Replication: Chetty, Looney, and Kroft (2009)

ECON 7670, Spring 2023
Instructor: Elliott Isaac

Name: _____

The intent of this assignment is to get you familiar with Stata's variable generation, regression, and table and figure automation functions.

Grading

The grading rubric is below:

	5 points	2 points	1 point	0 points
Stata replication	I am able to completely replicate your log file by changing to my own directory and hitting "Do" without any additional work	I am able to completely replicate your log file by changing to my own directory and hitting "Do" after correcting a few mistakes	I am able to replicate parts of your log file after correcting several mistakes	I am not able to replicate your log file
Replication accuracy		The output in your log file perfectly replicates the analysis and table and figure formatting	The output in your log file is close, but not identical to, the analysis and/or table and figure formatting	The output in your log file is very different than the analysis and/or table and figure formatting
LaTeX integration	I can perfectly construct a PDF from your LaTeX file by hitting "Compile" without any additional work	I can construct a PDF from your LaTeX file by hitting "Compile" after correcting a few mistakes	I can construct parts of a PDF from your LaTeX file by hitting "Compile" after correcting several mistakes	I am not able to construct a PDF from your LaTeX file

Submission

You will submit the following through Canvas:

1. Your .do file named lastname.do
2. Your .log file named lastname.txt
3. Your compiled LaTeX file named lastname.pdf

You will **not** submit the tables and figures you export from Stata. I will recreate your tables and figures by executing your .do file. You will also **not** submit the .tex file you use to compile your PDF (I have a copy). I will recreate your PDF by compiling the .tex file with the tables and figures your code generates.

Introduction

Obtaining replicable results and automating tables and figures will be invaluable in the future. Many journals now require authors to submit replication packages with their paper and some journals will refuse to publish your paper if they cannot perfectly replicate your results even if your paper has already been accepted.

Using Stata to automatically create tables and figures will greatly decrease time spent updating them if results change and greatly reduce the probability of incorrectly inputting results into the paper.

The purpose of this assignment is to teach you how to do the following in Stata:

1. Write a stand-alone .do file that replicates all analysis in a single step
2. Calculate summary statistics
3. Store estimation results with additional test statistics
4. Automatically build and export tables of summary statistics and regression results that can be immediately plugged into LaTeX
5. Automatically construct and export figures
6. Integrate your automated tables and figures into a LaTeX document

Part 1: Preparing Your Workspace

Concepts in this section:
- Directory
- File path
- Do-file
- Log file

Stata commands used in this section:
- cd (change directory)
- log using
- log close

Create a folder on your computer where you will store the raw data and all of your files for this assignment. This folder is called your **directory**. The location of your directory is called the **file path**. For example, my directory folder is called "clk2009_replication" and it is stored on my desktop, so the file path is:

```
/Users/elliottisaac/Desktop/clk2009_replication
```

In your directory, do the following:
1. Save your .do file as lastname.do
2. Save the dataset
3. Save your .tex file as lastname.do

In your .do file, do the following:
1. Copy your file path into the `cd` command
2. Change the name of the .log file to your own last name

Part 2: Building the Dataset

Stata commands in this section:

- `use`
- `tsset`
- `gen` with `ln()` function, `d.` notation, `()` function (true/false), and `round()` function
- `label variable`
- `tab` with `gen` option
- `egen` with `mean` function and `bysort` prefix

In your .do file, write code that:

1. Loads your dataset using the `use` command.
2. Sets the panel variable as *state* and time variable as *year* using the `tsset` command.
3. Generates the following variables:

Variable name	**Definition**
beer_per_cap	Beer consumption per-capita in cans. *c_beer* is total consumption in gallons, *population* is number of people, and there are $\frac{24}{2.25}$ cans of beer in 1 gallon.
d21	Variable equal to 1 if the state's drinking age is 21 or older. *da* is the state's drinking age in years.
anybac	Variable equal to 1 if the state uses either a 0.10 or 0.08 blood alcohol content (BAC) standard. bac_10 equals 1 if they use a 0.10 standard and *bac_08* equals 1 if they use a 0.08 standard.
policy_change	Variable equal to 1 if there is any state policy change. *D_driving*, *d_bac_t*, *d_lower*, *d_bac_10*, *d_bac_08*, and *d_admin* are equal to 1 if there is a state change in any of those policies.
st_income_cap	State income per capita. *st_income* is total state income, and *population* is number of people.
ln_btax_dollars	The natural log of 1+(*btax_dollars*/9.9). *btax_dollars* is the beer excise tax in dollars per 24-pack of beer and 9.9 is the average price of a 24-pack of beer between 1970-2003.
fraction_beer	The natural log of .045**c_beer*/(.045**c_beer*+.129**c_wine*+.41**c_spirits*).

	This is a calculation of the fraction of total ethanol consumption in each state and year that is accounted for by beer.
*ln_** variables	The natural log of the following variables: 1+(*beer_tax*/100), 1+(*salestax*/100), *beer_per_cap*, *population*, *st_income_cap*, and *st_uemp_rate*.
*dln_** variables	The 1 year difference of the following variables: *ln_beer_tax*, *ln_salestax*, *ln_beer_per_cap*, *ln_population*, *ln_st_income_cap*, *ln_st_uemp_rate*, *ln_btax_dollars*, and *ln_fraction* beer. After setting the panel and time variables you can use *d.* notation as follows: `gen dln_variable = d.ln_variable` to generate the 1 year difference.
*rnd_** variables	Rounded values of *dln_beer_tax* and *dln_salestax*, rounded to the nearest 0.001.
*meanbpc_** variables	Mean of *dln_beer_per_cap* within each value of *rnd_**
exempt	Variable equal to 1 if state has a statewide sales tax and exempted food from the sales tax. Alaska, Delaware, Montana, New Hampshire, and Oregon do not have sales tax.
*year** variables	A set of indicator variables for the year. You can use *i.* notation with the `reg` command, but you cannot use it with the `ivreg2` command, so you will have to create and include the set of indicator variables manually.

Part 3: Calculating summary statistics

Stata commands in this section:

`summarize` with the `estpost` and `eststo` prefixes

The `summarize` command will calculate summary statistics, but we also want to store those statistics so we can use the `estout` command to automatically generate a summary statistics table.

The estout command needs two things: 1) a regression output, and 2) for that output to be stored in Stata's memory.

Using the `estpost summarize` command will tell Stata to treat the summarize output as estimation output. Adding the `eststo: estpost summarize` prefix will additionally store the summarize output as a regression output in Stata's memory. You can also name your output with a custom name using `eststo customname: estpost summarize`.

`eststo` (part of the `estout` package) is user-written and you will need to download it.

In the Stata command window:

1. Install `estout` by typing `ssc install estout`

In your .do file, write code that:

1. Calculates summary statistics for *beer_per_cap*, *btax_dollars*, *beer_tax*, *salestax*, *d21*, *anybac*, and *policy_change* using the `eststo: estpost summarize` command

Part 4: Running Regressions for Table 6

Stata commands in this section:

- `reg`
- `test`
- `eststo` with `add` option

The `reg` command runs OLS regressions in Stata and the `ivreg2` command runs IV regressions. `ivreg2` is user-written and you will need to download it.

In the Stata command window:

1. Install `ivreg2` by typing `ssc install ivreg2`

Start with the regressions in Table 6 of Chetty, Looney, and Kroft (2009). The dependent variable is *dln_beer_per_cap*, but the set of control variables differ. Note that the alcohol regulation controls are *d_lower_limit*, *d_bac_teen_02*, *d_admin_license_rev*, *d_driving_age*, *d_bac_10*, and *d_bac_08*.

In your .do file, write code that:

1. Runs the first regression in Table 6, Column 1 using the `reg` command.
2. Tests whether the *dln_beer_tax* and *dln_salestax* coefficients are equal using the `test` command.
3. Stores the regression output and the test result using the `eststo` command with the `add` option (the p-value of the test result is stored as **r(p)** after you run the test).

Repeat steps 1-3 for the other regressions. Since the only difference in coding for the different regressions is the control variables, see if you can use a `foreach` loop so that you only type the `reg`, `test`, and `eststo` commands once. Using loops this way decreases the change that you forget to update a regression if you need to change something that is common to all regressions.

Part 5: Running Regressions for Table 7

Stata commands in this section:

- `reg` with *d.* notation and `if` statement
- `ivreg2`
- `test`
- `eststo` with `add` option

Now the regressions start to get different.

If you run an IV regression using the `ivreg2` command then you need to specify which variable is endogenous and which is your instrument by adding `(endogvar = instrumentvar)` at the end of your control variables.

If you want to use 3-year differences then you can specify that by typing *d3.varname* as your variable, which is possible because you set the panel and time variable using `tsset` earlier. In the `test` command, refer to these variables as *D3.varname*.

If you want to restrict the sample to a subsample, you can add `if` after your control variables followed by the conditions that must be met to be included in the sample.

In your .do file, write code that:

4. Runs the regressions in Table 7, Columns 1, 2, 4, and 5 using the `reg` or `ivreg2` command.
5. Tests whether the *dln_beer_tax* and *dln_salestax* coefficients are equal using the `test` command (or the *D3.ln_beer_tax* and *D3.ln_salestax* variables).
6. Stores the regression output and the test result using the `eststo` command with the `add` option (the p-value of the test result is stored as **r(p)** after you run the test).

There are still substantial similarities between these regressions even though the regression command, outcome variable, control variables, and `if` statement sometimes differ. See if you can figure out how to define **locals** for those things so that you can write another `foreach` loop and only type the `reg` or `ivreg2`, `test`, and `eststo` commands once.

Part 6: Creating and Exporting Tables

Stata commands in this section:

`estout`

List of estout options you probably need:

`cells()`
`stats()`
`keep()`
`rename()`
`varlabels()`
`mlabels()`
`mgroups()`
`collabels()`
`posthead()`
`indicate()`
`style(tex)`
`replace`

The `estout` command is a very power user-written command to create, format, and export tables. It contains a lot of options that allow for nuanced table formatting so that your exported tables can be immediately incorporated into LaTeX files without copy/pasting anything.

`estout` contains a sub-command, `esttab`, that has fewer formatting options but is more useful for exporting tables to incorporate into Excel files (which you can then link into Word files so they update automatically).

We will use `estout` and incorporate our tables into a LaTeX file. I encourage you to learn and use LaTeX because it has a lot of formatting options that make papers look professional and it is much easier to write math and equations than Word.

I have provided clk2009_replication_key.pdf, which displays what each of the tables and figures you create should look like.

In the Stata command window:

1. Type `help estout` to open the `estout` help file, which explains the syntax and options for the `estout` command.

In your .do file, begin with the following code, replacing sumstats with the name of your stored summary statistics output

```
estout sumstats using table5.txt, cells(mean() sd(par))
```

Open your LaTeX file. The LaTeX file is already set up to import table5.txt. Compile the LaTeX file and check out the table formatting.

Start simple. Gradually add the options listed above and experiment with them to adjust the formatting of the table until it matches the formatting in clk2009_replication_key.pdf.

When you are done with Table 5, move on to Tables 6 and 7 using your stored estimation output.

Some tips:

You can write in math mode in-line in LaTeX by enclosing your text in $$. For example, LaTeX will transform `$\Delta$` into Δ.

You can make a header cell span multiple columns using `\multicolumn{n}{c}{cell text}` where `n` is the number of column to span and `c` indicates centered alignment.

You can add a horizontal line across the entire table by adding `\hline` to the end of the line. Or you can add a horizontal line across only some columns using `\cmidrule(lr){x-y}`, where `x` is the first underlined column and `y` is the last underlined column.

You can add space between rows using the `elist` suboption for the `varlabels` option and adding `"[0.3cm]"` to the end of each line (or another value depending on how big of a gap you need).

Part 7: Creating and Exporting Figures

Stata commands in this section:

- `twoway scatter` with `msize` and `mcolor` options
- `twoway lfit` with `lcolor` option
- `graph export`

List of `twoway` options you probably need:

- `name()`
- `ytitle()`
- `xtitle()`
- `graphregion()`
- `ylabel()`
- `xlabel()`
- `legend()`

The twoway command is Stata's built-in graph command. It can create many different types of graphs and can overlay graphs on top of each other in a single figure (which is what we will do here).

I have provided clk2009_replication_key.pdf, which displays what each of the tables and figures you create should look like.

In the Stata command window:

1. Type `help twoway` to open the `twoway` help file, which explains the syntax and options for the `twoway` command as well as for each type of graph.

In your .do file, begin with the following code:

```
twoway (scatter meanbpc_dln_beer_tax rnd_dln_beer_tax) (lfit meanbpc_dln_beer_tax rnd_dln_beer_tax)
```

Enclosing `scatter` and `lfit` in their own parentheses with a single `twoway` command overlays the two graphs on top of each other so that we get a scatter plot with a line of best fit.

Start simple. Gradually add the options listed above and experiment with them to adjust the formatting of the figure until it matches the formatting in clk2009_replication_key.pdf.

Use the `graph export` command to save your graph as a .pdf file.

When you are done with Figure 2A, move on to Figure 2B.

Some tips:

The `msize`, `mcolor`, and `lcolor` options are for specific graphs and need to be enclosed within that graph's parentheses, but the other options are for the `twoway` command overall and need to come after the graphs in the parentheses. For example, `twoway (scatter y x, mcolor(black))` will create a scatter plot with black markers, but `twoway (scatter y x), mcolor(black)` will produce an error.

You should restrict your graphs to values with -0.2 < *rnd_dln_beer_tax* < 0.2 or -0.2 < *rnd_dln_salestax* < 0.2, respectively, to match the author's figures.

Part 8: Incorporating Tables and Figures into LaTeX

Download the `Student LaTeX Template.tex` file from Canvas. You can either open this in your own TeX editing software or open it with a text editor (e.g., Notepad in Windows or TextEdit in Mac) and copy/paste it to a blank document on Overleaf.com.

The `Student LaTeX Template.tex` file is already set up to directly import your tables and figures into the .tex file without you needing to change anything. I am providing this .tex file to you as a template for how you can set things up in the future.

A few things that this .tex file does that may be useful in the future:

1. Creates a bibliography using the biblatex package.
2. Expands the notes area underneath tables to be the page width even if the table is narrow.
3. Uses the `\input{}` and `\includegraphics{}` commands to pull tables and figures into the document without manually typing.
4. Uses the `\adjustbox{}` command to adjust the size of tables (not necessary in this case, but can be useful if you have very wide or tall tables to get them to fit on the page).

If you use your own TeX editing software on your computer:

1. Place the .tex file in your directory.
2. Make sure Stata is saving your table and figure output to your directory (not to a sub-folder).
3. Compile.

If you use Overleaf.com:

1. Copy the `Student LaTeX Template.tex` file into a blank document.
2. Upload your tables and figures to the project.
3. Compile.