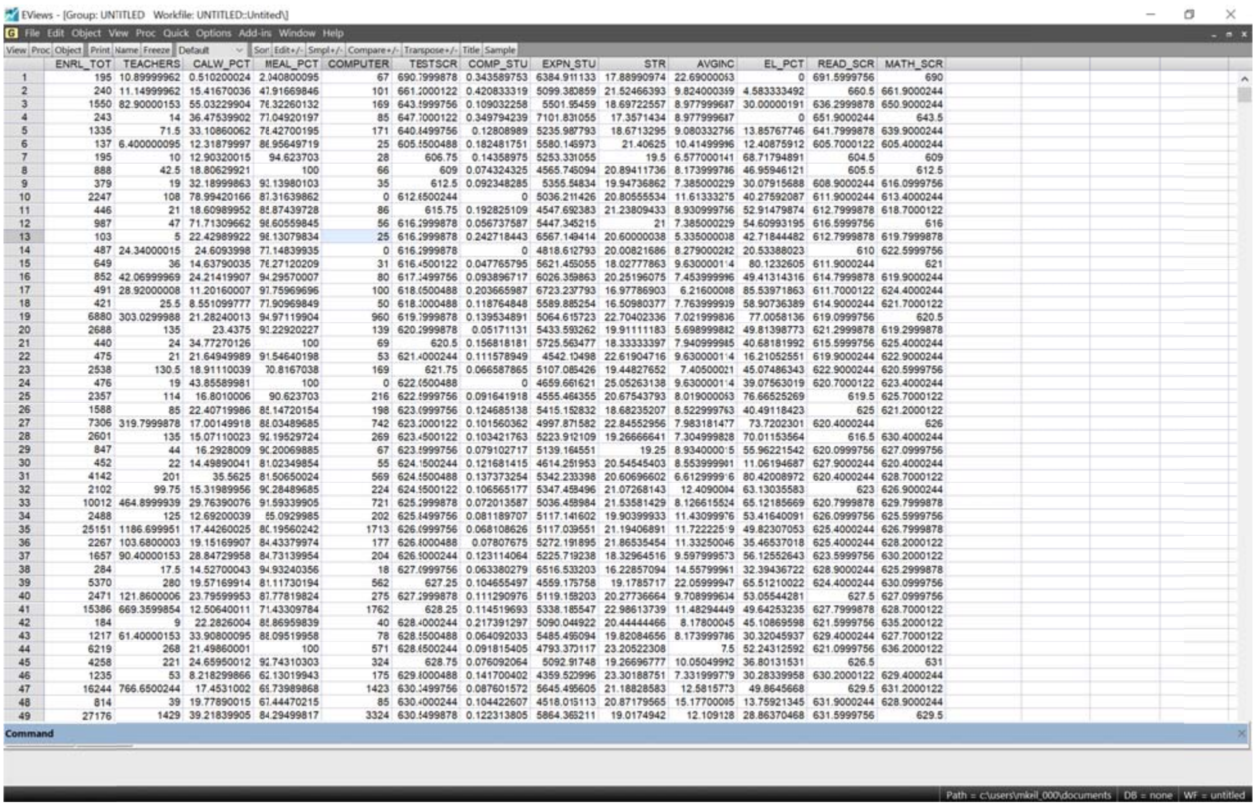
Note that the result for the 10 chosen school districts is quite different from the sample of all 420 school districts. However, this is a rather small sample and the regression R^2 is quite low. As a matter of fact, in Chapter 5 of your textbook, you will learn that the above slope is not statistically significant.

e) Entering Data from a Spreadsheet

So far you entered data manually. Most often you will work with larger data sets that are *external* to the EViews program, i.e., they will not be included in, or be part of, the program itself. This makes sense as data sets either become very large or are generated by another program, such as a spreadsheet.

Stock and Watson present the California test score data set in Chapter 4 of the textbook. Locate the corresponding Excel file *caschool.xls* and open it. Next, following the procedures discussed previously, open a new EViews workfile with 420 observations, and use the **Quick/Edit Group (Empty Series)** procedure. Return to the Excel file and mark F2:R421. Next, using the "*copy*" and "*paste*" commands common to Windows programs, move the data block to EViews. You presumably are familiar with this procedure. Make sure to select the grey box to the immediate right of "obs" before pasting (this will highlight that column). Next "rename" the 13 variables according to the name in the cells F1:R1.

This is what you should see in EViews:



When you are done, you are ready to save the workfile. Name it *caschool.wf1*.

You can now reproduce Equation (4.9) from the textbook. Use the regression command you previously learned to generate the following output ("**Freeze**" the output and use the **CellFmt** button to adjust the number of digits after the decimal point).

Dependent Variable: TESTSCR
Method: Least Squares
Date: 11/09/18 Time: 16:21
Sample: 1 420
Included observations: 420

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	698.93	9.47	73.82	0.0000
STR	-2.28	0.48	-4.75	0.0000

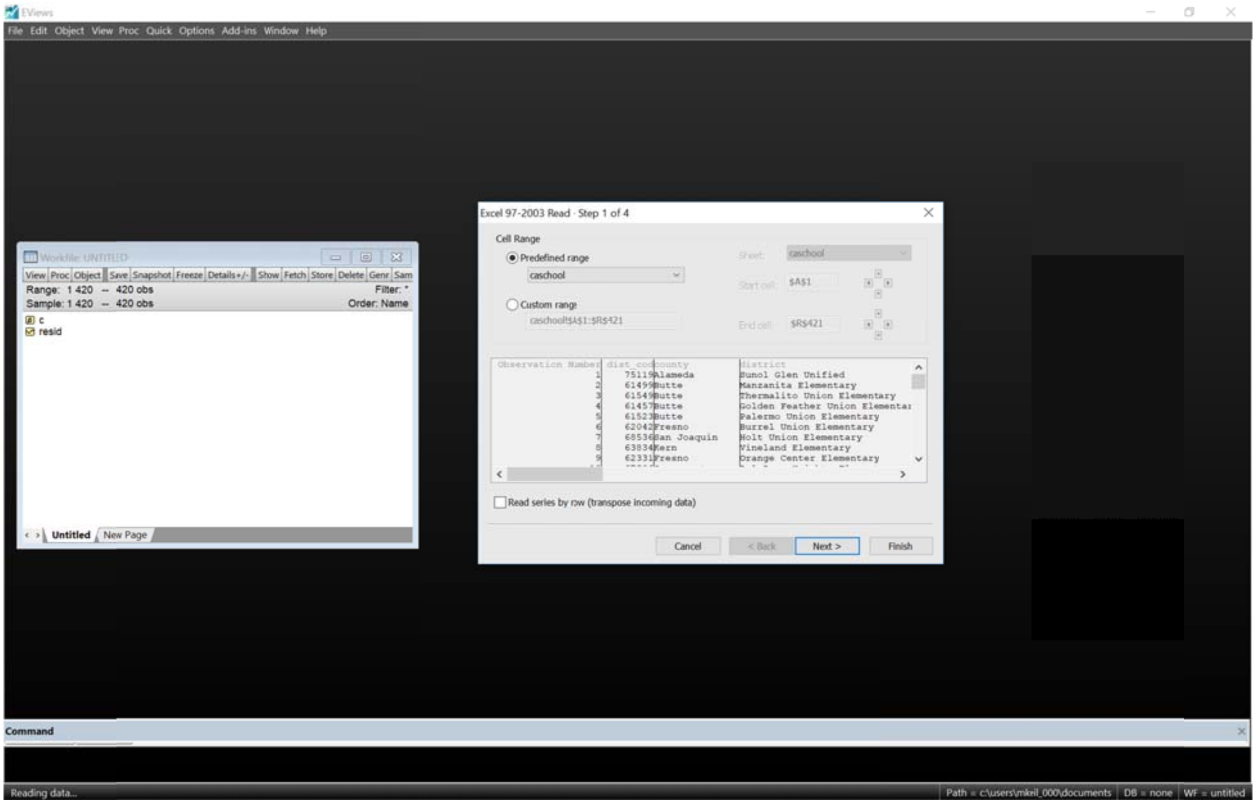
R-squared	0.05	Mean dependent var	654.2
Adjusted R-squared	0.05	S.D. dependent var	19.1
S.E. of regression	18.58	Akaike info criterion	8.69
Sum squared resid	144315.48	Schwarz criterion	8.71
Log likelihood	-1822.25	Hannan-Quinn criter.	8.69
F-statistic	22.58	Durbin-Watson stat	0.129
Prob(F-statistic)	0.000003		

(You can find the (homoscedasticity-only) standard errors on p. 165 of the Stock and Watson (2018) textbook. The regression R^2, sum of squared residuals (*SSR*), and standard error of the regression (*SER*) are presented in Section 4.3.)

*f) Importing Data Files directly into EView*s

Even though the *cut and paste* method seemed straightforward enough, there is a second, more direct way to import data into *EViews* from Excel, which does not involve copying and pasting data points.

Start again with a new workfile in EViews. Next press **Proc** /**Import /Import from File**. A dialog box will open, and you will first have to specify the location where your data file (caschool.xls) resides. After you double click on the file, another dialog box opens.



After you click through the "Next >" options, and finally "Finish," all the data including the variable names have been read in. You are good to go with the analysis. Note that *EViews* also allows you to import other types of data files, e.g. STATA files, although this may be a bit more complicated.

EViews will show that the data exist in the Workfile Window. You may want to check that the data were properly retrieved by typing the command "*show testscr str*" or running the same regression as before.

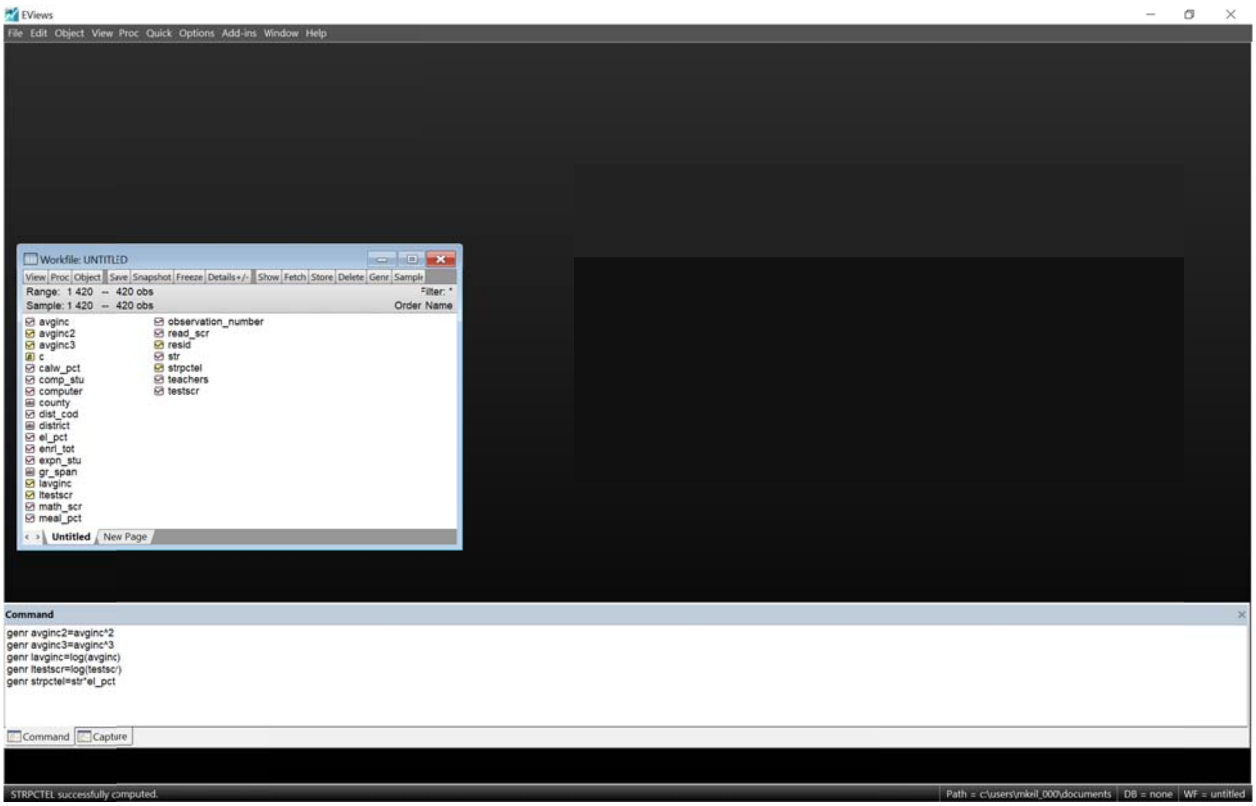
You can also save data in *ASCII*, spreadsheet, *STATA*, *SPSS*, and other formats by clicking on **File/ Save As** and then looking at the various options in "Save as type."

g) Data Transformations

So far, we have only used data that already existed in some file that we either created or used. Almost always, you will be required to transform some of the raw data that you received before you run a regression. In *EViews* you transform variables by using the "*genr*" (as in generate) command. For example, Chapter 8 of the Stock/Watson textbook introduces the polynomial regression model, logarithms, and interactions between variables. Let us reproduce Equations

(8.2), (8.11), (8.18), and (8.37) here. The following commands generate the necessary variables:[7]

```
genr avginc2=avginc^2
genr avginc3=avginc^3
genr lavginc=log(avginc)
genr ltestscr=log(testscr)
genr strpctel=str*el_pct
```



h) Multiple Regression Model

Economic theory most often suggests that the behavior of a certain variable is influenced not only by another single variable, but by a multitude of factors. The demand for a product depends not only on the price of the product but also on the price of other goods, income, taste, etc. Similarly, the Phillips curve suggests that inflation depends not only on the unemployment rate, but also on inflationary expectation and possibly supply shocks, etc.

---

[7] For example, I have generated a variable called "*avginc2*", and assigned it to be the square of the previously defined variable "*avginc*". Note that I am generating variable names that are somewhat self-explanatory. They could have been called "variable1", "variable2", "variable3", etc. but it is a good idea to create variable names that you can remember.

An extension of the simple regression model is the multiple regression model, which incorporates more than one regressor (see Equation (6.7) in the textbook on page 189).

$$Y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + ... + \beta_k X_{ki} + u_i\,,\ \ i = 1,\ldots,n.$$

To estimate the coefficients of the multiple regression model, you proceed in a similar way as in the simple regression model. The difference is that you now need to list the additional explanatory variables. In general, the command is:

ls(options) Y c X1 X2 … Xk

where *(options)* can be omitted, in which case the command becomes simply

ls Y c X1 X2 … Xk

This is the default version of the regression command, and it gives you homoskedasticity-only standard errors.

See if you can reproduce the following regression output, which corresponds to Column 5 in Table 7.1 of the textbook (page 224). The option used below is *(h)* to produce heteroskedasticity-robust standard error (*EViews* refers to these as "Huber-White-Hinkley (HC1) heteroscedasticity consistent standard errors and covariance").

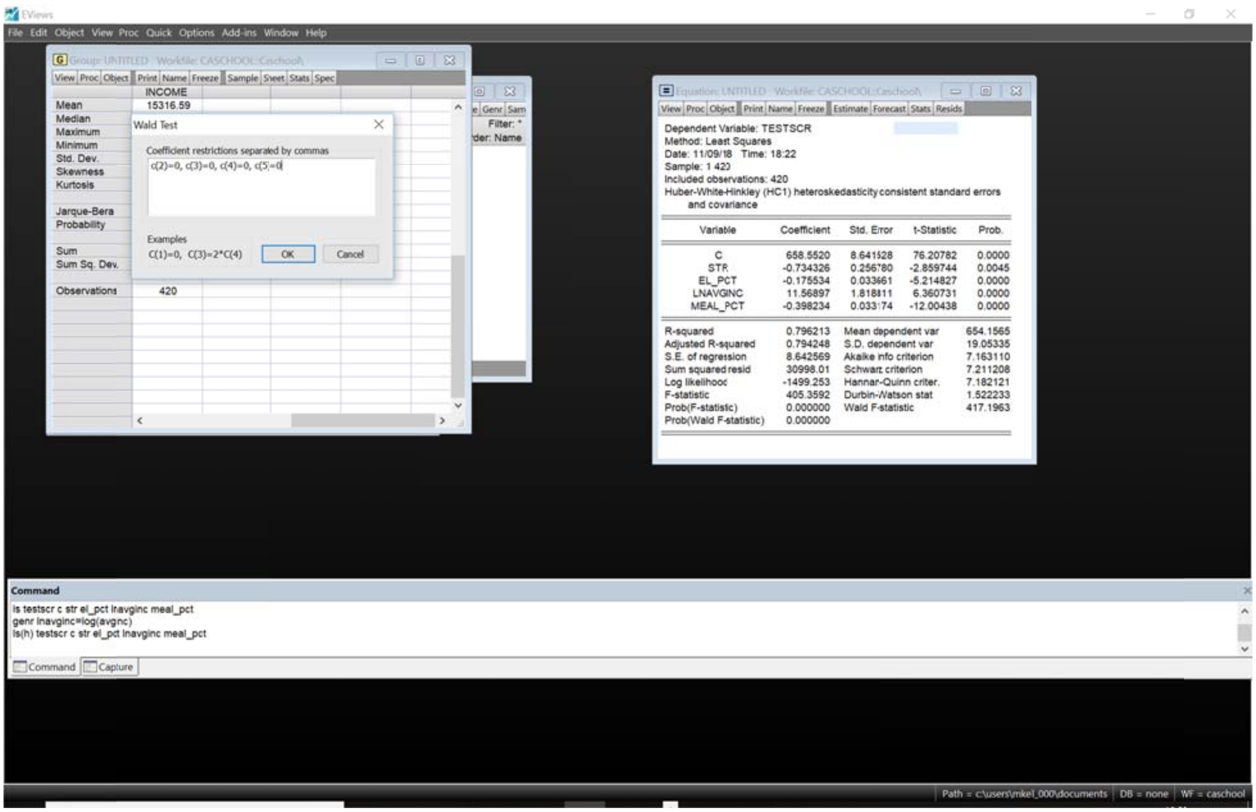
Dependent Variable: TESTSCR
Method: Least Squares
Date: 11/09/18 Time: 18:22
Sample: 1 420
Included observations: 420
Huber-White-Hinkley (HC1) heteroskedasticity consistent standard errors
and covariance

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	658.5520	8.641528	76.20782	0.0000
STR	-0.734326	0.256780	-2.859744	0.0045
EL_PCT	-0.175534	0.033661	-5.214827	0.0000
LNAVGINC	11.56897	1.818811	6.360731	0.0000
MEAL_PCT	-0.398234	0.033174	-12.00438	0.0000

R-squared	0.796213	Mean dependent var	654.1565
Adjusted R-squared	0.794248	S.D. dependent var	19.05335
S.E. of regression	8.642569	Akaike info criterion	7.163110
Sum squared resid	30998.01	Schwarz criterion	7.211208
Log likelihood	-1499.253	Hannan-Quinn criter.	7.182121
F-statistic	405.3592	Durbin-Watson stat	1.522233
Prob(F-statistic)	0.000000	Wald F-statistic	417.1963
Prob(Wald F-statistic)	0.000000		

The interpretation of the coefficients is equivalent to that of a controlled science experiment: it indicates the effect of a unit change in the relevant variable on the dependent variable, *holding all other factors constant* ("*ceteris paribus*").

Section 7.2 of the Stock and Watson (2018) textbook discusses the *F*-statistic for testing restrictions involving multiple coefficients. To test whether all of the above coefficients are zero with the exception of the intercept, click on **View/Coefficient Diagnostics/Wald Test-Coefficient Restrictions** (important: you will only see the options in ***View*** if you previously clicked on the regression output window). The regression coefficients are stored in a vector *c*(1) to *c*(*k*+1), where the number in parentheses indicates the order of appearance in the regression output. Thus in the example *c(1)* is the intercept or constant term, *c(2)* is the coefficient on STR, and so forth. To execute the above test, enter the following and press enter:



The computer will generate the following output:

Wald Test:
Equation: Untitled

Test Statistic	Value	df	Probability
F-statistic	417.1963	(4, 415)	0.0000
Chi-square	1668.785	4	0.0000

Null Hypothesis: C(2)=0, C(3)=0, C(4)=0, C(5)=0
Null Hypothesis Summary:

Normalized Restriction (= 0)	Value	Std. Err.
C(2)	-0.734326	0.256780
C(3)	-0.175534	0.033661
C(4)	11.56897	1.818811
C(5)	-0.398234	0.033174

Restrictions are linear in coefficients.

See if you can generate the *F*-statistic of 5.43 following Equation (7.6) in the Stock and Watson (2018) text and listed at the bottom of page 213 (restrict the coefficients of *STR* and *Expn* to be zero.[8]8

Next run the four regressions using the same technique as for multiple regression analysis. Finally save your workfile again and exit the workfile.

Exercise

One of the problems with the type of tutorial you are working on is that you just follow instructions without internalizing them. A typical student will finish the tutorial with few problems but then little is retained. If I asked you to retrieve a data set and to run a few regressions, for example, would you be able to do that? Or would you say "how do I do this?"

Let's see how much you understood. Go to the Stock and Watson website for the 3rd edition (http://www.pearsonhighered.com/stock_watson). Under *"Introduction to Econometrics, 3/e update" go* to the *Companion Website,* click on *Student Resources*, and download *CPS08_Data (Excel Dataset)*, which is the the CPS data set for Chapter 8 (*Data Sets for Replicating Empirical Results*: ***CPS Data Used in Chapter 8***). Then replicate the results for columns (1) from Table 8.1 on page 284 of the Stock and Watson (2018) textbook. Why do you think your results differ from those listed in the table? What if you found a way to restrict your sample to only include individuals who are at least 30 but not older than 64? To find a way to restrict your sample, look for **Help** and the *smpl if* command. Then restricting your sample to those individuals in that age group, replicate columns (1) to (3). For column (4), define *potential experience* as (*age – Years of education – 6*).

8 A word of caution here. In the above table, the *F*-statistic is 417.1963. In the regression output above, the same listed *F*-statistic is 405.3592, even though it tests for the same restrictions, namely that all slope coefficients are zero. Note that the latter statistic is the homoskedasticity-only *F*-statistic, even though the equation was estimated using heteroskedasticity-robust standard errors.

Batch Files

You will skip this section if you only have the Student Version, since this feature is not available for your version.

So far, you have either clicked on buttons in *EViews* or used the "Command Window" to type executable statements. But what if you wanted to keep a permanent record of all the transformations you made, regressions you tried, graphs you created, etc.? In that case, you would need to create a "program" that consists of line commands similar to those that you used in the "Command Window" previously. After having created such a program, you can then execute ("run") it and view the output afterwards (if you did not make any errors). Batch files can also include loops and conditional branching.

To create a program, click on **File** and then **New** and **Program**. This opens the "*Program*" box. Let's create a program. Type in the following commands exactly as they appear below:

```
open c:\StockandWatsons\caschool.wf1
genr income=avginc*1000
stats income
```

Here is the meaning of the three lines of this program:

Line 1: This line concerns the data set. As you learned earlier in the tutorial, datasets in *EViews* are called *wf1* files. The dataset which you will use here is *caschool.wf1*, which you downloaded earlier. The current line tells *EViews* the location and name of the dataset to be used for the analysis. Before you save the *Program-File*, replace the path in this line with the relevant path of the location where YOU saved *caschool.wf1* to.

Line 2: This line tells *EViews* to create a new variable call income. The new variable is constructed by multiplying the variable *avginc* by 1000. The variable *avginc* is contained in the dataset and is the average household income in a school district expressed in thousands of dollars. The variable *income* will be the average household income expressed in dollars instead of thousands of dollars.

Line 3: This line tells *EViews* to compute some summary statistics. *EViews* will produce the mean, standard deviation, etc.

As long as you have replaced the path in line 1 with the relevant path from the computer you are working on, and if you downloaded/saved the California Test Score Data Set, then we are good to go. Save the Program-file, using the *.prg* suffix and perhaps call the file *Tutorialch4.prg*. Next execute this *Program-File* by clicking on the **Run** button. This will "execute" or "run" the program. The results should produce the following information:

Date: 11/20/18
Time: 10:50
Sample: 1 420

	INCOME
Mean	15316.59
Median	13727.80
Maximum	55328.00
Minimum	5335.000
Std. Dev.	7225.890
Skewness	2.215156
Kurtosis	9.532125
Jarque-Bera	1090.186
Probability	0.000000
Sum	6432967.
Sum Sq. Dev.	2.19E+10
Observations	420

You now have an initial idea of how to work with *Program-Files* in *EViews*. The rest of this part of the tutorial will guide you through further commands and make the initial *Program-File* more complex. I suggest that you continue to work with the batch file you just created and then for you to add new lines to this program (if you use the *.pdf* version of this tutorial or have printed the tutorial using a color printer, then the new commands will appear in red).

```
'**********************************************************************************
' Stock and Watson, chapter 4-8 (EViews 10.0 Version)
'
' caschool.wf1 is the California School Data Set
' *********************************************************************************
' Read in the Dataset
' *********************************************************************************
open c:\StockandWatsons\caschool.wf1
' *********************************************************************************
' Transform Data and Create New Variables
' *********************************************************************************
' ****** Construct Average District Income in $s
genr income = avginc*1000
' *********************************************************************************
' Carry Out Statistical Analysis
' *********************************************************************************
' ******* Summary Statistics for Income
' *******************************************************************************
stats income
' *******************************************************************************
' End of Program
' *******************************************************************************
```

The new version of the Program-File carries out exactly the same calculations as before.

A new feature of the above *Program-File* is that many of the lines begin with a single quote (‘). EViews ignores the text that comes after ‘, so that these lines can be used for comments or to describe what the commands that follow are doing. Bottom line: Comments are useful if you want to remember later what you were doing or if you want others to understand your program. They do not affect the actual execution of the program

Next, change the previous version of the *Program-File* by adding commands until the new version looks as follows (again, new commands can be seen in red if your tutorial displays colors):

```
‘*****************************************************************************************
' Stock and Watson, chapter 4-8 (EViews 10.0 Version)
‘
‘ caschool.wf1 is the California School Data Set
‘ ****************************************************************************************
 ‘ Read in the Dataset
‘ ****************************************************************************************
open c:\StockandWatsons\caschool.wf1
‘ ****************************************************************************************
‘ Transform Data and Create New Variables
‘ ****************************************************************************************
‘ ****** Construct Average District Income in $s
genr income = avginc*1000
‘ ****** Define variables for subset of data
smpl if str<20
genr testscr_lo=testscr
smpl 1 420
smpl if str >= 20
genr testscr_hi = testscr
 ‘ ****************************************************************************************
‘ Carry Out Statistical Analysis
‘ ****************************************************************************************
' ******* Summary Statistics for Income and testscores
‘ ***************************************************************************************
smpl 1 420
stats income
stats testscr
group tab4_1 str testscr
tab4_1.stats
testscr.teststat(mean=0)
group testdiff20 testscr_lo testscr_hi
testdiff20.testbtw(mean)
‘ ***************************************************************************************
'* Repeat the Analysis using STR = 19
*****************************************************************************************
smpl if str<19
genr testscr_lo = testscr
smpl 1 420
smpl if str >= 19
```

```
genr testscr_hi = testscr
troup testdiff19 testscr_lo testscr_hi
testdiff19.testbtw(mean)

' ******************************************************************************************
' End of Program
```

There are three new features in this new version:

1) New variables are created using only a portion of the dataset. Two of the variables in the dataset are *testscr* (the average test score in a school district) and *str* (the district's average class size or student teacher ratio). The *Eviews* commands

```
smpl if str<20
genr testscr_lo=testscr
smpl 1 420
smpl if str >= 20
genr testscr_hi = testscr
```

generate new variables *testscr_lo* and *testscr_hi*. *testscr_lo* is only defined for districts that have an average class size of less than twenty students, that is, for which *str* < 20. Similarly, testscr_hi is only defined for districts that have an average class size of more than 20 students (*str* > 20).

The statement *str < 20* is an example of a "relational operation". *EViews* uses several relational operators:

<	**less than**
>	**greater than**
<=	**less than or equal to**
>=	**greater than or equal to**
<>	**not equal to**

2) There has been a small, but important, change in the summary statistics: you can have multiple variables displayed instead of showing them separately for income and testscores. However, if you want *EViews* to keep the summary statistics in your workfile, then you have to give the resulting output ("object") a name (here: *tab4_1*) and then ask *EViews* to display summary statistics for that object. The group command gives the object a name and then lists which variables below in the group. The *tab4_1.stats* command asks *EViews* to generate summary statistics for the variables in that group. After you run the program, you can click on the table you have generated in the workfile "caschool.wf1", and see the summary statistics output. If you save the workfile, the summary statistics table will be saved with it.

3) The *testscr.teststat(mean=0)* command asks *EViews* to test that the population mean is zero (in this example, the *t*-test that the population mean of test scores is equal to zero is not really of interest, but it shows you how *EViews* conducts *t*-tests for means in general).

The output is as follows:

Hypothesis Testing for TESTSCR
Date: 11/15/18 Time: 15:04
Sample: 1 420
Included observations: 420
Test of Hypothesis: Mean = 0.000000

Sample Mean = 654.1565
Sample Std. Dev. = 19.05335

Method	Value	Probability
t-statistic	703.6149	0.0000

The output shows the mean and the standard deviation of the variable testscr, and computes a t-test that the population mean is equal to zero. (Unfortunately it does not compute a 95% confidence interval for the population mean.)

4) The second command involving a t-test is *testdiff.testbtw(mean)*, which produces the following table:

Test for Equality of Means Between Series
Date: 11/15/18 Time: 15:23
Sample: 1 420
Included observations: 420

Method	df	Value	Probability
t-test	418	3.999193	0.0001
Satterthwaite-Welch t-test*	403.6070	4.042582	0.0001
Anova F-test	(1, 418)	15.99354	0.0001
Welch F-test*	(1, 403.607)	16.34247	0.0001

*Test allows for unequal cell variances

Analysis of Variance

Source of Variation	df	Sum of Sq.	Mean Sq.
Between	1	5605.547	5605.547
Within	418	146504.0	350.4881
Total	419	152109.6	363.0301

Executing the statement resulted in a test of the hypothesis that *testscr_lo* and *testscr_hi* come from populations with the same mean. That is, the command computes the *t*-statistic for the null hypothesis that the (population) mean of test scores for districts with class sizes less than 20 students is the same as the mean of test scores for districts with class sizes greater than 20 students. The "Satterthwaite-Welch t-test" reports the version of the test that assumes unequal variances in the two populations. *EViews* will also assume that the series are unpaired, meaning that the observations are for different districts: these are not panel data representing the same entity at two different time periods (see section 3.4 in Stock and Watson (2018)).

5) A last new feature in the *Program-File* is to show you how to replace data. This appears near the bottom of the file. Here, the analysis is to be carried out again, but using 19 as the cutoff for small classes. Note that the variables *testscr_lo* and *testscr_hi* already exist (they were defined by the *genr* command earlier in the program), *EViews* will "generate" variables with the same name by using the *genr* command and then the same variable name. There is no need for a separate replace command. In essence, the command instructs the program to overwrite the previously stored data.

You are now ready to execute ("run") the program as done before.

As before, change the previous version of the *Program-File* by adding commands until the new version looks as follows (again, new commands can be seen in red if your tutorial displays colors):

```
'*********************************************************************************************
' Stock and Watson, chapter 4-8 (EViews 10.0 Version)
'
' caschool.wf1 is the California School Data Set
' ********************************************************************************************
 ' Read in the Dataset
' ********************************************************************************************
open c:\StockandWatsons\caschool.wf1
' ********************************************************************************************
' Transform Data and Create New Variables
' ********************************************************************************************
' ****** Construct Average District Income in $s
genr income = avginc*1000
' ****** Define variables for subset of data
smpl if str<20
genr testscr_lo=testscr
smpl 1 420
smpl if str >= 20
genr testscr_hi = testscr
 ' ********************************************************************************************
' Carry Out Statistical Analysis
' ********************************************************************************************
' ******* Summary Statistics for Income and testscores
```

```
' ******************************************************************************************
smpl 1 420
stats income
stats testscr
'*****************************************************************************************
' ******* Table 4.1 ***********
'*****************************************************************************************
group tab4_1 str testscr
tab4_1.stats
' ****************************************************************************************
'******** Figure 4.2 and Figure 4.3 **********
'***************************************************************************************
group Fig4_2 str testscr
Fig4_2.scat
group Fig4.3 str testscr
Fig4_3.scat linefit
'***************************************************************************************
'********** Correlation *********
'***************************************************************************************
group cor_str_testscr str testscr
cor_str_testscr.cor
' ****************************************************************************************
' ********** Equation 4.11 and 5.8 ******
' ***************************************************************************************
equation eq4_11.ls(h) testscr c str
' ***************************************************************************************
' *********** Equation 5.18 ********
' ***************************************************************************************
genr dsize=0
smpl if str<20
genr dsize=1
smpl 1 420
equation eq4_33.ls(h) testscr c dsize
' ****************************************************************************************
group tab4_1 str testscr
tab4_1.stats
testscr.teststat(mean=0)
group testdiff20 testscr_lo testscr_hi
testdiff20.testbtw(mean)
' ******************************************************************************************
'* Repeat the Analysis using STR = 19
' *******************************************************************************************
smpl if str<19
genr testscr_lo = testscr
smpl 1 420
smpl if str >= 19
genr testscr_hi = testscr
group testdiff19 testscr_lo testscr_hi
testdiff19.testbtw(mean)

' ******************************************************************************************
' End of Program
```

The new commands reproduce some of the empirical results shown in Chapter 4 and 5 of Stock and Watson (2018). There are several features of *EViews* included in the new commands which have not been used in the previous examples.

1) The commands

```
group Fig4_2 str testscr
Fig4_2.scat
group Fig4.3 str testscr
Fig4_3.scat linefit
```

construct scatterplots of *testscr* versus *str* without (first two lines) and with (line 3 and 4) the estimated regression line for the simple regression of the California Test Score Data Set, shown in on pages 106 and 109 of Stock and Watson (2018).

2) The commands

```
group cor_str_testscr str testscr
cor_str_testscr.cor
```

tell *EViews* to compute the correlation between the student teacher ratio and test scores.

3) Next you will reproduce equations (4.11) and (5.8) in Stock and Watson (2018) by using the *ls* command:

```
equation eq4_11.ls(h) testscr c str
```

instructs *Eviews* to run an OLS regression with *testscr* as the dependent variable and *str* as the regressor. The robust (short *h*) option tells *EViews* to calculate heteroscedasticity-robust formulas for the standard errors of the regression coefficient estimators. Omitting this option results in the display of homoscedasticity-only standard errors. The output should looks as follows:

Dependent Variable: TESTSCR
Method: Least Squares
Date: 11/17/18 Time: 10:17
Sample: 1 420
Included observations: 420
Huber-White-Hinkley (HC1) heteroskedasticity consistent standard errors
and covariance

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	698.9330	10.36436	67.43619	0.0000
STR	-2.279808	0.519489	-4.388557	0.0000

R-squared	0.051240	Mean dependent var	654.1565
Adjusted R-squared	0.048970	S.D. dependent var	19.05335
S.E. of regression	18.58097	Akaike info criterion	8.686903
Sum squared resid	144315.5	Schwarz criterion	8.706143
Log likelihood	-1822.250	Hannan-Quinn criter.	8.694507
F-statistic	22.57511	Durbin-Watson stat	0.129062
Prob(F-statistic)	0.000003	Wald F-statistic	19.25943
Prob(Wald F-statistic)	0.000014		

4) The final innovation over the previous version of the *Program-File* is contained in the commands following the line Equation 5.18. First a binary (sometimes referred to as dummy or indicator) variable "d" is created suing the *EViews* commands

```
genr dsize=0
smpl if str<20
genr dsize=1
```

You first initialize the variable to take on the value of "0" for the entire sample, and then overwrite the observations with a "1" for which the class size is less than 20. The final *ls* command tells *EViews* to run a regression of test scores on the binary variable just created. The output reproduces equation (5.18) on p. xxx of Stock and Watson (2018).

Run the program now and look at the output.

The upcoming *Program-File* will be the last program for cross sectional data in this tutorial. Having understood all five should give you a solid grounding in programming. As before, there are several commands added to the previous version of the *Program-File*. Add these commands to your older version until the new version looks as follows (new commands can be seen in red if you tutorial displays colors):

```
'*******************************************************************************************
' Stock and Watson, chapter 4-8 (EViews 10.0 Version)
'
' caschool.wf1 is the California School Data Set
' *******************************************************************************************
```

```
 ‘ Read in the Dataset
‘ ****************************************************************************************
open c:\StockandWatsons\caschool.wf1
‘ ****************************************************************************************
‘ Transform Data and Create New Variables
‘ ****************************************************************************************
‘ ****** Construct Average District Income in $s
genr income = avginc*1000
‘ ****** Define variables for subset of data
smpl if str<20
genr testscr_lo=testscr
smpl 1 420
smpl if str >= 20
genr testscr_hi = testscr
 ‘ ****************************************************************************************
‘ Carry Out Statistical Analysis
‘ ****************************************************************************************
' ******* Summary Statistics for Income and testscores
‘ ***************************************************************************************
smpl 1 420
stats income
stats testscr
‘***************************************************************************************
‘ ******* Table 4.1 ***********
‘***************************************************************************************
group tab4_1 str testscr
tab4_1.stats
‘ *************************************************************************************
‘******** Figure 4.2 and Figure 4.3 **********
‘*************************************************************************************
group Fig4_2 str testscr
Fig4_2.scat
group Fig4.3 str testscr
Fig4_3.scat linefit
‘*************************************************************************************
‘********** Correlation *********
‘*************************************************************************************
group cor_str_testscr str testscr
cor_str_testscr.cor
‘ *************************************************************************************
‘ ********** Equation 4.11 and 5.8 ******
‘ ************************************************************************************
equation eq4_11.ls(h) testscr c str
‘ ************************************************************************************
‘ *********** Equation 5.18 ********
‘ ************************************************************************************
genr dsize=0
smpl if str<20
genr dsize=1
smpl 1 420
equation eq4_33.ls(h) testscr c dsize
‘ *************************************************************************************
group tab4_1 str testscr
tab4_1.stats
```

```
testscr.teststat(mean=0)
group testdiff20 testscr_lo testscr_hi
testdiff20.testbtw(mean)
' ******************************************************************************************
'* Repeat the Analysis using STR = 19
' ******************************************************************************************
smpl if str<19
genr testscr_lo = testscr
smpl 1 420
smpl if str >= 19
genr testscr_hi = testscr
group testdiff19 testscr_lo testscr_hi
testdiff19.testbtw(mean)
' ******************************************************************************************
' ********* Table 6.1 **********
' ******************************************************************************************
' initializing variables for table
' ******************************************************************************************
smpl 1 420
genr str_20 = 0
genr elq1 = 0
genr elq2 = 0
genr elq3 = 0
genr elq4 = 0
' ******************************************************************************************
' Creating Binary Variables
' ******************************************************************************************
smpl if str<20
genr str_20 = 1
smpl 1 420
smpl if str_20 = 1
genr ts_lostr = testscr
smpl 1 420
smpl if str_20 = 0
genr ts_histr = testscr
smpl 1 420
smpl if el_pct < 1.9
genr elq1 = 1
smpl 1 420
smpl if el_pct >= 1.9 and el_pct < 8.8
genr elq2 = 1
smpl 1 420
smpl if el_pct >= 8.8 and el_pct < 23.0
genr elq3 = 1
smpl 1 420
smpl if el_pct >= 23.0
genr elq4 = 1
' ******************************************************************************************
' running differenece in means tests
' ******************************************************************************************
smpl 1 420
group Table_6_1_test_1 ts_lostr ts_histr
Table_6_1_test_1.testbtw(mean)
smpl if elq1 = 1
```

```
group Table_6_1_test_2 ts_lostr ts_histr
Table_6_1_test_2.testbtw(mean)
smpl 1 420
smpl if elq2 = 1
group Table_6_1_test_3 ts_lostr ts_histr
Table_6_1_test_3.testbtw(mean)
smpl 1 420
smpl if elq3 = 1
group Table_6_1_test_4 ts_lostr ts_histr
Table_6_1_test_4.testbtw(mean)
smpl 1 420
smpl if elq4 = 1
group Table_6_1_test_5 ts_lostr ts_histr
Table_6_1_test_5.testbtw(mean)
‘ ****************************************************************************************
‘ ******* Equation 7.5 *************
‘ ****************************************************************************************
smpl 1 420
equation eq7_5.ls(h) testscr str el_pct
eq7_5.output
‘ ******************************************************************************************
‘ ******* Equation 7.6 **************
‘ ***************************************************************************************
genr expn_stu = expn_stu/2000
equation eq7_6.ls(h) testscr c str expn_stu el_pct
eq7_6.output
‘ **************************************************************************************
‘ Display Variance-Covariance Matrix
‘ ***************************************************************************************
eq7_6.coefcov
' ***************************************************************************************
' ******* F-test report in text
‘ **************************************************************************************
eq7_6.wald c(2) = 0, c(3) = 0
‘ ****************************************************************************************
' ******** Correlations reported in text
‘ ***************************************************************************************
group cor_str_otherrhsvar str expn_stu el_pct meal_pct calw_pct
cor_str_otherrhsvar.cor
‘ ***************************************************************************************
‘ ******** Table 7.1 *********
‘ ***************************************************************************************
‘ Column (1)
equation eq_tab7_1_col1.ls(h) testscr c str
eq_tab7_1_col1.output
‘ Column (2)
equation eq_tab7_1_col2.ls(h) testscr c str el_pct
eq_tab7_1_col2.output
‘ Column (3)
equation eq_tab7_1_col3.ls(h) testscr c str el_pct meal_pct
eq_tab7_1_col3.output
‘ Column (4)
equation eq_tab7_1_col4.ls(h) testscr c str el_pct calw_pct
eq_tab7_1_col4.output
```

```
' Column (5)
equation eq_tab7_1_col5.ls(h) testscr c str el_pct meal_pct calw_pct
eq_tab7_1_col5.output
' ***************************************************************************************
' ******* homoscedasticity only F-Statistic
' ***************************************************************************************
equation eq_homosk.ls testscr str expn_stu el_pct
equation eq_homosk1.ls testscr c el_pct
' **************************************************************************************
' * End of Program
' *************************************************************************************
```

The file produces several of the empirical results from Chapter 7 of Stock and Watson (2018).

In essence there is only one new command:

1) The first new command involves the test of restrictions in equation 7.6 (page xxx of Stock and Watson (2018)). The command

```
equation eq7_6.ls(h) testscr c str expn_stu el_pct
' ***************************************************************************************
' Display Variance-Covariance Matrix
' ***************************************************************************************
eq7_6.coefcov
```

instruct *EViews* to compute the regression. The command *coefcov* asks *EViews* to print out the estimated variances and covariances of the estimated regression coefficients. The command

```
eq7_6.wald c(2) = 0, c(3) = 0
```

gets *EViews* to carry out the joint test that the coefficients on *str* and *expn_stu* are both equal to zero.

The output will be as follows:

Wald Test:
Equation: EQ7_6

Test Statistic	Value	df	Probability
F-statistic	4301.658	(2, 416)	0.0000
Chi-square	8603.315	2	0.0000

Null Hypothesis: C(1) = 0, C(2) = 0
Null Hypothesis Summary:

Normalized Restriction (= 0)	Value	Std. Err.
C(1)	649.5779	15.45834
C(2)	-0.286399	0.482073

Restrictions are linear in coefficients.

You will find other examples of *Program-Files* in the accompanying Website for the Stock and Watson (2018) econometrics textbook. You can download *EViews Program-Files* from there to reproduce all of the analysis in Chapters 3-13. You will also find an *EViews* Program File for the time series chapters 15-17 there. *EViews* programming for time series is somewhat different than for cross-sectional of panel data, and *EViews* is particularly useful when working with time series data.

A summary of frequently used *EViews* commands is given at the end of the tutorial. As an exercise, start generating the equations, graphs, and tables of chapters 5-9 in Stock and Watson (2018).

3. TIME SERIES DATA

Many of you using *EViews* will do so since you are studying time series, or data of a single entity across time. An example may be consumption in the U.S. from 1955 to now. Time series analysis is used in Stock and Watson (2018) in chapters 15 to 17.

One major difference using time series data is that the order of the observations matter: you would lose much information if you "shuffled the deck" so to speak. In other words, while the order of observations does not matter in cross sectional data, with time series it is very important. Next, much information about the current behavior of a variable is contained in previous values of the same variable, or its "lags." We refer to "(t-1)" as a one period *lag* (similarly, "(t+1)" is a *lead*). In general, there are *(t-j)* lags that can be calculated (see Key Concept 15.1 on page 515 in Stock and Watson (2018)).

On the Stock and Watson (2018) website, you will find quarterly data for real GDP. Using the same methodology as discussed earlier in the tutorial, open the Excel spreadsheet and copy the data for real GDP into an EViews data file. In order to do this, open a new *workfile* ("Open a New Workfile"), chose **Frequency Quarterly** under **Data Specification**, and designate 1955:1 and 2017:4 as the start and end date. Next cut and paste the *Excel* data for the *GDPC1* column into the new group ("**Quick**" then "**Empty Group (Edit Series)**". Rename the variable *gdpr* (close the group, right click on *ser01*, then **rename**). Next save the workfile and name it "ch15usmacro.wf1".

The econometric task in Ch. 15 is to forecast the real GDP growth rate. So far, we have only read in the real GDP level and need to convert it into an annualized rate of GDP growth. We do this through the following transformation:

$$g_{Y,t} = \frac{GDP_t - GDP_{t-1}}{GDP_{t-1}} \times 400 = \left(\frac{GDP_t}{GDP_{t-1}} - 1 \right) \times 400 \approx 400 \times \left[\ln(GDP_t) - \ln(GDP_{t-1}) \right] = 400 \Delta \ln(GDP_t)$$

Here is the first program file for you to create:

```
'*********************************************************************************************
' Stock and Watson, chapter 15 (EViews 10.0 Version)
'
' ch15usmacro.wf1 is the California School Data Set
' ********************************************************************************************
 ' Read in the Dataset
' ********************************************************************************************
open c:\StockandWatsons\ ch15usmacro.wf1

' ********************************************************************************************
' Transform Data and Create New Variables
' ********************************************************************************************
' ****** Construct growth rate of Real GDP
' ****************************************************************************************
smpl 1955:1 2017:4
genr lgdpr = log(gdpr)
genr gdpgr = 400*(lgdpr-lgdpr(-1))
' ****************************************************************************************
' ****** Table 15.1 *******
' ****************************************************************************************
smpl 2016:4 2017:4
show gdpr lgdpr gdpgr gdpgr(-1)
' ***************************************************************************************
' * End of Program
' ***************************************************************************************
```

Differently from the cross sectional analysis, the sample command now indicates the year and the quarter. Hence we do our transformations for the entire sample period 1955Q1 to 2017Q4.

The next two lines create the log of real GDP, the lag of real GDP, and the growth rate as in equation (15.1) of the Stock and Watson (2018) textbook. To create past values of variables, you generate a lag by adding a "(-1)" after its variable name in the "*genr*" statement. In a spreadsheet, this amounts to copying an entire data series and pasting it into a new column one observation down: the first observation becomes the second observation, etc. The procedure generalizes to higher lags: X_{t-12} is X(-12).[9]

The final command reproduces the values of Table 15.1 on p. 516 of the textbook.

	GDPR	LGDPR	GDPGR	GDPGR(-1)
2016Q4	16851.42	9.732190	1.743040	2.742267
2017Q1	16903.24	9.735261	1.228157	1.743040
2017Q2	17031.085	9.742795	3.013954	1.228157
2017Q3	17163.894	9.750563	3.107115	3.013954
2017Q4	17271.702	9.756825	2.504579	3.107115

Add the following lines of code (listed in red) into the program-file and run it.

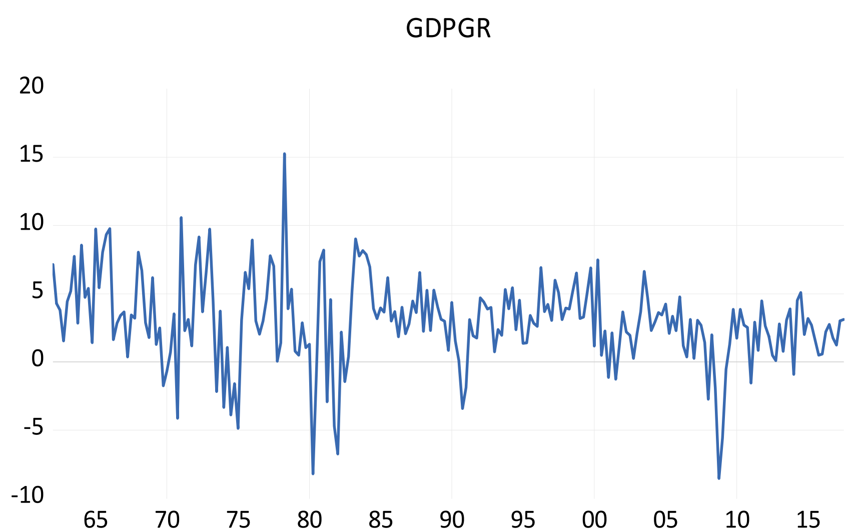
```
'**************************************************************************************
' Stock and Watson, chapter 15 (EViews 10.0 Version)
'
' ch15usmacro.wf1 is the California School Data Set
' ************************************************************************************
 ' Read in the Dataset
' ************************************************************************************
open c:\StockandWatsons\ ch15usmacro.wf1

' ************************************************************************************
' Transform Data and Create New Variables
' ************************************************************************************
' ****** Construct growth rate of Real GDP
' *********************************************************************************
smpl 1955:1 2017:4
genr lgdpr = log(gdpr)
genr gdpgr = 400*(lgdpr-lgdpr(-1))
' *********************************************************************************
' ****** Table 15.1 *******
' *********************************************************************************
smpl 2016:4 2017:4
show gdpr lgdpr gdpgr gdpgr(-1)
' ********************************************************************************
' ********** Figure 15.1
```

[9] In mathematics, a lag is defined (loosely) through the use of a "lag-operator" L, where $L^i x_t = x_{t-i}$. Similarly, the "difference operator" $\Delta = (1 - L)$, so that $\Delta x_t = x_t - x_{t-1}$. See Appendix 15.3 of the textbook for more details.

```
'*********************************************************************************************
smpl 1960:1 2017:4
group Fig15_1a lgdpr
Fig15_1a.line(m)
freeze(logRDGP) Fig15_1a
group Fig15_1b gdpr
Fig15_1b.line(m)
freeze(gdprg) Fig15_1b
' *****************************************************************************************
' AR(1) and AR(2) regressions
' ****************************************************************************************
smpl 1962:1 2017:3
equation eq15_9.ls gdpgr c gdpgr(-1)
eq15_9.output
equation eq15_11.ls gdpr c gdpgr(-1 to -2)
eq15_11.output
'****************************************************************************************
' * End of Program
' ****************************************************************************************
```

Here is what the new commands do:

1) *Fig15_1a* and *Fig15_1b.line(m)* produce a time series plot, meaning they plot the two series against time on the horizontal axis. The freeze command allows you to store the graph permanently, possibly to cut and paste it into a Word document. The graph should look as follows:



2) You should be familiar with the ls command from the cross sectional part of the tutorial.

As before, we have used heteroscedasticity-robust standard errors. You may want to compare the output to equation (15.11) in Stock and Watson (2018).

Dependent Variable: GDPGR
Method: Least Squares
Date: 11/19/18 Time: 10:18
Sample: 1962Q1 2017Q3
Included observations: 223
Huber-White-Hinkley (HC1) heteroskedasticity consistent standard errors and covariance

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	1.602751	0.371611	4.312978	0.0000
GDPGR(-1)	0.279235	0.076899	3.631174	0.0004
GDPGR(-2)	0.176734	0.077253	2.287720	0.0231

R-squared	0.145348	Mean dependent var	2.969795
Adjusted R-squared	0.137578	S.D. dependent var	3.244147
S.E. of regression	3.012731	Akaike info criterion	5.056933
Sum squared resid	1996.840	Schwarz criterion	5.102769
Log likelihood	-560.8480	Hannan-Quinn criter.	5.075436
F-statistic	18.70732	Durbin-Watson stat	2.005166
Prob(F-statistic)	0.000000	Wald F-statistic	13.36954
Prob(Wald F-statistic)	0.000003		

4. SUMMARY OF FREQUENTLY USED EVIEWS COMMANDS

The command '*genr*' creates new variables and modifies existing variables.

Examples:

```
genr expn=expn_stu/1000
```

generates the expenditure variable used in the textbook by dividing the original data by 1,000.

```
genr avginc2=avginc^2
genr lavginc=log(avginc)
```

create the square and log of average income, respectively.

Note that commands of the type

```
genr testscr = testscr/100
```

simply modify an existing variable.

The most frequently used operators are + (addition), - (subtraction), * (multiplication), / (division), ^ (exponentiation). Log(x) calculates the natural logarithm of x (see the above example) and exp(x) computes the exponent of x.

When working with time series data, lags are frequently used. EViews allows you to create these simply by entering (-i) immediately after the variable name:

```
genr dinf=inf-inf(-1)
genr yeardinf=inf-inf(-4)
```

The first command generates the quarterly change in the inflation rate (assuming that you work with quarterly data), while the second generates the annual change in the inflation rate.

The sample range is set through the 'smpl' command. The command is of the type: smpl n1 n2, where n1 and n2 are the beginning and end dates (first and last observations) for which EViews will execute the commands that follow.

Examples are

```
smpl 1 420
smpl 1959:1 2001:4
```

In the first case, EViews is instructed to use all 420 observations of the California Test Score Data Set used in Chapters 4-9. The second example restricts the sample to the first quarter of 1959 to the last quarter of 2001.

Note that you can work with a subsample by using relational operators.

```
smpl if str>=20
```

only looks at observations with a student-teacher ratio of less than 20.

The most frequently used *statistical operations* involve running regressions ('*ls*'), establishing the correlation between variables ('*cor*'), and graphing variables ('*line*').

EViews creates results by storing them in so-called objects. Initially, you will use the 'equation' object and the 'group' object most often, as in the following examples:

```
equation eq4_7.ls(h) testscr c str
equation eqtab5_2_5.ls(h) testscr c str el_pct meal_pct calw_pct
equation eq12_7.ls(h) dinf c dinf(-1)
```

In each case, an equation object is declared first and a name is assigned to it. '*ls*' then instructs EViews to use OLS estimation for the equation. The dependent variable appears first, followed by the regressors, where 'c' is used for the intercept ('c' is a reserved name in EViews, meaning that you cannot use it to generate a variable called 'c').

To create a line graph or to view the correlation between variables, you first must assign the variables to a group and name this group. Next, you execute the correlation and graphing through the 'cor' and 'line' command. Examples:

```
group cor_str_testscr str testscr
cor_str_testscr.cor
```

Here the variables *str* (student-teacher ratio) and *testscr* (test score) are assigned to a group called *cor_str_testscr* (the name was chosen to indicate what the group was used for, but we could have named it almost anything alternatively), and EViews is then instructed to calculate the correlation between the variables in the group. The group can contain more than two variables.

In the following example, *inf* (inflation) and *lhur* (unemployment rate) are assigned to a group called Fig12_1 and are then plotted (where '*m*' is an option that allows for the display of multiple graphs).

```
group Fig12_1 inf lhur
Fig12_1.line(m)
```

Another topic you may come across is statistical distribution functions. EViews provides functions that provide access to the density or probability functions, cumulative distribution, quantile functions, and random number generators for a number of standard statistical distributions. For example, the following command would tell show you that χ_4^2 $\Pr(Y \leq 7.78) =$

0.90:

```
genr result=@cchisq(7.78,4)
```

For a complete table and descriptions of distribution functions you can use, search for "statistical distribution functions" under the **Help** section.

5. FINAL NOTE

For a complete list of commands, consult the *EViews Command and Programming Reference* or the *User's Guide*. Alternatively, use the "**Help**" command inside *EViews*. Under the **Find** tab in **EViews Help Topics**, you can search for whatever you are looking for. As mentioned before, this tutorial is not intended to replace the *Reference* or *User's Guide*. The best way to learn how to use the program is to spend some time exploring and playing with it.

EViews replication batch files for all the results in the Stock/Watson textbook are available from the Web site. You are invited to download these and study them.